Mitsuru Ikeda
Kevin D. Ashley
Tak-Wai Chan (Eds.)

LNCS 4053

Intelligent Tutoring Systems

8th International Conference, ITS 2006
Jhongli, Taiwan, June 2006
Proceedings

Springer

Lecture Notes in Computer Science 4053

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Mitsuru Ikeda Kevin D. Ashley
Tak-Wai Chan (Eds.)

Intelligent Tutoring Systems

8th International Conference, ITS 2006
Jhongli, Taiwan, June 26-30, 2006
Proceedings

Springer

Volume Editors

Mitsuru Ikeda
Japan Advanced Institute of Science and Technology (JAIST)
School of Knowledge Science
1-1, Asahidai, Nomi, Ishikawa, 923-1292, Japan
E-mail: ikeda@jaist.ac.jp

Kevin D. Ashley
University of Pittsburgh
Learning Research and Development Center
Intelligent Systems Program and School of Law
3939 O'Hara Street, Pittsburgh, PA 15260, USA
E-mail: ashley+@pitt.edu

Tak-Wai Chan
National Central University
Graduate Institute of Network Learning Technology
300, Jhongda Road, Jhongli City,Taoyuan County 32001, Taiwan
E-mail: chan@cl.ncu.edu.tw

Library of Congress Control Number: 2006927186

CR Subject Classification (1998): K.3, I.2.6, H.5, J.1

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743
ISBN-10 3-540-35159-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-35159-7 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11774303 06/3142 5 4 3 2 1 0

Preface

The International Conference on Intelligent Tutoring Systems (ITS) provides a leading international forum for the dissemination of original results in the design, implementation, and evaluation of intelligent tutoring systems and related areas. The conference draws researchers from a broad spectrum of disciplines ranging from artificial intelligence and cognitive science to pedagogy and educational psychology. ITS 2006 (http://www.its2006.org/), the eighth in this series of biennial conferences, took place during June 26–30, 2006 at the National Central University in Jhongli, Taiwan. Previous ITS conferences have been held in Montreal, Canada (in 1988, 1992, 1996, and 2000), San Antonio, Texas, USA (in 1998), Biarritz, France and San Sebastian, Spain (in 2002) and Maceio, Alagoas, Brazil (in 2004).

The theme of ITS 2006, "Intelligent Tutoring Scales Up!", raises important issues for the future of ITS research. The conference explored intelligent tutoring systems' increasing real-world impact on a global scale. Improved authoring tools and learning object standards have enabled fielding systems and curricula in real-world settings on an unprecedented scale. Researchers have deployed ITSs in ever larger studies and increasingly have used data from real students, tasks, and settings to guide new research. With high volumes of student interaction data, data mining, and machine learning, tutoring systems are learning from experience and improving their teaching performance. The increasing number of realistic evaluation studies has also broadened researchers' knowledge about the educational contexts for which ITSs are best suited. At the same time, researchers have explored how to expand and improve ITS/student communications, for example, how to achieve more flexible and responsive discourse with students, help students integrate Web resources into learning, use mobile technologies and games to enhance student motivation and learning, and address multicultural perspectives.

The full papers and posters presented at ITS 2006 covered a wide range of ITS topics, including adaptive hypermedia, evaluation of instructional systems, learning environments, affect and models of emotion, human factors and interface design, machine learning in ITS, agent-based tutoring systems, instructional design, narratives in learning, natural language and discourse, architectures, instructor networking, pedagogical agents, assessment, intelligent agents, pedagogical planning, authoring systems, intelligent Web-based learning, situated learning, case-based reasoning systems, intelligent multimedia systems, speech and dialogue systems, cognitive modeling, Internet environments, student modeling, collaborative learning, knowledge acquisition, virtual reality, digital learning games, knowledge construction, Web-based training systems, distributed learning environments, knowledge representation, wireless and mobile learning, electronic commerce and learning, and learning companions.

This volume comprises all of the full papers and short papers associated with posters presented at ITS 2006. These 107 papers have survived a highly selective process. Of a total of 202 submissions, the Program Committee selected 67 full papers for oral presentation and 24 short papers for poster presentation. A later poster submission process yielded additional poster-related papers for a total of 40.

In addition to presentation of papers, ITS 2006 included a panoply of workshops, tutorials, posters, a student-track session, and invited keynote addresses from seven distinguished scholars: Yam San Chee, National Institute of Education, Singapore; Jim Greer, University of Saskatchewan, Canada; Ulrich Hoppe, University of Duisburg-Essen, Germany; Kinshuk, Massey University, New Zealand; Helen Pain, University of Edinburgh, UK; Rosalind Picard, Massachusetts Institute of Technology Media Laboratory, USA; and Ovid J.L. Tzeng, Academia Sinica, Taiwan.

Many people participated in making ITS 2006 a success. Tak-Wai Chan (National Central University, Taiwan) served as Conference Chair, with Mitsuru Ikeda (Advanced Institute of Science and Technology, Japan) and Kevin Ashley (University of Pittsburgh, USA) as Program Committee Co-chairs. A complete listing follows, but here we would especially like to thank: General Chair Wen-Lian Hsu (Academia Sinica, Taiwan), Local Organization Chair Richard Chih-Hung Lai (National Central University, Taiwan), Workshop Co-chairs Vincent Aleven (Carnegie Mellon University, USA) Chen-Chung Liu (National Central University, Taiwan) and Yao-Tin Sung (National Taiwan Normal University, Taiwan), Tutorial Co-chairs Oscar Lin (Athabasca University, Canada) and Wu-Yuin Hwang (National Central University, Taiwan), Panel Chair Tak-Wai Chan, (National Central University, Taiwan), Poster Co-chairs Tsukasa Hirashima (Osaka University, Japan) and Chih-Yueh Chou (Yuan Ze University, Taiwan) and Student Track Co-chairs Tanja Mitrovic (University of Canterbury, New Zealand) and Von-Wun Soo (National University of Kaohsiung, Taiwan). We thank the Program Committee and the External Reviewers for their thoughtful and timely participation in the paper selection process. The members of the Local Organizing Committee at the National Central University, Jhongli, Taiwan, worked especially long and hard: Financial and Registration Chair Jie-Chi Yang, Publicity Co-chairs Oscar Yang-Ming Ku and Magi, Exhibition Co-chairs Emily Ching and Legend Chang, Accommodation Co-chairs Jen-Hang Wang and Amy Yu-Fen Chen, Transportation Chair Zhi-Hong Chen, Website Chair Peter Chang, Technology Support Chair Eric Yu, Business Manager Hsiu-Ling Tsai and Social Event Co-Chairs Tzu-Chien Liu, Yi-Chan Deng, and Andrew Lee (Taipei Municipal Da-Hu Elementary School). As always, the ITS Steering Committee and its Chair, Claude Frasson (University of Montreal, Canada), provided steady guidance.

Finally, we gratefully acknowledge Springer for its continuing support in publishing the proceedings of ITS 2006 and the generous support of the Taiwan-based sponsors of ITS 2006, including the National Science Council, R.O.C., Ministry of Education, R.O.C., Taipei City Government, R.O.C. National Sci-

ence and Technology Program for e-Learning, and Taiwanese Association for Artificial Intelligence.

June 2006

Kevin Ashley and Mitsuru Ikeda
Co-chairs, Program Committee
Tak-Wai Chan
Conference Chair, ITS 2006

Organization

Conference Committee

Conference Chair:	Tak-Wai Chan (National Central University, Taiwan)
General Chair:	Wen-Lian Hsu (Academia Sinica, Taiwan)
Program Co-chairs:	Kevin Ashley (University of Pittsburgh, USA)
	Mitsuru Ikeda (Japan Advanced Institute of Science and Technology, Japan)
Local Organization Chair:	Richard Chih-Hung Lai (National Central University, Taiwan)
Workshop Chair:	Vincent Aleven (Carnegie Mellon University, USA)
Workshop Co-chairs:	Chen-Chung Liu (National Central University, Taiwan)
	Yao-Tin Sung (National Taiwan Normal University, Taiwan)
Tutorial Co-chairs:	Oscar Lin (Athabasca University, Canada)
	Wu-Yuin Hwang (National Central University, Taiwan)
Panel Chair:	Tak-Wai Chan (National Central University, Taiwan)
Poster Co-chairs:	Tsukasa Hirashima (Osaka University, Japan)
	Chih-Yueh Chou (Yuan Ze University, Taiwan)
Student Track Co-chairs:	Tanja Mitrovic (University of Canterbury, New Zealand)
	Von-Wun Soo (National University of Kaohsiung, Taiwan)
Financial and Registration Chair:	Jie-Chi Yang (National Central University, Taiwan)
Publicity Co-chairs:	Oscar Yang-Ming Ku (National Central University, Taiwan)
	Magi (National Central University, Taiwan)
Exhibition Co-chairs:	Emily Ching (National Central University, Taiwan)
	Legend Chang (National Central University, Taiwan)
Accommodation Co-chairs:	Jen-Hang Wang (National Central University, Taiwan)
	Amy Yu-Fen Chen(National Central University, Taiwan)
Transportation Chair:	Hon Zhi-Hong Chen (National Central University, Taiwan)
Website Chair:	Peter Chang (National Central University, Taiwan)

Technology Support Chair:	Eric Yu (National Central University, Taiwan)
Business Manager:	Hsiu-Ling Tsai (National Central University, Taiwan)
Social Event Co-chairs:	Tzu-Chien Liu (National Central University, Taiwan)
	Yi-Chan Deng (National Central University, Taiwan)
	Andrew Lee (Taipei Municipal Da-Hu Elementary School, Taiwan)

Program Committee

Mohammed Abdelrazzek (University of Cairo, Egypt)
Esma Aimeur (University of Montreal, Canada)
Vincent Aleven (Carnegie Mellon University, USA)
Lora Aroyo (Technische Universiteit Eindhoven, Netherlands)
Nicolas Balacheff (Centre National de la Recherche Scientifique, France)
Gautam Biswas (Vanderbilt University, USA)
Guy Boy (EURISCO, France)
Karl Branting (BAE Systems, USA)
Bert Bredeweg (University of Amsterdam, Netherlands)
Joost Breuker (University of Amsterdam, Netherlands)
Paul Brna (University of Glasgow, UK)
Peter Brusilovsky (University of Pittsburgh, USA)
Stefano Cerri (University of Montpellier II, France)
Kuo-En Chang (National Normal University, Taiwan)
Yam-San Chee (National Institute of Education, Singapore)
Nian-shing Chen (National Sum Yat-Sen University, Taiwan)
Weiqin Chen (University of Bergen, Norway)
Cristina Conati (University of Vancouver, Canada)
Evandro Barros Costa (University Federal de Alagoas, Brazil)
Vladan Devedzic (University of Belgrade, Serbia)
Ben du Boulay (University of Sussex, UK)
Isabel Fernandez de Castro (University of the Basque Country, Spain)
Claude Frasson (University of Montreal, Canada)
Gilles Gauthier (University of Quebec at Montreal, Canada)
Khaled Ghedira (ISG, Tunisia)
Guy Gouardères (University of Pau, France)
Art Graesser (University of Memphis, USA)
Jim Greer (University of Saskatchewan, Canada)
Neil Heffernan (Worcester Polytechnic Institute, USA)
Tsukasa Hirashima (Hiroshima University, Japan)
Ulrich Hoppe (University of Duisburg-Essen, Germany)
Ronghuai Huang (Beijing Normal University, China)

Lewis Johnson (University of Southern California, USA)
Yong Se Kim (Creative Design and Intelligent Tutoring System Research Center)
Ken Koedinger (Carnegie Mellon University, USA)
Lam-For Kwok (The Chinese University of Hong Kong, Hong Kong)
Fong Lok Lee (The Chinese University of Hong Kong, Hong Kong)
Chee-Kit Looi (Nanyang Technological University, Singapore)
Rose Luckin (University of Sussex, UK)
Gordon McCalla (University of Saskatchewan, Canada)
Erica Melis (DFKI, Germany)
Tanja Mitrovic (University of Canterbury, New Zealand)
Riichiro Mizoguchi (University of Osaka, Japan)
Jack Mostow (Carnegie Mellon University, USA)
Roger Nkambou (University of Quebec, Canada)
Toshio Okamoto (Electro-Communications University, Japan)
Helen Pain (University of Edinburgh, UK)
Ana Paiva (Instituto Superior Tecnico, Portugal)
Fábio Paraguau (University Federal de Alagoas, Brazil)
Jean-Pierre Pecuchet (INSA of Rouen, France)
Paolo Petta (Research Institute for AI, Austria)
Sowmya Ramachandran (Stotler-Henke, Inc., USA)
Carolyn Rose (Carnegie Mellon University, USA)
Khalid Rouane (University of Montreal, Canada)
Timothy Shih (Tamkang University, Taiwan)
Amy Soller (Institute for Defense Analyses, USA)
Elliot Soloway (University of Michigan, USA)
Dan Suthers (University of Hawaii, USA)
Pierre Tchounikine (University of Le Mans, France)
Gheorge Tecuci (George Mason University, USA)
Shian-Shyong Tseng (National Chiao Tung University, Taiwan)
Kurt VanLehn (University of Pittsburgh, USA)
Julita Vassileva (University of Saskatchewan, Canada)
Rosa Maria Viccari (Federal University of Rio Grande do Sul, Brazil)
Beverly Woolf (University of Massachussets, USA)
Kalina Yacef (University of Sydney, Austrlia)
David Yang (National Kaohsiung Normal University, Taiwan)

Steering Committee Chair

Claude Frasson (University of Montreal, Canada)

Steering Committee

Stefano Cerri (University of Montpellier II, France)
Isabel Fernandez-Castro (University of the Basque Country, Spain)

Gilles Gauthier (University of Quebec at Montreal, Canada)
Guy Gouardres (University of Pau, France)
Mitsuru Ikeda (Japan Advanced Institute of Science and Technology, Japan)
Marc Kaltenbach (Bishops University, Canada)
Judith Kay (University of Sidney, Australia)
Alan Lesgold (University of Pittsburgh, USA)
Elliot Soloway (University of Michigan, USA)
Daniel Suthers (University of Hawaii, USA)
Beverly Woolf (University of Massachussets, USA)

Referees

M.R. Abdel
E. Aïmeur
V. Aleven
I.M. Alexandre
M. Anwar
I. Arroyo
K. Ashley
R. Baker
C. Beal
K. Binsted
G. Biswas
E. Blanchard
B. Boulay
A. Bouwer
G. Boy
K. Branting
B. Bredeweg
J. Breuker
P. Brna
C. Brooks
P. Brusilovsky
S.A. Cerri
S. Chaffar
K.E. Chang
Y.S. Chee
N. Chen
W. Chen
X. Cheng
M. Constantino
E. Costa
C. Cristina
V. Devedzic
T. Dragon
I. Fernandez-Castro
C. Frasson
G. Gauthier
J.C. Gluz
J. Good
V. Goodkovsky
G. Gouarderes
S. Gouarderes
A. Graesser
J. Greer
R. Guizzardi
H.H. Ulrich
A. Harrer
N. Heffernan
T. Hirashima
R. Huang
G. Hwang
M. Ikeda
H. IP
P. Jaques
J. Johns
W.L. Johnson
J. Jovanovic
T. Kasai
A. Kashihara
Y.S. Kim
K. Koedinger
S.C. Kong
T. Konishi
L. Kwok
J. Labat
N.V. Labeke
H.C. Lane
F. Lee
J. Lee
Dr.K. Leelawong
B. Lefebvre
R. Luckin
A. Lum
H. Maclaren
N. Matsuda
G. McCalla
E. Melis
A. Mitrovic
R. Mizoguchi
R. Morales
J. Mostow
K. Muldner
R.C. Murray
J. Nicaud
R. Nkambou
G. Nobrega
T. Okamoto
H. Pain
A. Paiva
F. Paraguau
D. Pearce
J. Pecuchet
J. Pelle-tchetagni
P. Petta
N. Pinkwart
K. Porayska-Pomsta
R. Prada

S. Ramachandran
L. Razmerita
C. Rose
K. Rouane
D. Schwartz
L. Seixas
T. Shih
A. Soller
Y. Sung
P. Suraweera
D. Suthers
A. Takeuchi
P. Tchounikine
G. Tecuci
P. Tedesco
S. Tseng
J. Underwood
A. Valente
K. Vanlehn
J. Vassileva
R. Viccari
E. Wang
L.H. Wong
B. Woolf
K. Yacef
C. Zinn

Sponsors

National Science Council, R.O.C.
Ministry of Education, R.O.C.
Taipei City Government, R.O.C.
National Science and Technology Program for e-Learning
Taiwanese Association for Artificial Intelligence

Table of Contents

Assessment

Authoring Tools

Bayesian Reasoning and Decision-Theoretic Approaches

Case-Based and Analogical Reasoning

Cognitive Models

Collaborative Learning

Error Detection and Handling

Feedback

Gaming Behavior

Learner Models

Scaffolding

Simulation

Tutorial Dialogue and Narrative

Poster Papers

Keynotes

Automated Expert Modeling for Automated Student Evaluation

Robert G. Abbott

Sandia National Labs MS 1188
PO Box 5800
Albuquerque NM 87185, USA
Tel.: (505) 296-3446
rgabbot@sandia.gov

Abstract. This paper presents automated expert modeling for automated student evaluation, or AEMASE (pronounced "amaze"). This technique grades students by comparing their actions to a model of expert behavior. The expert model is constructed with machine learning techniques, avoiding the costly and time-consuming process of manual knowledge elicitation and expert system implementation. A brief summary of after action review (AAR) and intelligent tutoring systems (ITS) provides background for a prototype AAR application with a learning expert model. A validation experiment confirms that the prototype accurately grades student behavior on a tactical aircraft maneuver application. Finally, several topics for further research are proposed.

1 Introduction

This paper presents Automated Expert Modeling for Automated Student Evaluation (AEMASE, pronounced "amaze"). The dual use of the word "automated" denotes a degree of automation beyond that of existing approaches which automate student evaluation only after the expensive and time consuming process of implementing an expert system. To train the tutor, a subject matter expert demonstrates the desired behavior in a simulator or other instrumented environment. Next, machine learning techniques are used to build a model of the demonstrated behavior. Students working in the same task environment can then be graded by the expert model. Afterwards, the student and a human instructor can quickly review the training session by interacting with a plot of the time-dependent grade, which is linked to a video recording of the training session.

2 Background

This section reviews relevant aspects of After Action Review (AAR) and Intelligent Tutoring Systems (ITS). Both AAR and ITS are relevant because they may incorporate automated student evaluation. Most automated student evaluation

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 1–10, 2006.

research has been conducted in the field of ITS, but most ITS are designed to confer declarative knowledge. AAR (which is rooted in military practice) is particularly relevant because tactical and strategic knowledge are heavily dependent on spatiotemporal pattern learning, which is amenable to automated modeling.

2.1 After Action Review

After action review (AAR) is a general process for review and discussion of a training session in order to evaluate performance, diagnose problems, and build on successes. AAR is typically based on face-to-face interaction rather than a computer application.

The U.S. and Canadian Armies use the following AAR steps: [1] 1) *Plan:* select training objectives, important events, and arrange data collection for the exercise. 2) *Prepare:* stage data collection mechanisms, collect data during the exercise, and organize the presentation of the collected data. 3) *Conduct:* discuss training events and analyze student performance. The instructor should ensure that students feel free to participate. 4) *Follow Up:* adjust upcoming training to address issues raised during the AAR. The instructor maintains records of student competency in key areas.

The use of AAR was pioneered by the U.S. Army in the mid 1970's and has been adopted by many others in government and industry [2]. The U.S. Army Research Institute has developed several generations of AAR tools of increasing capability [3]. The first computer-based AAR tool was the Unit Performance Assessment System (UPAS), released in 1991. UPAS included some analytical capability based on relational database queries. Current AAR-related research focuses on developing automated measures of performance for AAR [4].

Outside of military applications, most research on automated student evaluation is related to intelligent tutoring systems (ITS). The link between AAR and ITS is a nascent development. AAR is traditionally used for live exercises with human instructors teaching students to perform as a coordinated team. This is a difficult setting for automated assessment tools because of the unconstrained environment and complex interactions between individuals in a group. Automated assessment is becoming feasible due to the adoption of simulation-based training.

2.2 Intelligent Tutoring Systems

This section briefly defines intelligent tutoring systems (ITS). AEMASE is a technique for automated student evaluation, which is critical for ITS.

Personal tutoring is the most effective way to teach students. Studies consistently conclude that tutoring improves average test scores [5] (reported in [6]). Unfortunately, individual tutoring is prohibitively labor intensive and expensive. Intelligent Tutoring Systems (ITS) are computer-based teaching or training systems which reduce cost by automating coursework selection, presentation, and student evaluation. Most ITS contain at least three components [7]: 1) the *Expert Module*, which contains the tutor's domain expertise and problem-solving capability; 2) the *Student Module*, which represents the student's strengths and weaknesses in the domain; and 3) the *Tutoring Module* which selects exercises and presents instruction.

Corbett [6] reviews the evidence that ITS can improve student test scores by approximately the same amount as human tutoring. However, ITS cannot provide cost savings over personal tutoring if the ITS require a large initial investment and are difficult to modify.

Murray [8] provides a comprehensive survey of authoring systems for ITS through 1999, including a categorization of ITS. AEMASE falls under "Expert Systems and Cognitive Tutors," which incorporate a cognitive model of domain expertise, compare student behavior to the model, and provide feedback when students vary from expected behavior. However, expert systems for ITS can be difficult to construct. Murray states, "Authoring an expert system is a particularly difficult and time-intensive task, and only certain tasks can be modeled in this manner." Automated expert model construction may lead to faster, cheaper ITS development cycles.

3 Automated Expert Modeling for Automated Student Evaluation

This section introduces automated expert modeling for automated student evaluation (AEMASE). All ITS automate student evaluation, but this technique also automates training the expert model responsible for evaluating students. To train the expert model, a subject matter expert demonstrates the desired behavior in a simulator or instrumented environment. Next, machine learning techniques are used to build a model of the demonstrated behavior. Then students working in the same task environment can be compared to the expert model. The objectives of AEMASE include: 1) broaden the scope of ITS to include training currently limited to subjective AAR by capturing implicit expert knowledge; 2) rapid development to accommodate evolving mission scenarios and tactics; 3) reduced development cost; and 4) raise the quality of training by delivering ITS for applications which were previously limited to textbook or group instruction.

The success of the approach depends on how well the model captures subject matter expertise and student knowledge. Effective knowledge capture depends on several factors including observability, the choice of modeling algorithm, and the difficulty of the training task.

3.1 Observability

In order to build an expert model, the system must be able to observe one or more experts performing relevant tasks. ITS normally assume observability because the simulation environment is a computer-based application, often purpose-built as an integral component of the tutoring system. With advances in sensing and artificial perception [9], the scope of ITS can be expanded to include less constrained training environments, which currently rely on manual AAR with little automation. Modern tactical aircraft already record trajectory information, which supports AAR of live training exercises and actual missions.

3.2 Modeling Algorithm

The field of machine learning offers many algorithms suitable for modeling expert behavior. Reasonable choices include decision tree induction [10], learning neural networks [11], Bayesian networks, and support vector machines. The AEMASE prototype uses a nearest neighbor algorithm [12]. At each time step, the model receives an observation from the simulator. When the expert model is being trained, the observations are stored in the knowledge base. When a student is being evaluated, each observation is compared to the knowledge base. Observations with no close match in the knowledge base are assumed to represent anomalous student behavior and are assigned a low score. This approach does not limit the model to a single correct solution for each situation; any expert behavior contained in the knowledge base is acceptable.

The system represents an observation as a real-valued vector. Each observation must contain all the information required to select an appropriate action; the Markov Property [13, 61] is assumed. In order to justify this assumption, the sequence of observations is augmented with derived features. For instance, an estimate of velocity can be derived from a sequence of positions. The system provides a rich set of feature extractors including some which incorporate memory and goals. For instance, an observation may contain a set of features which trace the recent trajectory of an entity.

If a very large amount of training data is available, the knowledge base may become too large to store and search conveniently. The system uses k-medoid clustering [12] to reduce the quantity and redundancy of data while preserving most of its variation. A weighting function can be specified to retain important observations while grouping together or discarding unimportant observations. The observations retained after clustering form the context set of the model.

The crucial parameters for automated student evaluation influence the distance function which compares observed student behavior to expert behavior. The system calculates the weighted Euclidean distance between observation vectors a and b:

$$d_{a,b}^2 = \sum_{k=1}^{K} \mathbf{w}_k \left(\mathbf{a}_k - \mathbf{b}_k\right)^2 \qquad (1)$$

where the vector $\mathbf{w}$ specifies a weight for each feature. The weight vector is a parameter to the system. Weights should be chosen to emphasize features which are essential to competent performance in the domain.

3.3 Task Difficulty

Different modeling approaches are suitable for different domains. The AEMASE prototype focuses on spatiotemporal pattern learning. This class of problems is both suitable to automated modeling, and challenging for manual knowledge elicitation. The following passage from Kornecki et al. [14] describes aspects of air traffic control (ATC) which present difficulty for manual knowledge elicitation and rule-based expert modeling:

> In real ATC sectors, a controller's actions are based on a subjective evaluation of the situation and a rough mental projection of the aircraft courses and computation of a future possible separation violation. There is no extensive or precise arithmetic computation involving the geometric relation between objects in three-dimensional space.

AEMASE uses behavioral modeling to capture the implicit, imprecise spatial and contextual relations that form the knowledge bases of experts in tasks such as air traffic control. Most existing ITS focus on declarative knowledge, which is less ambiguous and more suited to conventional knowledge elicitation. Automated expert model construction can compliment other techniques in a comprehensive ITS.

4 Automated After-Action Review Application

This section describes AEMASE-AAR, an automated after-action review (AAR) application which incorporates an expert model for evaluating students. Like most intelligent tutoring systems, the system automatically evaluates students' actions in real time by comparing them with the expert model. Unlike most ITS, development of the expert model is largely automated. The approach avoids manual knowledge elicitation and rule set authoring.

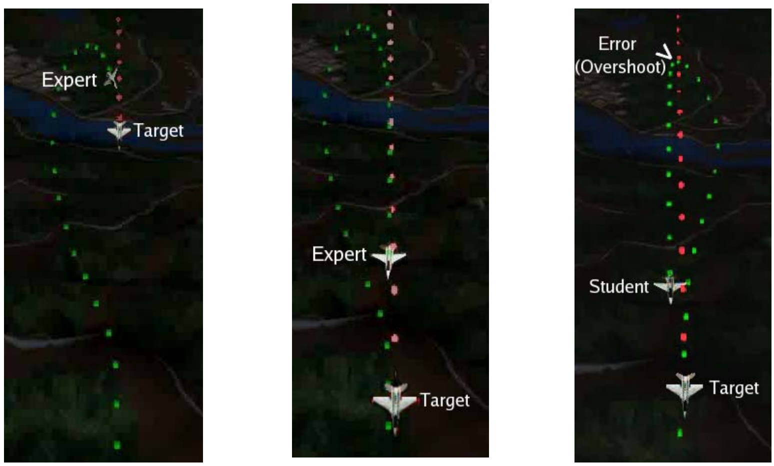
The AAR system is applied to tactical aircraft maneuver. This application was developed with the assistance of Jonathan Whetzel, Justin Basilico, and Kevin Dixon of Sandia National Laboratories. The AAR system has also been applied to an AWACS weapons director simulator [15] with little modification.

4.1 The Stern Conversion Training Exercise

In air combat, a stern conversion is any sequence of maneuvers which place the attacker behind the target, with the same altitude and heading, and within weapons range. The desired position is relatively safe for the attacker and very dangerous for the target.

Figure 1 shows three snapshots from stern conversion exercises. The pilots execute the maneuvers in a high fidelity F16 flight simulator. At each time step,the simulator reports its state, which includes the position of all entities (the green and red trails in Figure 1). In this scenario, the target flies down from the top of the screen. The pilot must execute a 180 degree turn (Figure 1(a)) to get behind the target. Figure 1(b) shows the trajectory of a USAF Lt. Col. F16 pilot. Figure 1(c) shows the trajectory of a student on the same scenario. The student offset to the right whereas the expert offset to the left, which is an acceptable variation. However, the student trajectory is flawed because he overshoots the target.

Given the headings and positions of the attacker and target, other state variables are derived including aspect angle, range, relative altitude, closure rate, and angle off tail (Figure 2). The behavior model defines a space spanning these features. The student is expected to keep the state of the scenario within the



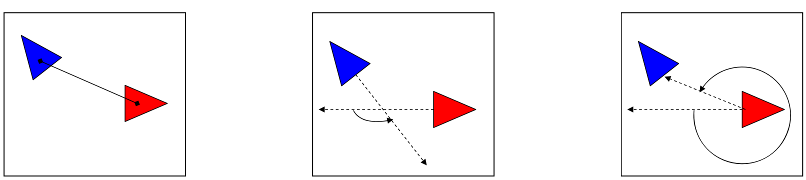
(a) Turning towards the target (b) An ideal stern conversion (c) A flawed stern conversion

Fig. 1. Examples of the stern conversion task

manifold defined by the example behavior of subject matter experts. In terms of the model, the goal of stern conversion is to drive the current system state into a subspace which represents the completion criteria.

After the exercise, the AAR system displays a plot of the time-varying score (Figure 3). A low score means that the student varied from expert behavior, driving the system into a state not encountered by experts. The system also records full-motion video of the student's computer display. The recorded video is linked to the score plot. By selecting any point in the plot, the student and instructor can immediately review the student's actions at the time in question.

Corbett [6] has shown that the learning benefits of ITS increase when the system can not only detect student errors, but suggest correct solutions. AEMASE-AAR offers analogy-based video recall to present relevant examples of expert



(a) Slant range: the Euclidean distance (including differences in altitude). (b) Angle off: the difference between aircraft headings. (c) Aspect angle: the angle from the target to the subject.

Fig. 2. Features extracted for evaluating student performance on the Stern Conversion training task. Red triangles represent target aircraft.

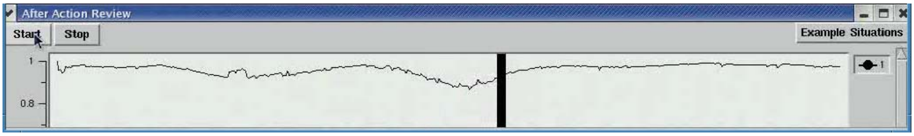


Fig. 3. Time-varying score for the flawed trajectory in Figure 1(c). The low point of the score indicates where the student overshoots the target.

behavior. The student uses the timeline (Figure 3) to select a point just prior to a decline in score. The student can then access a list of situations encountered by subject matter experts during automated knowledge elicitation. The situations are listed in order of similarity to the situation in question. Selecting an example situation initiates a video replay of relevant expert behavior. There is no need to manually index the catalog of expert video footage; the distance function determines situation relevance automatically.

5 An Experiment in Student Evaluation

The time and cost savings of AEMASE do not matter unless the technique is effective in evaluating students. This section presents an experiment designed to assess the discriminative ability of an expert model created through automated knowledge elicitation on the stern conversion task. The data for this experiment were collected for a study (not yet published) designed by Chuck Lance (University of Georgia), Brian Schreiber (U.S. Air Force Research Lab), and Lt. Col. (ret.) Mark Warner. The description of the data comes from private correspondence with Brian Schreiber.

In the study, subjects performed stern conversions in a high fidelity F16 flight simulator. Four subjects conducted two trials of nine different conditions for a total of 72 trials. In all conditions, the target began at a range of 15 miles and a speed of 450 knots, with the F16 traveling at 350 knots. The starting position and trajectory of the target varied between conditions. In each condition the target either flew in a straight line or performed a 180 degree turn. The simulator state was sampled at 20 Hz. Of the four subjects, two were Lieutenant Colonels and seasoned F16 pilots with extensive experience in the F16 simulator. The other two subjects were not pilots. They flew the simulator semi-frequently and had general declarative and procedural knowledge of the simulator controls and the stern conversion task, but rarely practiced.

One of the trials was incomplete (for all four subjects), leaving 68 trials. Each trial was graded a success or failure. A subject matter expert at the Air Force Research Lab evaluated a sample of trials, and we manually assessed the other trials. Trials were graded mainly by accomplishment of the desired end state – behind the target at a range of $3,000$ to $6,000$ feet with matching altitude and heading. A few trials matched this criteria but were graded failures because of undesirable trajectories. In air combat, desirable trajectories minimize stern conversion time and fuel consumption. 36 of the 68 trials (52.9%) were judged to be successful.

The stern conversion data were reformatted as AEMASE observations. AEMASE feature extractors were configured to convert the entity positions from latitude and longitude in degrees and altitude in feet to a 3d coordinate in meters. The AEMASE distance metric was configured with nonzero weights for only the slant range, aspect angle, and angle off tail features (Figure 2). The features were weighted equally after normalization. The features and weights were selected by trial and error. It is possible that other parameters would yield better results.

Finally we evaluated AEMASE using a "leave one out" procedure [16]. For each of the 68 trials, an AEMASE model was trained on all the successes among the other 67 trials. The set of successful trials included all starting conditions, so each model encompassed all the maneuvers and not just the maneuver in the test trial. The failed trials were not used in training. Each trial was then scored by its corresponding model. The penalty score is the distance in feature space from the current state of the simulation to the nearest observation in the model's knowledge base. This score varies over time. For compatibility with the manual pass/fail assessment of each trial, we then reduced the time-varying score to a pass or fail by thresholding the trial's maximum penalty score. This thresholding step is not part of the AEMASE AAR because the continuous time-varying score is more informative, but thresholding allowed direct comparison with the manual pass/fail assessments. The optimum threshold was determined by computing the classification accuracy with every unique maximum penalty score from the test set.

5.1 Experimental Results

After thresholding the time-varying penalty score to produce a pass/fail grade, AEMASE achieved 89.7% agreement with the manually assigned grades. The

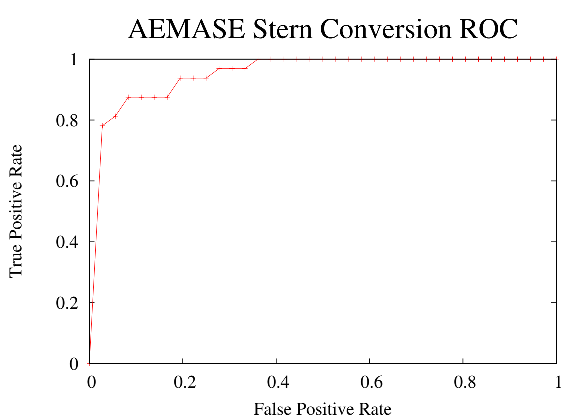


Fig. 4. Receiver Operating Characteristic Curve (ROC) for AEMASE on the stern conversion data set

baseline for this value is 52.9%, which would be achieved by grading every trial a success. At this level of accuracy, the false positive and true positive rates were 2.8% and 81.3%, respectively.

There is a tradeoff in selecting the threshold. A lower threshold is more likely to detect unexpected student performance (higher sensitivity), but will also raise more false alarms (less specificity). Figure 4 shows the Receiver Operating Characteristic (ROC) curve [17] for AEMASE on the stern conversion dataset. A false positive rate of 0 was not achieved with any true positive rate over 0. Achieving 100% true positive detection requires a false positive rate of 36%.

5.2 Discussion and Conclusion

In assigning a binary score to stern conversion trials, AEMASE achieved 89.7% agreement with a human grader. This level of accuracy should prove sufficient for many applications. We have not studied the level of agreement between two or more human graders on this task. A study of rooftop detection in satellite imagery by Ali et al. [18] compared results between a pair human analysts, and found true positive and false positive rates of 68.7% and 6.4%, respectively. Ali also considered self agreement (two evaluations by one analyst) and found true positive and false positive rates of 85.6% and 5.0%. While these results are not directly transferable to the stern conversion dataset, they provide evidence that the AEMASE true positive and false positive rates (81.3% and 2.8%) are not much worse than a pair of human graders would achieve.

Our experiment demonstrated high scoring accuracy on an application of interest. We are researching ways to model more complex domains and give more detailed feedback on student performance. We are also planning further research to validate the overall training benefit, efficiency gains over manual AAR, and cost savings over manual expert model implementation.

Given the effectiveness of our prototype application, we feel that automated expert modeling for automated student evaluation has great potential for intelligent tutoring systems. Potential benefits include more effective transfer of implicit knowledge, rapid development, and cost reduction.

References

1. Centre, C.A.L.L.: The after action review process, learning more from our training. Army Lessons Learned Centre Dispatches **6**(3) (1999)
2. DeGrosky, M.T.: Improving after action review (aar) practice. In: Eighth International Wildland Fire Safety Summit. (2005)
3. Meliza, L.: After action review: Methods and tools. http://www.hqda.army.mil/ari /news/factsheets.shtml, retrieved on Dec 2, 2005 (2004)
4. Jensen, R., Nolan, M., Chen, D.Y.: Automatic causal explanation analysis for combined arms training aar. In: Proceedings of the Industry/Interservice, Training, Simulation and Education Conference (I/ITSEC). (2005)
5. Cohen, P.A., Kulik, J.A., Kulik, C.C.: Educational outcomes of tutoring: A meta-analysis of findings. American Educational Research Journal **19** (1982) 237–248

6. Corbett, A.T.: Cognitive computer tutors: Solving the two-sigma problem. In: UM '01: Proceedings of the 8th International Conference on User Modeling 2001, London, UK, Springer-Verlag (2001) 137–147
7. Burns, H., Capps, C.: Foundations of intelligent tutoring systems: An introduction. In Polson, M.C., Richardson, J.J., eds.: Foundations of Intelligent Tutoring Systems. Lawrence Erlbaum Associates Publishers (1988) 1–18
8. Murray, T.: Authoring intelligent tutoring systems: An analysis of the state of the art. International Journal of Artificial Intelligence in Education **10** (1999) 98–129
9. Welch, G., Foxlin, E.: Motion tracking: no silver bullet, but a respectable arsenal. IEEE Computer Graphics and Applications **22**(6) (2002) 24–38
10. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1992)
11. Srinivasa, N., Ahuja, N.: A topological and temporal correlator network for spatiotemporal pattern learning, recognition and recall. IEEE Transactions on Neural Networks **10**(2) (1999) 356–371
12. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer (2001)
13. Sutton, R., Barto, A.: 3.5 The Markov Property. In: Reinforcement Learning. MIT Press (1998) 61–65
14. Kornecki, A., Hilburn, T., Diefenbach, T., Towhidnajad, M.: Intelligent tutoring issues for atc training system. IEEE Transactions on Control Systems Technology **1**(3) (1993) 204–211
15. Hess, S.M., MacMillan, J., Sefarty, D., Elliot, L.: From cognitive task analysis to simulation: Developing a synthetic team task for awacs weapons directors. Air Force Research Laboratory (1999)
16. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: IJCAI. (1995) 1137–1145
17. Zweig, Campbell: Receiver-operating characteristic (roc) plots: a fundamental evaluation tool in clinical medicine. Clin Chem **39**(4) (1993) 561–577
18. Ali, K., Langley, P., Maloof, M., Sage, S., Binford, T.: Improving rooftop detection with interactive visual learning. In: Proceedings of the Image Understanding Workshop, San Francisco, CA, Morgan Kaufmann (1998) 479–492

Multicriteria Automatic Essay Assessor Generation by Using TOPSIS Model and Genetic Algorithm

Shu-ling Cheng[1,2] and Hae-Ching Chang[1]

[1] Department of Business Administration, National Cheng Kung University, No.1, Ta-Hsueh Road, Tainan 701, Taiwan
emily@cc.fec.edu.tw, chc5728@ms54.hinet.net
[2] Department of Information Management, Far East College, No.49,Chung Hua Road, Tainan County 744, Taiwan

Abstract. With the advance of computer technology and computing power, more efficient automatic essay assessment is coming to use. Essay assessment should be a multicriteria decision making problem, because an essay is composed of multiple concepts. While prior works have proposed several methods to assess students' essays, little attention is paid to use multicriteria for essay evaluation. This paper presents a Multicriteria Automatic Essay Assessor (MAEA) based on combined Latent Semantic Analysis (LSA), Technique for Order Preference by Similarity to Ideal Solution (TOPSIS), and Genetic Algorithm (GA) to assess essays. LSA is employed to construct concept dimensions, TOPSIS incorporated to model the multicriteria essay assessor, and GA used to find the optimal concept dimensions among LSA concept dimensions. To show the effectiveness of the method, the essays of students majoring in information management are evaluated by MAEA. The results show that MAEA's scores are highly correlated with those of the human graders.

1 Introduction

Essay-based test has become more popular to assess candidate/examinee knowledge. It then has received wide application whether in the school or in the industry, e.g., essay-based exam for GMAT or GRE and for selecting a job candidate/examinee with appropriate domain knowledge for companies. Therefore, there is a growing demand on effective essay assessment methods. In the past, essay evaluation is usually executed by human, which is extremely time-consuming, labor intensive and subjective. One alternative is to grade an essay using automatic essay assessor which is time-saving, cheap and consistent [3, 18]. Basically, most current automatic essay assessors are based on artificial intelligence, computational linguistics, and cognitive science foundations, rather than on traditional psychometric and educational measurement [13].

In previous works on automatic essay assessment, essay words play a crucial role in assessment, e.g., the scoring based on the average sentence length and the number of words [4]. However, doubts are cast that the number of words a candidate/examinee can write may not implicate his/her knowledge. To improve the reliability of automatic essay assessor, some essay-scoring systems, such as e-rater and IEA, are developed by incorporating Natural language Processing (NLP) techniques.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 11 – 20, 2006.

Instead of measuring a candidate/examinee's essay by counting words, some other systems incorporated latent semantic analysis (LSA) [7,11,14,23,15] to measure the essay semantic content.

LSA [5] is firstly developed for use in the area of information retrieval to represent semantic space. The merits of LSA are to improve the computing efficiency by dimensionality reduction through the singular value decomposition (SVD) and to provide more reliable assessment by co-occurring terms onto the same dimensions of the reduced space, other than by words count. The basic assumption of LSA is the semantic relatedness between a candidate/examinee's essay on a certain topic and a grader's answer needs to be an effective and reliable measure of student knowledge. By calculating the cosine between two essay vectors, we can measure the similarity between two essays. Indeed, LSA is an effective and efficient automatic essay scoring technique. However, two deficiencies of LSA-based measurements [17,22] have been identified. One is the difficulty to find the optimal dimensionality to represent the original information matrix [17], and the other the lack of the discriminate capability [22].

In fact, essay scoring requires the most discriminative features about the essay topic rather than the most representative features. Hence, the sense of the largest singular values to find the scoring features cannot be used. This problem is also mentioned in some previous works about directionality problem [22] and optimal dimensionality [17]. To find the scoring dimensions in the semantic space, we propose using the multicriteria TOPSIS model with the genetic algorithm, which is used to find the optimal concept dimensions as the essay scoring model criteria.

Besides, some previous works has incorporated criteria into automatic assessor to evaluate an essay. For example, average sentence length and the number of words are regarded as criteria to calculate a student's essay score[4]. Another research uses keywords and weights as criteria to assess an essay [7]. From prior researches, there are two concise rules about how a human grader assesses an essay. Cognitively, a human grader firstly finds concepts mentioned in a candidate/examinee's essay to match with his/hers. Then he/she grades an essay depending on the coverage degree in terms of degree of understanding in depth and completeness for a certain topic. Under this situation, essay assessment should be a multicriteria decision making problem because an essay is composed of multiple concepts with various degree of importance in the topic. Hence, in this paper, we propose incorporating the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) method into the LSA-based automatic essay assessor. TOPSIS [10] is easy to use and only needs limited subjective information to make the decision. The method has been widely used to solve the large-scale nonlinear programming problems [1], evaluate airlines competitiveness [2], and intercompany performance comparison [6]. The basic assumption in it is that the selected best alternative should have the shortest distance from the ideal solution and the farthest distance from the negative-ideal solution.

The rest of the paper is organized as follows. In Section2, we begin with a review of the previous work about automatic essay assessor. In Section 3, we describe how the proposed automatic essay assessor based on GA and TOPSIS work. In Section 4, we provide the description of the experiments and results. Finally, the conclusion and the future challenges we may face are given in Section 5.

2 Automatic Essay Assessor

2.1 Related Work

Many researchers have devoted to develop effective and efficient automatic essay assessors. Currently, two kinds of automatic essay assessors are widely used, i.e., e-rater [19, 20] and IEA [12], briefly described below.

e-rater is developed by Educational Testing Service in mid-1990. Specifically, the method uses the vector-space model to measure semantic content and weight terms to reflect the degree of importance in an essay, and calculate the vector similarity between two essays. Basically, e-rater uses three features to evaluate an essay [20]: structure (indicated by the syntax of sentences), organization (various discourse features that occur throughout the extended text), and content (the prompt-specific vocabulary). The system is now widely applied to writing assessment in GRE and GMAT. The advantage of e-rater is its modular design, which allows the system to adapt to new essays much easilier because only content module needs to be re-trained for each new essay. Another advantage is that e-rater can provide prompt feedback. However, its drawback of e-rater is the system requires a large number of samples of pre-graded essays in a certain topic to provide a reliable assessment. Besides, the system cannot identify synonyms.

Intelligent Essay Assessor (IEA) is developed by Knowledge Analysis Technologies. IEA, based on LSA, measures an essay by three variables [13]: content, style and mechanics. Content is measured by LSA and mechanics used to find the misspelled words, which is handled by corpus. And style is used to measure essay's coherence derived by content and mechanics. Unlike e-rater, one benefit of using IEA is that the system dose not requires lots of pre-graded essays as the training data. On average, the correlation of IEA scores for content is 0.83, which is highly correlated with humans. Compared with content measurement, the correlation coefficient of IEA scores for style and mechanics are 0.68 and 0.66, respectively. The result shows that content measurement performs better than style and mechanics measurements. But it is with a difficulity in determining the optimal dimentionality for IEA.

2.2 Multicriteria Essay Assessment

Essay assessment should be a multicriteria decision making problem, because an essay is composed of multiple concepts. A human grader evaluates an essay according to his/her cognitive space in a certain topic, as illustrated in Figure 1. The human grader first identifies related concepts, and then assigns weight of those concepts to reflect their importance for a certain topic. The grader matches the related concepts with a candidate/examinee's answer and gives points to those related concepts. The score of the essay is to sum the values of the concepts multiplied by the weights of the different concepts. The following section will describe our methodology based on the multicriteria to evaluate essays.

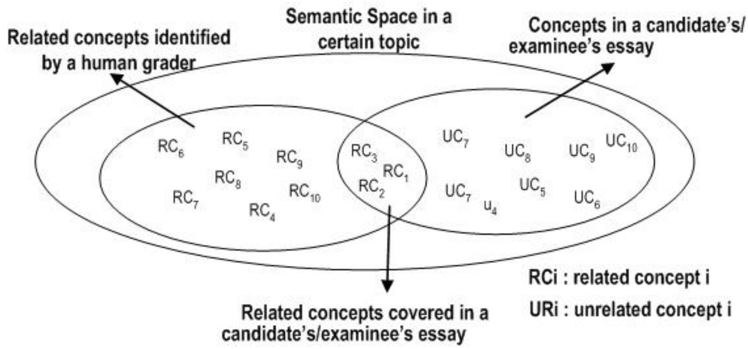


Fig. 1. Cognitive model for a human grader

3 Multicriteria Automatic Essay Assessor

Figure 2 shows the procedure of MAEA generation.

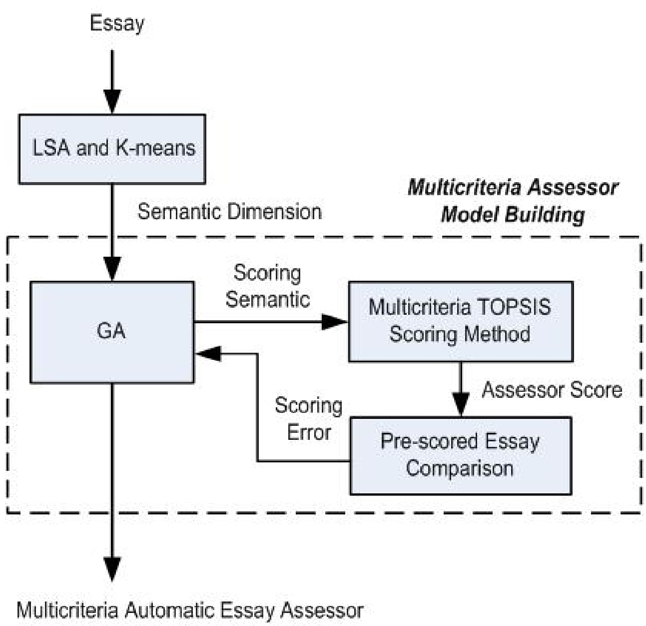


Fig. 2. Multicriteria Automatic Essay Assessor Generation

First, the essay dimensions are constructed by LSA with the training essays. Then, TOPSIS is incorporated to model the multicriteria essay assessor. Next, GA is applied to find the optimal concept dimensions among those of LSA as TOPSIS essay assessor criteria based on the grader's pre-scored essay. The generated MAEA is verified and improved by the grader. Finally, the automatic essay evaluation is performed based on the MAEA.

3.1 Concept Dimension Construction by Using LSA

Given a set of essays $E_1, E_2, \ldots, E_n$, they can be represented as a term-essay matrix $E = [E_1, E_2, \ldots, E_n]$. LSA decomposes the E using SVD as follows:

$$E = U\Sigma V^T \tag{1}$$

where $\Sigma = diag(\sigma_1, \sigma_2, \ldots, \sigma_r)$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq 0$, $U = [u_1, u_2, \ldots, u_r]$, and $V = [v_1, v_2, \ldots, v_r]$. σ_i is the i-th singular value of E, u_i and v_i the i-th left and right singular vectors, respectively. To best represent the original document space, LSA suggests that the most representative features with the k largest singular values σ_i (i=1~k) be kept to cover the original space:

$$E \approx E_k = U_k \Sigma_k V_k^T \tag{2}$$

By applying the K-means clustering method in the eigenvector spaces, the terms can be divided into many clusters denoted by a binary membership matrix M, where the element M_{ij} is 1 if the jth term belongs to cluster i and 0 otherwise. The concept vector C_i (i=1~r) is then obtained with the ideal essay vector G as follows:

$$C_{ij} = M_{ij} G_j \tag{3}$$

In addition, C_i is normalized to unit Euclidean length. Then, the concept vectors are orthonormal and can be used for dimension selection in assessor model building.

3.2 TOPSIS Assessor Model Selection

To find the scoring dimensions in the concept space, we propose using the multicriteria TOPSIS model with the genetic algorithm. Selecting a set of orthonormal concept features among concept dimensions as the scoring criteria $C_1, C_2, \ldots, C_m$, and given the weight of each criteria $w_1, w_2, \ldots, w_m$ (such that $\sum_{j=1}^{m} w_j = 1$), the sub-score x_{ij} of essay E_i on orthonormal concept C_j can be calculated as

$$x_{ij} = \cos(E_i^j, C_j) \tag{4}$$

where E_i^j is the similar concept on concept C_j and can be calculated as $E_{i\,k}^{j} = N_{jk} E_{ik}$. N is the binary membership matrix where the element N_{jk} is 1 if the k-th element of C_j does not equal to 0 and 0 otherwise. Note that we have constrained the input range of utility between 0 and 1 to avoid overvaluation if same words are repeated several times in an essay.

To calculate the essay score with the sub-scores x_{ij} and the weights w_i, TOPSIS first converts the various criteria dimensions into nondimensional criteria. The non-dimesnional sub-score r_{ij} is defined as

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{n} x_{ij}}} \tag{5}$$

And the weighted nondimesnional sub-score s_{ij} is calculated as

$$s_{ij} = w_j r_{ij} \tag{6}$$

Assuming each criterion has a tendency toward monotonically increasing utility, the ideal E^+ and the negative-ideal E^- solutions can be found as

$$E^+ = \left[s_1^+, s_2^+, \ldots, s_m^+\right] \tag{7}$$

$$\text{and } E^- = \left[s_1^-, s_2^-, \ldots, s_m^-\right] \tag{8}$$

where $s_j^+ = \max(s_{ij}, i = 1 \sim n)$ and $s_j^- = \min(s_{ij}, i = 1 \sim n)$. Finally, TOPSIS sets the score of the negative-ideal E^- solutions to be 0 and the score of the ideal E^+ solutions be 1. The score of the essay is computed by using the linear interpolation method as

$$S_i = \frac{d_i^-}{d_i^- + d_i^+} \tag{9}$$

where the distance of essay E_i to the ideal solution E^+ is $d_i^+ = \left\|E_i - E^+\right\|$ and that to the negative-ideal E^- solution $d_i^- = \left\|E_i - E^-\right\|$. Once the TOPSIS essays' scores model and the semantic dimensions are built, the GA can be used to find the optimal concept dimensions as the model criteria based on the grader's pre-scored essay.

3.3 Assessor Model Building by Using GA

For a set of pre-scored essays (scroes $S_{i,pre}$), the accuracy of the TOPSIS essay assessor model is given by the score prediction error, which is defined as

$$Err = \sqrt{\frac{\sum_{i=1}^{n} (S_i - S_{i,pre})^2}{n}} \tag{10}$$

When Err = 0, it represents that the model fits every training data perfectly. The primary goal of essay assessor model building is to minimize Err by selecting the optimal concept dimensions among LSA concept dimensions, so that the resulting assessor model can make accurate prediction. Note that high dimensions of LSA concept space may be demanded, raising the difficulty in searching the optimal semantic dimensions. To solve the problem in assessor model building, we use the GA to solve the nonlinear complex problem with a large search space, which has demonstrated its feasibility in information retrieval [9, 16].

GA initially randomly generates the population that contains individuals represented by feature-encoding chromosomes. Then, a next generation population is created by three operators: reproduction, crossover, and mutation. The reproduction operator stochastically selects individuals to become parents with the probability according to the fitness value. The crossover operator randomly swaps a portion of chromosomes between two chosen parents to produce offspring chromosomes. The mutation operator randomly flips a bit in chromosomes. For assessor model building, we set the chromosome with the form $[c_1, w_1, c_2, w_2, \ldots, c_m, w_m]$. And score prediction error function is used as the fitness function. Then, the GA can aim to build the assessor model with optimal concept dimensions and weights.

Based on the discussions above, a multicriteria automatic essay assessor generation algorithm (MAEA Generation Algorithm) is developed, described below.

Step 1: Construct the semantic dimensions with the pre-scored essays $E_1, E_2, \ldots, E_n$ by using LSA and generate the concept vectors by using the K-means clustering method with the terms.
Step 2: Initialize the GA parameters, including the number of generations, number of individuals, and bit string length of the chromosome, crossover rate, and mutation rate. And generate the initial population of chromosomes.
Step 3: Decode each chromosome for the concept dimensions and weights, and compute essay scores S_i by using the TOPSIS.
Step 4: Evaluate the fitness of each chromosome with the essay scores S_i and pre-scores $S_{i,pre}$.
Step 5: Create the new chromosomes by applying the operations of crossover and mutation.
Step 6: If the maximum number of generations is not reached, go to step 3 for the next generation; otherwise, stop and output the optimal semantic dimensions and weights.

4 Experiment

4.1 Data Collection

To evaluate the performance of the proposed method, we first collected essays from 120 college students who took the course of Business Intelligence in the department of Information Management for a semester. The essay topic is to write the related technologies of business intelligence in an evolutionary perspective. There are two human graders in the class to assess the essay. Before the essay-based examanation, students were told to prepare the examination based on three assigned magazine articles and one textbook related to the subject. The proposed method does not deal with open-ended questions. During one hour exam, students were asked to finish the essay with the book closed.

4.2 Experimental Procedure

For constructing the semantic space of the essay based on MAEA, we first use an ideal answer (provided by the grader) and 80 pre-graded essays with normalized scores uniformly distributed between 0 and 1 as the training data sets. We then initialize the GA parameters, which were set as follows:

- Number of generations: 200
- Number of individuals: 50
- Chromosome's bit string length: 200
- Crossover rate: 0.2
- Mutation rate: 0.05.

The optimal semantic dimensions and weights were found through TOPSIS model and GA.

4.3 Assessment Results

The average words in the exam. are about 250 words. The normalized mean score is 0.535 for a human grader and 0.5 for MAEA, respectively. Quantitative results show that related correlations between MAEA and two human graders are 0.861 ($p<0.01$) and 0.856 ($p<0.01$), respectively. Moreover, we make a comparison between our method and LSA-based method. The correlation coefficient between LSA-based method and two human graders are 0.826 ($p<0.01$) and 0.815($p<0.01$), respectively. Both correlation coefficients are better than those based on the LSA-only method. The result shows that MAEA performs better than LSA-only method. Furthermore, the correlation coefficient between first human grader and second grader is 0.84 ($p<0.01$) which is slightly lower than correlation coefficients between MAEA and human graders. Hence, MAEA can provide more consistent assessment rather than humans.

For further understanding the dimensions underling the concepts, we name each dimension. The ten ideal concept dimensions identified by a grader are interpreted as Data Mining (DM), Decision support system (DSS), Flexible Manufacturing (FM), Material Requirement Planning I (MARP I), Material Requirement Planning II (MARP II), Enterprise Resource Planning (ERP), Supply Chain Management (SCM), Customer relationship Management (CRM), Knowledge Management (KM) and eBusiness.

5 Conclusion

In this paper, we present a multicriteria automatic essay assessor to measure a candidate/examinee's essay. The proposed method is based on combined Latent Semantic Analysis, Technique for Order Preference by Similarity to Ideal Solution and Genetic Algorithm. The result shows that MAEA' scores are highly correlated with those of the human graders. Besides, MAEA can provide more consistent evaluation than humans. In the future, other grading criteria such as essay style or coherence can be considered in the method to better evaluate the student's essay. Besides, it would be interesting to provide the appropriate instructional training programs or materials depending on the candidate's/ examinee's domain knowledge.

References

1. Abo-Sinna, M. A. & Amer, A. H. Extensions of TOPSIS for multi-objective large-scale nonlinear programming problems. Applied Mathematics and Computation. 2005 (2005) 243-256
2. Chang, Y.-H. & Yeh, C.-H. Evaluating airline competitiveness using multiattribute decision making. Omega, 29(2001) 405-415
3. Clauser, B. E., Ross, L. P., Clyman, S. G., Rose, K. M., Margolis, M. J., Nungester, R. J., Piemme, T. E., Chang, L., El-Bayoumi, G., Malakoff, G. L., & Pincetl, P. S. Development of a scoring algorithm to replace expert rating for scoring a complex performance-based assessment. Applied Measurement in Education. 10(1997) 345-358
4. Dale, E. & Chall, J. S. A formula for predicting readability. Educational Research Bulletin, 87(1948) 11-20
5. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. Indexing by latent semantic analysis. Journal of the American Society for Information Science. 41(1990) 391-407
6. Deng, H., Yeh, C.-H., & Willis, R. J. Inter-company comparison using modified TOPSIS with objective weights. Computers and Operations Research. 10(2000) 963-973
7. Duwairi, Rehab M. A framework for the computerized assesment of university student essays. Computers in Human Behavior. 22(2006) 381-388
8. Foltz, P. W. Quantitative Approaches to Semantic Knowledge Representations. Discourse Processes. 25(1998) 127-130
9. Holland, J. H. Adaption in natural and artificial systems. Massachusetts: MIT Press (1992)
10. Hwang, C. L. & Yoon, K. 1981. Multiple Attribute Decision Making: Methods and Applications. Springer-Verlag, New York (1981)
11. Landauer, T. K. & Dumais, S. T. A Solution to Plato's Problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. Psychological Review. 104(1997) 211-240
12. Landauer, T. K., Laham, D., & Foltz, P. W. The intelligent Essay Assessor. IEEE Intelligent Systems. 15(2000) 27-31
13. Landauer, T. K., Laham, D., & Foltz, P. W. Automatic essay assessment. Assessment in Education: Principles, Policy & Practice. 10(2003) 295-308
14. Landauer, T. K. F., Peter W.& Laham, D. 1998. An Introduction to Latent Semantic Analysis. Discourse Processes. 25(1998) 259-284
15. Lemaire, B. & Dessus, P. A system to assess the semantic content of student essays. Journal of Educational Computing Research. 24(2001) 305-320
16. Martin-Bautista, M., Vila, Maria-Amparo, & Larsen, H. L. A fuzzy genetic algorithm approach to an adaptive information retrieval agent. Journal of the American Society for Information Science. 50(1999) 760-771
17. Miller, T. Essay Assessment with Latent Semantic Analysis. Journal of Educational Computing Research. 29(2003) 495-512
18. Page, E. B. 1966. The imminence of grading essays by computers. Phi Delta Kappan. 47(1966) 238-243
19. Page, E. B. 1994. Computer grading of student prose, using modern concepts and software. Journal of Experimental Education. 62(1994) 127-142
20. Page, E. B. & Petersen, N. S. The computer moves into essay grading: updating the ancient test. Phi Delta Kappan. (1995) 561-565

21. Powers, D. E., Burstein, J. C., Chodorow, M., Fowles, M. E., & Kukich, K. 2002. Stumping e-rater: challenging the validity of automated essay scoring. Computers in Human Behavior. 18(2002) 103-134
22. Rehder, B., Schreiner, M. E., Wolfe, M. B. W., Laham, D., Landauer, T. K., & Kintsch, W. Using Latent Semantic Analysis to Assess Knowledge: Some Technical Considerations. Discourse Processes, 25(1998) 337-354
23. Wolfe, M. B. W., Schreiner, M. E., Rehder, B., Laham, D., Foltz, P. W., Kintsch, W., & Landauer, T. K. Learning from Text: Matching Readers and Texts by Latent Semantic Analysis. Discourse Processes. 25(1998) 309-336

Better Student Assessing by Finding Difficulty Factors in a Fully Automated Comprehension Measure

Brooke Soden Hensler[1] and Joseph Beck[2]

Robotics Institute[1], Machine Learning Department[2]
Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213
joseph.beck@gmail.com

Abstract. The multiple choice cloze (MCC) question format is commonly used to assess students' comprehension. It is an especially useful format for ITS because it is fully automatable and can be used on any text. Unfortunately, very little is known about the factors that influence MCC question difficulty and student performance on such questions. In order to better understand student performance on MCC questions, we developed a model of MCC questions. Our model shows that the difficulty of the answer and the student's response time are the most important predictors of student performance. In addition to showing the relative impact of the terms in our model, our model provides evidence of a developmental trend in syntactic awareness beginning around the 2nd grade. Our model also accounts for 10% more variance in students' external test scores compared to the standard scoring method for MCC questions.

1 Introduction

The goal of intelligent tutoring systems (ITS) is to assist students in learning. But, the effectiveness of the tutoring can be difficult to determine as it is often difficult to assess how much the student is actually learning. Our goal is to better understand and to better score multiple choice cloze questions, and in doing so improve the accuracy and efficiency with which we assess students. We accomplish this by creating a generic, widely usable model of multiple choice cloze question assessment.

We chose to model multiple choice cloze (MCC) question assessment because it is a format that is commonly used in assessing students' comprehension of text, is amenable to ITS, can be used in any domain utilizing text, and is economical in that it uses very little time by humans to initiate and can then be created and scored by computer. An MCC question is created by deleting a word from text and asking the reader to select the deleted word from a set of response choices. MCC questions can be generated and presented in an ITS automatically. Such a format is used in the Project LISTEN Reading Tutor [1]. The Reading Tutor presents MCC questions to students with the goal of assessing reading comprehension.

Unfortunately, as with many assessments, especially newly constructed ones, it is very difficult to precisely infer students' knowledge based on question performance due to questions ranging widely in difficulty. For example, it would be inaccurate to equate raw score performance on ten easy questions to the raw score performance on ten difficult questions. The variance in question difficulty is especially acute within

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 21 – 30, 2006.

the Reading Tutor since questions are (semi) randomly generated in such a way that it is very rare for the same cloze question to appear multiple times. Therefore, we are working with a unique set of MCC questions to assess each student.

Given the disparity in question difficulty across students, our goal is to better extract information from the questions in order to assess the difficulty of the question and then use question difficulty information to better interpret the raw performance scores. By doing so, we hope to obtain more accurate student assessments. In addition to question difficulty, another factor influencing performance is students' motivation and engagement. Students' motivation and engagement vary tremendously, but past research has used MCC questions to model engagement [2].

ITS are prime candidates to develop and implement MCC assessments as they have a number of features which put them at a clear advantage over traditional pencil and paper-administered MCC assessments. First, ITS provide automatic question generation and scoring. Second, ITS enable us to consider precise response times and automatic part of speech identification for each question. These features allow us to consider factors which have not previously been part of MCC question assessment.

2 Data Description

Participants were 496 students in grades 1 through 6 (ages 5-12) in urban and suburban public schools in Pennsylvania and represented varying socio-economic statuses and ethnicities. Although there were participants from all six of these grades, MCC questions were only administered to students reading stories designated as 3rd grade reading level or above. Over the course of a school year, each student answered an average of 38 MCC questions for a total of 18,654 questions.

The Reading Tutor generated MCC questions by deleting a word (semi) randomly from the next sentence in the story the student was reading. The deleted word will be referred to throughout the paper as the "target word." The distractors were selected from the story being read and chosen to be words of similar frequency in English as the deleted word (see Fig. 1). The Reading Tutor read the sentence aloud (skipping over the deleted word) to the student and then read each response choice. The student's task was to select the word that had been deleted from the sentence. See [1] for additional details about how the cloze question intervention was instantiated in the Reading Tutor.

Fig. 1. Example of Multiple Choice Cloze Question presented on Reading Tutor

3 Modeling Approach

We developed our model of MCC question assessment by predicting whether a student would answer a *particular* MCC question correctly. Since our outcome measure is binary, we used multinomial logistic regression to calculate the relative impact of a number of terms. Our model includes *Part of Speech*, *Level of Difficulty*, *Response Time*, and *Student Identity* as factors. As covariates, we used *Tag-Primary POS Match*, *POS Confusability*, *Question Length*, *Deletion Location*, and *Syntactic Guess Rate* (see Table 1). We chose to model terms as covariates if we felt effects were generally linear and the clarity and interpretability benefited from a cleaner linear representation. Terms that were likely non-linear we treated as factors.

Although better student assessment was the impetus for creating this model, *Part of Speech* was the driving force behind it because it is amenable to computer interpretation and past research [3] has used part of speech (POS) as an indication of word knowledge. We felt that we could glean some of the students' response strategies and possibly students' knowledge from *Part of Speech*. Syntactic awareness, the idea that individuals are aware of the parts of speech of words and sensitive to the ordering of words in a sentence, has been demonstrated to be a developmental trend [4, p.275-276]. Further, studies have indicated that some parts of speech are learned before others [5, 6]. We hypothesized that more proficient readers would use syntactic cues while less proficient readers would not. For example, we suspected that more proficient readers would cue on the fact that a verb should fill the blank given the following MCC question: "We can ______ the stars in the sky." Given the response choices: *at*, *with*, *most*, *see*, a more proficient reader would likely choose "see" since it is the only verb.

To add syntax to the model, we utilized the Moby Part of Speech Database (available at http://www.dcs.shef.ac.uk/research/ilash/Moby/). The Moby POS Database identifies all possible parts of speech for words in its database and arranges these POS in decreasing order based upon frequency of use (much like the ordering of entries for each word in a standard dictionary). This POS information was useful, but could not tell us the specific POS of the target word as it was used in the cloze sentence. For example, the Moby POS Database indicates that the target word in Fig. 1, "fairy," could be a Noun or an Adjective. In order to determine the POS of the target word as it is used in the cloze sentence, we first replaced the blank with the target word. This process restored the original sentence so that the word was in context. Then, we used the TreeTagger [7] part of speech tagger to determine the target's part of speech in the sentence. The TreeTagger has demonstrated accuracy rates as high as 96.36% [7]. While we did not find the TreeTagger to be quite as accurate on our texts, spot checks of TreeTagger output on Reading Tutor data were performed by two humans and found to be acceptably accurate. Despite the TreeTagger's usefulness in its more accurate annotation of the target word as used in the cloze sentence, it is neither appropriate nor useful in determining the POS of the distractors. Therefore, we used the Moby POS Database to determine POS information for all of the response choices.

Table 1. Terms in Model

Factors	**Description of Term**	**Example Based on Fig. 1**
Part of Speech	Simplified part of speech classification of the target word as Noun, Verb, Adjective, Adverb, or Function Word.	Noun
Level of Difficulty	4 Levels of Difficulty based on frequency in English or special annotation.	Hard
Response Time	Response time rounded to the nearest second and capped at 9 seconds.	8 sec.
Student Identity	Unique Identification for each student.	Sally Student
Covariates		
Tag-Primary POS Match	Whether or not the tagged POS of the target word matched the most common POS the word could take on.	Yes
POS Confusability	The number of POS the target word can take on.	2
Question Length	Number of characters of the cloze question and the corresponding response choices.	95 characters
Deletion Location	Proportion of the sentence that is before the blank (location of word deletion).	0.19
Syntactic Guess Rate	Probability that the student could have answered the question using only part of speech information.	0.33

The first part of speech term in our model, *Part of Speech*, is the POS of the target word as annotated by the TreeTagger. To create the *Part of Speech* term, we converted the very specific part of speech tags returned by the TreeTagger (e.g., "WP$" is the tag returned for "Possessive wh-pronoun") to reflect a simplified classification system designating target words as Nouns, Verbs, Adjectives, Adverbs, or Function Words. We chose this granularity for POS classification since it is appropriate given our population, and it is consistent with other research (e.g.,[3]).

Our second part of speech term, *Tag-Primary POS Match*, is a covariate and could take on two values. *Tag-Primary POS Match* tests whether the target word's tagged POS (as computed by TreeTagger) is the same as its primary POS in the Moby POS Database. The motivation for the creation of this term was that the most common POS would probably be best known to the student, and that if a more obscure form of the word was used, the question would be more difficult.

Many words in English have multiple parts of speech (e.g., the word "pop" can, depending on contextual use, be all of our POS classifications), and we suspected that words that had multiple POS would be more difficult in that it is possible that the student may not have experienced the word used as a particular POS. Therefore, we added our third POS term, *POS Confusability*, as a covariate to account for the ambiguity of POS of the target word. *POS Confusability* is simply the number of POS the target word can take on.

The *Level of Difficulty* of the target word (and thereby all of the distractors) is classified in the Reading Tutor based on frequency in the English language [1]:

- "sight" words (the most frequent 225 words in a corpus of children's stories)
- "easy" words (the top 3,000 except for sight words)
- "hard" words (the next 22,000 words)
- "defined" words (words explicitly annotated with explanations)

Level of Difficulty was included as a factor because less proficient readers may not know the meanings of rare words. Word frequency has been used in other studies to select appropriate distractors for automatically generated MCC questions [8].

Student Identity was used as a factor to account for the overall individual performance on the questions to which she responded. Inclusion of *Student Identity* allowed us to more accurately estimate the relative impact of the other terms in our model by holding constant the impact of individual differences. This approach also accounts for the correlation among trials of a particular student, and properly calculates "N" for computing statistical significance [9].

Response Time was included as a factor because it has been demonstrated to account for engagement and performance in past studies [2]. The Reading Tutor recorded *Response Time* in milliseconds and it was later rounded to the nearest second in order to bin response times for a cleaner and more comprehensible analysis. Also, we truncated *Response Time* to nine seconds because we found that response times of greater than nine seconds were approximately the same for considering engagement.

We included *Question Length* as a covariate because we suspected that longer questions would be more difficult than shorter ones. *Question Length* was calculated as the number of characters of the sentence in addition to all of the response choices. The location of the deleted word in the sentence, *Deletion Location*, was included as a covariate because it was hypothesized that the cognitive load would be greater when the deleted word appeared early in the sentence, thereby making the question more difficult. *Deletion Location* was calculated as a percentage where the location of the blank (as measured by a count of the characters that preceded the beginning of the blank) was divided by the length of the question (as measured by the total number of characters of the sentence).

Syntactic Guess Rate, our final covariate, accounts for the probability that a student could have answered the question using only part of speech information. The ability to use part of speech information may not necessarily be an explicit strategy the student uses, but rather a skill that develops and is evidenced by response selection. For example, if the deleted target word were a noun, the blank within the sentence could syntactically take on a noun; if the student were solely using syntactic knowledge, she might consider only response choices that were nouns (even if she cannot explicitly identify these words as nouns). Presumably questions with many distractors able to take on the same part of speech as the answer, that is, words that might "fit," would be harder. *Syntactic Guess Rate* was calculated by counting the number of response choices that could take on the part of speech of the deleted target word, and then taking its inverse (e.g., In Table 1, since three response choices can take on the POS "noun," the *Syntactic Guess Rate* is one divided by three, or 0.33).

4 Results and Discussion

After training the model using multinomial logistic regression, we were able to get a relatively good fit (Nagelkerke $R^2 = 0.22$). We found that *Part of Speech*, *Level of Difficulty*, *Response Time*, *Student Identity, Question Length*, *Deletion Location*, and *Syntactic Guess Rate* have a statistically reliable impact on students' MCC question performance (see Table 2). The impact of each of these terms is reflected by the β coefficient, which is the impact each term is having on student performance when all of the other terms in the model are held constant. A positive β value indicates that as the corresponding feature increases, the student is more likely to answer the MCC question correctly. Note that since β coefficients are not normalized, it is inappropriate to compare the coefficients from different factors and covariates with each other (although it is appropriate to compare the β coefficients for various levels of a factor).

For *Part of Speech*, there are five β coefficients, one for each POS classification (e.g., Noun, Verb, etc.). Each *Part of Speech* β coefficient reflects the relative impact on student MCC performance when the target word was the particular POS. For *Part of Speech*, a positive β coefficient indicates that the particular POS was easier for the students, while a negative *Part of Speech* β coefficient indicates that particular POS was more difficult for students. Nouns were the easiest POS for students ($\beta = 0.100$), while Function Words were most difficult ($\beta = -0.191$).

Question Length affected student performance such that the longer the question (and its response choices), the more difficult the question was ($\beta = -0.006$, $p < 0.001$). It is very likely that longer questions were more difficult because students had more information to process which resulted in a higher cognitive load.

An increased cognitive load is also the most likely explanation for the impact of *Deletion Location*. Recall that *Deletion Location* was the percentage of the question that appeared before the blank (the deleted target word) in the cloze sentence. Students were more likely to answer a question correctly (i.e., the question was easier) when the *Deletion Location* score was high (i.e. when the blank appeared late in the sentence). The earlier in the sentence the blank appeared, the less likely the student would get the question correct ($\beta = 0.394$, $p < 0.001$).

Syntactic Guess Rate accounts for the chance that a question could have been answered correctly by relying on POS information. A high *Syntactic Guess Rate* score indicates fewer answer choices with a POS that would correctly fit in the blank, and thereby a higher chance that the student may have been relying on POS to answer correctly. The higher the *Syntactic Guess Rate* score, the more likely a student was to answer the question correctly ($\beta = 0.234$, $p = 0.003$).

Only two of the terms in our model, *Tag-Primary POS Match* and *POS Confusability*, did not have overall significance. However, when our population was divided based upon reading proficiency, *Tag-Primary POS Match* did have a varying effect depending on students' reading proficiency level. We will discuss two main findings in greater detail in the subsections below: a developmental trend of syntactic awareness, and using our model for more accurate student assessment.

Table 2. Impact of Terms in Model

Terms in Model	Relative Impact of each Term	Significance of Overall Effect of Term
Factors	**β**	**χ^2 p-value***
Part of Speech	-0.19 … 0.10	0.0001
Level of Difficulty	-0.96 … 0.17	1.01×10^{-46}
Response Time	-1.64 … 0.60	6.30×10^{-85}
Student Identity	-1.40 … 4.08**	4.50×10^{-171}
Covariates		
Tag-Primary POS Match	0.03	0.598
POS Confusability	-0.02	0.305
Question Length	-0.01	4.34×10^{-15}
Deletion Location	0.39	1.84×10^{-9}
Syntactic Guess Rate	0.23	0.003

* Significance of χ^2 is similar to the significance of β. χ^2 indicates the relative significance of the overall term, while, for factors, β p-values are only available for specific levels.

** For accuracy of relative impact of student performance, only students who answered 20 or more questions were included in this analysis (N=281).

4.1 Syntactic Awareness

In order to investigate students development of syntactic awareness, we used the students' Woodcock reading comprehension composite [10] test score to divide students into two groups. The Woodcock Reading Mastery Test is a standardized paper-based reading test administered by human testers that has several subtests which will be discussed in the next section. We had test scores for 373 students, and defined Low proficiency students as those who scored at the 2nd grade level or lower on the reading comprehension composite, and High proficiency as those who scored higher than the 2nd grade level. This split divided our population approximately in half.

Investigation of *Syntactic Guess Rate* revealed striking differences between High and Low proficiency readers. Our model found that students in the High group were sensitive to how many of the possible responses could take on the same part of speech as the correct answer for the cloze sentence ($\beta = 0.393$, $p = 0.002$), while students in the Low group were insensitive to this term ($\beta = 0.080$, $p = 0.467$). This result suggests that students' syntactic awareness, at least within the context of MCC questions, begins around the second grade.

Further evidence of High proficiency readers' greater awareness of syntax over that of Low proficiency readers is shown in *Tag-Primary POS Match*. *Tag-Primary POS Match* shows that High proficiency readers are affected very little by whether the POS of the target word as used in the cloze sentence is also the target word's most common POS ($\beta = -0.030$, $p = 0.709$). This could be indicative of higher proficiency readers' familiarity with multiple senses of some words. On the other hand, Low proficiency readers are possibly affected by *Tag-Primary POS Match*, and may do better when the target word is used in its most common, and likely most familiar, sense ($\beta = 0.104$, $p = 0.122$).

4.2 Using Difficulty Model for Student Assessment

We now discuss using our MCC question assessment model to estimate student reading proficiency. The approach we used was to consider the β parameter associated with each student in the logistic regression model. This parameter represents how student identity influences the probability she will correctly answer an MCC question. Therefore, β can be considered an estimate of how well the student has done answering MCC questions (holding other aspects of each question constant) and a possible estimate of the student's reading proficiency. Another, simpler commonly used, approach (e.g. [11]) to estimating student proficiency is to consider the percentage of cloze questions she answered correctly. This approach is a commonly used method of scoring cloze questions.

We compared these two approaches of estimating student MCC performance and determined how well they related to external tests of reading. Since it is difficult to assess students who have only answered few MCC questions, we restricted the analyses in this Section to students who answered at least 20 MCC questions and for whom we had Woodcock test scores. This restriction reduced our sample to 281 students.

To compare these two methods of assessing students, we computed the (square of the) correlation coefficient between each of those measures and various reading tests (see Table 3). For this comparison, we used the students' scores on relevant subtests of the Woodcock [10]. The subtests we used were decoding (ability to read words), vocabulary, and passage comprehension. We also used the reading comprehension score, a composite composed of vocabulary and passage comprehension, and total reading score, a composite of all of the Woodcock subtests. For every test, the student-specific β parameter extracted from our model as an assessment outperformed simply taking the percent of MCC questions that student answered correctly. Generally, β accounted for about 10% more variance (8% to 11%) in the test scores than did the percent correct. Therefore, β is a stronger assessment of student reading proficiency.

Table 3. Variance accounted for by logistic regression model and average percent correct

Test	r^2		Improvement in r^2
	Student β	Student % correct	
Decoding	0.34	0.23	0.11
Vocabulary	0.44	0.34	0.10
Passage comprehension	0.41	0.33	0.08
Reading comprehension composite	0.47	0.37	0.10
Total reading composite	0.43	0.32	0.10

5 Future Work

Our model uses word frequency to determine appropriate distractors for MCC questions where distractors are selected from the same word frequency classification as the target word. Our word frequency classification system breaks words into three possible categories, which can allow for words to be chosen as distractors from the same word frequency category that are actually relatively different in actual word frequency (e.g., since "Hard Words" includes 22,000 words, the 24,000th most frequent word in English could be matched with the 4,000th most frequent word). It may prove beneficial to use a more precise word frequency in selecting appropriate distractors and thereby provide a more accurate model of question difficulty.

The current model would benefit from testing on different populations. Our model relies on data of elementary school students from one geographic area. It would be interesting to test whether our model would be robust across populations. Further testing on a more diverse set of populations, especially those with a greater range of reading proficiency, could also reveal differences in the relative impacts of each term in the model, perhaps even necessitating additional terms. For example, a more extensive MCC assessment model could extrapolate Low versus High proficiency differences in older populations, such as college students. An extended MCC assessment model would likely reveal that college students are insensitive to some of the terms in our current model, such as POS, but are sensitive to other factors which we do not currently take into consideration.

6 Contributions and Conclusions

The initial goal of our model was to better understand student performance on MCC questions. In the past MCC questions have been interpreted in a crude way by looking at mean score performance (e.g.,[11]), or by using a very complex linear regression model with 54 terms [1]. Our model provides a more accurate assessment of students (by providing a β score for each student) than the standard interpretation of MCC scores, which is simply the mean score. A short-coming of our assessment is interpretability. While mean scores are easily interpreted as percentages and mapped onto familiar letter grades, our measure is more complicated and does not currently have an easily translated score, but is more accurate.

Another contribution is better understanding of the process of answering MCC questions by using our model to estimate direct effects and developmental trends. Such tasks are impossible looking at mean MCC scores as they offer no additional information beyond the score. Past models, such as the 54-term linear regression model [1], included so many factors and information external to the cloze question that it was not possible to determine just what MCC responses load on. Our model has few enough features that the effect of each term can be interpreted.

In conclusion, we have shown a domain independent MCC question assessment model that is broadly applicable as it can be on any text. (e.g., a web page). Further, we have presented a model of MCC performance. Our model enables us to determine what makes some questions hard, to examine developmental trends of students, and to more accurately assess students.

Acknowledgements. This work was supported by the National Science Foundation, ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Mostow, J., J. Beck, J. Bey, A. Cuneo, J. Sison, B. Tobin, and J. Valeri, *Using automated questions to assess reading comprehension, vocabulary, and effects of tutorial interventions.* Technology, Instruction, Cognition and Learning, 2004. **2**: p. 97-134.
2. Beck, J.E. *Engagement tracing: using response times to model student disengagement*. in *International Conference on Artificial Intelligence and Education*. 2005. p. *to appear*.
3. Schwanenflugel, P.J., S.A. Stahl, and E.L. McFalls, *Partial word knowledge and vocabulary growth during reading comprehension.* Journal of Literacy Research, 1997. **29**(4): p. 531-553.
4. Kamil, M.L., P.B. Mosenthal, P.D. Pearson, and R. Barr, eds. *Handbook of Reading Research, Volume III*. 2000, Lawrence Erlbaum Associates: Mahwah, New Jersey.
5. Gentner, D., *Some interesting differences between verbs and nouns.* Cognition and Brain Theory, 1981. **4**(2): p. 161-178.
6. Golinkoff, R.M., K. Hirsh-Pasek, L. Bloom, et al., *Becoming a word learner: A debate on lexical acquisition*. 2000, New York: Oxford University Press.
7. Schmid, H. *Probabilistic Part-of-Speech Tagging Using Decision Trees*. in *International Conference on New Methods in Language Processing*. 1994. p. 44-49.
8. Coniam, D., *A preliminary inquiry into using corpus word frequency data in the automatic generation of English language cloze tests.* CALICO Journal, 1997. **14**(2-4): p. 15-33.
9. Menard, S., *Applied Logistic Regression Analysis.* Quantitative Applications in the Social Sciences, 1995. **106**.
10. Woodcock, R.W., *Woodcock Reading Mastery Tests - Revised (WRMT-R/NU)*. 1998, Circle Pines, Minnesota: American Guidance Service.
11. Abraham, R.G. and C.A. Chapelle, *The meaning of cloze test scores: an item difficulty perspective.* The Modern Language Journal, 1992. **76**: p. 468-479.

Predicting State Test Scores Better with Intelligent Tutoring Systems: Developing Metrics to Measure Assistance Required

Mingyu Feng[1], Neil T. Heffernan[1], and Kenneth R. Koedinger[2]

[1] Computer Science Department, Worcester Polytechnic Institute
Worcester, MA, USA
{mfeng, nth}@wpi.edu
[2] Human-Computer Interaction Institute, Carnegie Mellon University
Pittsburgh, PA, USA
koedinger@cmu.edu

Abstract. The ASSISTment system was used by over 600 students in 2004-05 school year as part of their math class. While in [7] we reported student learning within the ASSISTment system, in this paper we focus on the assessment aspect. Our approach is to use data that the system collected through a year to tracking student learning and thus estimate their performance on a high-stake state test (MCAS) at the end of the year. Because our system is an intelligent tutoring system, we are able to log how much assistance students needed to solve problems (how many hints students requested and how many attempts they had to make). In this paper, our goal is to determine if the models we built by taking the assistance information into account could predict students' test scores better. We present some positive evidence that shows our goal is achieved.

1 Introduction

The limited classroom time available in middle school mathematics classes compels teachers to choose between time spent assisting students' development and time spent assessing students' abilities. To help resolve this dilemma, assistance and assessment are integrated in a web-based intelligent tutoring system ("ASSISTment") that offers instruction to students while providing detailed evaluation of their abilities to the teachers. In the 2004-2005 school year some 600+ students used the system about every two weeks to practice their skills on 8th grade Math items. These students were presented with randomly selected Massachusetts Comprehensive Assessment System (MCAS)[1] test items. If students got the *original item* correct they were given a new one, otherwise they were provided with a small "tutoring" session where they were forced to answer a few *scaffolding questions* that broke the problem down into steps. By doing this, the ASSISTment system is able to differentiate students who get the same original item wrong at first but need different levels of tutoring to get the problem correct eventually. For instance, suppose Tom, Dick and Harry all got the same original item wrong, but Tom needed to only ask for one hint to finish the whole item,

[1] http://www.doe.mass.edu/mcas

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 31 – 40, 2006.

Dick had to ask for 5 hints on one question and took a very large amount of time answering, while Harry needed no help on any of the scaffolding questions. Given these students asked for different amount of instructional assistance, we could expect Harry's MCAS score higher than Tom and Dick's. Essentially, our assistance metrics (measuring hint requests, timing information etc. as discussed in Section 3.1) are partial credit metrics and this paper asks if we can do a better job of predicting MCAS score using these assistance metrics. For those who are interested in knowing if students learn from the computer, please see [7] and [8], as this paper models primarily students' learning due to their classroom instruction.

Providing instructional assistance in the process of assessing students is the key feature of the ASSISTments. The hypothesis is that the ASSISTments can do a better job of assessing student knowledge than practice tests or other on-line testing approaches by using a "dynamic assessment" approach, thus providing a more precise prediction of student performance on the MCAS test. Feng, Heffernan and Koedinger [5] showed that by introducing the assistance students required as parameters, we were able to construct a better fitted regression model to predict students' performance on MCAS than simply using their performance on original items or on paper and pencil tests. Meanwhile, the longitudinal analysis approach[2] [9] has been applied to track student learning over time. In this paper, we propose a new method of MCAS score prediction by combining these two parts (i.e. regression model fitting plus longitudinal analysis). Specifically, our research question is:

Research question: Can we make a more precise prediction of students' performance on the MCAS by using assistance data longitudinally?

We had presented preliminary estimates of students' MCAS scores as a single column in one of our online teacher reports [4], the "Grade Book" report. The prediction was made based only upon student response on the original items. So it can not distinguish Tom, Dick and Harry in the example above. Besides, the predicted value was generated cumulatively: all past data was utilized equally while *time*, an important factor on student learning, was ignored. A positive answer to the research question would help us improve our reports.

Students' monthly performance on the original items was selected as the variable whose change we tracked longitudinally in our former work, while in this work we created two new variables **original_predicted_score** and **assistance_predicted_score** by applying regression models. The calculation of the two variables will be discussed in detail in Section 3. To answer our research question, we ran a longitudinal analysis to track the change of these two variables over time, obtained the prediction of students' MCAS scores in May 2005 and then compared the accuracy of the models as measured in *Median Absolute Difference* (MAD) – the average of absolute residual of the

---

[2] "Singer and Willet" style longitudinal data analysis is an approach for investigating change over time. It allows us to learn a slope (i.e., learning rate) and intercept (i.e. an estimate of incoming knowledge) for the group as a whole and for each individual student. This is achieved by fitting a multilevel model that simultaneously builds two sub-models, in which level-1 sub-model fits *within-person* change and describes how individuals change over time and level-2 sub-model tracks *between-person* change and describes how these changes vary across individuals This method extends well to allow us to ask questions like "Is student learning different in different schools, for different teachers or different classes?").

predicted score and students' real score in MCAS. In Section 3.5, we present the evidence that shows using the **assistance_predicted_score,** which took into account the amount of assistance students required, we did a better job estimating students' MCAS score.

2 Related Work

Other researchers have been interested in trying to get more assessment value by comparing traditional assessment (students getting an item marked wrong or even getting partial credit) with a measure that shows how much help they needed. Campione et al. [3] compared traditional testing paradigms against a dynamic testing paradigm. Grigorenko and Sternberg [6] reviewed relevant literature on the topic and expressed enthusiasm for the idea. In the dynamic testing paradigm, a student would be presented with an item and when the student appeared not to be making progress, would be given a prewritten hint. If the student was still not making progress, another prewritten hint was presented and the process was repeated. In this study they wanted to predict learning gains between pretest and posttest. They found that static testing was not as well correlated ($R = 0.45$) with student learning data as with their "dynamic testing" ($R = 0.60$) measure. Campione et al. suggested that this method could be effectively done by computer, but, as far as we know, their work was not continued. Beck et al. [2] examined using speech recognition measures (speed and correctness of reading) and student help requests to estimate reading proficiency and showed that a model can do a better job at estimating proficiency when taking into consideration student help requests. Luckily, the ASSISTment system provides an ideal test bed as it already provides a set of hints to students. In [5], we extended and tested Campione's hypothesis and replicated their finding of ASSISTment-style measures (including but not limited to student help requests) being effective and even better assessors.

3 Approach

Our new approach of MCAS score prediction combines assistance students required and the effect of time. It contains the following steps: a) Split the 494 students into training and testing sets; b) train regression models on the training set and obtain the variables entered in the models and their associated coefficients; c) apply regression models to the testing set and calculate the values of the variables **original_predicted_score** and **assistance_predicted_score** for each student for every month; d) longitudinally track student knowledge using original_predicted_score and assistance_predicted_score as an outcome variable; e) predict student MCAS score in May 2005 given the result of step d); f) compare the two outcome variables based on the MCAS score prediction result and answer the research question.

3.1 Data Source

For 2004 - 2005 school year, we collected data from 494 students who were using the ASSISTment system from September 16, 2004 through May 16, 2005 for an average of 249 minutes and finished an average of 135 items. Given the fact that the MCAS test was given on May 17, 2005, it would be inappropriate to use data after that day for the purpose of predicting MCAS scores. We also excluded data from the students' first day of using the ASSISTment system since they were learning how to use the system at that time. Though more than 600 students used our system, we were only able to collect integral data for 494 students as MCAS scores and/or the results of the paper practice test were missing for the rest. A student's raw MCAS score is out of 54 points, where each correct multiple choice or short answer question earns a point and a full correct answer to open response questions[3] earns 4 points. The paper practice test (we will refer to as *pretest)* was administered in September 2004. Students were asked to finish the test in two periods over two days (totally 80 minutes) and scores of this test were shown to be a significant predictor of MCAS scores in [5].

We constructed 15 "online measures" that we think indicate the amount of assistance a student needs to get an item correct. These online measures are:

- original_percent_correct – students' percent correct on original items only, which we often referred to as "static metric". Apparently, this measure correlates positively with knowledge.
- original_count – the number of original items students have done. This measures students' attendance and on-task-ness. The metric also reflects students' knowledge since better students have a higher potential to finish more items in the same period of time.
- percent_correct – students' percent correct over **all** questions (both original items and scaffolding questions). In addition to the original items, students' performance on the scaffolding questions is also a reasonable reflection of their knowledge. For instance, students who failed on the original items simply because of their lack of ability of forming problem-solving strategies will probably answer all the scaffolding questions correctly.
- question_count – the number of questions (both original items and scaffolding questions) students have finished. Similar to original_count, this is also a measure of attendance and knowledge but given the fact that scaffolding questions show up only if students failed the original items, it is not straightforward how this measure will correlate with students' MCAS scores.
- hint_request_count – how many times students have asked for hints.
- avg_hint_request – the average number of hint requests per question.
- hint_count – the total number of hints students received.
- avg_hint_count – the number of hint messages students received averaged over all questions.
- bottom-out_hint_count – the number of bottom-out[4] hint messages students got.
- avg_bottom_hint – the average number of bottom-out hint messages students got.
- attempt_count – the total number of attempts students made.
- avg_attempt – the average number of attempts made for each question.
- avg_question_time – on average, the length of time it takes for a student to answer a question, measured in seconds.

[3] Open response questions are not supported by the ASSISTment system currently.

[4] Since the ASSISTment system does not allow students to skip problems, to prevent students from being stuck, most questions were built such that the last hint message almost always reveals the correct answer. This message is referred to as a "bottom-out" hint.

- avg_item_time – on average, the length of time it takes for students to finish a problem (including all scaffolding questions if students answered the original items incorrectly).

The ten measures above are generally all *ASSISTment style*, dynamic assessment metrics indicating the amount of assistance students need to finish problems and the amount of time they spend to finish items. Therefore, we hypothesize all these measures would be negatively correlated with MCAS scores.

- total_minutes – the total number of minutes students worked on items in the ASSISTment system. Just like original_count, this metric is an indicator of attendance. Our hypothesis is that this measure will positively correlate with MCAS score with regard to the result we reported in [7] that students learned in the ASSISTment system.

3.2 Constructing Training and Testing Data Set

Among the 494 students, we selected approximately 50% as training individuals to train up regression models, leaving 244 students in the testing set. For the training individuals, we created a file of 250 rows with one row per student. Each row includes variables representing their associated real MCAS score, the student's pretest scores, and 15 "online measures" which we think indicate the amount of assistance a student needs to get an item correct.

In contrast to the training set, data for the 244 testing individuals are organized in the "person-period" style [9] to facilitate longitudinal analysis. To run a longitudinal data analysis, the first thing to decide is a sensible metric for time. Because a student only worked on the ASSISTments for one period (about 20 to 40 minutes, varies among schools) every time they came to the lab, rather than treating visiting days as the metric for time, we collapsed all data in one month and used *month* as the level of granularity to measure time to achieve more stable learning-over-time data. This variable for time is called "CenteredMonth" since it is centered around September 16, 2004 and it runs from 0 to 7. Rows in which CenteredMonth equals 0 contain data from Sep. 16 to October 16, and rows where CenteredMonth equals 1 contain data from October 17 to November 16 and so on. The "person-period" structured dataset contains on average 5 data waves for each student and values of all the online measures for each CenteredMonth were calculated.

To analyze data longitudinally, another important thing to determine is an outcome whose values change systematically over time. As mentioned in Section 1, traditionally students' percent correct on the original items was treated as an outcome. To mimic the real MCAS score, we multiplied the percent correct by 54 (the full MCAS score), which makes the outcome range change to 0~54. We will refer to this variable as **plain_predicted_score** to emphasize the fact that it is computed directly from students' monthly performance on original items without any correction. In addition, two new variables, referred to as **original_predicted_score** and **assistance_predicted_score**, will be calculated by applying the regression models that were trained using the training data set. The calculation of the variables will be discussed in detail in the following sections. All three predicted scores will be used as the outcome variable individually in our longitudinal data analysis and results will be compared.

3.3 Building Regression Models Based on Training Data

For a long time, we have observed that the ASSISTment system was consistently under-predicting student performance due to the following reasons. Firstly, when building the ASSISTments, authors changed the type of many questions from multiple choice to text input questions, which makes the ASSISTments on average harder than the actual MCAS items. Secondly, the ASSISTment system always allows students to ask for hints, which to some degree prevents students from trying their best to get the solution. Since hint requests were treated as false responses, this feature could impact students' evaluation. Thirdly, students did not take the ASSISTments as seriously as a real, high-stakes test such as the MCAS and finally they may behave differently when working on a computer because they like or dislike computers [1]. Therefore we want to take advantage of regression models to adjust the predicted scores.

First of all, we checked the correlations between MCAS scores and all independent variables (pretest and the 15 online measures) in the training dataset. All these factors except attempt_count turned out to be significantly correlated with MCAS scores ($p < 0.05$). The highest correlation ($r = 0.742$) occurs between MCAS score and pretest scores. Among all the online measures, original_percent_correct correlates best with MCAS score ($r= 0.709$). And the sign of the correlations verified our hypothesis about the relationships between the online measures and the MCAS score.

Table 1. Regression Models

Model	Parameter	Un-std. Coeff.	Std. Coeff.
Original_ Regression_ Model	(Constant)	4.753	
	pretest	.764	.496
	original_percent_correct	27.869	.367
Assistance_ Regression_ Model	(Constant)	26.035	
	pretest	0.64	.415
	percent_correct	24.205	.307
	avg_attempt	-10.56	-.202
	avg_hint_request	-2.283	-.125

We ran stepwise linear regressions to predict students' real MCAS scores using pretest scores plus original_percent_correct, and pretest scores plus all of the online measures respectively. The models, named Original_Regression_Model and Assistance_Regression_Model, are summarized in Table 1.

The interpretation of Table 1 is straightforward. Because of the lack of space we will only present the interpretation for Assistance_Regression_Model.

- Every one point increase in the pretest adds 0.64 points to the prediction of MCAS score. This is also the most significant parameter in both of the models according to standardized coefficients.
- It was percent_correct, not original_percent_correct that entered the model, which indicates that students' response to scaffolding questions should not be ignored when evaluating their knowledge. One percent increase on the percent correct earns student 0.24 points in the predicted MCAS score.

- The coefficient of the parameter avg_attempt is negative and thus consistent with our hypothesis about this measure. On average, if a student needs one more attempt to reach a correct answer for an item, he/she will lose 10.56 points in his/her predicted MCAS score.
- Similar to avg_attempt, avg_hint_request is also negatively correlated with MCAS score. The difference is that students' predicted score will be penalized for only 2.28 points for every hint request averaged over all questions.

3.4 Tracking Two Outcomes Longitudinally

Given Table 1, we constructed the following formulas to compute values for the two new variables that represent student knowledge in a certain month:

original_predicted_score = 4.753 + pretest * 0.764 + original_percent_correct * 27.869
assistance_predicted_score = 26.035 + pretest * 0.64 + percent_correct * 24.205 - avg_attempt * 10.56 -avg_hint_request * 2.283

It is worth pointing out that using the above formula, **assistance_predicted_score** takes into account student performance on scaffolding questions together with the amount of assistance, in particular, the number of attempts and hints, students need on average to get an item correct.

Given this data set, we fit mixed-effect models ([9], also referred to as *multilevel linear models* in sociological research) on the testing data set and continuously track **original_predicted_score** and **assistance_predicted_score** respectively. The modeling was conducted in SPSS. In [5], *school* was discovered to be a significant predictor of both students' initial knowledge status and learning rates. Hence here we introduced school as a predictor again. To facilitate our discussion, we will refer to the two models as *Original_Mixed_Model* when original_predicted_score was picked as the outcome variable and *Assistance_Mixed_Model* when assistance_predicted_score was used as the outcome variable respectively. Each model gave two parameters for any individual student, intercept (representing initial knowledge status in the first month) and slope (denoting learning rate across the 8 months).

3.5 Which Is the Best Model That Will Predict MCAS Scores?

Recall that our research question asked whether a more precise prediction can be achieved by taking into account the assistance information. To investigate this question, we computed the MAD result from the above models. Naturally, the predicted scores for the last month (i.e. CenteredMonth = 7) were adopted as the predicted MCAS score.

With predicted MCAS scores available, we can calculate MAD for both models. For the Original_Mixed_Model, we got a MAD of 6.20, with a standard deviation equal to 4.72 while for the Assistance_Mixed_Model the MAD is 5.533 with standard deviation being 4.40. Consequently, we claim that the Assistance_Mixed_Model, by utilizing the dynamic online metrics, helps to improve the correctness of the prediction on MCAS score. The paired t-test comparing absolute residuals of each student indicates the improvement is statistically significant ($p = 0.011$).

3.6 More Results

Sharp readers may have noticed that in Section 4.2, no quadratic terms or interactions between factors were included when building regression models. As a matter of fact, we suspected that there might be a non-linear relationship between the online measures and MCAS scores and therefore such a regression model was also trained and assistance_predicted_score computed. Though the R^2 of the non-linear model is higher than that of the Assistance_Regression_Model, it led to significantly larger MAD. The non-linear model probably over-fitted the training data and was thus disregarded. In both regression models presented in Table 1, pretest was a significant parameter. We wondered how much the tutoring and assistance information can help without pretest because pretest scores are not always available every school year. We replicated the whole process without using pretest. A comparison of evaluation measures to corresponding values in the above sections shows that pretest is an important predictor and without it, the precision of prediction degrades; meanwhile, the model involving tutoring and assistance information still exhibits its superiority and the difference in MAD is almost statistically significant ($p = 0.055$).

3.7 Can We Do Better, or Are We Done?

In Section 4.4, we presented that we achieved a MAD of 5.533 when predicting MCAS score using the Assistance_Mixed_Model, which is about 10.2% of the full score. To see how good the prediction is, we compare this prediction to the prediction reached by 3 other approaches as measured by MAD scores.

Among other things, pretest scores alone could be used for prediction purposes. So we did a simple regression to predict student's real MCAS scores using associated pretest scores and ended up with a MAD of 6.57 that was statistically significantly higher ($p<0.05$) than the 5.533 scores from Section 4.4.

For a second comparison we looked at the predictions in the "Grade Book" reports to teachers on our current web site (Shown in Figure 1). The prediction was primitive and was simply a linear function of percent correct on original items. For students in the testing data set, this approach gave a MAD equal to 7.47.

In yet a third comparison, we can compare it to using the *plain_predicted_score* as an outcome variable in the longitudinal analysis which brought on a MAD of 9.13. Obviously, all three of these comparisons show higher MAD values, thus indicated that they are not as good at predicting MCAS scores.

Note that the comparison between pretest-prediction-method and the ASSISTment approach confounds total time during the assessment (80 vs. 249 minutes) in the sense that it took only about 80 minutes to do the paper and pencil pretest. However, we argue that this is a fair comparison, because our schools (6 schools have adopted the system this year) say they are willing to use the ASSISTments often because they think that students are learning during their use of the ASSISTment web site.

In Section 4, we found we had reduced the MAD to 5.533, but can we do better? Should we be dissatisfied unless we can get a MAD of zero? We want to investigate what a reasonable comparison should be. Ideally, we wanted to see how good one MCAS test was at predicting another MCAS test. We could not hope to do better than that. We did not have access to data for a group of kids that took two different versions of the MCAS test to measure this, but we could estimate this by taking students' scores on MCAS, randomly splitting the test in half, and then using their score on the first half to predict the second half. We excluded open response questions from the MCAS 2005 test and kept the remaining 34 multiple-choice and short answer questions with regard to the fact that open response questions are not supported in the ASSISTment system. Then the 34 items were randomly split into two halves and student performance on one half was used to predict their performance on the other half. This process was repeated 5 times. On average, we got MAD of 1.89, which is about 11% of the full score (17 points with one point for each item). Thus we drew the conclusion that using the new approach, our prediction of MCAS score is as good as the real MCAS test itself, with the caveat that only 34 items were utilized in the process here, while our prediction models were built based on students' work on 135 ASSISTment items over eight months.

4 Conclusion and Future Work

In this paper, we continued our work in [5] and proposed a new method of MCAS score prediction by integrating timing information and the amount of assistance a student needs. To evaluate the method we compared this new method to some traditional methods. Evidence was presented that the new method did a better job of predicting student knowledge than traditional methods which only looked at students' performance on original items because items can be broken down into steps and students' responses to those steps are taken into consideration in the prediction. As our future work, we will evaluate the method further using this year's data and improve the teacher reporting system utilizing the new method.

Acknowledgements

This research was made possible by the US Dept of Education, Institute of Education Science, "Effective Mathematics Education Research" program grant #R305K03140, the Office of Naval Research grant #N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. All of the opinions in this article are those of the authors, and not those of any of the funders.

We thank Professor Brian Junker from Carnegie Mellon University for helpful comment on this work. This work would not have been possible without the assistance of the 2004-2005 WPI/CMU ASSISTment Team including Nathaniel O. Anozie, Elizabeth Ayers, Brian Junker, Andrea Knight, Meghan Myers, Carolyn Rose all at CMU, Steven Ritter at Carnegie Learning, Tom Livak, Abraao Lourenco, Michael Macasek, Goss Nuzzo-Jones, Kai Rasmussen, Leena Razzaq, Terrence Turner, Ruta Upalekar, and Jason Walonoski all at WPI.

References

1. Baker, R.S., Roll, I., Corbett, A.T., Koedinger, K.R. (2005) Do Performance Goals Lead Students to Game the System? In *Proceedings of the 12th International Conference on Artificial Intelligence and Education*, 57-64.
2. Beck, J. E., Jia, P., Mostow, J. (2004). Automatically assessing oral reading fluency in a computer tutor that listens. In *Technology, Instruction, Cognition and Learning*, 2, 61-81.

3. Campione, J.C., Brown, A.L., & Bryant, N.R. (1985). Individual differences in learning and memory. In R.J. Sternberg (Ed.). In *Human abilities: An information-processing approach*, 103-126. New York: W.H. Freeman.
4. Feng, Mingyu, Heffernan, N.T. (2005). Informing Teachers Live about Student Learning: Reporting in the ASSISTment System. *The 12th Annual Conference on Artificial Intelligence in Education Workshop on Usage Analysis in Learning Systems*, 2005.
5. Feng, M., Heffernan, N.T, Koedinger, K.R. (in press). Addressing the Testing Challenge with a Web-Based E-Assessment System that Tutors as it Assesses. *The 15th International World Wide Web Conference*, Scotland. http://web.cs.wpi.edu/~mfeng/pub/www06.pdf
6. Grigorenko, E.L & Sternberg, R.J. (1998). *Dynamic Testing.* In *Psychological Bulletin*, 124, 75-111.
7. Razzaq, L, Feng, M., Nuzzo-Jones, G., Heffernan, N.T. et al. (2005). The ASSISTment Project: Blending Assessment and Assisting. In *Proceedings of the 12th Annual Conference on Artificial Intelligence in Education.*
8. Razzaq, L., & Heffernan, N. T. (in press). Scaffolding vs. Hints in the ASSISTment System. *The 8th International Conference on Intelligent Tutoring Systems*.http://nth.wpi.edu/pubs_and_grants/ITS2006/Submissons/Leena/razzaq.doc
9. Singer, J. D. & Willett, J. B. (2003). *Applied Longitudinal Data Analysis: Modelling Change and Occurrence*. Oxford University Press, New York.

Authoring Constraint-Based Tutors in ASPIRE

Antonija Mitrovic, Pramuditha Suraweera, Brent Martin,
Konstantin Zakharov, Nancy Milik, and Jay Holland

Intelligent Computer Tutoring Group
University of Canterbury, Christchurch, New Zealand
{tanja, psu16, brent, kza10, nmi14,
jah130}@cosc.canterbury.ac.nz

Abstract. This paper presents a project the goal of which is to develop ASPIRE, a complete authoring and deployment environment for constraint-based intelligent tutoring systems (ITSs). ASPIRE is based on our previous work on constraint-based tutors and WETAS, the tutoring shell. ASPIRE consists of the authoring server (ASPIRE-Author), which enables domain experts to easily develop new constraint-base tutors, and a tutoring server (ASPIRE-Tutor), which deploys the developed systems. Preliminary evaluation shows that ASPIRE is successful in producing domain models, but more thorough evaluation is planned.

1 Introduction

Building a constraint-based tutor, like any other ITS, is a labour-intensive process that requires expertise in constraint-based modelling (CBM) and programming. While ITSs contain a few modules that are domain-independent, their domain model, which consumes the majority of the development effort, is unique. Our goal is to reduce the time and effort required for producing ITSs by building an authoring system that can generate the domain model with the assistance of a domain expert and produce a fully functional system. We also envisage that the authoring system would enable teachers, with little or no expertise in CBM, to build their own ITSs.

This paper presents ASPIRE, an authoring system that assists in the process of composing domain models for constraint-based tutors and automatically serves tutoring systems on the web. The proposed system is an enhancement of WETAS [4, 5], a web-based tutoring shell that facilitates building constraint-based tutors. WETAS is a prototype system that provides all the domain-independent components for text-based ITSs. The main limitation of WETAS is its lack of support for authoring domain models. ASPIRE guides the author through a semi-automated process for building the domain model and seamlessly deploys the resulting domain model to produce a fully functional web-based tutoring system.

The paper commences with a brief introduction to related authoring systems for building ITSs. Section 3 details the ASPIRE authoring system, including an outline of the domain authoring process and the architecture of the system. We also include an overview the constraint generation algorithms, the central component of the authoring process. Finally, Section 4 presents conclusions and the directions of future work.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 41 – 50, 2006.

2 Related Work

Murray [10] classified ITS authoring tools into two main groups: pedagogy-oriented and performance-oriented. Pedagogy-oriented systems focus on instructional sequencing and teach relatively fixed content. On the other hand, performance-oriented systems focus on providing rich learning environments, where students learn by solving problems while receiving dynamic feedback on their progress. These systems have a deep model of expertise, which enables the tutor to correct the student as well as provide assistance on problem solving. Authoring systems thus need to support the acquisition of domain models. Typically, sophisticated machine learning techniques are used for acquiring domain rules with the assistance of a domain expert.

Only a few authoring systems are capable of generating domain models. Disciple, developed by Tecuci and co-workers [15, 16], is an example of a learning agent shell for developing intelligent educational agents. A domain expert teaches the agent to perform domain-specific tasks, similar to a manner of an expert teaching an apprentice, by providing examples and explanations. The expert is also required to supervise and correct the behaviour of the agent. Disciple acquires knowledge using a collection of complementary learning methods including inductive learning from examples, explanation-based learning, learning by analogy and learning by experimentation. A completed Disciple agent can be used to interact and guide students in performing tasks of the domain.

The Cognitive Tutor Authoring Tools (CTAT) [1, 2] assist in the creation and delivery of ITSs based on model tracing. The main goal of these tools is to reduce the amount of artificial intelligence (AI) programming expertise required. The system allows authors to create two types of tutors: ‘Cognitive tutors’ and ‘Pseudo tutors’. ‘Cognitive tutors’ contain a cognitive model that simulates the student's thinking to monitor and provide pedagogical assistance during problem solving. In contrast, ‘Pseudo tutors’ do not contain a cognitive model: to develop a tutor of this kind, the author needs to specify a recording of possible student actions and corresponding feedback messages. Although ‘Pseudo tutors’ do not require AI programming, they are specific to the demonstrated set of problems, and cannot deal with student actions’ which are not pre-specified by the author.

3 ASPIRE

ASPIRE assists with the creation and delivery of constraint-based tutoring systems. It generates constraints that make up the domain model with the assistance of the domain expert, minimising the programming expertise required for developing a new constraint-based tutor. The system also provides all the domain-independent functionality of constraint-based ITSs.

3.1 Authoring Process

Authoring a constraint-based tutor in ASPIRE is a semi-automated process, carried out with the assistance of the domain expert. The authoring process, summarised in Figure 1, consists of nine distinct phases. Initially, the author specifies general features of the chosen instructional domain, such as whether the domain consists of a sub-domains focusing on specific areas, and whether the domain is procedural or not. In the case of procedural domains, the author is required to enumerate the problem-solving steps. As an example, let us consider the procedural domain of adding fractions. The problem-solving procedure can be broken down into four steps, as outlined in Figure 2. Initially, it is necessary to check whether the two fractions have the same denominator; if that is not the case, the lowest common denominator must be found. Step two involves modifying the two fractions to have the lowest common denominator (when needed). After that, the two fractions are added, which may result in an improper fraction. Finally, the result is to be simplified, if appropriate.

1. Specifying the domain characteristics
2. Composing the domain ontology
3. Modelling the problem and solution structures
4. Designing the student interface
5. Adding problems and solutions
6. Generating syntax constraints
7. Generating semantic constraints
8. Validating the generated constraints
9. Deploying the tutoring system

Fig. 1. The phases of the authoring process

In the second phase, the author develops an ontology of the chosen instructional domain, which plays a central role in the authoring process. ASPIRE-Author provides an ontology workspace for visually modelling ontologies (Figure 3). A domain ontology describes the domain by identifying important concepts and relationships between them. The ontology outlines the hierarchical structure of the domain in terms of sub- and super-concepts. Each concept might have a number of properties, and may be related to many other domain concepts. A preliminary study conducted to evaluate the role of ontologies in manually composing a constraint base showed that constructing a domain ontology assisted the composition of constraints [13]. The study showed that ontologies support authors to reflect on the domain, organise constraints into meaningful categories and produce more complete constraint bases.

1. Find the lowest common denominator (LCD)
2. Convert fractions to LCD as denominator
3. Add the resulting fractions
4. Simplify the final result

Fig. 2. Problem-solving procedure for fraction addition

An ontology for the domain of adding fractions is illustrated in Figure 3. It contains *Number* as the most generic concept, which has two specialisations, *Whole-number* and *Fraction*. *Whole-number* is further specialised into lowest common denominator (*LCD*), while *Fraction* is specialised into *Improper* and *Reduced*. The specialization/generalization relationships between domain concepts are visually represented as arrows between concepts. Figure 3 shows three additional relationships defined for the *Reduced Fraction* concept: *whole number*, *numerator* and *denominator*. While *numerator* and *denominator* and mandatory relationships, *whole number* may only occur if the resulting fraction needs to be simplified.

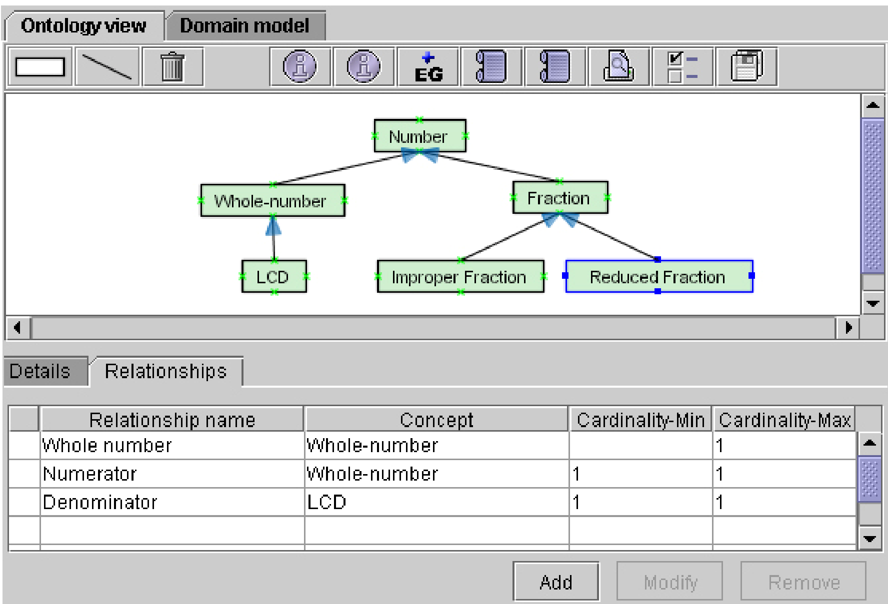


Fig. 3. Ontology for adding fractions

In the third phase, the author specifies the problem/solution structures. Problems can consist of components (textual or graphical) and a problem statement. In our example domain, problems contain a common statement ("Add these two fractions"), and the problem to be solved (e.g. "1/3 + 1/5"). Student solutions may also consist of several components. The overall structure of solutions depends on whether the domain is procedural or declarative. A declarative task requires a single solution that may consist of a number of components, whereas a procedural task requires a solution for each step of the procedure. As the result, the structure of solutions for each step has to be modelled. The solution structure for fraction addition is outlined in Figure 4, showing also the corresponding domain concepts.

The student interface needs to be designed next. The final outcome of this phase is a form-based interface that can be used by students to compose their solutions. The system initially generates a default interface, placing an input area for each component defined in the solution structure [9]. The domain expert can rearrange the interface components in order to provide a more intuitive interface for students. An example of an interface for adding fractions is shown in Figure 5.

After designing the student interface, the author enters example problems and their solutions. For each problem, the author enters a problem statement, and one or more correct solutions. In order for the authoring system to learn about different ways of solving a problem, the expert is required to provide multiple solutions to a problem depicting different ways of solving it. These solutions are used by the authoring system for generating semantic constraints.

Problem solving step	Solution component	Concept
1. Find LCD	LCD	LCD
2. Convert fractions to LCD	Fraction 1 numerator Fraction 1 denominator Fraction 2 numerator Fraction 2 denominator	Improper fraction
3. Sum of improper fractions	Improper sum numerator Improper sum denominator	Improper fraction
4. Final reduced sum	Final sum whole number Final sum numerator Final sum denominator	Reduced fraction

Fig. 4. Solution structure for adding two fractions

Once example problems and their solutions are available, ASPIRE-Author generates the domain model. The syntax constraint generator analyses the domain ontology and generates syntax constraints directly from it. These constraints are generated by translating the restrictions on the properties and relationships of concepts specified in the ontology, as detailed in Section 0. The constraint generator produces an extra set of syntax constraints for procedural domains that ensure that the student progresses correctly in the problem solving process.

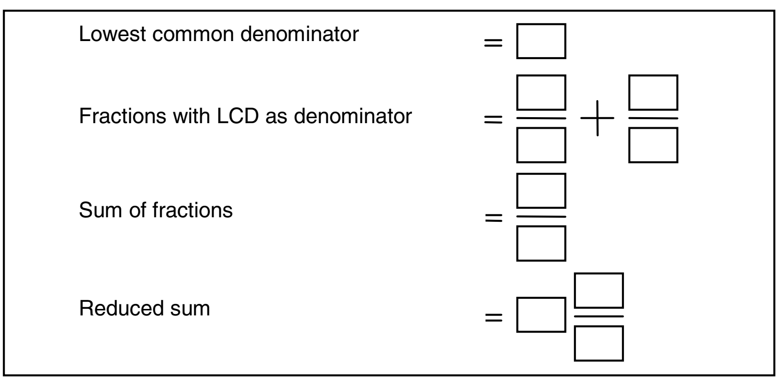


Fig. 5. Student interface for adding two fractions

Semantic constraints are generated using a machine learning algorithm that learns from the solutions provided for each problem. It analysing pairs of solutions to identify similarities and differences between them. Section 0 provides more details on the semantic constraint generation algorithm.

The generated domain model is validated during the penultimate phase of authoring the domain model. The author requests the system to identify errors in an incorrect solution. If errors are identified incorrectly, further example problems and solutions have to be provided by the domain expert. The author may also examine a high-level description of each generated constraint and dispute them by providing counter examples.

Finally, the domain model is deployed as a tutoring system during the final phase of the authoring process. A new instance of a tutoring system is started in ASPIRE-Tutor, which can be tested by the domain expert and made available to students. The domain expert can evaluate the effectiveness of the domain model by analysing the learning curves for constraints produced by ASPIRE-Tutor.

3.2 Architecture

ASPIRE consists of an authoring server (ASPIRE-Author) for assisting with the development of new systems, and a tutoring server (ASPIRE-Tutor) for delivering tutors. Both servers are implemented in Allegro Common Lisp [3] as web servers for users to interact through a standard web browser. All required domain-dependent information, such as the domain model and other configuration details produced by ASPIRE-Author, are transferred to ASPIRE-Tutor as an XML database.

3.2.1 Authoring Server

The authoring server consists of a set of modules, where each module is assigned a specific set of responsibilities in generating constraint-based tutors. The basic architecture of the ASPIRE-Author, as depicted in Figure 6, consists of a web interface, authoring controller, constraint generator, constraint validator and the domain model manager [8]. The domain expert interacts with each component of the web interface to generate the domain model.

The Authoring Controller manages the process and guides the author. This module receives all requests from the interface layer, initiates processes within other modules and returns the results to the relevant interface component.

The Syntax Constraint Generator is responsible for generating syntax constraints by analysing the domain ontology. Semantic constraints are generated by the Semantic Constraint Generator using a machine learning algorithm that learns from problems and their solutions. The Constraint Validator is responsible for carrying out all the necessary operations required for validating the constraints generated by the constraint generators.

The Domain Model Manager contains the necessary classes for storing the components of domain models. It is responsible for creating and updating domain model components such as ontology, problem solution structure, problems, solutions etc. The Domain Model Manager is also capable of producing XML representations of all domain model components for data transfer.

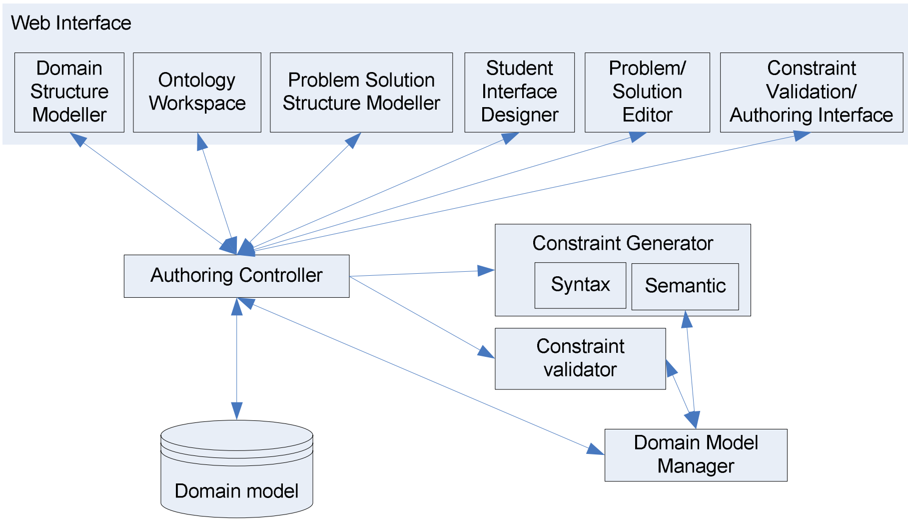


Fig. 6. The architecture of ASPIRE-Author

3.2.2 Tutoring Server

ASPIRE-Tutor (Figure 7) is also designed as a collection of modules, based on the typical ITS architecture. ASPIRE-Tutor is capable of serving a collection of tutoring systems in parallel. Each tutoring system served by ASPIRE-Tutor would have its own unique URL. Students can access the tutoring system relevant to them by pointing their browser to the appropriate URL.

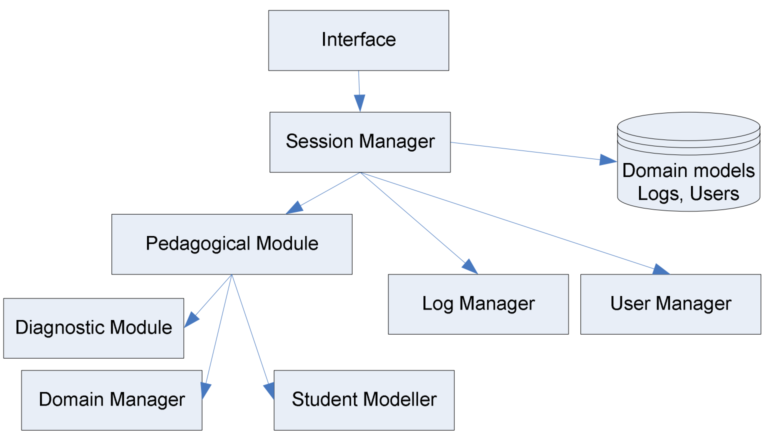


Fig. 7. The architecture of ASPIRE-Tutor

The interface module is responsible for producing an interface for each tutoring system deployed on the server. The interface provides features such as login/logout, select/change problem, submit solution for evaluation etc.

The session manager is responsible for maintaining the state of each student during their interaction. The current state of a student is described by information such as the selected domain, sub-domain and problem number. The session manager also acts as the main entry point to the system, invoking the relevant modules to carry out necessary tasks. For example, when a student submits a solution to be validated, the session manager passes on all information to the pedagogical module, which returns the feedback to be presented to the student.

The Pedagogical Module (PM) decides how to respond to each student request. It is responsible for handing all pedagogy-related requests including selecting a new problem, evaluating a student's submission and viewing the student model. In the event of evaluating a student's submission and providing feedback, the PM delegates the task of evaluating the solution to the diagnostic module and decides on the appropriate feedback by consulting the student model. The student modeler maintains a long term model of the student's knowledge.

3.3 Syntax Constraints Generation

An ontology contains a lot of information about the syntax of the domain. Composing a domain ontology is a much easier task for the author than composing constraints that check whether the student has used correct syntax. The goal of syntax constraint generator is to extract all useful syntactic information from the ontology and translate them into syntax constraints for the domain model.

Syntax constraints are generated by analysing relationships between concepts and properties of concepts specified in the ontology. The algorithm extracts the restrictions specified for relationships and properties and generates syntax constraints by translating them into constraints. These constraints are applicable to both procedural and non-procedural domains. An extra set of constraints are generated for procedural domains to ensure that the student adheres to the correct problem-solving procedure. These constraints are generated by analysing the solution structure modelled during stage three of the authoring process. The syntax constraints generation algorithm is detailed in further in [12, 14].

ASPIRE-Author produced 11 constraints for fraction addition from the ontology in Figure 3 and the solution structure in Figure 4. For example, constraint 7 is relevant while the student is carrying out the first problem solving step ('Find LCD') and its satisfaction condition ensures that the student has entered the answer. As the domain does not contain any complicated syntax restrictions, and inputs are restricted by the student interface, the generated constraints are sufficient to ensure that students use the correct syntax and the correct problem-solving procedure.

The syntax constraint generation algorithm has been evaluated in a number of domains. The evaluations carried out for the domains of ER modelling and database normalisation produced promising results. All syntax constraints that were hand-crafted in KERMIT [7, 11], a successful constraint-based tutor for ER modelling were generated by ASPIRE. Furthermore, the algorithm produced all but two syntax constraints that existed in NORMIT [6, 7], an effective tutoring system for database normalisation.

3.4 Semantic Constraints Generation

Semantic constraints ensure that a student's solution satisfies all semantic requirements of a problem, by comparing the student's and ideal solution. They are generated by a machine learning algorithm. Problems and solutions provided by the author are used as examples for semantic constraint generation. Multiple solutions for a problem depict different ways of solving it, and enable the algorithm to generate constraints that can identify all correct solutions, regardless of the student's approach.

The algorithm generates new semantic constraints by analysing a pair of correct solutions for the same problem. Constraints are generated by identifying similarities and differences between two solutions. The process of generating constraints is iterated until all pairs of solutions are analysed. Each new pair of solutions can lead to either generalising or specialising previously generated constraints. If a newly analysed pair of solutions violate a previously generated constraint, its satisfaction condition is generalised in order to satisfy the solutions, or the constraint's relevance condition is specialised for the constraint to be irrelevant for the solutions. This algorithm is discussed in [12]. Evaluations performed show that the semantic constraints generator produced 85% of the semantic constraints found in KERMIT. Moreover, the generated constraints for the domain of database normalisation covered all the semantic constraints that exist in NORMIT.

39 semantic constraints were generated for fraction addition, from only two example problems. As each problem in this domain has only a single valid solution, semantic constraints check that the student's solution matches the ideal solution. For example, constraint 1 ensures that if the student is currently doing the first problem solving step ('Find LCD'), the LCD component of their solution is not empty (i.e., the student has specified the LCD) and the ideal solution contains an LCD (i.e. it is necessary to find the LCD for the current problem), then the student's answer needs to be equal to the one specified in the ideal solution.

The majority of generated semantic constraints ensure that relationships, such as fractions having a numerator and a denominator, exist in student solutions. As the interface implicitly forces these relationships, some semantic constraints are trivially satisfied. However, we believe that it is still necessary for the domain model to contain such constraints, because the author may design a less restrictive interface. Only two example problems were needed to generate semantic constraints for fraction addition, as the domain is very simple.

4 Conclusions

We provided an overview of ASPIRE, an authoring system that assists domain experts in building constraint-based ITSs and serves the developed tutoring systems over the web. ASPIRE follows a semi-automated process for generating domain models, and produces a fully functional web-based ITS, which can be used by students. We also outlined the constraint generation algorithms, which produced promising results during preliminary evaluations. ASPIRE-Author produced a satisfactory domain model for fraction addition, consisting of 11 syntax and 39 semantic constraints. The generated domain model can be used to power a tutoring system for students with minor modifications.

ASPIRE will be completed in July 2006, and then we will conduct a thorough evaluation of the system's effectiveness. We also intend to develop a tutorial outlining the authoring process to assist novices in building constraint-based tutoring systems using ASPIRE, especially modelling domain ontologies.

Acknowledgements. The ASPIRE project is supported by the eCDF grant from the Tertiary Education Commission of New Zealand. We thank all members of ICTG for their support.

References

1. Jarvis, M., Nuzzo-Jones, G., Heffernan, N., Applying Machine Learning Techniques to Rule Generation in Intelligent Tutoring Systems. In *ITS 2004*, (Maceio, Brazil, 2004), Springer, 541-553.
2. Koedinger, K., Aleven, V., Heffernan, N., McLaren, B. and Hockenberry, M., Openning the Door to Non-programmers: Authoring Intelligent Tutor Behavior by Demonstration. In *ITS 2004*, (Maceio, Brazil, 2004), Springer, 162-174.
3. Allegro Common Lisp (www.franz.com)
4. Martin, B., Mitrovic, A. Authoring Web-Based Tutoring Systems with WETAS. Kinshuk, R. Lewis, K. Akahori, R. Kemp, T. Okamoto, L. Henderson, C-H Lee (eds) Proc. ICCE 2002 (Auckland, 2002), 183-187.
5. Martin, B., Mitrovic, A. Domain Modelling: Art or Science? In: U. Hoppe, F. Verdejo & J. Kay (ed) Artificial Intelligence in Education 2003, 183-190.
6. Mitrovic, A. The Effect of Explaining on Learning: a Case Study with a Data Normalization Tutor. In: C-K Looi, G. McCalla, B. Bredeweg, J. Breuker (eds) Proc. Artificial Intelligence in Education, 2005, IOS Press, 499-506.
7. Mitrovic, A., Suraweera, P., Martin, B., Weerasinghe, A. DB-suite: Experiences with Three Intelligent, Web-based Database Tutors. *Journal of Interactive Learning Research*, 15, 2004, 409-432.
8. Mitrovic, A., Martin, B., Suraweera, P., Zakharov, K., Milik, N., Holland, J. ASPIRE: Functional Specification and Architectural Design**.** Tech. Report TR-COSC 05/05, University of Canterbury, 2005.
9. Mitrovic, A., Martin, B., Suraweera, P., Zakharov, K., Milik, N., Holland, J. ASPIRE: Student Modelling and Domain Specification. Tech. Report TR-COSC 08/05, University of Canterbury, 2005.
10. Murray, T. An Overview of Intelligent Tutoring System Authoring Tools: Updated analysis of the state of the art. *Authoring tools for advanced technology learning environments*. 2003, 491-545.
11. Suraweera, P., Mitrovic, A., An Intelligent Tutoring System for Entity Relationship Modelling. *Artificial Intelligence in Education*, 14, (2004), 375-417.
12. Suraweera, P., Mitrovic, A., Martin, B., A Knowledge Acquisition System for Constraint-based Intelligent Tutoring Systems. In: C-K Looi, G. McCalla, B. Bredeweg, J. Breuker (eds) Artificial Intelligence in Education, 2005, IOS Press, 638-645.
13. Suraweera, P., Mitrovic, A., Martin, B., The role of domain ontology in knowledge acquisition for ITSs. In *Intelligent Tutoring Systems 2004*, (Maceio, Brazil, 2004), Springer, 207-216.
14. Suraweera, P., Mitrovic, A., Martin, B., The use of ontologies in ITS domain knowledge authoring. in *2nd Int. Workshop on Applications of Semantic Web for E-learning SWEL'04, ITS2004*, (Maceio, Brazil, 2004), 41-49.
15. Tecuci, G. *Building Intelligent Agents: An Apprenticeship Multistrategy Learning Theory, Methodology, Tool and Case Studies*. Academic press, 1998.
16. Tecuci, G., Keeling, H. Developing an Intelligent Educational Agent with Disciple. *Artificial Intelligence in Education*, 10, 1999, 221-237.

A Teaching Strategies Engine Using Translation from SWRL to Jess

Eric Wang and Yong Se Kim

Creative Design and Intelligent Tutoring Systems Research Center
Sungkyunkwan University, Suwon, Korea
{ewang, yskim}@skku.edu

Abstract. Within an intelligent tutoring system framework, the teaching strategy engine stores and executes teaching strategies. A teaching strategy is a kind of procedural knowledge, generically an if-then rule that queries the learner's state and performs teaching actions. We develop a concrete implementation of a teaching strategy engine based on an automatic conversion from SWRL to Jess. This conversion consists of four steps: (1) SWRL rules are written using Protégé's SWRLTab editor; (2) the SWRL rule portions of Protégé's OWL file format are converted to SWRLRDF format via an XSLT stylesheet; (3) SweetRules converts SWRLRDF to CLIPS/Jess format; (4) syntax-based transformations are applied using Jess meta-programming to provide certain extensions to SWRL syntax. The resulting rules are then added to the Jess run-time environment. We demonstrate this system by implementting a scenario with a set of learning contents and rules, and showing the run-time interaction with a learner.

1 Introduction

We are developing an intelligent tutoring system framework, in which learning contents and learner data are stored in ontologies. Within this framework, the teaching strategy engine stores and executes teaching strategies to determine the teaching actions. Teaching strategies are a kind of procedural knowledge, including assessment algorithms and decision procedures. Traditionally, such knowledge has been difficult to represent within an ontology in a format that supports automatic execution. However, recent advances in integrating rules and ontologies, and the development of tools to support them, present an opportunity to develop a teaching strategy engine using ontology-based rule representations. We have developed a conversion from SWRL rules written in Protégé, to a run-time engine in Jess, using several standard open-source tools. The benefit of representing teaching strategies as SWRL rules is that the strategies' computations would be explicitly represented in the ontology, and could be viewed and edited, as well as reasoned about by other applications.

2 Tools

Our conversion method uses the following standard tools.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 51 – 60, 2006.

2.1 SweetRules

SweetRules 236 is a toolkit that provides a suite of converters between several standard rule formats, including RuleML, Courteous Logic Programs, and SWRL, and several execution engines, including Jess, Jena, and XSB Prolog.

2.2 Semantic Web Rule Language

Semantic Web Rule Language (SWRL) 4 combines a rich OWL ontology + description logic (DL) with a subset of first-order logic (FOL) syntax. SWRL rules are always defined on top of an OWL ontology. Syntactically, a SWRL rule is function-free (no user-defined functions), Datalog (no functions within terms), Horn (no negation or disjunction), and has no explicit quantifiers, but with implicit universal quantification for all variables. SWRL atoms include class and property atoms whose names refer to classes and properties in the ontology, and a library of built-in function atoms that implement fundamental math, string, and date operations.

Class and property atoms in a SWRL rule body represent queries into the knowledge base. A class atom with a ground argument (individual name or bound variable) performs a class membership test. If the argument is an unbound variable, i.e. this atom is the first occurrence of the variable in the rule, then a class atom conceptually iterates over every individual of that class. (This iteration actually arises from the implicit universal quantification over all variables, which ensures that every rule will be applied to every individual in the knowledge base.) Property atoms are analogous, but perform property membership tests.

In a rule head, class and property atoms represent new conclusions. According to the SWRL semantics, an ontology with rules is consistent iff the head facts are true whenever the bodies are true. The standard implementation of this semantics is to *make* the head facts true, i.e. to add them to the knowledge base. SWRL has no intrinsic facilities for performing side-effects, including modifying existing knowledge.

The SWRL standard 4 defines two concrete XML-based formats for SWRL rule representation: an XML concrete syntax, and an RDF concrete syntax. SweetRules denotes these formats as SWRLXML and SWRLRDF, respectively, and provides some translation tools to and from these formats.

2.3 Protégé OWL + SWRLTab

Protégé 5 is a standard, open-source ontology editor, with support for Web Ontology Language (OWL) via an OWL plugin. Recent builds of Protégé OWL include a SWRLTab view, which provides convenient editing of SWRL rules. The SWRLTab editor uses SWRL's "human readable" syntax 4, which is comparable to Prolog in conciseness and readability. Another key benefit is that the SWRL rule editor is closely integrated with the OWL ontology.

From Protégé's perspective, the SWRL language syntax is represented as an ontology, and a SWRL rule is simply an OWL individual that is instantiated using classes from this ontology. The SWRLTab editor automates the laborious details of instantiating the tree-like structure for each SWRL rule. Protégé saves the OWL ontology and SWRL rules into the same .owl file. (This causes some complications for the conversion to Jess, as will be discussed below.)

2.4 Jess Rule Engine

Jess (Java Expert System Shell) 1 is a rule engine written in Java. It provides a Lisp-like syntax and interpreter, with forward-chaining rules using the rete network algorithm. It is based on the earlier CLIPS rule language, and is still largely forward-compatible with CLIPS, in that most CLIPS programs are also valid Jess programs. It is free for research use.

Jess is able to use standard Java reflection to import Java libraries, call any Java code, and directly manipulate Java objects. This gives it great flexibility as a general-purpose execution environment. We exploit this capability to run XSLT and SweetRules from within Jess.

3 Conversion from OWL Ontology to Jess Facts

An OWL ontology consists of classes and properties, OWL restrictions to specify their semantics, and individuals defined using these elements. SweetRules provides a translation path (a sequence of translator tools) from OWL to Jess, which can be used to convert a knowledge base of facts. In particular, we can use this to convert the learning contents ontology to Jess facts.

An OWL ontology file is converted to Jess by executing the SweetRules command:

translate owl jess owl-input-path jess-output-path

This produces a Jess file in which OWL individuals are converted to Jess facts (simple assertions), and OWL properties and restrictions are converted to Jess rules.

Note that we must exclude SWRL rules from consideration here, since this translation path has expressiveness limitations that conflict with Protégé's handling of SWRL rules.

- SweetRules's OWL-to-Jess translation path is restricted to a subset of Protégé OWL's expressiveness. The OWL *functional* property attribute, which denotes that a property may have at most one value, must be avoided, as it is a kind of cardinality restriction, which this translation path can't support.
- However, to edit SWRL rules in Protégé, the user must "activate" SWRL, which imports the SWRL ontology definition. The SWRL ontology itself uses functional properties internally.

Hence, a Protégé .owl file that has "activated" the SWRL editing capabililty can no longer be translated via SweetRules' OWL-to-Jess path. It follows that to use the OWL-to-Jess translation requires two separate Protégé .owl files: (1) an "ontology-only" file that defines all individuals, but does not activate SWRL, which is converted as described here; and (2) a "SWRL-only" file that defines the SWRL rules, which is converted separately, as described in the next section. These two files must share the same class and property definitions, and declare the same xml:base URI, to ensure that individuals generated from one file will match rules generated from the other.

4 Conversion from SWRL Rules to Jess Rules

In this section, we describe our conversion from Protégé's SWRL rules to Jess code.

4.1 Conversion from Protégé OWL to SWRLRDF

Within Protégé's .owl file format, SWRL rules are saved in a syntax that is similar to SWRL's RDF concrete syntax (SWRLRDF format), but with some specific differences. We have developed an XSLT stylesheet to convert the SWRL rule subset of a Protégé .owl file to SWRLRDF. This conversion includes the following steps.

- **Convert lists to collections.** In Protégé OWL, swrl:body and swrl:head atoms are defined as lists, using a Lisp-like recursive list structure that provides explicit sequential ordering. Each swrl:body or swrl:head atom has a single swrl:AtomList child node, which itself has exactly two child nodes: an rdf:first node whose child is the first element of the (sub)list, and an rdf:rest node whose child is the remainder of the list, recursively represented as another swrl:AtomList, or by the special end-of-list symbol #nil. Since the rdf:first and rdf:rest nodes are explicitly named, their order within the file is arbitrary.

  SWRLRDF defines ruleml:body and ruleml:head atoms as collections of multiple nodes, with implicit sequential ordering between them. Specificially, these atoms have an rdf:parseType="Collection" attribute, and can have any number of child nodes.

  We convert the swrl:AtomList format to the rdf:parseType="Collection" format, preserving the sequence of the elements.
- **Lift variable declarations to top of file.** Every variable used in any SWRL rule must be declared once as a swrl:Variable atom. In SWRLRDF, it is preferred to list all such declarations *a priori*, in a block at the top of the file, before any SWRL rule definition. This convention simplifies subsequent conversion processes by automated tools.

```
<?xml version="1.0"?>
<rdf:RDF ...>     Root node
  <swrl:Variable rdf:ID="x" />  Variable declarations
  ...
  <swrl:Imp rdf:ID="rule-1">          Rule definitions
    ...
</rdf:RDF>
```

  In contrast, OWL supports an enhanced syntax with "just-in-time" declarations, in which the first occurrence of an identifier in the .owl file is written as a child node below the node where it is first used, with an rdf:id=*name* attribute, which acts as a declaration. All subsequent occurrences of the same identifier in this file use an rdf:resource=#*name* attribute on the node that uses it, which acts as a reference to the previous declaration. This means that swrl:Variable declarations may appear anywhere in a Protégé OWL file, nested at any depth within a SWRL rule's definition, and that every usage of a swrl:Variable has two alternative syntactic forms, which complicates any processing. The following excerpt shows an OWL "just-in-time" declaration for `swrl:argument2`'s "U" variable, and a reference for `swrl:argument1`'s "H" variable.

```
<swrl:IndividualPropertyAtom>
  <swrl:argument2><swrl:Variable rdf:ID="U"/>
  </swrl:argument2>
  <swrl:propertyPredicate
rdf:resource="#hasLearner"/>
  <swrl:argument1 rdf:resource="#H"/>
</swrl:IndividualPropertyAtom>
```

We convert swrl:Variable atoms in two steps. (1) When converting the `rdf:RDF` root node (which is always the first node in the file, by definition), we "look ahead" and extract all swrl:Variable atoms, at any depth in the file, and copy them as immediate children of the root node. (2) All "just-in-time" declarations are rewritten by copying the child node's rdf:ID attribute as an rdf:resource reference.

- **Filter out OWL ontology atoms.** All class, property, and individual atoms that pertain to the OWL ontology are not used by SWRLRDF. These are filtered out by simply not copying them.

  In a sense, the conversion from OWL to SWRLRDF "loses" all information about the OWL ontology. The OWL ontology and individuals are assumed to be converted separately, as described in Section 0. It is the user's burden to ensure that the results of these two separate translations remain consistent with each other.

4.2 Conversion from SWRLRDF to CLIPS

The SweetJess component of SweetRules includes a translation path from SWRLRDF to CLIPS. This conversion is achieved by executing the SweetRules command:

translate swrlrdf clips swrlrdf-input-path clips-output-path

As Jess uses CLIPS syntax, the result is also valid as a Jess file. We distinguish the "ontology-only" Jess file produced in Section 0, from the "rules-only" Jess file produced here, by assigning them distinct extensions ".jess" and ".clp", respectively.

By default, a SWRL rule represents new conclusions only, i.e. new facts that are "made true" by adding them to the knowledge base. This is typically implemented by converting every SWRL rule head to a Jess `assert` statement. Consider a simple SWRL rule, written in the Protégé SWRLTab:

Student(?S) → Person(?S) "A student is a person"

SweetRules conversion from SWRLRDF to CLIPS produces the following Jess rule. (Here and hereafter, Jess rules and facts are shown in a terse format, with namespace prefixes omitted, for clarity.)

(defrule rule-1 (triple type ?S **Student**) => (assert (triple type ?S **Person**)))

This is a valid Jess rule, which is triggered by existing facts about `Students`, and responds by `asserting` new facts about `Persons`.

4.3 Extending SWRL Via Rule Transformations in Jess

SWRL provides a restricted expressiveness to ensure decidability. Jess rules are often more concisely expressed using constructs not available in SWRL. We have devised a generic mechanism to extend SWRL rules with additional constructs based on syntactic rule transformations in Jess. Syntactically, a SWRL rule must be a flat list of atoms, but it does guarantee that sequence is preserved. We find that judicious insertion of new, reserved class and property atoms is sufficient to add new keywords and even block structure. We define the following extension keywords:

SWRLx Atom	Where	Why	Effect
__name(*str*)	body	SweetRules workaround	Sets the Jess rule name to *str*. (SweetRules ignores Protégé's encoding of a SWRL rule's name, and generates default names.)
__naf(?)	body	Expressive	Converts to a Jess (**not** …) block.
__all(?A) __end(?A)	body	Expressive	Expands to a "guarded not" block 7, which handles the implicit iteration for an "if all" test.
__bind(?R)	body	Jess	Expands to a Jess pattern binding.
__modify(?R)	head	Jess	Changes the immediately following clause from an (assert …) to (modify ?R …).
__call(?)	head	Effecting	Changes the immediately following clause from an (assert (P …)) to a Jess function call (P …).

Our extended keywords are edited normally using the SWRL rule editor, and are converted verbatim by SweetJess. We treat the resulting CLIPS file as an intermediate quasi-rule format, convert each quasi-rule to a nested-Vector format that supports text processing, and apply the above rule transformations as a set of functions written in Jess, thereby achieving our extended semantics. The transformed rules are then evaluated in Jess, which updates Jess's working memory.

5 Implementation Issues for Automatic Conversion

We now discuss some pragmatic issues in implementing the conversion capability.

5.1 SweetRules Installation

SweetRules's considerable power is offset by its intimidating installation requirements. It depends on many third-party open-source software components, all of which must be installed concurrently. We note the following installation pitfalls, which could deter a typical user.

- **Version dependencies.** Some parts of SweetRules suffer from hard-coded dependencies to specific versions of other components. In our experience, some of these components can be upgraded to the latest versions without harm[1]. Other components *must* use the stated versions, else SweetRules quickly fails and throws many Java exceptions. More recent versions of these components have typically renamed some internal Java .jar files, which breaks SweetRules' hard-coded dependencies.

The following table summarizes SweetRules' software components, their stated version requirements, and the allowed and forbidden upgrades, based on our empirical testing. All version requirements are as stated by SweetRules 2.1's own installation program, which is the latest version of SweetRules.

Table 1. SweetRules software components, and allowed version upgrades

Source/ Vendor	Component	Stated Version	Can Upgrade To
---	SweetRules	2.1	---
Sun	Java SDK	1.4.2	5.0 Update 5, 6
Sandia	Jess	6.1p7	6.1p8
IBM	CommonRules	3.3	---
Sun	Java Web Services Development Pack (JWSDP)	1.5	**Must use 1.5**
SourceForge	dom4j	1.5	1.6.1
Apache	log4j	1.2.8	1.2.12
SourceForge	Junit	3.8.1	---
Apache	Xalan	2.6.0	**Must use 2.6.0**
Declarativa	InterProlog	2.1.1	**Must use 2.1.1**
SourceForge	XSB Prolog	2.6	2.6-fixed
SourceForge	KAON DLP	0.5	---
SourceForge	Saxon	8.1.1	**Must use 8.1.1**
HP	Jena	2.2	**Must use 2.2**

- **Installer error.** The final step of the SweetRules 2.1 installer should produce a Windows command script, "runsr.cmd", which is the most convenient way to launch SweetRules, as it properly fills in the extremely long[2] Java classpath. But it fails due to an unfortunate bug. The installer's final input screen prompts the user to enter the SweetRules installation directory path, which it uses to compose a Java command line that will produce the script. However, it assumes that the user included double-quotes around the SweetRules path, and blindly strips the first and last characters (without checking to see whether they actually are double-quotes!). This results in an invalid directory path, which causes the Java command to fail.

  The workaround is straightforward: Edit the installation log file, copy the final Java command line, manually fix all occurrences of the SweetRules directory path, and manually execute it from a Windows command prompt.

---

[1] More precisely, we can say that these upgrades have *not yet* caused any observed problems for the OWL-to-Jess and SWRLRDF-to-CLIPS translation paths.

[2] For the author's runsr.cmd file, the classpath argument is 4,006 characters long.

5.2 Managing SweetRules as a Child Process

SweetRules presents a simple command-line interface. We run SweetRules as a child process under Jess, using the standard Java ProcessBuilder class, and spawn worker threads to read SweetRules' output buffers asynchronously as a work-around for its tendency to block during translations. This reduces the entire SweetRules conversion step to a single function call.

6 Example Scenario

We demonstrate the teaching strategy engine using the following simple scenario.

- The learning contents includes a Problem1, which has a Solution1a.
- Among all learners, there is a learner named Alan.
- Alan has scored 58.0 on Problem1, which is recorded in a HistoryDatum h1.

We define the base ontology separately, then use OWL import statements to ensure that the "individuals-only" and "rules-only" ontology files share it. The base ontology is shown in Figure 1.

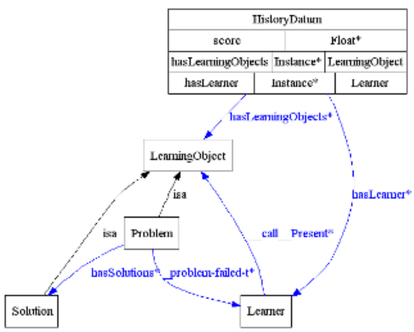


Fig. 1. Ontology classes and properties for scenario 1

The "individuals-only" ontology file adds individual definitions corresponding to the scenario data described above. This file is converted via the OWL-to-Jess translation path to a set of Jess statements, including the following fact:

```
(assert (triple score h1 58.0))                    (a)
```

This scenario uses one teaching macro-strategy: "If a learner fails a problem, show a solution immediately" 8, where a problem is considered to be failed if its score is less than 75%. This strategy is decomposed into three SWRL rules, organized in a bottom-up manner:

r-score-low	"A score less than 75% is a *low score*."
r-problem-failed	"A learner fails a problem if the learner has a *low score*."
s-strategy1	"If a learner fails a problem, show a solution immediately."

Editing of these rules in Protégé SWRL is shown in Figure 2.

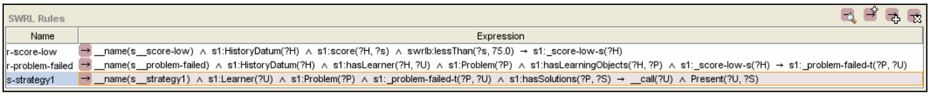


Fig. 2. Protégé SWRL rules for scenario 1

Having edited the rules, the teacher converts the rules to Jess by opening the "rules-only" .owl file. This automatically invokes XSLT, SweetRules, and rule transformations in Jess to achieve the conversion, detects which rules have changed, and updates only those rules in Jess's working memory, as shown in Figure 3.

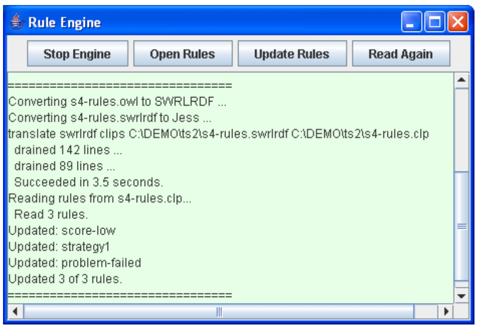


Fig. 3. Rule engine interface for automatic conversion from Protégé SWRL to Jess

With the scenario facts and rules loaded, the running Jess rule engine produces the following results:

- Fact (a), with h1's score of 58.0, satisfies the bottommost rule **r-score-low**. This rule fires, and asserts a new fact:

  (triple type h1 _score-low-s) (b)

- Fact (b) satisfies the intermediate rule **r-problem-failed**, which fires and asserts a new fact

  (triple _problem-failed-t Problem2 Alan) (c)

- Fact (c) satisfies the topmost rule **s-strategy1**, which fires and executes the Jess function call

  `(Present Alan Solution2b)`

We may assume that the `Present` function displays the solution in some manner that is integrated with the system's visual interface.

7 Summary

We have developed a teaching strategy engine in Jess, based on an automatic conversion mechanism from OWL individuals and SWRL rules in Protégé to Jess facts and rules. Our conversion mechanism leverages existing tools for rule language conversions, and extends their applicability to include Protégé's OWL format. More broadly, we have established a framework for representing teaching strategy knowledge as rules in a standard ontology editor. This supports a pedagogical development environment where a teacher can incrementally edit rules in the ontology, and quickly reload them into the teaching strategy engine for testing. We have also demonstrated a syntax-based extension to SWRL that supports more flexible and expressive rules, which supports the development of practical rules that can encode interactions with a human learner.

Acknowledgements

This research was supported by the Korean Ministry of Science & Technology through the Creative Research Initiative Program.

References

1. Friedmann-Hill, E. (2003). Jess in Action. Manning, Greenwich.
2. Grosof, B. N. (1997a). Building Commercial Agents: An IBM Research Perspective. IBM Research Report RC20835.
3. Grosof, B. N., Gandhe, M. D., and Finin, T. W. (2003). SweetJess: Inferencing in Situated Courteous RuleML via Translation to and from Jess Rules, *Unpublished working paper*, http://ebusiness.mit.edu/bgrosof/#sweetjess-basic.
4. Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., and Dean, M. (2004). SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission, http://www.w3.org/Submission/SWRL/.
5. Protégé, http://protege.stanford.edu/.
6. SweetRules, http://sweetrules.projects.semwebcentral.org/.
7. Wang, E., Kim, S. A., and Kim, Y. S. (2004). A Rule Editing Tool with Support for Non-Programmers in an Ontology-Based Intelligent Tutoring System, Workshop on Semantic Web for E-Learning (SW-EL), 3rd Int'l. Semantic Web Conf. (ISWC), Hiroshima, Japan.
8. Wang, E., and Kim, Y. S. (2006). Teaching Strategies Using SWRL. Submitted for publication in *Journal of Intelligent Information Systems*. Revised version of: Wang, E., Kashani, L., and Kim, Y. S. (2005), Teaching Strategies Ontology Using SWRL Rules, Int'l. Conf. on Computers in Education (ICCE), Singapore.

The Cognitive Tutor Authoring Tools (CTAT): Preliminary Evaluation of Efficiency Gains

Vincent Aleven, Bruce M. McLaren, Jonathan Sewall, and Kenneth R. Koedinger

Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
aleven@cs.cmu.edu, bmclaren@cs.cmu.edu, sewall@cs.cmu.edu, koedinger@cmu.edu

Abstract. Intelligent Tutoring Systems have been shown to be effective in a number of domains, but they remain hard to build, with estimates of 200-300 hours of development per hour of instruction. Two goals of the Cognitive Tutor Authoring Tools (CTAT) project are to (a) make tutor development more efficient for both programmers and non-programmers and (b) produce scientific evidence indicating which tool features lead to improved efficiency. CTAT supports development of two types of tutors, Cognitive Tutors and Example-Tracing Tutors, which represent different trade-offs in terms of ease of authoring and generality. In preliminary small-scale controlled experiments involving basic Cognitive Tutor development tasks, we found efficiency gains due to CTAT of 1.4 to 2 times faster. We expect that continued development of CTAT, informed by repeated evaluations involving increasingly complex authoring tasks, will lead to further efficiency gains.

1 Introduction

Intelligent Tutoring Systems can be very effective in improving student learning (e.g., [8, 18]). However, few ITSs are used regularly in real educational settings. E-Learning courses are created by the hundreds, but ITSs are seldom, if ever, seen as embedded components. A prime reason is that ITSs are typically hard to author. Estimates of development time have varied from 200-300 hours of authoring for one hour of instruction [4, 13, 19]. One way to make ITSs more widespread is to create authoring tools that speed up tutor development. A wide range of authoring tools have been built [1, 6, 13, 14, 17], and some of these have been used to build successful real-world systems [16]. Others have seen extensive evaluations focused on better understanding the authoring process and desired tool properties [1].

We report on an on-going project to create a set of authoring tools that supports the development of two types of tutors: Cognitive Tutors, which rely on a rule-based cognitive model and have been successful in improving students' math proficiency in American high schools [8], and Example-Tracing Tutors, a relatively novel type of tutors that provide the same core tutoring functionality as Cognitive Tutors but do not require any programming [6]. (Previously, these tutors were called Pseudo Tutors.) CTAT aims to increase the efficiency of authoring by means of an example-based

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 61 – 70, 2006.

approach to authoring. In this approach, an author demonstrates both correct and incorrect problem-solving behavior, which is recorded by the tool. The author then either generalizes and annotates the recorded examples, so they can serve as the basis for Example-Tracing Tutors, or uses them to guide the development and testing of a cognitive model for use in a Cognitive Tutor. CTAT aims to support a broad range of users: ITS/Ed Tech researchers, researchers in the field of the learning sciences interested in using ITSs as a vehicle for learning science experiments, on-line course developers, and computer-savvy college professors.

In evaluating the efficiency gains afforded by new tools, development time estimates derived from real-world tutor projects, such as those mentioned above, are helpful but are potentially subject to wide variability in terms of the experience of the developers, the subject matter for which tutors were built, and the scale of the project. Therefore, it is important that such estimates are supplemented with results from rigorous experiments that provide insight into the tool features most conducive to efficient authoring. We conducted such an experiment: a preliminary, small-scale ablation study in which we compared the authoring efficiency with the full CTAT tool suite to a version that had CTAT's novel tools taken out. To the best of our knowledge, this kind of evaluation focused on authoring efficiency has not been reported before in the ITS literature.

In this paper, which is meant to be a companion paper to an earlier paper focused on Example-Tracing Tutors [6], we present an overview of the CTAT tools used for developing Cognitive Tutors, illustrate hypothesized advantages of these tools in terms of authoring efficiency, and present the results from the small scale experiment.

2 Overview of CTAT

Cognitive Tutors and Example-Tracing Tutors, the two types of tutors supported by CTAT, represent different trade-offs between ease of authoring on the one hand and generality and flexibility of the resulting tutors on the other. Cognitive Tutors are rooted in the ACT-R theory of cognition and learning [4]. They interpret student problem-solving behavior using a cognitive model that captures, in the form of production rules, the skills that the student is expected to learn [8]. The tutor applies an algorithm called "model tracing" to monitor a student involved in a problem: it compares the students' actions against those that are appropriate according to the model. Developing a cognitive model for a Cognitive Tutor is a time-consuming task that requires AI programming. The upside is that the model works across a range of problems, and has flexibility, since it allows the tutor to recognize multiple student solution strategies and deal with subtle dependencies among solution steps.

Example-Tracing Tutors provide key elements of Cognitive Tutor behavior but are created "by demonstration" rather than by programming [6]. That is, an author demonstrates to the system how students are expected to solve each assigned problem and what errors they are expected to make. Compared to Cognitive Tutors, more per-problem-authoring is needed as solutions need to be demonstrated and annotated for each problem separately. However, a key advantage is that no AI programming is needed. We have seen repeatedly during workshops that people new to CTAT learn to build their first Example-Tracing Tutor in less than an afternoon.

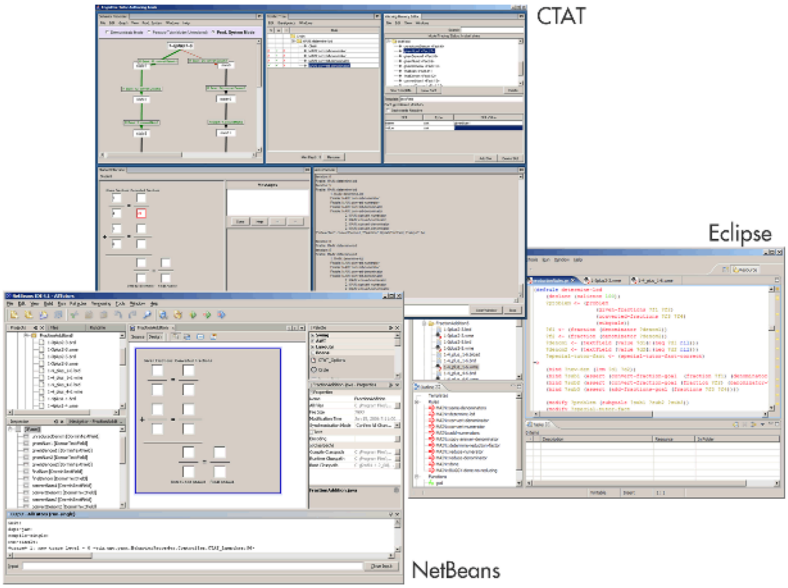


Fig. 1. The Cognitive Tutor Authoring Tools

The Cognitive Tutor Authoring Tools, depicted in Figure 1, comprise three separate applications: an external GUI Builder (typically, NetBeans or Macromedia's Flash), a set of core tools for demonstration-based task analysis and for testing and debugging cognitive models, and an external editor for cognitive models (typically Eclipse). The following tools are used prominently when authoring Cognitive Tutors:

GUI Builder – used to create a **Student Interface,** a problem-solving environment in which the student interacts with the tutor. The GUI Builder supports interface building without programming: the author arranges interface widgets on a canvas by drag and drop techniques (see Figure 1, bottom left). The GUI Builder is an external, off-the-shelf tool enriched with "tutorable" widgets developed for CTAT. We have used both Macromedia's Flash and Java development environments such as NetBeans and IntelliJ. In projects focused on providing tutoring within an existing simulator or problem-solving environment, the Student Interface is replaced with the external environment, which typically can be done without extensive effort [3, 12]. The use of a GUI Builder in an ITS authoring tool is not novel, but to the best of our knowledge, it is novel to have plug-and-play compatibility with standard GUI Builders.

Behavior Recorder – a central tool with three key functions. First, it records examples of correct and incorrect behavior demonstrated by the author, in the Student Interface, in the form of a Behavior Graph. Second, it implements the example-tracing function. Third, it provides support for planning and testing of cognitive models.

Working Memory Editor – used for cognitive model development; allows an author to inspect and modify the contents of the cognitive model's "working memory," which is frequently needed during model development. The Jess rule engine we use does not itself come with such an editor.

Conflict Tree and Why Not Window – tools for debugging the cognitive model, which provide information about rule activations and partial activations explored by the model-tracing algorithm. The Conflict Tree is specific to model tracing, but the Why Not Window is useful for general production rule programming. However, we know of no existing production rule system that offers a tool like this.

Jess Console – enables an author to interact directly with the Jess interpreter via the command line, which is helpful to carry out debugging strategies not directly supported by CTAT. It is similar to simple Jess tools such as JessWin.

External Editor–used to edit the Jess rules for the cognitive model. The Jess plug-in for Eclipse (shown on the right in Figure 1) provides syntax checking and auto-completion features. Other editors can be used, since no tight link exists with CTAT.

3 Hypothesized Advantages of Authoring with CTAT

Using fraction addition as an example domain, we illustrate four hypotheses about how CTAT facilitates cognitive model creation:

1. the Behavior Recorder supports the planning of a cognitive model;
2. CTAT auto-generates working memory content to facilitate modeling;
3. the Behavior Recorder facilitates testing, by providing automatic snap shots of all problem states and a facility for regression testing;
4. the Conflict Tree and Why Not Window facilitate error localization.

Prior to developing a cognitive model, the author creates a Student Interface suitable for solving fraction addition problems, using the GUI Builder shown at the bottom left of Figure 1. She also creates worked-out example problems with the Behavior Recorder. The examples will guide the model development efforts and serve as test cases. For example, our author may demonstrate two ways of solving the fraction addition problem 1/4 + 1/6: by converting the fractions to denominator 12, or by converting them to 24. Both are acceptable strategies that students are likely to employ. As the author demonstrates the steps, the Behavior Recorder records them in a "Behavior Graph," shown at the top left in Figure 1, with separate paths corresponding to each strategy. The author then labels each step in the recorded problem solutions with names for relevant skills. This activity is a form of cognitive task analysis, because the author determines how the overall problem-solving skill breaks down into smaller components. At the same time, it is a way of planning the cognitive model, since the author will later create production rules corresponding to each identified skill. This planning step could be done on paper, but doing it with CTAT has the advantage that the demonstrated examples are more likely to be complete and can later serve as semi-automated test cases for the cognitive model.

Having created a Student Interface and annotated examples, the next step for the author is to create a working memory representation for the cognitive model. In Jess, working memory is a collection of "facts" whose attributes (or "slots" in Jess terminology) must first be declared by means of "templates." CTAT helps by creating initial working memory content for the author. The structure generated by CTAT mirrors the Student Interface – it contains a fact for each element (i.e., widget) in the interface. This organization is useful in particular when the external representation of a problem (as captured in the interface) reflects its internal structure. Even if the

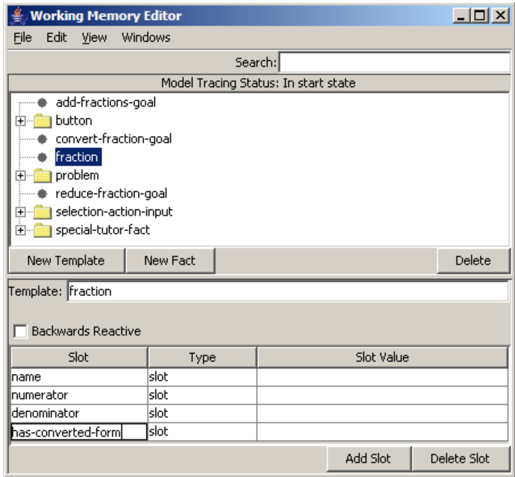


Fig. 2. Augmenting Working Memory using CTAT's Working Memory Editor

interface elements do not fully reflect the internal problem structure, facts corresponding to interface elements are still useful for model tracing. In the fraction addition example, the representation generated by CTAT contains one working memory fact per text field in the interface. This representation is a start, but the author must add the information that each text field represents a particular part of a particular fraction (numerator, denominator) and must also represent the role in the problem that each fraction plays (e.g., given fractions, converted fractions, sum fractions). In the process, she might create a new template to represent fractions, with slots for the numerator and denominator, using CTAT's Working Memory Editor (see Figure 2).

Next, the author needs to write the production rules for each skill in the problem. The actual editing of the production rules is done with a standard editor such as Eclipse. We plan to make the writing of production rules easier by means of structured editing techniques and automated rule stub generation – we implemented some of these facilities for the TDK production rule language [7] but not yet for Jess. We are also working on a machine learning approach to rule creation [9].

When it comes time to test the production rules, the Behavior Recorder is helpful in two ways. First, the Behavior Recorder essentially provides automated snapshots for all recorded problem states. That is, it can be used to move working memory and the Student Interface to any state recorded in the Behavior Graph, just by clicking on the state. This capability makes it easier to test rules involved in a particular step in the problem, since it saves the author from having to provide input in the Student Interface for all previous steps. This kind of manual input would be time-consuming, especially for more complex problems with many steps, and especially considering that it needs to be done for every edit-test-debug cycle. In addition, the Behavior Recorder supports semi-automated regression testing in which the cognitive model is tested against a full Behavior Graph as test case (i.e., as a specification of how the model should behave on the steps of the given problem). CTAT indicates by means of color coding whether the tutor (applying its model-tracing algorithm) produces the expected result for each link in the graph. If not, then typically the cognitive model is to blame; one or more rules are not yet working as intended. This kind of testing is especially useful when rules that have been authored for an earlier tutor problem need to be modified for a later problem; such modifications sometimes introduce errors in problems on which previously the rule worked correctly.

To help localize errors in a cognitive model, the Conflict Tree window shows the space of rule activations explored in the process of model tracing (shown on the left in Figure 3). When a student submits a problem-solving step to the tutor, the model-tracing algorithm searches for a sequence of rule activations that produce the same action as the student. Showing the search space graphically, as a tree of rule activations, helps an author fully understand the model's behavior, which is often useful for debugging or for getting to know models built by others. The Why Not window (shown on the right in Figure 3) provides further detail about each search node depicted in the Conflict Tree. It shows both full and partial rule activations that were generated at any given node. This information is useful particularly when a rule that was expected to fire did not – hence the name "Why Not". Experienced modelers typically like these tools. Without them, an author would have to use the Jess command-line interface to extract the information in the Conflict Tree in piecemeal fashion. It is not possible to extract the information that is presented in the Why Not window using the Jess command-line. An author would have to resort to inserting print statements in rule conditions or other cleverness to infer this information.

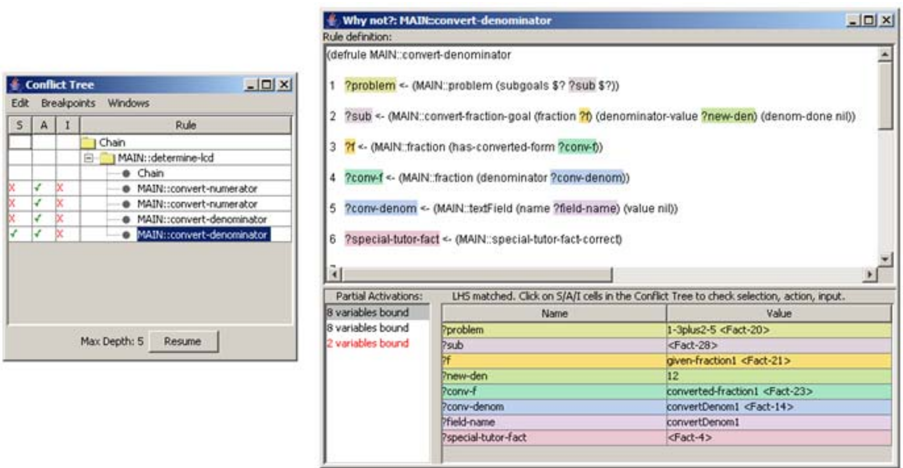


Fig. 3. The Conflict Tree and Why Not Window: debugging tools

4 Preliminary Evaluation of Efficiency Gains

We conducted a small-scale experiment to test if the novel tools in CTAT lead to more efficient tutor development. The experiment was an ablation study, in which we compared the full CTAT suite ("Full Tool Set") to a version in which the novel tools were removed, namely, the Behavior Recorder, the Conflict Tree, the Why Not window, and the Working Memory Editor. This "Reduced Tool Set" is essentially a standard Jess environment, akin for example to JessWin, but augmented with the model-tracing algorithm and a fully-integrated Student Interface. Thus, the control condition in this experiment begins at a point far ahead of an author implementing a model-tracing tutor from scratch. The goal of the experiment was to get an indication of CTAT's efficiency and to identify areas for improvement. In particular, we wanted to see whether the last three of the hypothesized advantages of CTAT would materialize. Due to the small number of subjects, the experiment does not allow for statistically significant results. Nonetheless, it is valuable as a formative evaluation.

The experiment focused on a simple, semi-realistic modeling activity, in which the participants were asked to create a cognitive model consisting of 6 rules, given detailed statements of the form of "If … then … " that mapped quite directly onto the rule conditions and actions to be implemented. Four subjects participated, all of whom were students at CMU who had used CTAT for a class project. The subjects thus had some experience but were not expert cognitive modelers. All subjects did the modeling task twice, two of them first with the Full Tool Set and then again, approximately a week later, with the Reduced Tool Set, the other two with the order reversed. Each task lasted 3.5 hours at most – less if the subject finished before that amount of time had elapsed. We measured the number of rules completed.

As shown in Figure 4, the decrease in time per rule was greater when the subjects switched from the Reduced Tool Set to the Full Tool Set than when they used the tool sets in the opposite order. Thus, the increase in efficiency between the first and second time the task was performed is not likely to be due solely to the subjects' greater familiarity with the task. There is an effect of the tools that suggests that the Full Tool Set improves the efficiency of modeling. Overall, the time per rule with the Full Tool Set was a factor of 1.4 faster than with the Reduced Tools. That effect is less than the 2x improvement that we reported previously for an experiment with a single participant [7]. That earlier experiment involved a similar ablation design as the current but involved CTAT tools that support modeling not in Jess, but in TDK, the modeling language used to develop the Algebra and Geometry Cognitive Tutors. In the earlier experiment, the difference between the Full and Reduced Tool Sets was greater, since the CTAT tools to support TDK had features that have not yet been implemented in the CTAT/Jess tools, which may explain the higher efficiency gain.

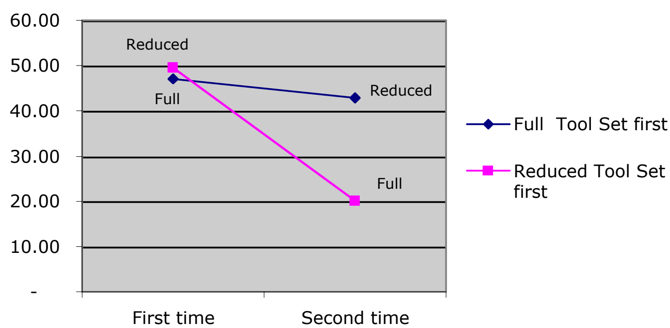


Fig. 4. Comparison of the time to author production rules with CTAT versus an ablated version of CTAT from which its novel tools were removed

In addition to evaluating the overall efficiency of rule creation, we undertook a detailed quantitative analysis of subjects' actions with the tools, recorded with Camtasia, to evaluate CTAT's hypothesized efficiency advantages. The experiment seemed to confirm that the auto-generation of working memory saves time, as measured by the amount of time spent editing the working memory content (7 mins on average with the Full Set, 14 mins with the Reduced Set). The time savings were very modest, but it is worth pointing out that the measure used does not include any savings in development time that may have resulted from starting with a solid working memory representation. In other words, the auto-generation may provide useful scaffolding for inexperienced tool users that is hard to quantify – and this may indeed be its main advantage. The results were more surprising with respect to the hypothesized time savings due to the Behavior Recorder's automated snap shot facility. Even though all subjects knew about this facility, three of the four hardly used it, with one subject accounting for almost 60% of its use. The limited use may reflect the simplicity of the modeling task that the subjects were given (a six-step problem), or it may be that when testing a cognitive model, it is more natural to work in the Student Interface than in the Behavior Graph. In the Student Interface the details of the current problem state are always clearly visible, whereas in the Behavior Graph only non-descriptive names for the states are shown, without details, which may hamper the intended nimble navigation among problem states. If the author's attention is naturally anchored in the Student Interface, then it makes sense that the commands for navigating the Behavior Graph should be issued from this window, for example, by means of arrow keys or "bookmarks". This solution would, we expect, retain any efficiency advantages due to the snapshots. Finally, there was evidence that the debugging tools (the Conflict Tree and the Why Not Window) were useful. The subjects used the Why Not window regularly (31.25 times on average). We do not know how often they used the Conflict Tree, since the tool is usually visible without requiring interaction by the author. Importantly, there was evidence of a higher number of edit-test-debug cycles with the Reduced Tool Set. While the time spent editing rules was about the same in each condition, the number of editing episodes was higher (81 v. 59 on average) with the Reduced Tool Set. Further, the subjects using the Reduced Tool Set spent more time testing, as distinct from debugging (26 mins v. 16 mins), and had many more testing actions (164 v. 93). While we cannot attribute these numbers solely to any greater diagnostic power of the CTAT debugging tools, they are certainly consistent with the notion that with better debugging tools one needs fewer edit-test-debug cycles.

In spite of the modest scale of the experiment, analyses such as those presented above are very useful in guiding future tool redesign and development efforts. They underscore the importance of getting the HCI right in designing interactive tools and lead to specific suggestions for improvement. Further qualitative analysis of the errors in the subjects' production rules will also help in that regard. In interpreting the efficiency results, it is important to keep in mind that the modeling tasks in these experiments were simple. The task involved only 6 straightforward rules of which detailed English versions were given, and thus was significantly less complex than a typical real-world modeling task. Viewed in that light, a 1.4-2x gain in efficiency is encouraging, even if our eventual goal is to achieve higher gains.

5 Conclusion

In CTAT, the authoring of Example-Tracing Tutors and Cognitive Tutors is organized around examples of demonstrated behavior. These examples include alternative strategies for solving problems and errors students are expected to make, and can be recorded conveniently with CTAT's Behavior Recorder tool. They can be used as the basis for Example-Tracing Tutors to provide guidance to students. The examples can also be used as planning cases and semi-automatic test cases for cognitive models, if an author is developing a Cognitive Tutor.

The preliminary experiment described in this paper suggests that authoring with CTAT is becoming more rapid. Holding ourselves to a high scientific standard of rigorous laboratory experimentation with a high-bar control condition (all but the newest features), we have shown a modest efficiency improvement estimate of 1.4 to 2 times faster, compared to standard tools for model tracing. We are aiming for higher overall efficiency gains, but it is nonetheless encouraging that a speed-up was attained on a small and easy task. An interesting finding was that CTAT seems to lower the number of edit-test-debug cycles needed to create a cognitive model. The preliminary experiment led to a number of ideas for tool improvement, focused on improving the HCI of the tools. To improve the efficiency of the tools, we are also developing semi-automated techniques, such as machine learning [9] and bootstrapping [10]. We expect that the advantages of CTAT will be more pronounced in a more complex modeling task and as we continue to improve CTAT.

So far, CTAT has been used by over 220 users in a number of workshops, graduate courses, summer schools, and tutorials. We estimate that 30-40% of these users were non-CMU people. CTAT is being used to develop a set of tutors for introductory college-level genetics (http://www.cs.cmu.edu/~genetics), which have been piloted in various colleges across the country, and has been used in learning science experiments in the domains of thermodynamics [3], stoichiometry [11], French culture [15], and Chinese character recognition. Clearly, these numbers indicate that there is a need for authoring tools and that CTAT is offering useful functionality that can be applied in a range of domains. CTAT is available free of charge for research and educational purposes (http://ctat.pact.cs.cmu.edu). It is our hope that through tool development and other efforts by the ITS community, intelligent tutoring systems will become more widespread and will one day be staples of on-line courses.

Acknowledgments

Brett Leber, Grace Lee Leonard, and the anonymous reviewers helped make this a better paper. The research is sponsored by ONR Award N000140410107, by NSF Award SBE0354420 to the Pittsburgh Sciences of Learning Center, and by a grant from the Grable Foundation. Their contributions are gratefully acknowledged. The views expressed in the paper may not be the official views of the funding agencies.

References

1. Ainsworth, S.E & Fleming, P.F. (2005) Evaluating a mixed-initiative authoring environment: is redeem for real? In *Proceedings of the 12th International Conference on Artificial Intelligence in Education* (pp. 9-16). Amsterdam, IOS Press.

2. Aleven, V., & Koedinger, K. (2002). An effective metacognitive strategy: learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science, 26*, 147-179.
3. Aleven, V., & Rosé, C. (2005). Authoring plug-in tutor agents by demonstration: Rapid, rapid tutor development. In *Proceedings of the 12th International Conference on Artificial Intelligence in Education, AIED 2005* (pp. 735-737). Amsterdam: IOS Press. (Poster).
4. Anderson, J. R. (1993). *Rules of the Mind.* Hillsdale, NJ: Erlbaum.
5. Friedman-Hill, E. (2003). *Jess in Action*. Manning Publications Co.
6. Koedinger, K. R., Aleven, V., Heffernan, N., McLaren, B., & Hockenberry, M. (2004). Opening the door to non-programmers: authoring intelligent tutor behavior by demonstration. In *Proceedings of Seventh International Conference on Intelligent Tutoring Systems, ITS 2004* (pp. 162-174). Berlin: Springer Verlag.
7. Koedinger, K. R., Aleven, V. & Heffernan, N. T. (2003). Toward a rapid development environment for Cognitive Tutors. In *Proceedings of the 11th International Conference on Artificial Intelligence in Education, AI-ED 2003* (pp. 455-457). Amsterdam: IOS Press.
8. Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education, 8,* 30-43.
9. Matsuda, N., Cohen, W. W., & Koedinger, K. R. (2005). Applying programming by demonstration in an intelligent authoring tool for Cognitive Tutors. In AAAI Workshop on Human Comprehensible Machine Learning (pp. 1-8). Menlo Park, CA: AAAI Association.
10. McLaren, B. M., Bollen, L., Walker, E., Harrer, A., & Sewall, J. (2005). Cognitive Tutoring of collaboration: developmental and empirical steps toward realization, *In Proceedings of the Conference on Computer Supported Collaborative Learning (CSCL-05),* Taipei, Taiwan.
11. McLaren, B. M., Lim, S., Gagnon, F., Yaron, D., & Koedinger, K. R. (in press). Studying the effects of personalized language and worked examples in the context of a web-based intelligent tutor. 8th International Conference on Intelligent Tutoring Systems, ITS 2006.
12. Murray, T. (1999). Authoring intelligent tutoring systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education, 10,* 98-129.
13. Murray, T., Blessing, S., & Ainsworth S.E. (Eds.) (2003). *Tools for Advanced Technology Learning Environments.* Amsterdam: Kluwer Academic Publishers.
14. Nuzzo-Jones, G., Walonoski, J.A., Heffernan, N.T., Livak, T. (2005). The eXtensible Tutor Architecture: A new foundation for ITS. In *Proceedings of the 12th Artificial Intelligence In Education* (pp. 902-904). Amsterdam: IOS Press.
15. Ogan, A., Aleven, V., & Jones, C. (2005). Improving intercultural competence by predicting in French film. In *Proceedings of E-Learn 2005*. Norfolk, VA: AACE.
16. Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., et al. (2005). The Assistment Project: Blending Assessment and Assisting. In *Proceedings of the 12th Artificial Intelligence In Education* (pp. 555-562). Amsterdam: IOS Press.
17. Suraweera, P., Mitrovic, A. & Martin, B. (2005). A Knowledge Acquisition System for Constraint-based Intelligent Tutoring Systems. In *Proceedings of the 12th international conference on Artificial Intelligence in Education,* (pp. 638-645). Amsterdam, IOS Press.
18. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., et al. (2005). The Andes physics tutoring system: five years of evaluations. In *Proceedings of the 12th international conference on Artificial Intelligence in Education.* Amsterdam, IOS Press.
19. Woolf, B. P., & Cunningham, P. (1987). Building a community memory for intelligent tutoring systems. AAAI 1987 (pp. 82-89).

A Bayesian Network Approach for Modeling the Influence of Contextual Variables on Scientific Problem Solving

Ronald H. Stevens[1] and Vandana Thadani[2]

[1] UCLA IMMEX Project, 5601 W. Slauson Ave. #255, Culver City, CA 90230
immex_ron@hotmail.com
[2] Loyola Marymount University, Dept of Psychology, Los Angeles, CA 90045
vthadani@lmu.edu

Abstract. A challenge for intelligent tutoring is to develop methodologies for transforming streams of performance data into insights and models about underlying learning mechanisms. Such modeling at different points in time could provide evidence of a student's changing understanding of a task, and given sufficient detail, could extend our understanding of how gender, prior achievement, classroom practices and other student/contextual characteristics differentially influence performance and participation in complex problem-solving environments. If the models had predictive properties, they could also provide a framework for directing feedback to improve learning.

In this paper we describe the causal relationships between students' problem-solving effectiveness (i.e. reaching a correct solution) and strategy (i.e. approach) and multiple contextual variables including experience, gender, classroom environment, and task difficulty. Performances of the IMMEX problem set *Hazmat* (n ~ 33,000) were first modeled by Item Response Theory analysis to provide a measure of effectiveness and then by self-organizing artificial neural networks and hidden Markov modeling to provide measures of strategic efficiency. Correlation findings were then used to link the variables into a Bayesian network representation. Sensitivity analysis indicated that whether a problem was solved or not was most likely influenced by findings related to the problem under investigation and the classroom environment while strategic approaches were most influenced by the actions taken, the classroom environment and the number of problems previously performed. Subsequent testing with unknown performances indicated that the strategic approaches were most easily predicted (17% error rate), whereas whether the problem was solved was more difficult (32% error rate).

1 Introduction

Strategic problem solving is a complex process with skill development being influenced by the task, the experience and knowledge of the student, the balance of cognitive and metacognitive skills possessed by the student and required by the task, gender [7], ethnicity, classroom environment [23] and overall ability constructs such as motivation and self efficacy [18]. The variable contributions of these influences helps account for why it is so challenging for teachers to identify which students are using the knowledge

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 71 – 84, 2006.

and critical thinking skills presented in class to solve real-world problems, and distinguish them from other students that may require interventional supports [16]. These analyses are further complicated as the acquisition of problem solving skills is a dynamic and often gradual process characterized by transitional changes over time as experience is gained and learning occurs [13]. Given the nature of novice learning, student trajectories are likely to be complex with regard to the heterogeneity of strategies, the pace of learning, and the level of expertise obtained. [6].

To address these challenges we have been developing probabilistic models of learning trajectories that can begin to position students' scientific problem-solving skills upon a continuum of experience. These models provide estimates of student ability (Item Response Theory (IRT) analysis), describe the strategy used on any particular problem solving session (Artificial Neural Networks (ANN)) and define trajectories of progress as multiple problems are performed (Hidden Markov Modeling (HMM)) [31].

One consistent finding of our studies across the domains of chemistry, molecular genetics, genetics, medicine and K-12 science is that after a period of practice students stabilize with a level problem solving competency characterized by particular approaches [27] [30] [31] [34] [40]. Furthermore, once such stabilization has occurred, many students will use these approaches when presented with similar problems up to 3 months later. Unfortunately, not all students will stabilize with efficient and/or effective approaches indicating that experience alone is not sufficient for some students to make progress, a finding reported by others [17]. The challenge therefore, is to rapidly identify students who are unlikely to make progress on their own and then begin to target deliberate practice [6], teacher guidance, and/or interventions such as pedagogical feedback or collaborative group learning [42] to improve the level of competency.

To enable predictive modeling it will be important to better understand how the diverse set of individual and contextual variables associated with complex problem solving differentially contribute to the adoption and persistence of strategies. In this paper we describe the construction and preliminary validation of descriptive Bayesian networks that can serve both as an analytic workbench to better understand the interactions among these variables, as well as an engine for developing support decisions for future problem solving and learning activities.

2 Methods

IMMEX (Interactive Multi-Media Exercises) is an online problem solving environment and layered analytic system that delivers problem solving tasks that require students to analyze descriptive scenarios, judge what information is relevant, plan a search strategy, gather information, and eventually reach a decision(s) that demonstrates understanding [35]. While IMMEX problem solving supports the three cognitive components described by [36] as important for problem solving (e.g. understanding of concepts, understanding the principles that link concepts, and linking of concepts and principles to procedures for application), evaluation studies suggest that the second and third components are emphasized by the IMMEX format.

Since online delivery of these cases began 5 years ago, over 500,000 problems have been performed by students spanning middle school to medical school. One, of several problem sets researched extensively is *Hazmat*, which provides evidence of students' ability to conduct qualitative chemical analyses [31]. A multimedia presentation is shown to the students, explaining that an earthquake caused a chemical spill in the stockroom and their task is to identify the unknown chemical by gathering information using a 22 item menu containing a Library of terms, a Stockroom Inventory, and different Physical or Chemical Tests (e.g. a precipitate test as shown in Figure 1). This problem set contains 38 cases that can be performed in class, assigned as homework, or used as quizzes.

To follow students' performance and progress we have developed analytic models of how strategies are constructed, modified and retained as students learn to solve problems like *Hazmat* [31].

2.1 Model 1. Item Response Theory (IRT) Estimates of Student Ability

The 38 Hazmat cases include a variety of acids, bases, and compounds giving either a positive or negative result when flame tested. As expected, the flame test negative compounds are more difficult for students because both the anion and cation have to be identified by running additional chemical tests. As students perform multiple cases, refined estimates of their ability can be obtained by IRT analysis by relating characteristics of items and individuals to the probability of solving a given case [15]. Overall, the problem set presents an appropriate range of difficulties to provide reliable estimates of student ability [34]. In the subsequent BN models we refer to these values as **IRT**.

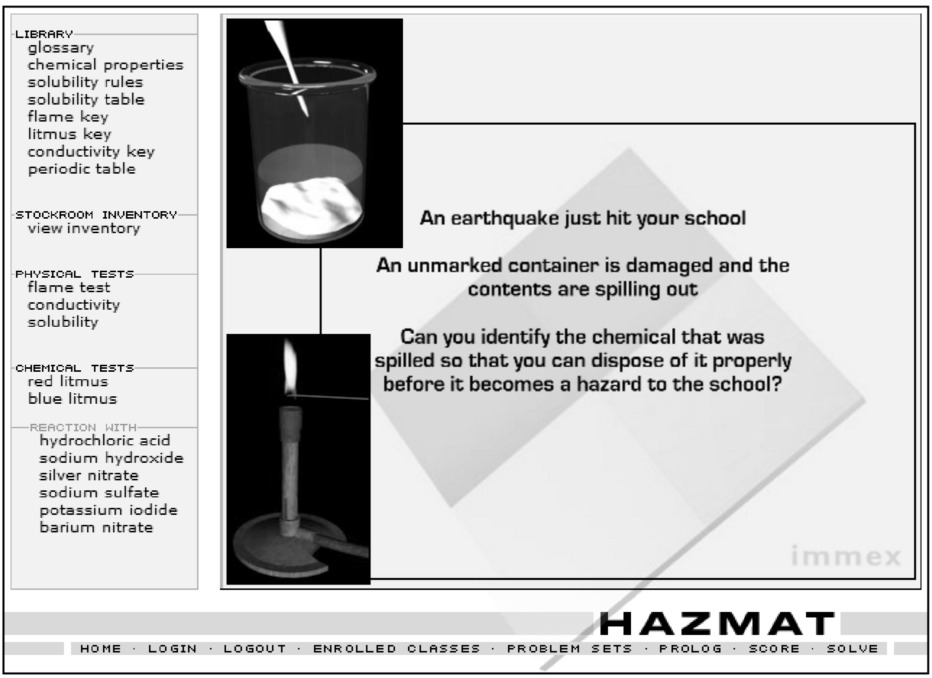


Fig. 1. ***Hazmat*.** This composite screen shot of *Hazmat* illustrates the challenge to the student and shows the menu items on the left side of the screen. Also shown are two of the test items available, a precipitation reaction and the result of flame testing the unknown.

2.2 Model 2. Artificial Neural Network (ANN) Classification of Strategies

While useful for ranking the students by the effectiveness of their problem solving, IRT does not provide strategic measures of this problem solving. Here, we use ANN analysis. As students navigate the problem spaces, the IMMEX database collects timestamps of each student selection. The most common student approaches (i.e. strategies) for solving *Hazmat* are identified with competitive, self-organizing artificial neural networks [12] [33] [30] using these time stamped actions as the input data. The result is a topological ordering of the neural network nodes according to the structure of the data where geometric distance becomes a metaphor for strategic similarity. Often we use a 36-node neural network and the details are visualized by histograms showing the frequency of items selected for student performances classified at each node (Figure 2 A). **Strategies** so defined consist of actions that are always selected for performances at that node (i.e. with a frequency of 1) as well as ones ordered variably.

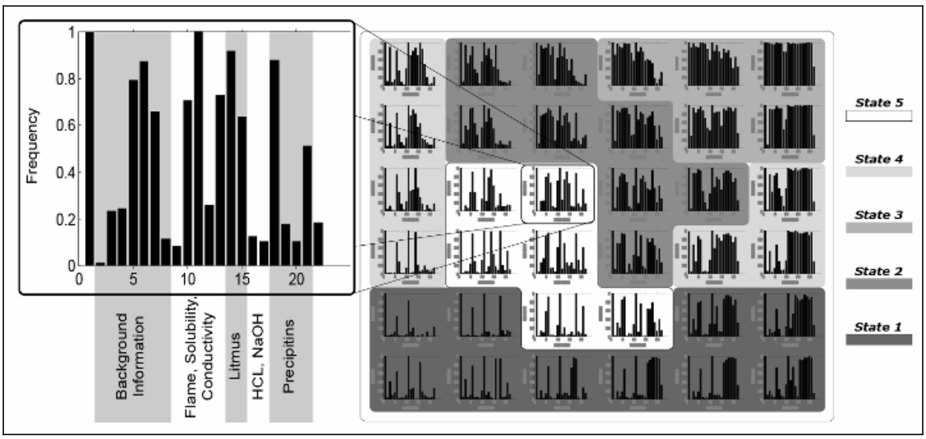


Fig. 2. Sample Neural Network Nodal Analysis. a.) The selection frequency of each action (identified by the labels) is plotted for the performances at node 15, and helps characterize the performances clustered at this node and for relating them to performances at neighboring nodes. The nodes are numbered in rows, 1-6, 7-12, etc. **b.)** This figure shows the item selection frequencies for all 36 nodes.

Figure 2B is a composite ANN nodal map that shows the topology of performances generated during the self-organizing training process. Each of the 36 matrix graphs represents one ANN node where similar student's problem solving performances have become competitively clustered, and as the neural network was trained with vectors representing student actions, it is not surprising that a topology developed based on the quantity of items. For instance, the upper right of the map (nodes 6, 12) represents strategies where a large number of tests were ordered, whereas the lower left contains strategies where few tests were ordered.

2.3 Model 3. Hidden Markov Model (HMM) Strategic Progress Models

On their own, artificial neural network analyses provide point-in-time snapshots of students' problem solving. More complete models of student learning should also account for the changes of student's strategies with practice. Here we postulate that students will pass through a number (3-5) of **States** as they shift their problem solving strategies over time. In these models students perform multiple cases in the 38-case Hazmat problem set, and each performance is classified with the trained ANN. Predictive models of student progress are then developed from sequences of these strategies with HMM [24] [21]. This results in a Transition Matrix, and an Observation Matrix representing the resulting model. This approach is shown in Figure 3 where students solved 6 Hazmat cases. One level (stacked bar charts) shows the distribution of the 5 HMM states across the 6 performances. On the first case, when students are framing the problem space, the two most frequent states are States 1 and 3. Moving up an analytical layer from HMM states to ANN nodal strategies (the 6 x 6 histogram matrices) shows that State 3 represents strategies where students ordered all tests, and State 1 where there was limited test selection. With experience the students transited from State 3 (and to some extent State 1), through State 2 and into States 4 and 5, the more effective states. By the fifth performance the State distributions stabilized after which time students without intervention tended not to switch their strategies, even when they were ineffective. Stabilization with ineffective strategies is of concern as students tend to retain their adopted strategies over at least a 3-months period [33].

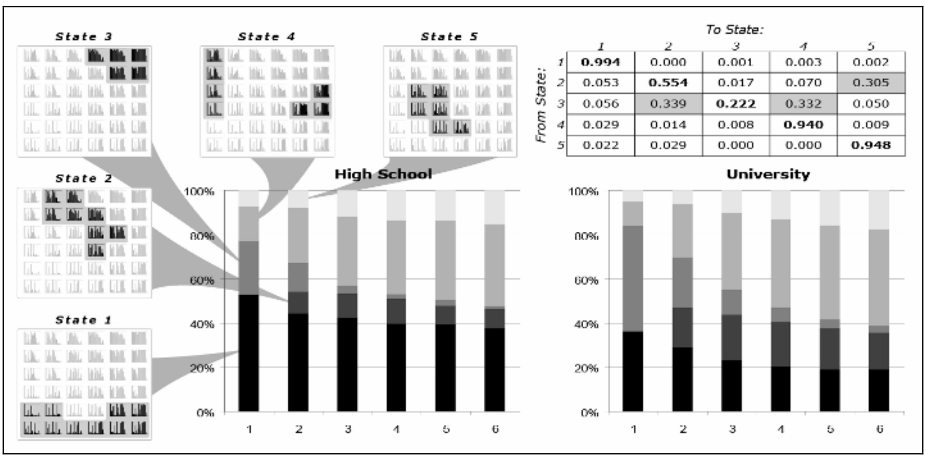


Fig. 3. Modeling Individual and Group Learning Trajectories. This figure illustrates the strategic changes as high school and university students gain experience in *Hazmat* problem solving. Each stacked bar shows the distribution of HMM states for the students (N=7290) after a series (1-6) of performances. These states are also mapped back to the 6 x 6 matrices which represent 36 different strategy groups identified by self organizing ANN. The highlighted boxes in each neural network map indicate which strategies are most frequently associated with each state. From the values showing high cyclic probabilities along the diagonal of the HMM transition matrix (upper right), States 1, 4, and 5 appear stable, suggesting once adopted, they are continually used. In contrast, students adopting State 2 and 3 strategies are more likely to adopt other strategies (gray boxes).

3 Results

Crosstabulation analyses of over 75,000 student performances across multiple domains have repeatedly shown significant associations among student and contextual variables that influence both the problem-solving performance (solve rate, IRT ability estimates) as well as the approaches (ANN and HMM classifications) students adopt [26] [30] [32]. These variables include gender, the number of prior cases performed, the experience of the student (regular high school, AP high school, university), teacher and classroom effects as well as the conditions under which problem solving is performed (individual vs. collaborative).

Using commercial Bayesian network (BN) software (Netica, Inc), we have captured these interactions into belief networks to better understand the dependencies of the different variables. A sample BN is shown in Figure 4 where the network was initialized with a model of student problem solving based on a dataset of >33,000 *Hazmat* performances.

Cross tabulation analysis has shown that dependencies exist among multiple nominal variables related to problem solving. The variables can be divided into two major categories: 1) dependent outcome measures that consist of whether or not the problem was solved (Solved/Not Solved in Figure 4), along with how the problems were solved (Strategy, State), and 2) contextual variables that include gender, the problem cases, student experience, the learning environment (individual vs. collaborative) and the class/teacher.

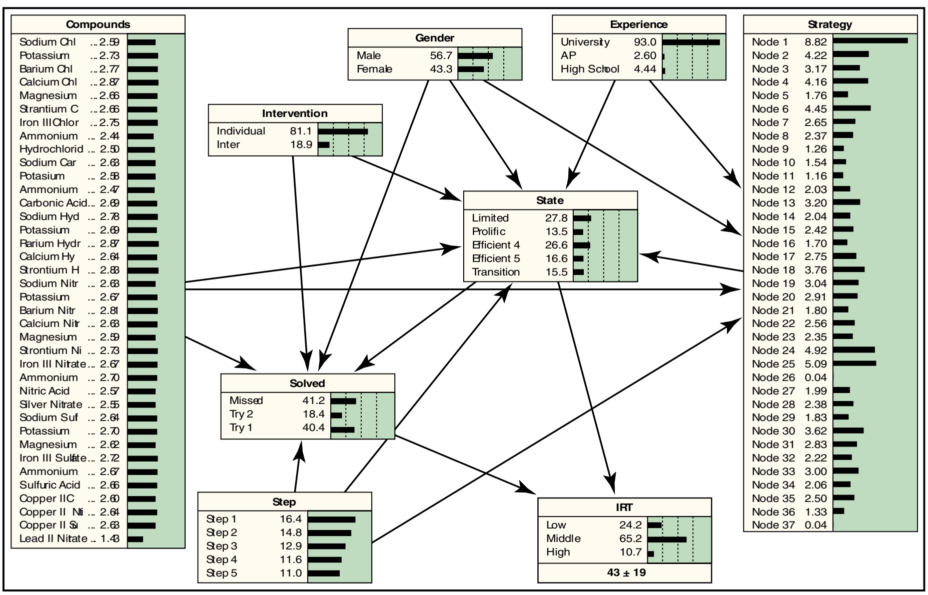


Fig. 4. A Sample *Hazmat* Belief Network. Each of the variables being investigated has been divided/discretized into categories, and the bar charts indicate the proportion of the sample in each category. This figure also provides a representation of the dataset composition. For instance, there are similar numbers of males and females, but there are more university students than high school students.

Starting from the left, **Compound** represents the unknown in the case that is being solved. **The Compound → Solved** dependency acknowledges the relative difficulty of the cases given the nature of the compounds (acids, bases, salts, flame test +/-). There is also a **Compound → Strategy** association (ANN node, 1-36 from Figure 2) as would be expected as flame test negative compounds require more extensive testing than flame test positive compounds. The **Compound → State** (HMM State) link reflects the correlation between certain HMM hidden states and different compounds. In this figure, the *limited* value equals State 1 in Figure 3, *prolific* = State 3, *transition* = State 2 and *efficient_4* and *efficient_5* = States 4 and 5 respectively. This association between Strategy and State is identified from the HMM emission matrix and confirmed by X^2 analysis.

The **Intervention** variable indicates whether the problem was solved by an individual or through an intervention, which here is placing the students in collaborative groups. We have previously shown that students working in groups stabilized their strategies more rapidly than did individuals, solved a greater proportion of the problems, and used different approaches [32]. These dependencies are reflected in the **Intervention → Solved** and **Intervention → State** links.

Previous studies have also shown that while the overall problem solution frequency (Solved) is similar across gender there are significant gender differences in the Strategies and States used during the problem solving process that account for the **Gender → Solved** and **Gender → State** links [26].

More educationally advanced students represented by the **Experience** node, solve problems more effectively (**Experience → Solved)** and efficiently (**Experience → State)** [31]. As shown in this figure, this dataset primarily contains university students. Nevertheless, given the size of the dataset, a limited comparison can be made between university and high school students (Figure 3).

It is also possible to include a **Classroom** identifier that allows a finer granularity of classroom practices to be included.

The Step variable in the lower left corner acknowledges the changes in Strategies (**Step → Strategies**), States (**Step → States**), and Solved (**Step → Solved)** as students perform a series of *Hazmat* problems. As the problems are randomly delivered to students there are no links to **Compounds**.

The final variable included is **IRT** which is the estimate of overall student ability modeled by Item Response Theory analysis after students have solved a series of problems of varying difficulty. As the input data for IRT analysis is whether or not a problem was solved, IRT is closely correlated with the **Solved** variable (**IRT → Solved)**. Students with different IRT abilities stabilize their strategies at different rates and with different proportions of the States [30].

As most of the students performed between 5 and 10 *Hazmat* problems, the dataset contains performance data on subsequent problems allowing the incorporation of nodes for the predicted performance state (**Prediction**), as well as predictions as to whether or not a subsequent problem will be solved (**PSolved**). Such analyses can become quite refined given the ability to isolate particular values of the different variables. For example, in the Figure 5 it can be seen that if high ability students using an *efficient* strategy are given a difficult case, Carbonic Acid as their second case, they are likely to miss the case, but are likely to solve (**PSolved**) the next case with a good strategy (**Prediction**).

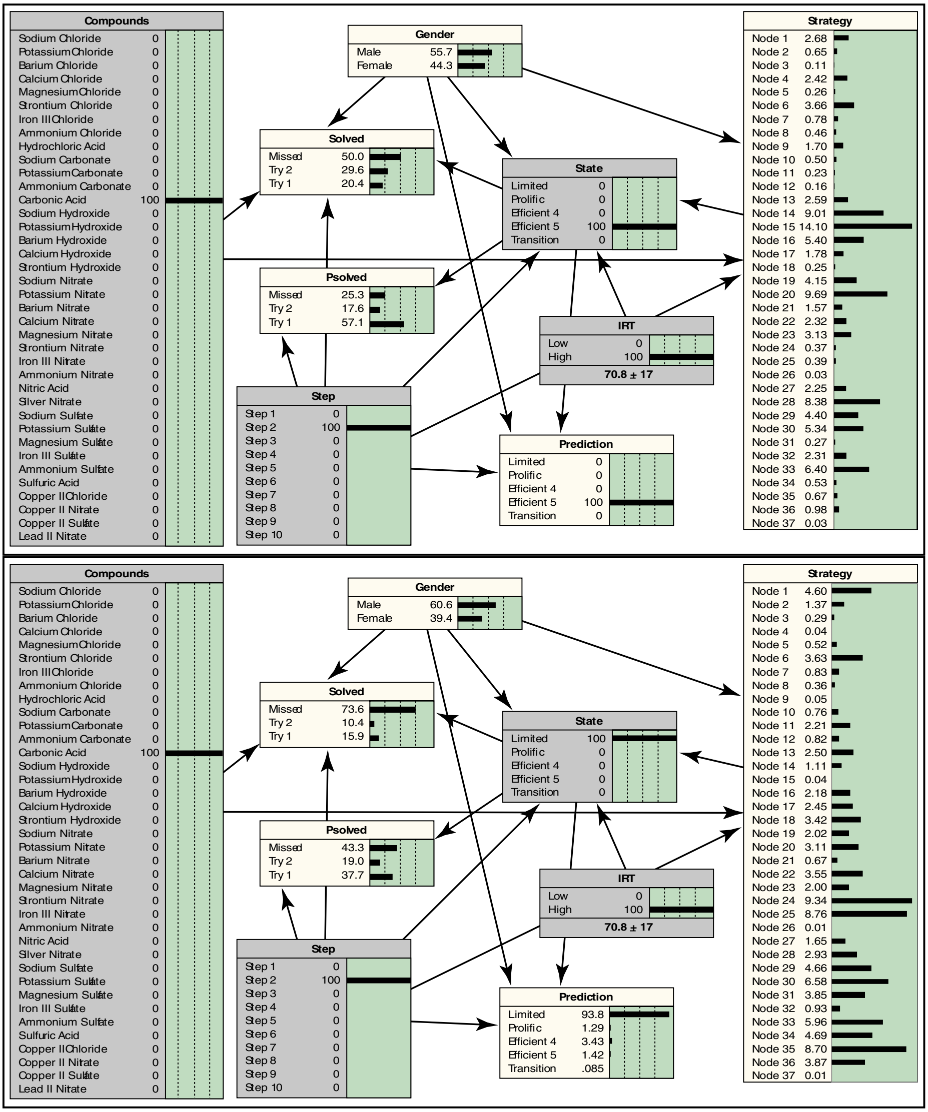


Fig. 5. A Hazmat Belief Network Outcome Analysis. In this analysis the variables Compound, Step, and IRT were fixed and the State was alternated between *efficient_5* (Fig. 5a-top) and *limited* (Fig. 5b-bottom) to determine the outcomes for the Solved, PSolved and Prediction variables.

However, were the student to adopt a *limited* **State** under the same circumstances, then the solve rate would be lower, and the predicted solve rate on the subsequent case would also be lower (Figure 5b). This example illustrates how the output of the descriptive network could serve as a controller engine, which together with a module for sequencing tactics, could deliver pedagogical feedback between cases to improve subsequent performances. In the first example, it is unlikely that feedback would be needed, whereas in the second it would be better justified.

As part of the validation process, we analyzed the sensitivity of the findings for **State** and **Solved** with other evidence nodes (Table 1.). The goal here is to determine which of the other tests provides the best information about the State and Solved node values. A single number that is often used to best describe the sensitivity of one node to another is termed entropy which reflects the uncertainty in a probability mass. The reduction in entropy at the query node by the findings at the test nodes provides a measure of the strength of interactions.

Table 1. Sensitivity of *State* and *Solved* Nodes to Findings at Other Nodes

Findings at:	Entropy Reduction (%) at State	Findings at:	Entropy Reduction (%) at Solved
State	100	Solved	100
Strategy	17.8	Compounds	9.16
Class	7.5	Class	1.33
Step	3.2	Strategy	1.04
Solved	0.25	State	0.56
Compounds	0.23	Step	0.23
Intervention	0.2	Gender	0.04
Experience	0.07	Intervention	0.02
Gender	0.04	Experience	0.01

Such an analysis for **State** indicates that the **Strategy** contributed most to entropy reduction which makes sense as the ANN nodes constituting **Strategy** are the input symbols for the HMM analysis. Similarly, **Step** was the third highest contributor to entropy reduction, and again, this was not surprising as the HMM modeling resulting in the **State** outputs is conducted over a series of cases, i.e. a progress metric. What was less expected was that **Class** was the second highest contributor suggesting that the environmental context under which the problem solving occurred may be an important contributor to the strategy eventually used. This is consistent with earlier correlation data reported for a molecular genetics problem set [29].

A similar analysis (Table 1) of whether or not the case was **Solved** showed that **Compounds** contributed most to entropy reduction, which makes sense given the spectrum of compounds of varying difficulties [30]. The **Class** variable was the second highest contributor again pointing to the importance of the instructional environment.

To evaluate where the model is/is not functioning properly testing was performed with ~1000 student performances that were randomly removed from the dataset before the BN learning. For **State** (Table 2) there was 17% error rate with the *limited* State being the most predictable with an error rate of 9% and the *transition* state being the least predictable with an error rate of 30%. For the **Solved** variable (Table 3) the overall error was 32% and was similar for both the *Missed* and the *solved (Try_1)* values.

Table 2. Classification Error Rates for *States* When Tested with Randomly Selected Unknown Performances

Predicted					
Limited	*Prolific*	*Efficient_4*	*Efficient_5*	*Transition*	Actual
218	4	8	3	3	*Limited*
3	82	20	5	26	*Prolific*
6	6	218	5	8	*Efficient_4*
8	3	3	136	2	*Efficient_5*
4	10	7	18	90	*Transition*
Error rate = 16.96%					

Table 3. Classification Error Rates for *Solved* When Tested with Randomly Selected Unknown Performances

Predicted		
Missed	*Try_1 (Solved)*	Actual
280	152	*Missed*
135	338	*Try_1 (Solved)*
Error rate = 31.71%		

4 Discussion

These studies were motivated by the large number of statistically significant correlations we have observed between different performance metrics and a spectrum of nominal contextual variables including gender, whether the IMMEX cases were performed individually or in groups, student's academic experience, the classroom environment, and overall problem solving ability. These associations have been observed in scientific problem solving situations from middle school through the university and were obtained from relatively large datasets (12-33,000 performances).

The BN models being developed appear consistent with prior Chi square analyses in that the **Solved** variable is most influenced by the **Compound** being identified while the **State** variable was most influenced by the **Strategy** variable followed by **Class** and **Step**. Each of these variables would be expected to influence how the problem is framed and approached in different ways, **Compound** because of the diversity of compounds in the dataset, **Class** in that the way the problem solving is modeled for the students is likely to affect the student's own approach, and **Step** because the problem solving approaches are expected to change as experience is gained..

The most unusual finding was the only distant relationship between the **Solved** and **State** variables suggesting from a learning perspective that these two outcomes may represent separable aspects of the problem solving process, e.g. having a correct model of a concept, and correctly applying the model, which may not only have different cognitive foundations, but may also have implications for supporting student learning.

The strategic approaches for instance, may be best represented by theories of skill acquisition. Across many observable human activities it is apparent that most individuals do not continually improve their performance. As experience is gained and students' approaches to performing the task become more routine, the incremental gains in their skills become smaller and eventually appear to stabilize. Our prior studies and those reported in Figure 2 show that on scientific problem solving tasks this skill stabilization may be accompanied by the stabilization of strategies [41] [30]. However, the strategies with which students stabilize are often not effective, indicating that, experience alone is not sufficient for some students to progress, a finding reported by others [6]. In the current study sensitivity analysis has also shown a limited dependency between whether or not the problem was solved and the approach taken during the process.

While the theory of skill acquisition helps account for the stabilization of strategic approaches, the factor influencing whether or not the problem was solved is less clear, but may relate to variables outside the scope of those currently being collected. In particular they may relate more to attribution theory and the ways that students assign causality to their actions. Whether students relate their performance to internal factors, such as their own level of intelligence or to external factors such as the teacher could have significant effects on motivation, behavior and eventual outcomes. We are currently conducting parallel survey information to begin to probe these contributions.

From the sensitivity analysis, the contribution of the classroom environment (**Class**) was one of the highest contributors to both solved and strategy being the second largest contributor for each. IMMEX is a complex tool and such complex problem solving is likely to be most effective when it is facilitated by strong instructional practices. That teaching plays a crucial role in successful use of the program was documented in a recent study of repeated classroom observations of students engaged in IMMEX problem solving. In some classrooms, what began as open-ended, multi-faceted problem solving became a series of algorithmic procedures; in others the problem unfolded more richly (V. Thadani, manuscript in preparation). A similar finding has been documented in the video component of the Third International Math and Science Study, which examined differences in the mathematics problems assigned by teachers during instruction [10]. That study also found that tasks unfolded quite differently across classrooms, regardless of how they were initially posed to students. It became clear that, though they started with rich problems, teachers were representing those problems quite differently to students.

We are currently developing classroom practice codes that will facilitate the incorporation of such variables to expand our BN architectures. These codes capture instructional events or strategies that may predict students' performance and progress on IMMEX. For instance, one set of variables examines phases of IMMEX lessons on the hypothesis that particular IMMEX events -- such as the presence of an extended IMMEX "sharing" phase (during which the class discusses and reflects on a just-solved problem) -- will be positively correlated with higher performance and more effective strategy use. A second set of variables examines functions of teacher tasks (or directives) and questions; here the hypothesis is that some tasks and questions – for instance, those that promote student metacognition -- will similarly predict greater student gains.

Finally the visual interface developed through the Bayesian modeling is a valuable visualization and training tool for helping to understand the complex contributions of multiple variables to problem solving outcomes.

Acknowledgement. Supported in part by grants from the National Science Foundation (NSF-ROLE 0231995, DUE Award 0126050, ESE 9453918) and the U.S. Department of Education (R305H050052).

References

1. Arroyo,I., & Woolf, B. (2005). Inferring learning and attitudes from a Bayesian Network of log file data. Artificial Intelligence in Education, C.-K. Looi *et al* (Eds.). IOS Press.
2. Alexander, P. A. (2003). The Development of Expertise: The Journey from Acclimation to Proficiency. Educational Researcher, 32 (8), 10-14.
3. Brown, A. & Palincsar, A. (1989). Guided, Cooperative Learning and Individual Knowledge Acquisition. In L. Resnick (Ed.), Knowing, learning, and instruction: Essays in honor of Robert Glaser. Hillsdale, NJ: Lawrence Earlbaum Associates.
4. Case, E.L. (2004). The effects of collaborative grouping on problem solving in first-year chemistry. Ph.D. thesis, Clemson University.
5. Chung, G.K.W.K, deVries, L.F., Cheak, A.M., Stevens, R.H., and Bewley, W.L. (2002). Cognitive Process Validation of an Online Problem Solving Assessment. Computers and Human Behavior, 18: 669.
6. Ericsson, K. A. (2004). Deliberate Practice and the Acquisition and Maintenance of Expert Performance in Medicine and Related Domains. Academic Medicine. 79 (10), S70-S81.
7. Fennema, E. Carpenter, T., Jacobs, V., Franke, M., and Levi, L. (1998). Gender differences in mathematical thinking. Educational Researcher, 27, 6-11.
8. Gallagher, A.M., and De Lisi, R. (1994). Gender differences in scholastic aptitude test mathematics problem solving among high ability students. Journal of Educational Psychology, 86(2), 204-214.
9. Gerosa L., Giordani A., Ronchetti M., Soller A., & Stevens R., (2004). Symmetric Synchronous Collaborative Navigation. In: IADIS International Conference WWW/Internet 2004. A cura di Isaias P., Karmakar N., IADIS Press, Lisboa, Portogallo: 2004. Vol. 1, p. 748-754.
10. Haider, H., and Frensch, P.A. (1996). The role of information reduction in skill acquisition. Cognitive Psychology 30: 304-337
11. Hiebert, J., and Stigler, J. (2000). A Proposal for Improving Classroom Teaching: Lessons from the TIMSS Video Study. The Elementary School Journal, 101, 3-21.
12. Kohonen, T., (2001). Self Organizing Maps. 3rd extended edit. Springer, Berlin, Heidelberg, New York. ISBN 3-540-67921-9, ISSN 0720-678X
13. Lajoie, S.P. (2003). Transitions and Trajectories for Studies of Expertise. Educational Researcher. 32: 21-25.
14. Lepper, M. R., Woolverton, M., Mumme, D., & Gurtner, J. (1993). Motivational Techniques of Expert Human Tutors: Lessons for the Design of Computer-Based Tutors. S. P. Lajoie & S. J. Derry (Eds.), Computers as cognitive tools pp. 75-105. Hillsdale NJ: Erlbaum.

15. Linacre, J.M. (2004). WINSTEPS Rasch measurement computer program. Chicago. Winsteps.com
16. Marshall, S. (1995). Schemas in Problem Solving. Cambridge University Press, ISBN: 0521430720.
17. Mayer, R. E. (2004). Should There Be A Three-Strikes Rule Against Pure Discovery Learning? American Psychologist, 59, 14-19
18. Mayer, R.E., (1998). Cognitive, Metacognitive and Motivational Aspects of Problem Solving. Instructional Science 26: 49-63.
19. Mislevy, R.J., Steinberg, L.S., Breyer, F.J., Almond, R.G., & Johnson, L. (1999). A cognitive task analysis, with implications for designing a simulation-based assessment system. Computer and Human Behavior, 15, 335-374.
20. Moreno, R., & Duran, R. (2004). Do Multiple Representations Need Explanations? The Role of Verbal Guidance and Individual Differences in Multimedia Mathematics Learning. Journal of Educational Psychology, 96, 492-503.
21. Murphy, K. (2004). Hidden Markov Model (HMM) Toolbox for Matlab. Available online at: http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html.
22. O'Regan, K. (2003). Emotion and e-Learning. Journal of Asynchronous Learning Network. 7(3). Pp 78-92.
23. Olson, A., & Loucks-Horsley, S. (Eds). (2000). Inquiry and the National Science Education Standards: A guide for teaching and learning. Washington, DC: National Academy Press
24. Rabiner, L., (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. Proceedings: IEEE, 77: 257-286.
25. Schunn, C. D., & Anderson, J. R. (1999). The Generality/Specificity of Expertise in Scientific Reasoning. Cognitive Science. 13, 337-370.
26. Soller, A., & Lesgold, A. (2003). A Computational Approach to Analyzing Online Knowledge Sharing Interaction. Proceedings of Artificial Intelligence in Education, 2003, 253-260, Sydney Australia.
27. Soller, A., Wiebe, J., & Lesgold, A. (2002). A Machine Learning Approach to Assessing Knowledge Sharing During Collaborative Learning Activities. Proceedings of Computer Support for Collaborative Learning. Boulder, CO, pp. 128-137
28. Stevens, R. H, & Palacio-Cayetano, J. (2003). Design and Performance Frameworks for Constructing Problem-solving Simulations. Cell Biology Education, 2, 162-179.
29. Stevens, R. H., Ikeda, J., Casillas, A., Palacio-Cayetano, J., & Clyman, S. (1999). Artificial Neural Network-based Performance Assessments. Computers in Human Behavior, 15, 295-314.
30. Stevens, R. H., Wang, P., & Lopo, A. (1996). Artificial Neural Networks Can Distinguish Novice and Expert Strategies During Complex Problem Solving. JAMIA, Vol. 3, 2 pp 131-138.
31. Stevens, R., Johnson, D. F., & Soller, A. (2005). Probabilities and Predictions: Modeling the Development of Scientific Competence. Cell Biology Education, 4: 42-57. The American Society for Cell Biology.
32. Stevens, R., Soller, A., Cooper, M., and Sprang, M. (2004) Modeling the Development of Problem Solving Skills in Chemistry with a Web-Based Tutor. Intelligent Tutoring Systems. Lester, Vicari, & Paraguaca (Eds). Springer-Verlag Berlin Heidelberg, Germany. 7th International Conference Proceedings (pp. 580-591).
33. Stevens, R.H., & Najafi, K. (1993). Artificial Neural Networks as Adjuncts for Assessing Medical Students' Problem-solving Performances on Computer-based Simulations. Computers and Biomedical Research, 26, 172-187.

34. Stevens, R.H., Ikeda, J., Casillas, A., Palacio-Cayetano, J., and S. Clyman (1999). Artificial neural network-based performance assessments. Computers in Human Behavior, 15: 295-314
35. Stevens, R.H., and Soller, (2005). Implementing a Layered Analytic Approach for Real-Time Modeling of Student's Scientific Understanding. Proceedings of the Artificial Intelligence in Education Conference, 2005.
36. Sugrue, B., (1995). A theory-based framework for assessing domain-specific problem solving ability. Educational Measurement: Issues and Practice, 14(3), 29-36.
37. Topping, K.J., & Ehly, S. (Eds.) (1998). Peer-assisted learning. Mahawah, NJ: Erlbaum.
38. Underdahl, J., Palacio-Cayetano, J., and Stevens, R., (2001). Practice makes perfect: assessing and enhancing knowledge and problem-solving skills with IMMEX software. Learning and Leading with Technology. 28: 26-31
39. VanLehn, K., (1996). Cognitive Skill Acquisition. Annual Review. Psychology. 47: 513-539
40. VanLehn, K., Ohlsson, S., & Nason, R. (1994). Applications of Simulated Students: An Exploration. Journal of Artificial Intelligence in Education. 5(2), 135-175.
41. Vendlinski, T., Stevens, R. (2002). A Markov Model Analysis of Problem-Solving Progress and Transfer. Journal of Technology, Learning, and Assessment, 1(3).
42. Webb, N. (1992). Testing a Theoretical Model of Student Interaction and Learning in Small Groups. In R. Hertz-Lazarowitz and N. Miller (Eds.), Interaction in Cooperative Groups: The Theoretical Anatomy of Group Learning. (pp 102-119). New York: Cambridge University Press.

A Decision-Theoretic Approach to Scientific Inquiry Exploratory Learning Environment

Choo-Yee Ting[1], M. Reza Beik Zadeh[2], and Yen-Kuan Chong[3]

[1] Faculty of Information Technology
[2] Faculty of Engineering
[3] Center for Multimedia Education and Application Development
Multimedia University, 63100 Cyberjaya Malaysia

Abstract. Although existing computer-based scientific inquiry learning environments have proven to benefit learners, effectively inferring and intervening within these learning environments remain an open issue. To tackle this challenge, this article will firstly address the issue on learning model by proposing *Scientific Inquiry Exploratory Learning Model*. Secondly, aiming at effective modeling and intervening under uncertainty in modeling learner's exploratory behaviours, decision-theoretic approach is integrated into INQPRO. This approach allows INQPRO to compute a probabilistic assessment on learner's scientific inquiry skills (*Hypothesis Generation* and *Variables Identification*), domain knowledge, and subsequently provides tailored hints. This article ends with an investigation on the accuracy of proposed learner model by performing a model walk-through with human expert and field trial evaluation with a total number of 30 human students.

1 Introduction

Recent studies have demonstrated the positive implications of employing scientific inquiry as a learning strategy to engage learners actively in learning science [1,2,3,4,5]. To date, computer-assisted scientific inquiry learning environments such as the *Belvedere* [6], *BGuiILE* [1], *KIE* [2], *SCI-WISE* [5], *SimQuest* [7], and *Smith-Town* [8] have focused on employing scientific inquiry for both domain knowledge and problem solving skills acquisition. Although there is an attempt to employ learner model in *Instruction In Scientific Inquiry Skills* [9], how and when to provide tailored pedagogical interventions that enhance scientific inquiry skills remains difficult. To tackle these complexities, researchers have employed Decision-Theoretic approach [10], an extension of Bayesian networks [11], to model and intervene under uncertainty. This approach has been employed in tutoring systems such as the *DT-Tutor* [12], *iTutor* [13] and *CAPIT* [14] to handle vagueness in exploratory behaviours. However, none of these systems rooted on a specific learning model and built particularly for scientific-inquiry learning environment. In this light, a Decision-Theoretic approach for modeling scientific inquiry skills and generating adaptive pedagogical interventions under a sound learning model is proposed.

In this article, the discussions are center around (i) development of INQPRO learning environment rooted on *Scientific Inquiry Exploratory Learning Model* as learning

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 85 – 94, 2006.

model, (ii) generation of probabilistic learner model that iteratively assesses learner's scientific inquiry skills throughout the learning process, and (iii) tailoring pedagogical interventions to coach learners through Decision-Theoretic approach.

2 Scientific Inquiry Exploratory Learning Model

Kuhn et al. [15] has reported that learners rarely used causal mechanisms to account for the hypotheses they generated and most often just generate a hypothesis out of the blue without even trying to give any substantiation for it. In addition to that, recent study [16] has also found out that learners do most experiments without expressing a hypothesis that is specific enough to guide the inquiry process. Moreover, most hypotheses are formulated without substantiation, neither from experimental data nor from an assumed causal mechanism.

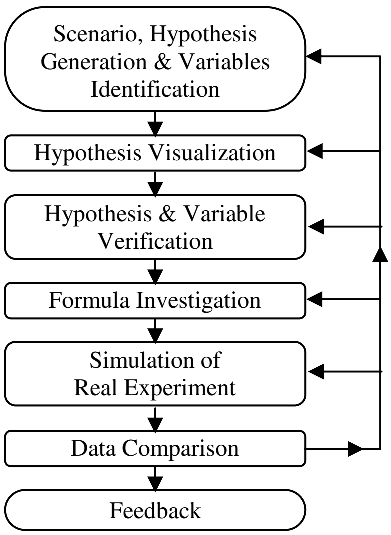


Fig. 1. Scientific Inquiry Exploratory Learning Model

By carefully considering the above constraints, a refined scientific inquiry learning model based on our previous work [17] is proposed (Fig. 1). This learning model is built specifically for the development of INQPRO. The INQPRO interfaces consist of user-interaction components such as the *drag-and-drop elements*, *option buttons*, *checkboxes*, *trackbars*, and *drop down listboxes* to provide a rich learner control experiences. Learners are required to firstly select a scenario upon logging into the learning environment. Having the scenario studied, learners are then requested to identify the different types of variables and subsequently constructing a hypothesis.

INQPRO provides computer simulations that allow learner to infer the characteristics of the model underlying the simulation. By varying the inputs (hypotheses, variables, and graph) to INQPRO, the resulting changes in outputs can be visualized. By observing the computer simulations, it is predicted that learners will reinvestigate into the scenario and verify the hypotheses generated earlier. In short, learners going through *Scientific Inquiry Exploratory Learning Model* will eventually be trained to construct and validate hypotheses, identifying correct and suitable variables, and subsequently generalizing the relationships between the variables.

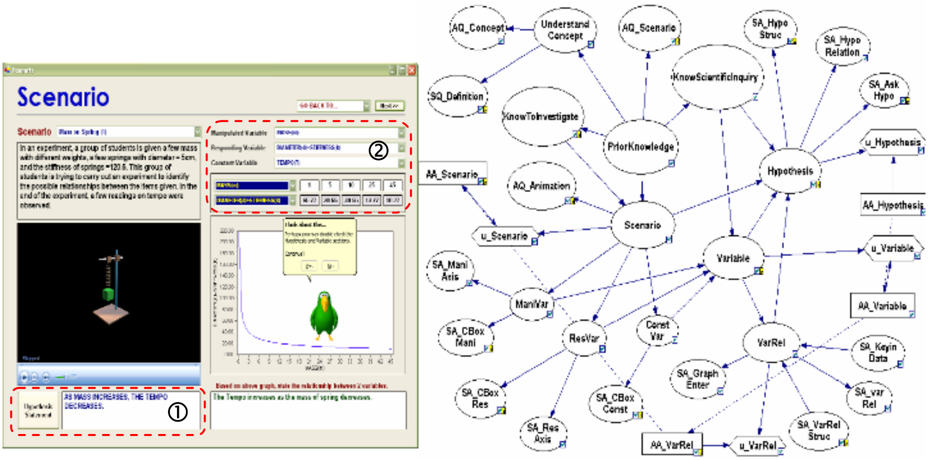
In the following section, the integration of Decision-Theoretic approach into the INQPRO learning environment is described by firstly present an overview of each INQPRO interface and followed by the consideration taken during the generation of Decision network.

3 Decision-Theoretic Approach for INQPRO

Learners involve in reasoning and addressing their own misunderstanding by actively asking their own questions, engaging in hypothesis generation, making and testing predictions about prior concepts. However, study shows that learners might encounter difficulties in performing these activities [18]. In this study, we tackle these challenges by allowing learners explicitly visualize their hypotheses through the "*Hypotheses Visualization*" steps (Fig. 1). Although explicitly implementing this step is a novel approach, modeling learner's reasoning during the learning process remains difficult due to the low bandwidth in inferring learner's exploratory behaviours [19]. Inaccuracy learner modeling will further prevent the pedagogical agent from providing tailored pedagogical interventions. Therefore, Decision-Theoretic approach was employed in INQPRO aiming at handling the uncertainty inherent in learner modeling. We shall now discuss now this approach is integrated into two of the INQPRO interfaces mainly due to space limitation.

3.1 The *Scenario* Interface

Fig. 2(a) depicts the INQPRO *Scenario* interface. A learner is requested to firstly select a scenario upon logging into the learning environment. While studying the given scenario, a computer animation that acts as advance organizer is presented. Having the scenario studied, the learner proceeds with hypotheses generation (Fig. 2(a)-①) sections and later proceed to variables identification (Fig. 2(a)-②). INQPRO does not required learners to draw graph, as drawing of a graph by itself is not a trivial task and has been the object of instruction in itself [20]. Therefore, with the appropriate information on *x*-axis, and *y*-axis provided by learners, INQPRO will plot the graph automatically. This facility aims at providing learners with the opportunity to interpret and relate the graph with selected variables and hypothesis constructed. At the variable relationship column, again, learners are expected to relate the selected variables shown in the graph.



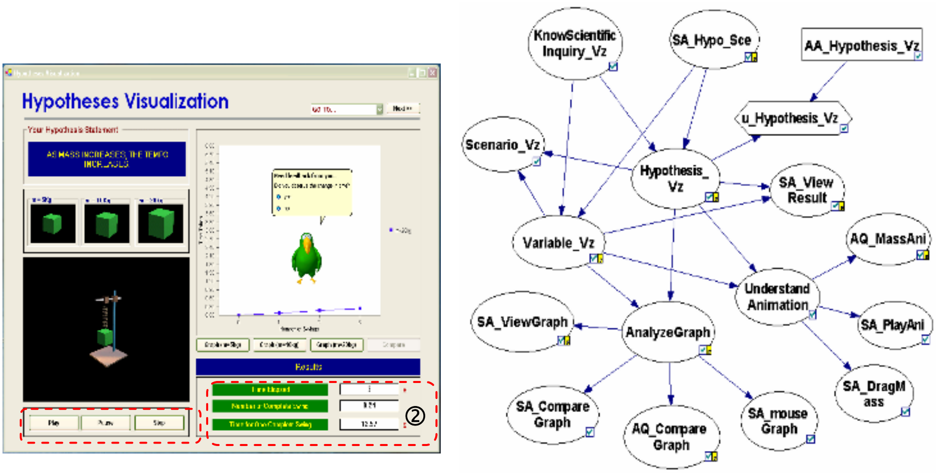
(a) **(b)**

Fig. 2. The Scenario interface and its Decision network

Fig. 2(b) depicts the Decision network integrated into the *Scenario* interface. By iteratively capturing learner's interactions as evidences, the network performs both *diagnostic* and *predictive* reasoning to predict learner's ability in hypothesis generation and variable identification. By performing *diagnostic* reasoning, a learner's ability in hypothesis generation is directly observed from the following evidences: (1) the correctness of hypothesis structure (node *SA_HypoStruct*), (2) the correctness of variable relationships (node *SA_HypoRelation*), and (3) explicit help request from the agent (node *SA_AskHypo*). The *predictive* reasoning, conversely, offers an indirect assessment through the propagation of probability values from nodes *KnowScientificInquiry, Variables*, *Scenario*, and *VarRel*. This is shown by the arcs directed to node *Hypothesis*. The mastery level of variables can be inferred from learner's interaction with INQPRO through the *Variable Identification* section. It is predicted that the degree to which a learner is considered to have mastered the variables (node *Variable*) relies on whether or not s/he has correctly select the variables (nodes *ManiVar*, Re*sVar*, and *ConstVar*). The node *Variable* and nodes *ManiVar*, Re*sVar*, and *ConstVar* are *d-separated* for two reasons: first, the nodes *ManiVar*, Re*sVar*, and *ConstVar* will never been instantiated; second, correctly identify one variable does not guarantee understanding the other variables. Apart from performing probabilistic assessment on learner's ability in hypotheses generation, the Decision-Theoretic approach provides tailored metacognitive interventions (e.g. *AA_Hypothesis*, *AA_VarRel*, *AA_Scenario*, and *AA_Variable*) during learning process. For instance the *AA_Hypothesis* node, it aims at tackling difficulties that can be categorized into (1) unable to find new hypotheses, (2) unable to state or adapt hypotheses on the basis of data gathered, (3) avoid stating hypotheses due to *fear of rejection* [18]. These nodes aims at providing metacognition help instead of providing direct answers.

3.2 The Hypotheses Visualization Interface

Fig. 3(a) depicts the *Hypotheses Visualization* interface of INQPRO. The uniqueness of this interface is that it helps learners explicitly visualize their hypotheses through computer simulations. In other words, depending on the hypothesis generated in *Scenario* interface, the computer simulation presented might not be similar to what the learner has in mind. It is predicted that the naïve concepts can be made explicit which subsequently allows the uncovering of learner's misconceptions can be maximized. Making learner's mental model explicit has been reported to be a vital precursor to mental model restructuring [21]. There are three masses (*m=50g*, *m= 100g*, *m=200g*) where a learner can choose from to investigate the relationships between mass and tempo. Having the mass chosen, learners will proceed to the computer simulation control section (Fig. 3(a)-①). A corresponding computer animation will be displayed upon clicking the *play* button. Conversely, once the *stop* button is clicked, the graph halted and detail analysis of the simulation is reported in the *Results* section (Fig. 3(a)-②).



(a) (b)

Fig. 3. The *Hypotheses Visualization* interface

Fig.3(b) depicts the Decision network for *Hypotheses Visualization* interface. To infer the degree to which a learner is considered to have mastered the variables (node *Variable_Vz*) and hypothesis (node *Hypothesis_Vz*) relies on whether or not s/he is able to analyze the graph (node *AnalyzeGraph*) and understanding of simulation (node *UnderstandAnimation*). However, both *AnalyzeGraph* and *UnderstandingAnimation* cannot be observed directly from the interface. To obtain the posterior probability values for these two nodes, the network performs *diagnostic* reasoning given the instantiation of evidential nodes (nodes *AQ_MassAni*, *SA_PlayAni*, *SA_DragMass*, *SA_ViewGraph*, *SA_CompareGraph*, and *AQ_CompareGraph*).

4 Preliminary Evaluation

In order to gain an insight into how the INQPRO Decision networks behave, we conducted a study to produce an empirical evaluation of its accuracy in predicting the learners' scientific inquiry skills (nodes *hypothesis* and *variable*). However, evaluating the learner model directly is difficult as it requires assessing of learners' actual scientific inquiry skills that may vary from one interface to other during the learning process. Thus, in this study, we performed a model walk-through with human expert and field trial evaluation to study the classification accuracy. In this section, we shall focus on the evaluation for *Scenario* Decision network only as the similar evaluation process repeated for the rest of INQPRO Decision networks.

4.1 Model Walk-Through with Human Expert

The walk-through of INQPRO learning environment and Decision networks with 2 domain expert that served two purposes. The walk-through process aimed at identifying the suitability of questions prompted by agent (nodes begin with *AQ* in Decision networks) at each interface. This was easily identified by expert while interacting with INQPRO. The verification process was eased by a marking sheet containing all the questions prompted by pedagogical agent. Table 1 shows the results of the appropriateness of agent questions by counting the unnecessary questions prompted. There are three *AQ* nodes and the *AQ_Animation* is considered unnecessary. The reason given is that learners will hardly view the computer animation for more than 2 times. The average accuracy is 67%.

Table 1. Inappropriate agent interventions in the *Scenario* interface

INQPRO interface	# inappropriate agent intervention(s), error rate (name of nodes)	
	Domain Expert 1	**Domain Expert 2**
Scenario	1, 33% (*AQ_Animation*)	1, 33% (*AQ_Animation*)
Average error rate	**33%**	

The second analysis carried out was a *predictive* evaluation. It aimed at studying the accuracy in classifying learners' acquisition level of *Hypothesis* and *Variables*. To serve this purpose, we have developed our own version of *Artificial Student* [22] technique. A total of 500 *artificial students* were generated by taking into consideration of different abilities in generating hypothesis, and identifying variables. The *artificial students* are categorized into three categories (*High*, *Moderate*, and *Low*) of hypothesis and variable understanding. From the 500 generated *artificial students* (n_{Low}= 36, n_{Moderate}= 316, n_{High}= 121), 15 were chosen randomly from each category. The evaluation aimed at investigating the accuracy in predicting the mastery level for *Hypothesis* and *Variable*. For each *artificial student*, an instantiated Decision network was printed and presented to the expert. Our study has found out that expert preferred the presentation of *artificial students* in graphical way rather than tabulated format. Table 2 shows the predictive accuracy of *Scenario* Decision network. The overall average accuracy for *Hypothesis* is 75.56% while 59.96% for the variable. The accuracy values for all the categories were slightly low due to the random instantiation of nodes. The random instantiation of evidential nodes has neglected the weights of dependencies between the query nodes and evidential nodes that were predefined in CPT and subsequently, resulting in low accuracy.

Table 2. Expected mastery level for *Hypothesis* and *Variable*

# Artificial students	**# correct classification for *Hypothesis***	**% of accuracy**	**# correct classification for *Variable***	**% of accuracy**
High(n=15)	11	73.3	8	53.3
Moderate(n=15)	12	80.0	10	66.6
Low(n=15)	11	73.3	9	60.0
Average	***11.3***	***75.56***	***9***	***59.96***

It was found out that the accuracy of a particular query node depends on its adjacent query nodes. Thus, to improve the accuracy for classification with respect to node *Variable*, one of the solutions is to reduce its adjacent query nodes. In the following section, we further evaluate the behaviour of *Scenario* Decision network by carrying out the field trial evaluation.

4.2 Field Trial Evaluation

A total of 30 subjects participated in field test. All the subjects were first-year university students who had gone through O'level Science/Physics course during their secondary school. Subjects participated in a session that lasted at most 120 minutes which consisted of a pre-test, a session with INQPRO, and a post-test. The pre-test and post-test both consisted of 23 multiple choices questions. From the 23 questions, there are 6 questions on *Hypothesis* and 17 on *variables*. Before the pre-test and post-test were administrated, the learners were requested to elicit their mastery level of *hypothesis* and *variable*. To help in the elicitation process, a 3-rank scale (*full mastery*, *partial mastery*, *non mastery*) was given to the learners. Apart from the model walk-through with expert as one of the means for evaluating the *Scenario* Decision network, we performed 2 comparative evaluations between the classifications of the *Scenario* Decision network to (1) the pre-test results, (2) learners' self-evaluated rankings. Comparing the classification of network to pre-test results should also be reflected through node *ScientificInquirySkills* (Fig. 5). The comparison is valid under the assumption that *Scenario* is the first interface in INQPRO and thus, actual learning is yet to fully occur. Table 4 shows the comparative evaluation between the *Scenario* Decision network and results from Pre-Test. The network achieved 80% of correctly classifying the learners' *hypothesis* and *variable* understanding. We found out that the current version of network has a tendency in returning either "*full mastery*" or "*non mastery*". The possible reason is that the both nodes Hypothesis and Variable returned "*mastery*" or "*non mastery*" given all the evidential nodes are instantiated. This is shown by the lowest accuracy for *Moderate artificial student* category in Table 3.

Table 3. Comparison of learner's pre-test and self-ranking classification accuracy

Category of Human learners	# correct classification based on pre-test results	Accuracy (%)	self-ranking results	Accuracy (%)
High (n=2)	2	100	1	50
Moderate (n=8)	5	62.5	7	87.5
Low (n=20)	17	85	13	70
Average Accuracy		***80***		***70***

In this article, we end our discussion on evaluating the Scenario *Decision* network by performing comparative evaluation on learner self-ranking results. Table 3 shows the comparative accuracy with respect to learners' self-ranking results. There are 13 out of the 20 low performance learners ranked as "*non mastery*" while 7 of them ranked themselves "*partial mastery*". Although the overall accuracy is relatively low, learner self-ranking is no doubt a method for eliciting current understanding of *hypothesis* and *variables*. This is particularly important as assessing learner's hypothesis and variable understanding during the learning session is difficult as traditional educational assessments are done by formative and summative evaluation.

5 Conclusion and Future Work

The aim of this paper is to create a probabilistic learner model for inferring learner's scientific inquiry skills acquisition level and subsequently providing adaptive feedback that enhance learning. This work addresses a methodology approach for overcoming one of the major limitations of existing scientific inquiry learning environments by integrating probabilistic learner model into a sound proposed instruction model, the *Scientific Inquiry Exploratory Learning Model*. In this study, a Decision-Theoretic approach has been integrated into INQPRO aiming handling the large amount of uncertainty involved in acquiring scientific inquiry skills (hypothesis generation and variables identification), and providing tailored pedagogical interventions. In this study, we employed *Artificial Students* technique to investigate the network behaviour by performing case-based evaluation.

The next step of our work will involve evaluating refining the Decision networks to model and support metacognition (Self-Regulation). The current version of the Decision networks does not take into consideration of *time* as a factor in modeling learners. However, results obtained from the expert and human learners reflect that there is a need to incorporate *time* to enhance the accuracy of classification. In addition, to represent the evolving learner's mastery level of scientific inquiry skills, the *Dynamic Decision Networks* shall be employed and examined.

Acknowledgement

This study is carried out supported by the Microsoft Research Fellowship, Malaysia. The Decision Networks models mentioned in this paper were created using the GeNIe

and SMILE modeling application developed by the Decision Systems Laboratory of the University of Pittsburgh (http://www.sis.pitt.edu/~dsl).

References

1. Frederiksen, J. R., White, B. Y.: Teaching and learning generic modeling and reasoning skills. Journal of Interactive learning Environment. **5** (1998) 33-52
2. Linn, M. C.: Designing the knowledge integration environment. International Journal of Science Education. **22** (2000) 781-796
3. Pryor, A., Soloway, E.: Foundation of Science: Using Technology to Support Authentic Science learning. (1997). http://hi-ce.eecs.umich.edu/papers/
4. Reiser, B. J., Tabak, I., Sandoval, W. A., Smith, B., Steinmuller, F., Leone, T. J.: BGuILE: Stategic and Conceptual Scaffolds for Scientific Inquiry in Biology Classrooms. In: S.M. Carver & D. Klahr (Eds.). Cognition and Instruction: Twenty five years of progress. Mahvah, NJ: Erlbaum (2001)
5. Shimoda, T. A., White, B. Y., & Frederiksen, J. R.. Student goal orientation in learning inquiry skills with modifiable software advisors. Science Education. **86** (2002) 244-263
6. Paolucci, M., Suthers, D. & Weiner, A.. Automated advice giving strategies for scientific inquiry. In C. Frasson & G. Gauthier & A. Lesgold (Eds.). 3rd International Conference on Intelligent Tutoring Stystems. (1996)
7. Veermans, K, van Joolingen, W. R.: Combining Heuristics and Formal Methods in a Tool for Supporting Simulation-Based Discovery Learning. 7th Intelligent Tutoring Systems, Brazil (2004)
8. Shute, V.J., & Glaser, R. A large-scale evaluation of an intelligent discovery world: Smithtown. Interative Learning Environments. **1** (1990) 51-77
9. Miller, T., Kretschmer, M., Meyer, T., Steuck, K., Pesthy, C.: Teaching Scientific Inquiry Skills with an Intelligent Tutoring System. International Conference on Mathematics / Science Education and Technology, USA (1999)
10. Jameson, A.: Numerical Uncertainty Management in User and Student Modeling: An Overview of Systems and Issues. User Model. User-Adapt. Interact. **5** (1995) 193-251
11. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: networks of Plausible Inference. San Mateo, CA: Morgan-Kaufmann (1998)
12. Murray, R.C., VanLehn, K., & Mostow, J.: Looking ahead to select tutorial actions: A decision-theoretic approach. International Journal of Artificial Intelligence in Education. **14** (2004) 235-278
13. Pek, P.K., Poh, K.L.: Making decisions in an intelligent tutoring system. International Journal of Information Technology and Decision Making **4** (2005) 207-233
14. Mayo, M and Mitrovic, A.: Optimizing ITS behaviour with Bayesian networks and decision theory. International Journal of Artificial Intelligence and Education. **12** (2001) 124-153.
15. Kuhn, D., Black, J., Keselman, A., & Kaplan, D.: The development of cognitive skills to support inquiry learning. Cognition and Instruction. **18** (2000) 295-523
16. Löhner, S., Van Joolingen, W.R., Savelsberg, E.R., Van Hout- Wolters, B.: Students' reasoning during modeling in an inquiry learning environment. Computers in Human Behavior. **21** (2005) 441-461
17. Ting, C.Y.: Integrating Animated Pedagogical Agent into Scientific Inquiry Model: A Cognitive Tools Perspective. IEEE International Conference on Computers in Education 2002, Auckland, New Zealand, December 3-6, 2002

18. de Jong, Ton, van Joolingen, Wouter R.: Scientific discovery learning with computer simulations of conceptual domains.Review of Educational Research. **68** (1998) 179-201.
19. VanLehn, K. Student Modeling. In: Polson, M. C, Richardson, J. J. (eds.) Foundations of Intelligent Tutoring Systems, pg. 55-78. Erlbaum, Hillsdale, NJ (1988)
20. Karasavvidis, I., Pieters, J. M., Plomp, T.: Exploring the mechanisms through which computers contribute to learning. J. Comp. Assisted Learning. **19** (2003) 115-128
21. Chi, M.T.H., Roscoe, R.D.: The processes and challenges of conceptual change. In: M. Limon, M., Mason, L. (Eds.), Reconsidering conceptual change: Issues in theory and practice. Kluwer, Netherlands (2002)
22. Ting, C.Y., M. Reza Beik Zadeh, & Liau, B. H. Artificial Student Toolkit for Assessing Bayesian Student Model. Proceedings of IEE International Conference on Intelligent Systems (2005)

Conceptual Change Modeling Using Dynamic Bayesian Network

Choo-Yee Ting[1] and Yen-Kuan Chong[2]

[1] Faculty of Information Technology
[2] Center for Multimedia Education and Application Development
Multimedia University
63100 Cyberjaya Malaysia

Abstract. Modeling the process of conceptual change in scientific inquiry learning environments involves uncertainty inherent in inferring learner's mental models. INQPRO, an intelligent scientific inquiry exploratory learning environment, refers to a probabilistic learner model aims at modeling conceptual change through the interactions with INQPRO Graphical User Interface (GUI) and Intelligent Pedagogical Agent. In this article, we first discuss how conceptual change framework can be integrated into scientific inquiry learning environment. Secondly, we discuss the identification and categorization of conceptual change and learner properties to be modeled. Thirdly, how to construct the INQPRO learner model that employs Dynamic Bayesian networks (DBN) to compute a temporal probabilistic assessment of learner's properties that vary over time: awareness of current belief, cognitive conflict, conflict resolution, and ability to accommodate to new knowledge. Towards the end of this article, a sample assessment of the proposed DBN is illustrated through a revisit of the INQPRO *Scenario* interface.

1 Introduction

Recent years have demonstrated a flourishing of studies on scientific-inquiry learning environments such as the *BGuiILE* [1], *KIE* [2], and *SCI-WISE* [3] to engage learners in learning science while systems like *Help-Tutor* [4], and *ACE* [5] aim at modeling metacognition. However, little attention has been given to employ probabilistic approach to model conceptual change in scientific inquiry exploratory learning environment. Although a framework for conceptual change within scientific inquiry has been proposed in *SpacePlanting* [6], explicitly modeling of the process is particularly difficult as one system has to deal with the high level of uncertainty inherent in inferring learner's mastery level of scientific inquiry skills, prior knowledge, cognitive states (*cognitive conflicts*, *metacognition*, and *accommodation*). To handle the uncertainty in learner modeling, Bayesian networks [7] are employed by researchers [5,8]. Bayesian networks are mostly applied to assess user's properties that remain unchanged during a session. However, learner properties often vary with time, e.g., learner's cognitive states, scientific inquiry skills, and domain knowledge evolve across time in INQPRO learning environment. Thus, to model the evolving learner's properties, researchers have leveraged DBN [5,9]. In this light, due to the fact that

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 95 – 103, 2006.

conceptual change is a process that varies with time, we employed DBN. The work presented in this paper is an attempt to (1) propose how conceptual change approach can be integrated into Scientific Inquiry Exploratory Learning Model, (2) identify and establish causal relationship between variables in conceptual change, and (3) applying DBN approach to model conceptual change process by taking the INQPRO *Scenario* interface as sample assessment.

2 Conceptual Change in INQPRO

Learning involves altering one's existing conceptual framework in the light of new experience. Conceptual change is thus considered to be a process of progressively reconstructing mental representations of events in one's environment [10,11,12,13]. In this study, the conceptual change process is fostered through the iterative scientific-inquiry process in INQPRO, an intelligent computer-based scientific inquiry exploratory learning environment (Fig. 1).

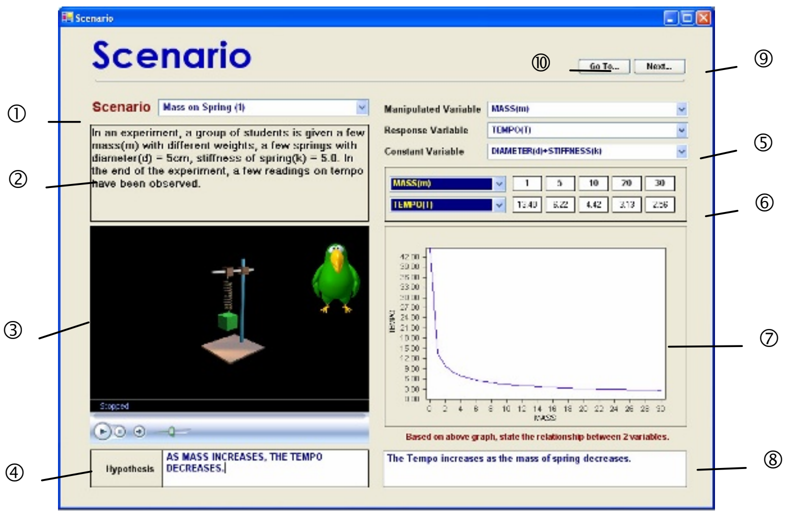


Fig. 1. INQPRO *Scenario* Interface

In this study, the proposed conceptual change framework within INQPRO scientific inquiry learning environment consists of the following three phases and has been modify from conceptual change framework discussed in [14]:

Phase 1: Acknowledging and Assimilating the Selected Scenario
The initial step towards conceptual change is to foster the acknowledgement of selected *Scenario* (Fig. 1-①). For this purpose, the computer animation (Fig. 1-③) that acts as advance organizer will help the new idea to be dressed up enough to gain learner's attention and activate prior knowledge. By iteratively interact with Intelligent Pedagogical Agent, and making connection between the computer simulation

(Fig. 1-③) and scenario, it is predicted that the naïve concepts can be made explicit which subsequently allows the uncovering of learner's misconceptions can be maximized. Making learner's mental model explicit has been reported to be a vital precursor to mental model restructuring [10,11].

Phase 2: Evaluating and Accommodating the Possible Discrepant Event
Learners often do not see the reason to change their beliefs as they provide good explanations of their everyday experiences, function adequately in the everyday world, and are tied to years of confirmation. In order for them to re-examine their initial explanations of prompted new ideas, the learning environments seem to play vital role. The learning environments have to theoretically related and provide meaningful experiences. One of the strategies is to create cognitive conflict. Together with INQPRO GUI and Intelligent Pedagogical Agent, the discrepant events have to be invoking (1) *Dissatisfaction*. Learner must first realize that there are some inconsistencies and that their way of thinking does not solve the problem at hand. Metacognition awareness was promoted in INQPRO learning environment by encouraging learners to make their ideas overt, to test them and compare them with those of other learners and to give scientific explanations. (2) *Intelligibility*. The new information must not only make sense and learner must be able to regurgitate the argument, (3) *Plausibility*. The conception must be plausible for it to be accommodated. It must be able to be integrated to existing prior knowledge to solve the problem, (4) *Fruitfulness*. To achieve fruitfulness in this study, different interface has different activities to achieve this target. The new concept is then been employed to open up new areas of inquiry.

Phase 3: Reevaluation and Generalization
Solving an instance of a problem once can hardly create a new and robust mental structure. The newly formed mental structure might still be rather fragile and might easily become disintegrated. Thus, apart from questions prompted by the Intelligent Pedagogical Agent at the end of the learning session, learners are encouraged to further with new scenario. Ultimately, the new knowledge needs to be integrated into the beliefs system.

In the following sections we will describe how the above conceptual change framework can be model by Bayesian networks and adding time factor to explicitly represent the learner's cognitive states that evolve across time.

3 Conceptual Change Modeling Using DBN

Cognitive research often focuses on descriptions of the cognitive performance of subjects at different ages and at different levels of expertise rather than on the implicit mechanisms explaining how conceptual change happens. There are two main reasons: (1) modeling of learner cognitive states such as cognitive abilities during the learner-computer interaction are a task frequently permeated with uncertainty. Significant cause for this uncertainty is that often the identical situation can greatly induce a variety of different cognitive states in different learners, and (2) the process of conceptual change is a slow and gradual affair that evolves across time [10]. To handle the high level of uncertainty in this modeling task, in next subsection, we explicitly represent the probabilistic nature of the relations between learner cognitive states, their causes and effects for conceptual change using Bayesian Network (BN). Due of space limitations, we have to limit the description of conceptual change Bayesian networks model to the INQPRO *Scenario* interface. However, the same modeling approach is applicable to other INQPRO interfaces.

3.1 Model Structure and Variables

Fig 2 depicts the fine-grained construction of conceptual change Bayesian networks for INQPRO *Scenario* interface. The network is generated real-time when the INQPRO *Scenario* interface is activated by learners. The network consists of 3 submodels: (1) *Acknowledgement & Assimilation submodel*, (2) *cognitive Conflict submodel*, and (3) *Cognitive Resolution & Accomodation submodel*. For each of the submodels, nodes are categorized into groups of *Scientific-Inquiry Exploration* nodes, *Agent Interaction* nodes, and *Conceptual Change* nodes. We now describe each submodel in detail:

- *Acknowledgement & Assimilation Submodel*
  The relevancy of the new information presented in the selected scenario to a learner is very much depends on the level of prior knowledge. It is inferred that the higher relevancy of selected scenario to learner's existing prior knowledge will increase the assimilation of new information, however, decreasing the possibility of being a discrepant event. In order to predict the degree to which the learner is considered to have assimilated the new information, apart from relevancy of selected scenario, is influenced by whether or not he/she has demonstrated misconceptions. The misconceptions can be inferred from the degree of correctly construct hypothesis (Hypothesis submodel), determine types of variables, manipulate graphs, and identify the variable relationships (Variable submodel). The details about *Hypothesis* and *Variable* submodels can be found in [16]. Form this submodel, it is predicted that interaction with agent and learner's metacognition awareness have influenced towards assimilation of new information.

- *Cognitive Conflict Submodel*
  A learner who is experiencing cognitive conflict is predicted to interact more frequently with the Intelligent Pedagogical Agent and time is spent on *Hypothesis* components (Fig 1-④). This interactions and time spent may serve as evidence for the system to infer that the learner is actively revising existing mental model, and subsequently provide an important clue to further interpret learner's attempt to object or accept the new information.

- *Cognitive Resolution & Accommodation Submodel*
  Before accommodating the new information, the process of cognitive resolution occurs. The degree to which cognitive resolution is considered high is influenced by cognitive conflict. Learner who experiences cognitive conflict will have the tendency to resolve it, thus suggesting that higher probability of cognitive conflict will consequence in higher probability of cognitive resolution. By observing the evidence from changing the variables (*c_var* node), hypothesis (*c_hypo* node), manipulating graph (*c_graph* node), and stating the variables relationship

(*c_varRel* node), the low bandwidth issue [17] for cognitive resolution is reduced. Having high possibility of cognitive resolution will in turn increase the possibility of accommodating new information. The causal dependencies are shown in Fig 2 that the arrows are pointing from *Plausibility* and *Intelligibility* nodes to *accommodate* node. The evidence for supporting whether or not the new information is considered to be plausible and intelligible is provided through learner's iterative interaction with the Agent (*Agent Prompt* node). The evidence of a learner who tries to accommodate can be obtained from the correctness of hypothesis construction, identification of variables, inferring the graph, and stating the relationship between variables before click *next* button (Fig 1-⑨) and leave the *Scenario* interface.

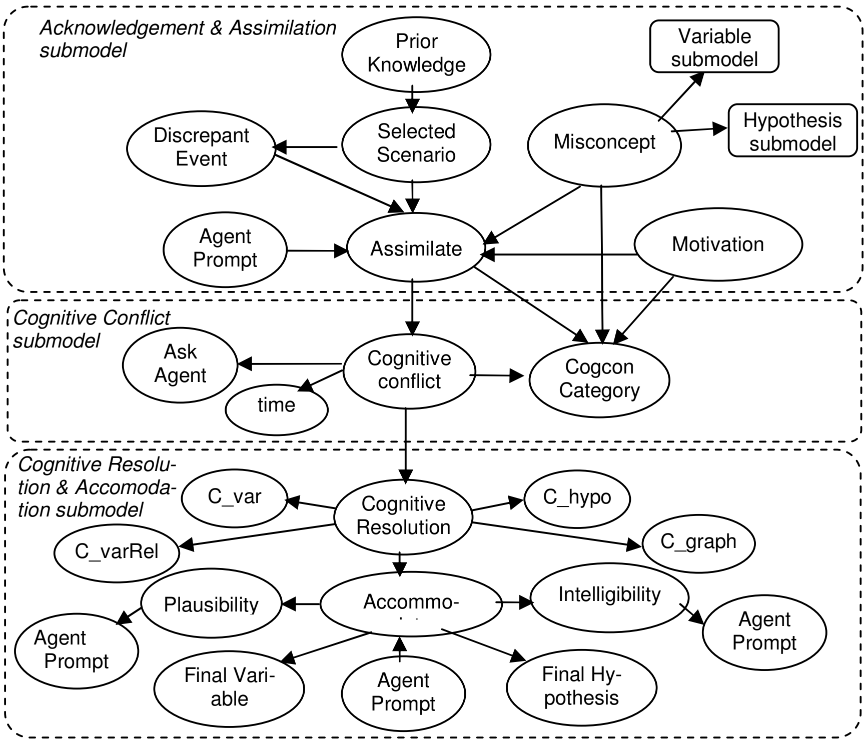


Fig. 2. Conceptual Change Bayesian Networks Submodels for *Scenario* Interface

3.2 Probabilistic Dependencies Between Conceptual Change Variables

The Bayesian network depicted in Fig 2 does not provide direct mechanism for representing temporal dependencies. In attempting to add temporal dimension into the existing BN model, Dynamic Bayesian Network (DBN) is employed. Fig 3 depicts the high-level presentation of DBN employed for presenting the conceptual change for the INQPRO *Scenario* Interface. The time slice t_0 represents the present state of learner while the time slice t_1 describes immediate subsequent state. These time slices are interconnected by temporal relations, which are demonstrated by arcs joining variables. The temporal causal dependencies between the time slices are then defined through the Conditional Probability Tables (CPTs). The variables represented by *Dynamic nodes* (D) evolve over time, while variables that exist in only one time slice are referred to as *Temporary nodes* (T) [12].

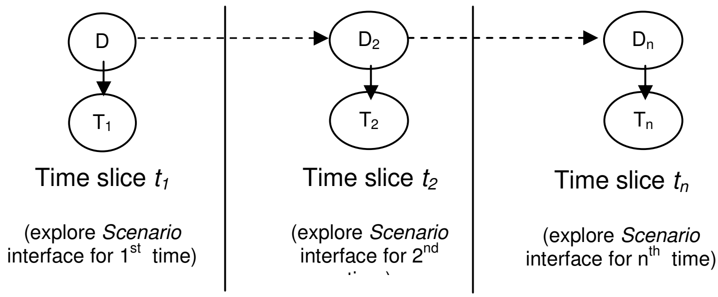


Fig. 3. High-level presentation of conceptual change DBN for *Scenario* Interface

Fig 4 depicts the details of conceptual change time slice employed to represent snapshot of the evolving temporal process. The DBN consists of a sequence of BNs each representing the INQPRO *Scenario* interface activated by learner at a particular time-point.

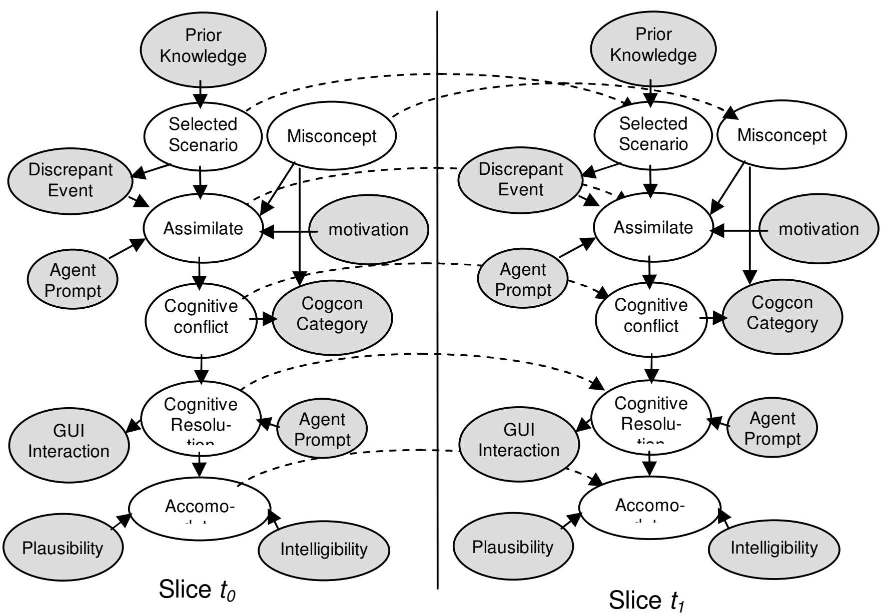


Fig. 4. Two time slices of the DBN model of conceptual change in *Scenario* Interface

Thus, from Fig. 5, we can tell that a learner has explored the INQPRO *Scenario* interface for the second time. In the DBN, there are links between these nodes: *Selected Scenario*, *Assimilate*, *Cognitive Conflict*, *Cognitive Resolution*, and *Accommodate*. The shaded nodes represent random variables that are temporal nodes (*T*) where some of which evidences are available to update the learner model at a given time slice t_i. These evidence nodes include the *Agent Prompt* nodes, as well INQPRO *Scenario GUI Interaction* nodes. The links between the two *assimilate* nodes, for example, model the fact that a learner is likely to assimilate new information at time t_1 if s/he is predicted to have high possibility of assimilation at time t_0.

The links between *Cognitive Conflict* encode that the value of cognitive conflict increases at time t_1 if at time t_0, there is no evidence that the learner has attempt to resolve the conflict. The value *Cognitive Resolution* nodes, however, are expected to increase as the time slices increase representing that the learner is able to resolve the cognitive conflicts through interacting with the evidence nodes (*GUI Interaction* node, *Agent Prompt* node). Learner who demonstrates high possibility to accommodate new information (through *Accommodate* node) at time t_0 is predicted to have better formularizing the new knowledge at time t_1.

4 Sample Assessment

In this section, we illustrate the probabilistic assessment of learner's conceptual change states through the DBN as depicted in Fig 5. The propagation of available evidence within and among the time slices allows the model to incrementally refine the assessment on the user's conceptual change states. Assume that initially at t_1 a learner is assessed to possess four properties: (i) a low prior domain knowledge, (ii) high possibility of misconceptions, (iii) unable to correctly answer agent questions, and (iv) unmotivated, is exploring the INQPRO *Scenario* interface. Due to these four initial properties, it is predicted that the *assimilate* node returns low probability value (0.2) suggesting that the learner is facing with a discrepant event and experiencing difficulty in assimilating new information. The low probability value of assimilation will in turn infer that the learner is experiencing high cognitive conflict (0.7). Under normal circumstances, it is predicted that a learner who is experiencing cognitive conflict will try to resolve it by interacting with Intelligent Pedagogical Agent and INQPRO *Scenario* interface. Therefore, the *cognitive resolution* node has a higher value (0.4) than *accommodate* node. However, in the first time slice t_1, it is predicted that plausibility and intelligibility of new information has yet to achieve and thus suggesting a low probability value for *accommodate* node (0.3). After the second time exploring *Scenario* interface (t_2), assuming that this time the learner is able to answer agent's questions (*agent prompt* node), and therefore suggesting that a higher probability value for *assimilate* node (0.7). From the evidence obtained from reconstructing the hypotheses (*c_hypo* node), reidentifying variables (*c_var* node), manipulating the graph (*c_graph* node), and restating the variables relationships (*c_varRel* node), the probability for *cognitive resolution* node is updated, and subsequently increase the probability for *cognitive resolution* node (0.6). In addition to that, the fruitful agent-learner interactions (*Agent Prompt* nodes) suggest a higher *intelligibility* and

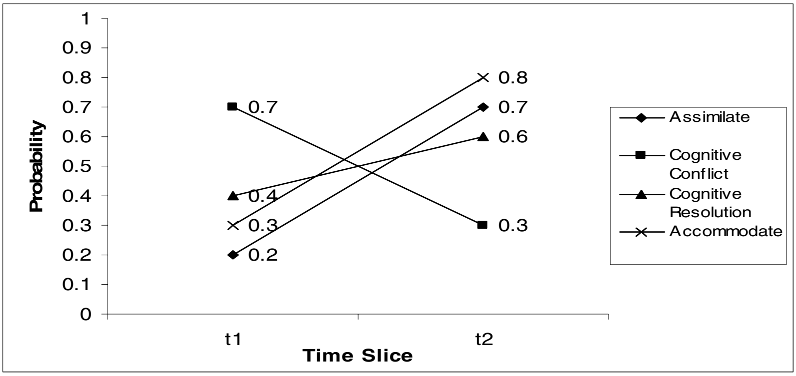


Fig. 5. Two time-slice conceptual change DBN for INQPRO *Scenario* interface

plausibility. From Fig 5, the *accommodate* node demonstrates a high value (0.8) due to the direct causal influence from *cognitive resolution* node, *intelligibility* node, and *plausibility* node.

5 Conclusion

In this article, we proposed the integration of conceptual change framework into INQPRO Scientific Inquiry Exploratory Learning environment. To support the uncertainty inherent in learner modeling throughout the exploration session, INQPRO has moved from static Bayesian network to employing Dynamic Bayesian Network (DBN). DBN allows the system to model learner's cognitive states that evolve across time. In section 3, we highlighted the methodological approach for identifying and explicitly categorizing the conceptual change process into Bayesian network subnetworks. These subnetworks represent the three proposed conceptual change phases within scientific inquiry learning environment. Preliminary investigation has been carried out suggesting that the proposed DBN is capable of capturing learner's interactions and infers the implicit cognitive states soundly.

In the next step of this study, we shall evaluate the conceptual change learner model with human participants, allowing us to verify the assumptions made during the modeling process. We are also working on look-ahead decision-theoretic agent that aims at maximizing accommodation of new information in order to achieve effective learning.

Acknowledgement

This study is carried out supported by the Microsoft Research Fellowship, Malaysia. The Bayesian Networks models mentioned in this paper were created using the GeNIe and SMILE modeling application developed by the Decision Systems Laboratory of the University of Pittsburgh (http://www.sis.pitt.edu/~dsl).

References

1. Reiser, B. J., Tabak, I., Sandoval, W. A., Smith, B., Steinmuller, F., Leone, T. J.: BGuILE: Stategic and Conceptual Scaffolds for Scientific Inquiry in Biology Classrooms Carver, S. M. & Klahr, D (Eds.) Cognition and Instruction: Twenty five years of progress. Mahvah, NJ: Erlbaum (2001)
2. Linn, M. C.: Designing the knowledge integration environment. International Journal of Science Education. 22 **8** (2000) 781-796.
3. Shimoda, T. A., White, B. Y., Frederiksen, J. R.: Learner goal orientation in learning inquiry skills with modifiable software advisors. Science Education. **86** (2002) 244-263
4. Aleven, V., Roll, I., McLaren, B., Ryu, E.J., Koedinger, K.: An architecture to combine meta-cognitive and cognitive tutoring: Pilot testing the Help Tutor. AIED 2005, Netherlands (2005)
5. Bunt A., Conati C and Muldner K.: Scaffolding self-explanation to improve learning in exploratory learning environments. 7th International Conference on Intelligent Tutoring Systems, Maceio, Brazil. (2004)
6. Wichmann, A., Gottdenker, J., Jonassen, D., Milrad, M.: Developing a Framework for Conceptual Change within Scientific Inquiry. ICALT 2003, Greece (2003)
7. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: networks of Plausible Inference. San Mateo, CA: Morgan-Kaufmann (1988)
8. Jameson, A. Numerical Uncertainty Management in User and User Modeling: An Overview of Systems and Issues. User Model. User-Adapt. Interact **5** (1995)
9. Murray, R.C., VanLehn, K., Mostow, J.: Looking ahead to select tutorial actions: A decision-theoretic approach. International Journal of Artificial Intelligence in Education. **14** (2004)
10. Chi, M.T.H. & Roscoe, R.D.: The processes and challenges of conceptual change. In Limon, M and Mason, L (Eds.), Reconsidering conceptual change: Issues in theory and practice. Kluwer, Netherlands (2002)
11. Vosniadou, S.: On the nature of naive physics. In M. Limon & L. Mason (Eds.), "Reconsidering conceptual change: Issues in theory and practice. Dordrecht: Kluwer (2002)
12. Carey, S.: Conceptual Change in Childhood. Cambridge, MA: Bradford Books, MIT Press. (1985)
13. Posner, G. J., Strike, K. A., Hewson, P. W., Gertzog, W. A.: Accommodation of a scientific conception: Towards a theory of conceptual change. Science Education. **66** (1982)
14. Zirbel, L.E.: Framework for Conceptual Change. Http::/cosmos.phy.tufts.edu/~zirbel/ScienceEd/Framework-for-Conceptual-Change.pdf. (2005)
15. Schafer, R, Weyrath, T.: Assessing Temporally Variable User Properties With Dynamic Bayesian Networks", User Modeling: Proceedings of the Sixth International Conference. Vienna, New York: Springer Wien New York (1996)
16. C.Y. Ting and M.R. Beik Zadeh. Modelling and Supporting Learners in Scientific Inquiry Exploratory Learning: A Bayesian Approach. ED-MEDIA 2005. Montreal, Canada (2005)
17. VanLehn, K.: Student Modeling. In M. C. Polson and J. J. Richardson, editors, Foundations of Intelligent Tutoring Systems.Erlbaum, Hillsdale, NJ (1988)

A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems

Kai-min Chang, Joseph Beck, Jack Mostow, and Albert Corbett

Project LISTEN, School of Computer Science
Carnegie Mellon University, Pittsburgh PA 15213, USA
kkchang@cs.cmu.edu
http://www.cs.cmu.edu/~listen

Abstract. This paper describes an effort to model a student's changing knowledge state during skill acquisition. Dynamic Bayes Nets (DBNs) provide a powerful way to represent and reason about uncertainty in time series data, and are therefore well-suited to model student knowledge. Many general-purpose Bayes net packages have been implemented and distributed; however, constructing DBNs often involves complicated coding effort. To address this problem, we introduce a tool called BNT-SM. BNT-SM inputs a data set and a compact XML specification of a Bayes net model hypothesized by a researcher to describe causal relationships among student knowledge and observed behavior. BNT-SM generates and executes the code to train and test the model using the Bayes Net Toolbox [1]. Compared to the BNT code it outputs, BNT-SM reduces the number of lines of code required to use a DBN by a factor of 5. In addition to supporting more flexible models, we illustrate how to use BNT-SM to simulate Knowledge Tracing (KT) [2], an established technique for student modeling. The trained DBN does a better job of modeling and predicting student performance than the original KT code (Area Under Curve = 0.610 > 0.568), due to differences in how it estimates parameters.

1 Introduction

Intelligent Tutoring Systems (ITS) derive much of their power from having a student model [3] that describes the learner's proficiencies at various aspects of the domain to be learned. For example, the student model can be used to determine what feedback to give [4] or to have the students practice a particular skill until it is mastered [2]. Unfortunately, assessing student knowledge is difficult because 1) we can only infer student knowledge from observation of student performance, 2) student performance may not be a perfect reflection of student knowledge (e.g. performance is prone to guessing and slipping), and 3) the state of student knowledge changes over time.

Dynamic Bayes Nets (DBNs) [5] address these difficulties in assessing student knowledge by providing a powerful way to represent and reason about uncertainty in time series data [4, 6]. Section 2 demonstrates how DBNs can model

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 104–113, 2006.

student knowledge and performance. Unfortunately, constructing DBNs typically requires a complicated coding effort. Section 3 describes BNT-SM, a tool designed to reduce the cost of developing and evaluating Bayesian student models. Section 4 illustrates how to use BNT-SM to simulate Knowledge Tracing, an established technique for student modeling. Section 5 evaluates BNT-SM by comparing it to the original Knowledge Tracing code. Finally, section 6 summarizes BNT-SM's contributions and limitations.

2 Dynamic Bayes Nets and Student Modeling

We will begin by applying DBNs to an example. Suppose we want to assess the probability of a student knowing a skill. Since we cannot read the student's mind, we can only infer the knowledge state from a set of observable events, such as student performance (whether the student applies the skill correctly) and tutor intervention (whether the tutor gives assistance). We might have certain assumptions about how the latter two factors interact with student knowledge. For example, we might know that student performance on a task is affected by student knowledge and moreover, tutor intervention affects both student knowledge (by helping students to learn) and student performance (by scaffolding the student's current attempt without necessarily causing long-term learning). Figure 1 depicts such causal relationships in a conditional independence graph where:

K = student knowledge state (whether the student knows the skill or not)
H = tutor intervention (whether the tutor gives help or not)
C = observed student performance (whether correct or incorrect)

In DBNs, we usually model a latent variable (e.g. the unobserved student knowledge state) as a changing state in time series data. A state is represented as a node in the graph (e.g., K), while a time slice consists of a set of nodes that

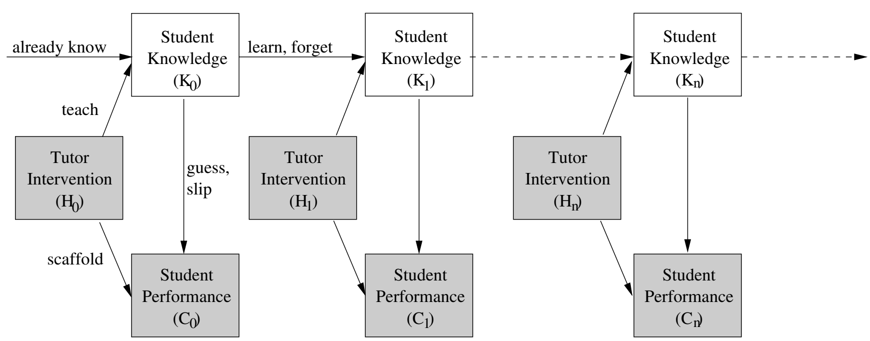


Fig. 1. Dynamic Bayes Net for a student model. Unshaded nodes represent latent variables; shaded nodes represent observed variables.

are modeled at a point of time (e.g. the set $\{K, H, C\}$). Normally, we assume that the parameters (or more precisely, the conditional probability distributions) in a DBN do not change over time; rather, what changes is the value of the latent variable. In the case of student modeling, we assume that student knowledge changes over time, but not the parameters of the model itself. Thanks to this assumption, we can model any number of time slices without estimating an infinite number of parameters.

The conditional independence graph is a graphical representation of the joint probability distribution specified in Equation 1. The joint probability formulation ensures that inferences (estimates of the values of latent variables) are mathematically sound and computationally efficient. This representation makes it easy to include certain kinds of parameters in the model. For example, the guess and slip rates for the vertical edges in Figure 1 are modeled as $P(C = true|K = false)$ and $P(C = false|K = true)$, respectively.

Given the conditional independence graph and the values of the observed variables (evidence), DBNs can infer the values of the latent variables. For example, the DBN uses the observed evidence to compute the posterior probability of the student knowing a skill by applying the Bayes rule as in Equation 2.

$$P(K, L, C) = P(H) * P(K|H) * P(C|K, H) \tag{1}$$

$$P(K = true|H, C) = \frac{P(H, K = true, C)}{P(H, K = true, C) + P(H, K = false, C)} \tag{2}$$

In summary, DBNs provide a powerful way to represent and reason about uncertainty in time series data, and are therefore well-suited to model student knowledge. Indeed, others have applied DBNs to student modeling [4, 6].

3 BNT-SM: Bayes Net Toolkit for Student Modeling

Many general-purpose Bayes net packages have been implemented and distributed. For example, BNT[1], BUGS[2] and GMTK[3] are three popular Bayes net packages that implement different inference and learning algorithms. We decided to use BNT because it supports the most inference algorithms (including junction tree, variable elimination, and Gibbs sampling) and learning algorithms (including both parameter learning and structure learning). More importantly, it supports DBNs, which are essential to student modeling. This software is distributed under the GNU Library General Public License and is implemented using Matlab, a widely used and powerful mathematical software package.

We have extended BNT to reduce the cost to develop and evaluate student models. We call this extension BNT-SM. A researcher can examine the factors that may affect student knowledge simply by specifying the hypothesized causal relationships in an XML specification file and providing empirical data.

[1] BNT is available at http://bnt.sourceforge.net

[2] BUGS is available at http://www.mrc-bsu.cam.ac.uk/bugs/

[3] GMTK is available at http://ssli.ee.washington.edu/˜bilmes/gmtk

By hiding most of the coding detail in constructing and training DBNs, BNT-SM lets the researcher focus on the modeling aspect of the problem and quickly experiment with alternative models. BNT-SM reduces the coding overhead by providing a simpler language to specify DBN than the generic BNT code. BNT-SM inputs a student model specified in an XML file. BNT-SM outputs BNT Matlab code to train and evaluate this model. Its XML input is shorter than its BNT output by a factor of at least 5, based on the model that we constructed.

4 Modeling Knowledge Tracing with BNT-SM

To provide a real example of using BNT-SM, we first introduce Knowledge Tracing, a student modeling technique that is related to DBNs.

4.1 Knowledge Tracing

Knowledge Tracing (KT) [2] is an established technique for student modeling and was first used in the ACT Programming Languages Tutor [2]. The goal of KT is to estimate the student's knowledge from his or her observed actions. At each successive opportunity to apply a skill, KT updates its estimated probability that the student knows the skill, based on the skill-specific learning and performance parameters and the observed student performance (evidence).

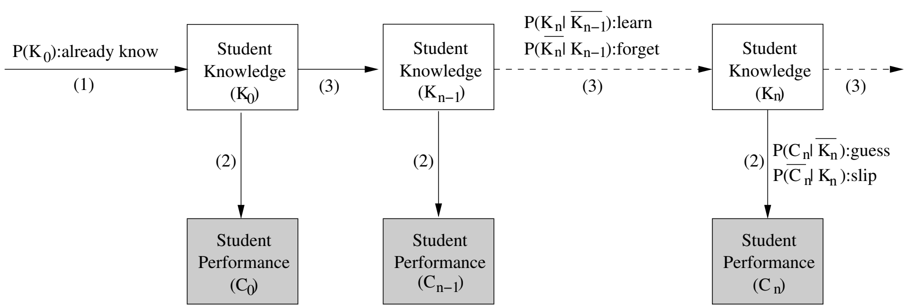


Fig. 2. A DBN that simulates Knowledge Tracing. (1), (2), and (3) refer to identifiers in the XML specification.

Reye [6] showed that KT is special case of a DBN which assumes parameters do not change across time slices. More specifically, the conditional independence graph of KT can be drawn as Figure 2. (We rename the original KT variables $L0$ and t as *already know* and *learn* for consistent naming with other sections.)

4.2 Example of Simulating KT with BNT-SM

Given that KT is a special case of DBNs, we now provide an example of simulating KT with BNT-SM. To use BNT-SM, a researcher needs to follow four steps:

1. Specify the data source in an XML specification[4].
2. Specify the network structure in an XML specification.
3. Specify and initialize parameters in an XML specification.
4. Call `RunBnet.m` in Matlab.

There are two sections in the XML specification file. First, we specify the data source as a tabulated file where each row represents a student's attempt to apply a skill. The columns of the data source file specify the values of the observed variables (e.g. `asr_corr` provides data for the *correct* node) according to empirical data and leave the values of the latent variables (e.g. *pknow*) as a question mark (so that the input and output files follow the same format).

```
user    utterance_start_time  skill  pknow  asr_corr
mTR4-6  2002-08-14_10:22:56   my     ?      1
mTR4-6  2002-08-14_10:22:56   cat    ?      1
...
```

After we have specified the data source, we specify the network structures. As Figure 2 shows, KT has two nodes per time slice. The first node, named *pknow*, represents student knowledge as a latent binary variable. The second node, named *correct*, represents student performance as an observed binary variable. *Pknow* has a `within` time slice connection to *correct* since we assume that student knowledge affects student performance. Moreover, *pknow* has a `between` time slice connection to *pknow* in the next time slice since we model student knowledge as a changing state in the time series data. The following XML (in two column format) instantiate the preceding discussion.

```
<nodes>
 <node>
  <name>     pknow    </name>
  <values>   2        </values>
  <type>     discrete</type>
  <observed>latent   </observed>
  <within>   correct </within>
  <between>  pknow   </between>
 </node>
 <node>
  <name>     correct </name>
  <values>   2       </values>
  <type>     discrete</type>
  <observed>asr_corr</observed>
  <within></within>
  <between></between>
 </node>
</nodes>
```

After we have specified the graphical model, we specify the parameters. For example, we define the parameter named *guess* as the conditional probability $P(C = true|K = false)$ (abbreviated `P2(T|F)`) and initialize it to a random value. `P1`, `P2`, and `P3` denote the conditional probability distributions labeled (1), (2), and (3) in Figure 2. Notice that some parameters are initialized to something like $1 - P1(T)$ such that probability distributions sum to one. Also, KT assumes there is no forgetting once a student learns a skill, so the $forget$ parameter is set to 0.

[4] For the detailed Document Type Definition of our XML specification file, please see http://www.cs.cmu.edu/~listen/BNT-SM.

```
<eclasses>
 <eclass>
  <id>     1 </id>
  <values> 2 </values>
  <eq>     P1(pknow) </eq>
  <cpd>
   <eq>    P1(T)   </eq>
   <init>  rand    </init>
   <param> already know </param>

   <eq>    P1(F)   </eq>
   <init>  1-P1(T) </init>
   <param> null    </param>
  </cpd>
 </eclass>

 <eclass>
  <id>     2 </id>
  <values> 4 </values>
  <eq> P2(correct|pknow) </eq>
  <cpd>
   <eq>    P2(T|F)   </eq>
   <init>  rand      </init>
   <param> guess     </param>

   <eq>    P2(F|T)   </eq>
   <init>  rand      </init>
   <param> slip      </param>

   <eq>    P2(F|F)   </eq>
   <init>  1-P2(T|F) </init>
   <param> null      </param>

   <eq>    P2(T|T)   </eq>
   <init>  1-P2(F|T) </init>
   <param> null      </param>
  </cpd>
 </eclass>

 <eclass>
  <id>     3 </id>
  <values> 4 </values>
  <eq> P3(pknow|pknow) </eq>
  <cpd>
   <eq>    P3(T|F)   </eq>
   <init>  rand      </init>
   <param> learn     </param>

   <eq>    P3(F|T)   </eq>
   <init>  0.000     </init>
   <param> forget    </param>

   <eq>    P3(F|F)   </eq>
   <init>  1-P3(T|F) </init>
   <param> null      </param>

   <eq>    P3(T|T)   </eq>
   <init>  1-P3(F|T) </init>
   <param> null      </param>
  </cpd>
 </eclass>
</eclasses>
```

After we have specified the graphical model in an XML specification file and provided the empirical data, we then call the `RunBnet.m` script in Matlab to start the training (estimating the parameters) and evaluating procedure (estimating the values of the latent variables). The student model's estimation of student knowledge is output to a file similar to the data source file, except that the question marks are replaced by the estimates. The resulting skill-specific parameters are output as follows:

skill	#students	#cases	already know	learn	guess	slip
my	179	4490	0.904	0.034	0.605	0.073
cat	178	1579	0.953	0.045	0.274	0.074
...						

To demonstrate BNT-SM's flexibility and ease of use, we point out that it is easy to extend our two node student model (KT) to the three node student model in Figure 1. To model the new tutor intervention node and its causal relationships to student knowledge and student performance, it suffices to add just 13 lines to the XML specification file. In contrast, KT code is limited to the simple 2-node model.

5 Evaluation of Model Fit

KT and DBNs estimate model parameters differently. The original KT code treats parameter estimation as a curve-fitting problem and uses a conjugate gradient search method by Powell [7]. In contrast, DBNs typically use Expectation Maximization to estimate the parameter values. We now perform an empirical comparison of the two parameter estimation procedures.

5.1 Data Collection

Our data came from 360 children between six and eight years old who used Project LISTEN's Reading Tutor [8] in the 2002-2003 school year. Over the course of the school year, these students read approximately 1.95 million words (as heard by the automatic speech recognizer). On average, students used the tutor for 8.5 hours.

During a session with the Reading Tutor, the tutor presented one sentence (or fragment) at a time for the student to read aloud. The student's speech was segmented into utterances delimited by silences. Each utterance was processed by the Automatic Speech Recognizer (ASR) and aligned against the sentence. This alignment scored each word of the sentence as either accepted (read correctly) or rejected (misread or omitted). For modeling purposes, this paper treats each English word as a separate skill.

5.2 Evaluation Metric

Since student knowledge is a latent variable that cannot be directly observed, we have no gold standard to compare against. Instead, we used the trained student model to predict whether the ASR would accept or reject a student's next reading of the word. An ROC (Receiver Operating Characteristic) curve measures the performance of a binary classifier by plotting the true positive rate against the false positive rate of varying decision threshold. The area under the ROC curve (AUC) is a reasonable performance metric for classifier systems, assuming no knowledge of the true ratio of misclassification costs.

To evaluate the model fits, we computed the correlation and the AUC between the student model's estimate of the student knowledge and the actual performance (as scored by the ASR).

5.3 Method

To determine which parameter estimation method provided a better model fit to student performance data, we first separated the training and testing set by splitting the students into two groups. The split was done by sorting the students according to their amount of Reading Tutor usage and alternately assigning students to the two sets. Then, we set up a DBN using BNT-SM to simulate KT and compared it against the curve fitting code of the original KT code.

The curve fitting algorithm is a general purpose routine for the minimization of a function in several variables. The algorithm implemented is a modification of conjugate gradient search by Powell [7]. The legacy code was part of the original KT code[5].

For DBN, we used the Expectation Maximization (EM) algorithm to optimize the data likelihood (i.e. the probability of observing our student performance data). EM is the standard algorithm used in the machine learning community to estimate DBN parameters when the structure is known and there exist latent variables. EM is guaranteed to converge to a local maximum on the likelihood surface. We used the junction tree algorithm for exact inference.

After training with each method, we cross validated by using the resulting student models to predict student performance in the testing set. We then computed the correlation and AUC between predicted student knowledge and observed performance.

5.4 Results

As Table 1 shows, EM outperforms the original KT's curve fitting code on both the correlation and the AUC evaluation metrics. Their two correlation coefficients are reliably different at $p < 0.01$. The values of model fit appear low because we are predicting individual student performance data rather than aggregated performance. It is difficult to predict a student's individual responses. To determine an upper bound on the best possible correlation, we did a cheating experiment that can peek at the future data when it makes a wrong prediction. The cheating experiment further assumed monotonicity constraints. That is, a correct response always increases the student model's estimate of student knowledge, whereas an incorrect response decreases the estimate. The cheating experiment revealed that the maximum correlation of any model that obeyed monotonicity constraints was only 0.5. Note that this maximum performance requires peeking at the data to be predicted and is not necessarily attainable by any actual model.

As Table 2 shows, EM estimates much higher *already know* and lower *learn* parameters than curve fitting. EM's *already know* estimate of 0.68 is more plausible than curve fitting's estimate of 0.33. The 20 most frequent words of English account for about 33% of text words. Since we are using the weighted average for parameter values (weighted by the number of times students encountered the word in the Reading Tutor), an initial knowledge of 0.33 would be comparable to six through ten year old knowing only these 20 words–an unlikely proposition at best. EM's estimate of 0.68 would be comparable to students knowing only the 431 most frequent words in English, which is much more believable.

[5] Source code is courtesy of Albert Corbett, http://www.cs.cmu.edu/˜rsbaker/curvefit.tar.gz

Table 1. KT vs DBNs Model Fit

Method	Correlation	AUC
Curve-fitting	0.07	0.568
DBN's EM	0.16	0.610

Table 2. KT vs DBNs Parameters

Method	already know	learn	guess	slip
Curve-fitting	0.33	0.19	0.72	0.07
DBN's EM	0.68	0.14	0.64	0.07

6 Conclusion and Future Work

In summary, this paper describes how DBNs provide a powerful way to model student knowledge in an Intelligent Tutoring System. We introduce a tool called BNT-SM that helps training and evaluating DBNs. The main advantages of BNT-SM are that it is 1) flexible, 2) easy to use, and 3) provides a better model fit and a more plausible model parameters, at least on our data set. We invite the student modeling community to download BNT-SM at
http://www.cs.cmu.edu/˜listen/BNT-SM.

Currently, BNT-SM handles Bayes nets with any number of latent and observed variables, but the variables are limited to discrete values. Moreover, there exist some restrictions on the possible causal relationships that a researcher may model. For instance, BNT-SM does not allow links that go backward in time or that skip forward past the next time slice. Although generality was a top priority when we designed BNT-SM, we have tested it on only three simple networks (two of them discussed in this paper), so it may have limitations of which we are unaware. Thus, we encourage researchers to try out BNT-SM and provide valuable feedback for us to improve its generality.

As future work, we wish to explore the generality of BNT-SM by providing XML specification files for various popular student models. Moreover, we wish to investigate more complex models such as models with continuous variables, as well as hierarchical models. Hierarchical Bayes nets allow researchers to model more than one level of factors, where a higher level factor (such as overall reading proficiency) affects multiple lower level factors (such as individual words).

Another issue that we would like to address in the future is the computational complexity of training the DBN. Since we're using an exact inference engine (junction tree), training time is rather long. As it currently stands, EM takes 25 hours to estimate parameters for the DBN that simulates KT, while the curve-fitting code in the original KT code takes only 2 hours. There are several ways to speed up training. For instance, we can use approximate inference algorithms that trade off accuracy for efficiency. BNT already implements several approximation algorithms , including Gibbs Sampling and variational methods. Another way to speed up the training process is to formulate the network using some specialized, well-studied formalism such as Hidden Markov Models, for which more efficient algorithms exist.

Acknowledgement. This work was supported by the National Science Foundation, ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation or the official policies, either expressed or implied, of the sponsors or of the United States Government. We also acknowledge members of Project LISTEN who contributed to the design and development of the Reading Tutor, and the schools that used the tutor.

References

1. Murphy, K.: Bayes Net Toolbox for Matlab. <http://bnt.sourceforge.net> Accessed 2006 March 21.
2. Corbett, A., Anderson, J.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction **4** (1995) 253-278
3. Woolf, B.: Artificial Intelligence in Education. Encyclopedia of Artificial Intelligence. John Wiley & Sons: New York (1992) 434-444
4. Conati, C., Gertner, A., VanLehn, K.: Using Bayesian networks to manage uncertainty in student modeling. User Modeling and User-Adapted Interaction **12** (2002) 371-417
5. Dean, T., Kanazawa, K.: A model for reasoning about persistence and causation. International Journal of Computational Intelligence **5** (1989) 142-150
6. Reye, J.: Student modeling based on belief networks. International Journal of Artificial Intelligence in Education **14** (2004) 1-33
7. Powell, M.: An efficient method for finding the minimum of a function in several variables without calculating derivatives. Computer Journal **7** (1964) 155-162
8. Mostow, J., Aist, G.: Evaluating tutors that listen: An overview of Project LISTEN. Smart Machines in Education. K. Forbus and P. Feltovich, Editors, MIT/AAA Press: Menlo Park, CA. (2001) 169-234

A Comparison of Decision-Theoretic, Fixed-Policy and Random Tutorial Action Selection

R. Charles Murray[1] and Kurt VanLehn[2]

[1] Carnegie Learning, Inc., Frick Building, 20th Floor,
437 Grant St., Pittsburgh, PA 15219
cmurray@carnegielearning.com
[2]Computer Science Department & Learning Research and Development Center
University of Pittsburgh, Pittsburgh, PA 15260
vanlehn@cs.pitt.edu

Abstract. *DT Tutor* (DT), an ITS that uses decision theory to select tutorial actions, was compared with both a Fixed-Policy Tutor (FT) and a Random Tutor (RT). The tutors were identical except for the method they used to select tutorial actions: FT employed a common fixed policy while RT selected randomly from relevant actions. This was the first comparison of a decision-theoretic tutor with a non-trivial competitor (FT). In a two-phase study, first DT's probabilities were learned from a training set of student interactions with RT. Then a panel of judges rated the actions that RT took along with the actions that DT and FT would have taken in identical situations. DT was rated higher than RT and also higher than FT both overall and for all subsets of scenarios except help requests, for which DT's and FT's ratings were equivalent.

1 Introduction

Intelligent tutoring systems (ITSs) that coach students as they attempt tasks often emulate the turn taking observed in human tutorial dialog [1; 2]. The tutor's main task can be seen as deciding what action to take on its turn, or *tutorial action selection*. Selecting tutorial actions involves inherent difficulties.

A significant source of difficulty is that the tutor is uncertain about the student's internal state because it is unobservable and changes over time (e.g., as the student learns). Furthermore, the tutor is uncertain about the effects of the tutor's actions on the student. Many ITSs [see, e.g., 3] have modeled the tutor's uncertainty in terms of probability using Bayesian techniques [4] for sound yet relatively efficient inference.

Another difficulty is that just what constitutes effective tutorial action depends upon the tutor's objectives and priorities. The tutor's objectives are likely to include increasing the student's knowledge, helping the student complete tasks, bolstering the student's affective state, and being a cooperative discourse partner. It may not be possible to maximize attainment of all objectives at once, in which case the effectiveness of the tutorial action alternatives depends upon the tutor's priorities. Tutors must often strike a "delicate balance" among multiple competing objectives [5, p.280; 6; 7].

Decision theory extends probability theory by considering, in addition to uncertainty, the decision-maker's objectives and priorities as a rational basis for making decisions [8]. *DT Tutor* (DT) [9] uses decision-theory to decide the tutor's actions.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 114 – 123, 2006.

For each alternative, DT computes (1) the probability of every possible outcome of that tutorial action, (2) the utility of each possible outcome relative to the tutor's objectives and priorities, and then (3) the alternative's *expected utility* by weighting the utility of each possible outcome by the probability that it will occur. DT then selects the tutorial action with maximum expected utility. This approach unifies considerations regarding (1) the tutor's uncertain beliefs about the changing tutorial state and the tutor's effects upon it, and (2) the tutor's objectives and priorities.

One advantage of a decision-theoretic approach is the capability to balance multiple tutorial objectives in a principled way. DT leverages this capability by simultaneously considering the student's knowledge, focus of attention, and affective state, along with joint task progress and the student-tutor discourse state. Another advantage is that by looking ahead to anticipate student difficulties and the influence of the tutor's actions, DT can provide *proactive* help to attempt to prevent student difficulties in addition to the *reactive* help that most ITSs provide.

While many ITSs have used Bayesian networks for reasoning under uncertainty [3], decision-theoretic approaches remain rare, and comparisons of decision-theoretic approaches with non-trivial competitors are rarer still. CAPIT [10] and iTutor [11] appear to be the only other decision-theoretic tutors that have been implemented and evaluated. However, these tutors were compared only with no tutoring at all [10], with consulting the teacher when required [11], and with randomized action selection [10]. Our work does not directly assess effectiveness with students, but it does compare decision-theoretic tutoring against a higher standard: a Fixed-Policy Tutor (FT) that selects tutorial actions by emulating the fixed policies employed by successful tutors such as Andes1 [12] and the Cognitive Tutors [13], which are theory-based [14], widely-used and highly effective [15]. DT and FT were also compared with a Random Tutor (RT), which selects randomly from among relevant tutorial actions.

2 The Three Tutors: DT, FT and RT

DT, FT and RT shared the same user interface and help messages. All of the tutors gave immediate flag feedback by highlighting correct responses in green and errors in red. The only difference between the tutors was the method they used to select from among the same tutorial action alternatives, which consisted of deciding whether to provide a help message, and if so, which message to provide. The differences in their performance were thus due solely to their action selection methods.

This study encompassed three different types of help situations. Most ITSs provide help (1) in response to help requests and (2) after some number of errors. Also included were (3) *step starts*, which are opportunities for the tutor to provide proactive help at the start of a step before the student has reached an impasse or made an error. Few ITSs provide help at step start, but human tutors sometimes do [e.g., 6; 16]. Help provided at step start or after an error is proactive since the student has not asked for help and may not even want it. Responses to help requests are reactive.

The tutors could choose either to provide no help message (a *null* help message) or to provide one of four different kinds of help messages: *prompt, hint, teach*, or *do*. The *prompt* message pointed out pertinent information that was already available in the interface without providing any new information. The *hint* message provided partial information about the step – not enough to teach the student how to do the step but perhaps enough to either remind the student how to do the step or help the student figure it out. The *teach* message provided all the information that the student needed to understand the domain rule related to the step, including at least one example, and thus to help the student complete the step correctly by learning the rule. The *do* message told the student exactly what to do for the current step (e.g., what text to enter) without teaching anything about the related rule. These help messages are ranked in order of increasing explicitness about what to do for the current step, from *prompt* (the least explicit,), through *hint*, *teach*, and *do* (the most explicit).

RT randomly provided help relevant to the current step as follows: For proactive help opportunities, RT decided randomly whether to provide proactive help. For reactive help opportunities, RT always provided help. When RT decided to provide help, it decided in advance a random order for the four types of help messages and then returned help types in that order, repeating the order cyclically if necessary.

FT, consistent with Andes1 [12] and the Cognitive Tutors [13], always provided help after help requests, never provided help for step starts, and provided help after *n* errors (two in this study). When FT provided help, it followed a strong *successive explicitness* constraint: It always provided a help message that was minimally more explicit than any help already provided. In other words, first it provided a *prompt* message, then *hint*, then *teach*, before bottoming out with *do*. If the student continued to request help after that, the *do* message was repeated.

DT's help selection cannot be described in terms of a simple policy because it simultaneously considers multiple aspects of the tutorial state. However, one of these aspects, the student-tutor discourse state (modeled so that DT can be a cooperative discourse partner) did constrain DT's help selections. The two discourse constraints that DT followed were (1) to always provide help for help requests and (2) a weak successive explicitness constraint: It never provided less explicit help than had already been provided (so, e.g., if the student requested more help after receiving a *teach* message, the student wouldn't be disappointed with a *prompt* message). Constraint (1) is the same as FT and RT. Constraint (2), weak successive explicitness, is different than FT in that DT does not have to select the help message that is *minimally* more explicit than any help already provided. This means, for instance, that DT can provide a *teach* message as the first help provided, or progress directly from a *prompt* message to a *teach* message. The other difference between DT and FT's help selection is that DT always considers providing proactive help. It should be noted that DT can easily be configured to emulate FT (and therefore Andes1 and the Cognitive Tutors, among others) by considering only the discourse state (giving it a utility of 1 while giving all other aspects of the tutorial state a utility of 0) and increasing DT's discourse constraints to (1) never provide help for step starts, (2) provide help after errors only after the *n*th error, and (3) follow a strong successive explicitness constraint.

Because DT and FT both followed a successive explicitness constraint, a subset of the help opportunities were especially relevant for revealing differences between the help selection strategies of DT and FT. These were *first-message-opportunity* scenarios (FMOs), in which the tutor has the opportunity to select the first help message to be displayed for the current step. For FMOs, which occurred in over half the 350 scenarios in the study described below, DT had free reign over which help message to select (if any) while FT adhered to its fixed policy. First-message-opportunity scenarios were sometimes partitioned according to student performance on the pretest problem that corresponded to the rule required to complete the problem step: *pretest-wrong* and *pretest-right*. The idea behind this partitioning is that students who get a pretest problem wrong are more likely than those who get it right to need help during tutoring on steps that require knowledge of the rule tested by the pretest problem. This is by no means a perfect test – e.g., the student might have merely slipped on the pretest problem, or the student might have learned the rule since the pretest – but one advantage is that it does not require subjective judgment by the experimenter. It must be noted that DT was not given information about the pretest performance of students in the test set. However, DT could glean information about the likelihood that a particular student in the test set knew a rule in two ways: (1) by the percentage of the training set students who got the corresponding pretest problem correct (learned during phase I of the study as prior probabilities), and (2) by the student's performance during tutoring on steps related to the rule.

3 Study Design

A two-phase study design was employed. In the first phase, *data collection and tuning*, 60 students took a pretest, solved the same five multi-step calculus problems using RT, and then took a posttest. The students used RT so that we could collect data about the effects of *individual* tutorial actions while statistically controlling for the effects of *sequences* of tutorial actions by randomizing over the sequences in which the individual actions occurred. Students were allowed as much time as they needed to complete the problems and most took about an hour. The student data was partitioned into training and test sets of 30 students, which were matched according to pretest scores. Logged student-tutor interactions from the training set, along with pre- and posttest performance, were used to learn probabilities about student knowledge, student behavior, and the effects of tutorial actions. The data collection and tuning phase is described in detail elsewhere [17]. This paper focuses on the *assessment* phase.

During the assessment phase, we replayed logged student-tutor interactions from the test set while recording the responses that DT and FT would provide in the same tutorial situations. When the actions selected by RT and DT differed, the action selected by RT was replayed in order to preserve the fidelity of the replay, and DT updated its model of the tutorial state to include the action actually provided by RT. A similar process was undertaken to record the actions that FT would have taken for the same situations. A panel of judges then rated the actions selected by RT, FT and DT in a large sample of test set situations.

While this study design cannot provide conclusive information about the bottom line – which tutor is most effective with students – it has other advantages. First, it provided data for learning many of DT's key probabilities. Second, it allowed us to compare the action selections of different tutoring approaches in identical situations. Third, it can provide information that is much more detailed than the bottom line about what makes the tutors' actions effective or not in particular situations [18], information that can be used to improve not only DT but other tutors as well. Advantages two and three allowed us to decrease the grain size of the analysis from the student to the scenario – i.e., to help opportunities. With an estimated effect size for DT over FT of 0.2 standard deviations, we needed about 320 samples from each of three conditions (RT, FT and DT) using the conventional parameters of $\alpha = .05$ and $\beta = .20$ [19], and 960 students was more than we could afford. Reducing the grain size to the scenario allowed us to use many fewer students.

4 The Comparative Assessment

4.1 Subjects

Three paid judges were recruited from among graduate mathematics students who had extensive experience tutoring calculus as well as other mathematics to college and high school students. These judges were considered skilled, although not necessarily expert, because of their extensive mathematical knowledge and tutoring experience.

4.2 Materials

For each scenario to be assessed, judges were given a detailed printed description that included, among other things:

- A screen shot showing the student interface at the moment of the scenario.
- A description of the scenario, including whether the student had just requested help or made an error, along with the correct entry for the current step.
- The student's current number of correct entries, errors and help requests as a general indicator of the student's problem-solving performance and help usage.
- The student's performance on the pretest problem related to the current step.
- *Relevant Action History*: Previous student-tutor interactions on (1) any previous steps that use the rule related to the current step, and (2) the current step.

Each scenario listed the five possible tutorial responses in random order (the help messages corresponding to help types *prompt*, *hint*, *teach* and *do*, plus "no message" for the *null* response) and the judges rated each on a scale of 1 (worst) to 5 (best). The judges also chose their preferred response for each scenario.

Scenario Types and Stratified Sampling. For assessing the performance of the different tutorial action selection methods, 350 scenarios were to be selected from the test set. The intention was to select the scenarios randomly so as not to introduce any bias. However, a completely random selection would have produced a highly skewed sampling among help requests, errors and step starts. Of 5009 scenarios in the test set, 57% were step starts, 35% were errors, and 8% were help requests. This was just the opposite of what was desired for assessing help selection, for which help provided for help requests is arguably most important, help provided for errors is probably next most important, and it is debatable whether help should be provided for step starts at all. With a completely random distribution, the judges' ratings of the tutors would be dominated by their ratings for step start scenarios and only weakly influenced by their ratings for help request scenarios. Therefore, a stratified sample was selected with the sample for each stratum randomly selected from among all the scenarios in that stratum: 175 help requests, 100 errors and 75 step starts.

One additional criteria was employed for selecting scenarios: Since RT selected relevant actions randomly, it might for a specific step select, say, *do* help (the most explicit), and then if the student was unsuccessful, select *prompt* (the least explicit help). Such sequences of help messages violated even the weaker successive explicitness constraint. It was unclear just how DT or FT should respond following sequences of help messages that violated their own constraints. A related concern was that it was unclear just how such seemingly odd (indeed, random) sequences of help messages would affect the judges' intuitions about what kind of help to provide next. Therefore, scenarios whose Relevant Action History included sequences of tutorial actions that violated the weaker successive explicitness constraint were excluded from the sample.

4.3 Procedure

The judges rated all possible responses for 350 scenarios. Note that with this design it is possible to use the same judges' ratings to assess the tutorial action selections of still more tutors, or updated versions of the same tutors, as long as they use the same student interface and select from the same pool of help messages. The judges were told that they were rating scenarios in order to provide information about what help messages would be best to provide for various situations. They had no idea which tutor provided which responses or that their ratings would be used to compare tutors.

For comparing the tutors, we constructed composite judges' ratings to better represent the population of skilled tutors. Our goal was to discount ratings that were outside the norm without excluding any ratings. To this end, we used the median rating for each response. The median discounts the effect of the magnitude of outlying ratings while still taking their existence into account. With outlying ratings for individual responses thus discounted, composite ratings for sets of responses were computed as the mean of the median ratings for each response.

Ratings for All Three Tutors by Scenario Type. Table 1 displays results of paired-sample t-tests comparing RT vs. DT and FT vs. RT for all scenarios and for each scenario type, along with effect sizes and mean composite ratings. Effect sizes were calculated as the difference in means divided by the standard deviation of the control group: either RT or FT as applicable in their comparisons with DT.

As the table shows, the judges' ratings for DT were higher than their ratings for RT overall and for help requests, errors and first message opportunities, significant at level $p<.01$ with effect sizes ranging from .33 to .49. Only for step start scenarios was DT not rated significantly higher than RT after the Bonferroni correction for multiple comparisons. However, the significance before the Bonferroni correction was $p=.012$ and the Bonferroni correction is known to be very conservative to protect against Type I errors. The effect size for step starts was still a healthy .30.

Table 1. Tutor x Scenario Type, paired t-tests: RT vs. DT, FT vs. DT

Comparison			df	t	Sig.	Bonferroni Sig.*	Effect Size
RT vs. DT	RT Mean	DT Mean					
All	2.94	3.43	349	5.746	<.001	<.01	.35
Help Req	3.23	3.66	174	3.937	<.001	<.01	.33
Errors	2.31	2.95	99	3.324	.001	.010	.49
Step Starts	3.11	3.55	74	2.572	.012	.120	.30
FMOs	2.99	3.54	187	5.057	<.001	<.01	.40
FT vs. DT	FT Mean	DT Mean					
All	3.08	3.43	349	5.251	<.001	<.01	.24
Help Req	3.59	3.66	174	1.078	.282	1.0	.06
Errors	2.10	2.95	99	4.693	<.001	<.01	.61
Step Starts	3.19	3.55	74	3.222	.002	.020	.22
FMOs	3.12	3.54	187	4.351	<.001	<.01	.28

* Significance with Bonferroni correction for 10 t-tests (Sig. x 10)

The judges' ratings for DT were higher than their ratings for FT overall and for the scenario types of errors, step starts and first message opportunities, all with significance p=.02 or less and with effect sizes ranging from .22 to .61. For help requests, however, DT, with mean 3.66, and FT, with mean 3.59, were rated approximately equivalently with a .06 effect size and a significance level (with Bonferroni correction) of approximately p=1.0.

DT vs. FT for Help Requests. Since DT's and FT's ratings were approximately the same for help requests, one might expect that DT and FT selected mostly the same tutorial responses in the same situations. However, their patterns of responses were significantly different, with a Pearson's chi-square test of association of $\chi^2(3)=60.8$, $p<.001$. FT and DT also behaved significantly differently for FMO help requests, for which FT always selected the *prompt* response according to its fixed-policy. DT's response selections varied: For the pretest-wrong scenarios, DT selected *prompt* only 34% of the time and *teach* 66% of the time, receiving a mean composite rating of 4.00 while FT received a mean composite rating of 3.55. This difference was significant, $t(28) = 2.218$, $p=.035$. For pretest-right scenarios, DT selected *prompt* slightly more often, 44% of the time, and received a mean composite rating of 3.80 while FT's responses (always *prompt*) received a higher mean composite rating, 4.02, although this difference was not quite significant, $t(44) = 1.634$, $p=.109$. Apparently, the judges generally preferred the *teach* response when the student was more likely to need explicit help and the *prompt* response when the student was less likely to need explicit help. DT adjusted its response selections according to the same preference structure but did not adjust them enough when the student was less likely to need explicit help.

Table 2. FMO scenarios, paired t-tests: FT vs. DT

Comparison	FT	DT	df	t	Sig.	Bonferroni Sig.*	Effect Size
Pretest wrong	2.51	3.24	74	4.606	p<.001	p<.002	.55
Pretest right	3.53	3.73	112	1.776	p=.079	p=.158	.14

* Significance with Bonferroni correction for 2 t-tests (Sig. x 2)

DT vs. FT for Errors. FT's ratings for errors were significantly lower than DT's because it always selects a *null* response the first time the student makes an error. All of the judges gave low ratings to *null* responses after errors. 68 out of the 100 error scenarios involved the student's first error, so FT received a low rating for most of the error scenarios. For first errors, DT's mean composite rating, 2.88, was significantly higher than FT's rating of 1.35, t(67)=8.516, p<.001, with a large effect size of 2.58. On the 32 error scenarios that did not involve the student's first error, FT, with a mean composite rating of 3.69, was rated higher than DT, which had a mean of 3.09, t(31) = 2.094, p=.044. This was in turn due to DT replying *null* on 13 of these 32 scenarios, for which it received a mean rating of only 1.23 compared to FT's mean of 3.10. The bottom line is that our judges did not like *null* responses to errors.

DT vs. FT for Step Starts. As with errors, FT received lower ratings than DT for step starts because of *null* responses. Per its fixed policy, FT always selected *null* responses for step starts. DT did not reply *null* on 21 of the 75 step start scenarios, and for these, DT's mean composite rating, 3.67, was significantly higher than FT's mean composite rating of 2.38, t(20) = 3.959, p=.001, effect size .92. DT's significant advantage in ratings when it did not reply *null* led to a significant advantage over FT in ratings for step scenarios overall, 3.55 versus 3.19, p=.020.

DT vs. FT for First-Message-Opportunity Scenarios (FMOs). FT always provided either the *null* or the *prompt* response for FMO scenarios, while DT also included the *teach* response and varied its responses according to the likelihood that the student needed explicit help. These differences paid off as DT was rated significantly higher than FT for FMOs, p<.01, effect size .28. A closer look shows that DT was nominally rated more highly than FT both for pretest-wrong and for pretest-right scenarios, as shown in Table 2. For pretest-wrong scenarios, DT's mean composite rating is significantly higher, p<.01, with effect size .55. For pretest-right scenarios, DT's mean composite rating is not significantly higher after the Bonferroni correction, p=.158.

5 Discussion

Fixed-policy tutors such as FT use a time-tested and proven, even theoretically-based [13] policy for selecting the response type for tutorial actions. However, this policy considers only (1) whether the student has just made a help request or the *n*th error, and (2) the most recent response type for the current step. The result is response selections that are all the same regardless of other attributes of the tutorial situation.

FT emulates the policies of Andes1 and the Cognitive Tutors, which follow a strong successive explicitness constraint and volunteer help only after *n* errors [13; 20]. This policy was based on psychological research showing that students remember material better when they generate it themselves. However, even the architects of the Cognitive Tutors and the theory behind them admit that "these may not be the best choices" since, for example, "[s]ome students stubbornly refuse to seek help even when they need it" and "students are often annoyed with the vague initial messages and decide there is no point in using the help facility at all" [13, p. 199]. Once students begin clicking past vague initial help messages, as many as 82-89% of students click all the way through to bottom-out help [20].

DT, like human tutors [1; 5; 6], considers multiple tutorial state attributes to decide when and how to provide help. These attributes include the student's knowledge, affective state and focus of attention, along with task progress and the discourse state. DT's resulting sensitivity to the tutorial state was demonstrated, for instance, in its responses to first-message-opportunities, for which not only did DT's responses vary significantly for pretest-right versus pretest-wrong scenarios, but its ratings were higher for both, significantly so for pretest-wrong scenarios. DT's greater sensitivity paid off in generally higher ratings from the judges.

A major reason why DT surpassed FT in the judges' ratings was DT's use of proactive help, which FT never provides for step start and first error scenarios. Proactive help when a student would otherwise flounder can save time, prevent confusion, provide valuable information at a time when the student is prepared and motivated to learn it, and avoid the negative affective consequences of frustration and failure.

DT's consideration of multiple tutorial state attributes and the variability of its responses is more like human tutors, for whom the timing of feedback appears "to depend critically on the consequences of the particular error or impasse encountered" [5, p.283]. When considering proactive help, human tutors "sometimes seek to forestall errors, sometimes intervene as soon as errors occur; at other times they may allow errors to occur" [6, p.85]. Indeed, the very effectiveness of human tutorial help "may arise because of the contingency of feedback style and content" [1, p. 346].

The bottom line in choosing a method for selecting tutorial actions is which technology delivers the desired capabilities for the least cost (in time and money). If the desired behavior of the tutor is unambiguously defined and only simple capabilities are required, fixed policy is best, no contest, because of its ease of implementation. However, as more sensitivity is required, as the need for flexibility increases, or as the desired behavior of the tutor becomes more ambiguous, decision-theoretic tutoring becomes more attractive. Decision theory can enable a tutor to respond in a principled way to an unlimited variety of situations – even unanticipated situations – without having to come up with a fixed-policy for every combination of uncertain beliefs.

References

1. Merrill, D.C., B.J. Reiser, S.K. Merrill, and S. Landes (1995). Tutoring: Guided learning by doing. *Cognition and Instruction*, 13(3): 315-372.
2. Graesser, A.C., N.K. Person, and J.P. Magliano (1995). Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*, 9: 495-522.
3. Jameson, A. (1996). Numerical uncertainty management in user and student modeling: An overview of systems and issues. *User Modeling and User-Adapted Interaction*, 5(3-4): 193-251.

4. Pearl, J. (1988). Probabilistic reasoning in intelligent systems: Networks of plausible inference. San Mateo, CA: Morgan-Kaufmann.
5. Merrill, D.C., B.J. Reiser, M. Ranney, and J.G. Trafton (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. *The Journal of the Learning Sciences*, 2(3): 277-306.
6. Lepper, M.R., M. Woolverton, D.L. Mumme, and J.-L. Gurtner (1993). *Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors*, in Computers as Cognitive Tools, S.P. Lajoie and S.J. Derry, editors. Erlbaum. p. 75-105.
7. Reye, J. (1995). A goal-centred architecture for intelligent tutoring systems. In J. Greer (eds.), *7th World Conference on Artificial Intelligence in Education*, p. 307-314.
8. Russell, S. and P. Norvig (1995). Artificial Intelligence: A Modern Approach. Englewood Cliffs, NJ: Prentice Hall.
9. Murray, R.C., K. VanLehn, and J. Mostow (2004). Looking ahead to select tutorial actions: A decision-theoretic approach. *International Journal of Artificial Intelligence in Education*, 14(3-4): 235-278.
10. Mayo, M. and A. Mitrovic (2001). Optimising ITS behaviour with Bayesian networks and decision theory. *International Journal of Artificial Intelligence in Education*, 12: 124-153.
11. Pek, P.-K. (2003). Decision-Theoretic Intelligent Tutoring System. PhD dissertation, National University of Singapore, Department of Industrial & Systems Engineering. ftp://ftp.medcomp.comp.nus.edu.sg/pub/pohkl/pekpk-thesis-2003.pdf
12. Conati, C., A. Gertner, and K. VanLehn (2002). Using Bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, 12(4): 371-417.
13. Anderson, J.R., A.T. Corbett, K.R. Koedinger, and R. Pelletier (1995). Cognitive Tutors: Lessons Learned. *The Journal of the Learning Sciences*, 4(2): 167-207.
14. Anderson, J.R. and C. Lebiere (1998). The atomic components of thought. NJ: Erlbaum.
15. Koedinger, K.R., J.R. Anderson, W.H. Hadley, and M.A. Mark (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8: 30-43.
16. Fox, B.A. (1993). The Human Tutorial Dialogue Project: Issues in the Design of Instructional Systems. Hillsdale, NJ: Lawrence Erlbaum Associates.
17. Murray, R.C. (2005). An evaluation of decision-theoretic tutorial action selection. PhD dissertation, University of Pittsburgh, Intelligent Systems Program. http://etd.library.pitt.edu/ETD/available/etd-08182005-131235/
18. Mostow, J., C. Huang, and B. Tobin (2001). Pause the Video: Quick but quantitative expert evaluation of tutorial choices in a Reading Tutor that listens. In J.D. Moore, C.L. Redfield, and W.L. Johnson (eds.), *10th International Conference on Artificial Intelligence in Education*, p. 343-353.
19. Cohen, J. (1988). Statistical Power Analysis for the Behavioral Sciences. Erlbaum.
20. Aleven, V. and K.R. Koedinger (2000). Limitations of student control: Do students know when they need help? In G. Gauthier, C. Frasson, and K. VanLehn (eds.), *Intelligent Tutoring Systems, 5th International Conference, ITS 2000*, p. 292-303.

Evaluation of a System That Generates Word Problems Through Interactions with a User

Kazuaki Kojima and Kazuhisa Miwa

Graduate School of Information Science, Nagoya University
464-8601 Furo-cho, Tikusa-ku, Nagoya city, Japan
{koj, miwa}@cog.human.nagoya-u.ac.jp

Abstract. In mathematical learning, it is important to give learners a number of problems that have various features in both surface problem situations and deep mathematical solution structures. In this study, we implement a system that generates various word problems by using episodes, which are knowledge regarded as cases of problem generation. Our system interacts with a teacher as a user to acquire the common knowledge needed to generate word problems. We performed experimental evaluations to verify problem generation by our system, with the results indicating that our system can successfully expand the variety of problems from the initial ones stored in the system. We also found that our system needs interactions with a knowledgeable user because novice users cannot necessarily provide the system with effective knowledge.

1 Introduction

In general mathematical learning, a teacher first presents a solution method by example problems, and students then learn by solving problems regarded as analogical instances. Since a number of problems is needed in such mathematical learning, teachers generally use multiple workbooks to provide their students with a variety of problems. Thus, the production of problems is considered as a crucial task in research on educational systems.

Mathematical problems can be categorized into several types such as word problems and geometry problems. Word problems, one of the most typical types of problems in mathematical learning, have problem situations denoting contextual settings expressed in texts such as "*purchase of goods*" and "*Transfer by vehicles*." In mathematical learning, such problem situations are as important as mathematical structures of problems. It has been pointed out that problem solving by novice students is strongly influenced by superficial characteristics of problems such as problem situations (e.g., see [1,8]), and it has been well recognized as crucial to provide students with problems that have various features in both surface problem situations and deep mathematical structures [2]. However, automatic generation of word problems is extremely difficult since natural language processing and generation, as well as commonsense reasoning, are needed.

Several systems that generate problems in mathematical learning have already been implemented. For example, Kanenishi's WBT system [5] generates calculation problems in qualifying examinations for the Information-Technology Promotion Agency

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 124 – 133, 2006.

Japan (IPAJ) by using given domain knowledge. Shirota [9] developed a system that automatically generates Web-based learning materials based on problem definitions in the domain of economical optimization problems. Martin et al. [7] implemented automatic problem generation by using constraint-based modeling that represents student and domain models in their SQL tutoring system. Problem generation by these systems aims to reduce costs in production of learning materials; however, since problem situations are not crucial in their domains, the systems do not need to generate various problem texts.

Although many programs pose word problems, most only provide the same type of problems by changing the parameters included in the problem texts. Hirashima et al. [4] developed a Web-based environment for authoring and sharing knowledge base of word problems. Their system represents basic problems using a schema and creates new problems that have new problem situations and solutions by replacing concepts in the schema and giving operational knowledge to the schema. However, that system does not automatically generate new problems without manual operation by a user. Takano's WBT system [10] also poses word problems; however, her system can generate only a limited variety of problems composed of solutions and problem situations prepared in the system. Therefore, it is an important challenge to implement a system that can generate various word problems controlled by both features of problem texts and structures of solutions.

In this study, we implement a system that generates various mathematical word problems based on control of problem situations and their solutions. Our system aims to expand the variety of problems by generating new ones from the initial problems stored in the system, but to generate word problems, some technical issues such as commonsense reasoning must be overcome. To do these issues, we adopt the following two approaches: (1) our system forms and uses episodes of problem generation, and (2) our system acquires common knowledge through interactions with a teacher as a user. Each episode is knowledge regarded as a case where a new problem is generated from an example one, and it contains these two problems and relationships between them. Thus, an episode can be considered as a meta case comprising two problems as cases.

2 A System That Generates Word Problems

Our system stores initial mathematical problems and generates new problems from them. First, it acquires problem data and common knowledge through interactions with a user, which are then used in problem generation. Our study focuses on problem generation for a teacher as the user, not on support for students' learning. Of course, in the future our system will support learning by presenting problems to students after it has acquired sufficient knowledge.

In the current study, we select a problem domain of word problems, each of which contains information about two objects and their attribute values, and is solved by simultaneous equations. These problems are used in mathematical education in Japanese middle schools.

2.1 Basic Idea of Problem Generation

Our system forms episodes and generates new problems by using the episodes from initial ones stored in the system. Each episode is knowledge comprising a single base example problem (*base*) and a single new analogical instance (*new instance*), which is regarded as a case where a *new instance* is generated from a *base*. Relationships between a *base* and a *new instance* are used to generate new problems. In problem generation, a new output problem (*output*) is generated from a given input example problem (*input*) by mapping relationships between a *base* and a new *instance* in an episode.

Fig. 1 shows our basic idea of problem generation by using an episode. In Fig. 1, the horizontal axis represents the features in solutions of problems and the vertical axis represents the features in problem situations. Each cell (e.g., A-1 and A-2) represents a category of problems that have the same solution and situation. Each category is called a *problem pattern*. Initially, the system has four problems in *problem patterns* A-1, B-1, A-2 and A-3. Suppose that a valid episode *E* is formed from a problem in A-1 as a *base* and a problem in B-1 as a *new instance*. If relationships in *E* can be adapted to a problem in A-2 as *input*, then a new problem in B-2 is generated as *output*. Therefore, problem generation by our system is regarded as the generation of *output* from *input* through the solving of the four-term analogy "*base*: *new instance* = *input*: *output*." In the same way, a problem in B-3 is generated from a problem in A-3 by using *E*. Our system generates problems in B-2 and B-3 that have the same solution and different situations; this means that our system adds two new *problem patterns*.

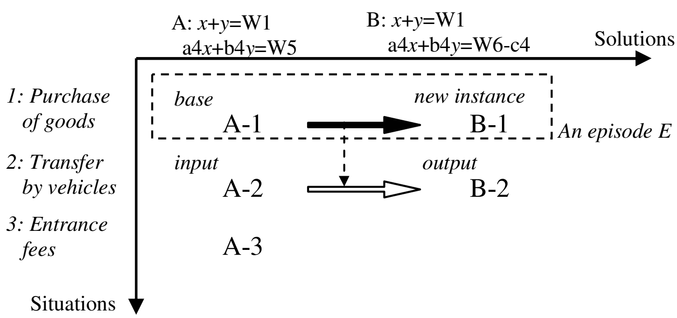


Fig. 1. Basic idea of problem generation by using an episode

With this basic idea, our system increases the number of problems and expands their variety by forming and using episodes.

Here, problem generation that brings a new *problem pattern* is defined as *novel generation*, and that which presents problems in existing *problem patterns* is defined as *modifying generation*, such as generation of a problem in B-1 as *output* from another problem in A-1 as *input*. In *modifying generation*, the system can automatically create texts for generated problems, though, the system requires the help of a user to create texts in *novel generation*.

2.2 Construction of the System

Our system comprises four main components: an *input-analysis interface*, a *dictionary database*, a *casebase*, and a *production engine*. The *input-analysis interface* constructs problem data from problems input by a user. The *dictionary database* has a *conceptual dictionary* with a thesaurus, and an *ontology database* that contains knowledge of words used in past word problems. The dictionary provides the word knowledge needed in problem generation. The *casebase* consists of two different databases, a *problem database* and an *episode database* that stores knowledge used to generate new problems. The *production engine* generates new problems using the *dictionary database* and the *casebase*. Fig. 2 shows the architecture of our system.

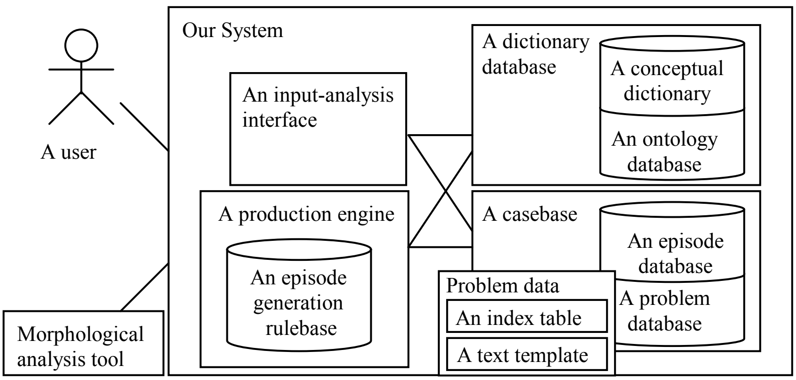


Fig. 2. Architecture of our system

The system acquires word knowledge in the *ontology database*, and problem data in the *problem database* through interactions with a user. An increase of such knowledge means an increase in the number of episodes formed validly, words used as entity objects in problem texts, and *text templates*. Each *text template* in problem data is knowledge formed from a problem text, which is used to create texts of new problems generated by the system. The increase of knowledge thus expands the variety of problems generated by our system.

2.3 Summary of the System's Problem Generation Procedures

We briefly explain the system's problem generation through interactions with a user. For details on the implementation of the generation, see our proceeding article [6].

1. Storing problem data: In problem generation by our system, a user first stores problem data in the system. The user inputs a problem text and its solution in plain-text format. The *input analysis interface* then analyzes them and constructs problem data. After that, the system requires the user to add new information or modify the problem data that the system cannot understand correctly.
2. Forming episodes: Each episode is formed from two problems that a user selects as the *base* and the *new instance* from problems stored in the system. The *production*

engine forms episodes by providing procedures called *altering actions* that describe which indexes of problem data are altered and how in problem generation based on comparisons between indexes of the *base* and the *new instance*.
3. Problem generation: Our system generates new problems (*output*) from given problems (*input*) by using the episodes. The basic idea of problem generation has already been illustrated in Fig. 1. The *production engine* generates *output* by adapting *altering actions* in each episode to a given *input*. The system presents *output* to a user and simultaneously requires the user to revise it, if needed. Since any problem can be generated through interactions with a user, every new generated problem is always evaluated and revised by the user.

3 Experimental Evaluations

Our system was evaluated in regard to the following viewpoints: (1) whether it can appropriately generate new problems and expand the variety of problems, and (2) whether it can acquire effective knowledge through interactions with a user. To verify the first viewpoint, we conducted performance tests to evaluate problems automatically generated by our system from initial problems, and to verify the second viewpoint, we conducted application tests to confirm the appropriateness of the system's interactions with general users.

3.1 Performance Tests

3.1.1 Procedures and Materials

In the performance tests, we first stored initial problems in the system, and second repeated the cycles where the system performed automatic generation and then interacted with a teacher as the user. In the automatic generation, (i) the system formed episodes comprising every effective pair of problems that were selected from the initial problems and then excluded duplicate identical episodes[1], and (ii) the system generated problems by adapting all episodes formed in (i) to every adaptable *input* in the stored problems. In the tests, the first author interacted with the system as the user.

Four sets of initial problems stored in the system were prepared from three general workbooks, Wordbook 1, 2, and 3, used in actual mathematical education. Additionally, a merge set was also produced from the three workbooks. Table 1 shows the contents of each set in detail.

The first viewpoint was verified by evaluating in the following two criteria: (1) the rates of emerged new *problem patterns* after generation, and (2) the appropriateness of the generated problem texts.

Table 1. Sets of initinal problems

	Workbook 1	Workbook 2	Workbook 3	Marge set
# of problems	9	34	34	77
# of *problem patterns*	6	22	26	41

[1] They are those that have an identical set of *altering actions*.

3.1.2 Results

Fig. 3 shows the rates of emerged new *problem patterns* after generation from each set of initial problems. As the figure indicates, new *problem patterns* were never generated from the initial problems of Workbook 1. In contrast, from Workbook 2, the number of *problem patterns* increased to twice as many as the initial number. These results indicate that our system can expand the variety of problems, but the degree of expansion depends on the nature of the initial problems.

Our system basically generates some problems in each problem pattern. The numbers of unique problems generated from the merge set were 303 in the first generation, 584 in the second and 592 in the third.

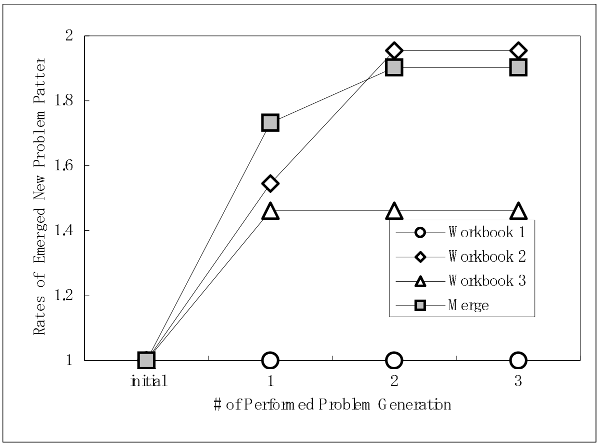


Fig. 3. Rates of emerged new problem patterns

Fig. 4 demonstrates the appropriateness of problem texts generated from the merged set. The figure shows the proportions of each of (1) problems generated in *modifying generation* whose problem texts were created appropriately ("appropriate" in Fig. 4), (2) those in *modifying generation* whose texts needed to be partially revised based on common knowledge[2] ("needed revision"), (3) and those in *novel generation* whose texts needed to be input by a user ("needed input") in all generated problems. The proportion of problems that need input can be automatically computed by the system. Other proportions were computed based on evaluation of each problem text by the first author. As indicated in Fig. 4, problems that needed the user's input disappeared in the third generation. This is because no problems in novel *problem patterns* were generated in the third generation, as indicated in Fig. 3. These results indicate that our system can relatively appropriately generate new problems through interactions with a user.

[2] E.g., to revise a text "I bought some bottles of juice in a flower shop" into "I bought some bottles of juice in a store."

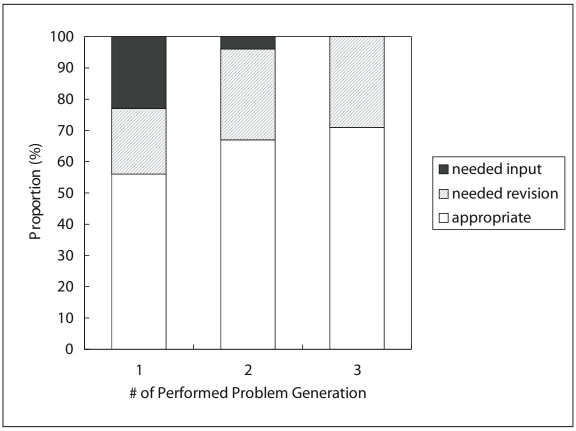


Fig. 4. Proportions of appropriate problems

3.1.3 Discussion

The results indicated in Figs. 3 and 4 confirmed that our system can relatively appropriately generate new problems and can expand the variety of problems from the initial problems.

As Fig. 4 indicates, problems whose texts needed revision did not decrease. This arises from the following technical issue: when creating *text templates*, the system cannot extract and alter terms other than those which are directly related to solutions. Therefore, our system necessarily requires the user to evaluate every generated problem text.

3.2 Application Tests

In the application tests, we asked participants as users to interact with our system. The participants were general undergraduates in Test 1, and a mathematics teacher and graduates majoring in mathematics in Test 2. These tests were conducted to verify whether our system can acquire effective knowledge through interactions with users.

3.2.1 Test 1

In Test 1, the system with which each participant interacted stored one of two sets of initial problems, Set 1 and Set 2, shown in Table 2. Set 1 contained simple problems, and Set 2 contained both simple problems and more complicated ones.

From each set, the system generated and presented nine problems each of which was included in one of nine different *problem patterns*. The participants were asked to evaluate and revise those problems. Especially, they were instructed to input texts with natural sentences for problems in novel *problem patterns*. Each participant was always able to ask the first author questions about operations of the system throughout the test session.

Eighteen undergraduates participated in Test 1. Half of them interacted with the system that stored Set 1, and the other half, Set 2. The participants were asked to solve three problems included in each set prior to interactions with the system. The results confirmed that all of them have appropriate abilities to solve problems in the target domain.

Table 2. Sets of initinal problems (Set 1 on the left, and Set 2 on the right)

		Solutions			Solutions		
		A	B	C	A	D	E
Situations	1	A-1	B-1		A-1	D-1	
	2	A-2		C-2	A-2		
	3	A-3			A-3		E-3

3.2.2 Results of Test 1

Table 3 shows the numbers of participants who appropriately revised each problem or input an appropriate text for each problem. As Table 3 shows, all participants successfully revised problems generated in *modifying generation*, however, some participants could not input appropriate texts for problems generated in *novel generation*. These results did not confirm that our system can acquire effective knowledge through interactions with general undergraduates.

Table 3. The numbers of participants who provided effective knowledge (Set 1 on the left, and Set 2 on the right)

		Solutions			Solutions		
		A	B	C	A	D	E
Situations	1	9	9	1*	9	9	5*
	2	9	9*	9	9	5*	3*
	3	9	9*	1*	9	6*	9

Cells marked „*“ had a problem generated in *novel generation*, and the others in *modifying generation*.

3.2.3 Test 2

As experts of the target domain, a mathematics teacher and three graduates majoring in mathematics participated in Test 2.

A set of initial problems stored in the system was created by merging Set 1 and Set 2, shown in Table 2. In Test 2, the system presented only problems generated in *novel generation*. The participants were asked to input problem texts for problems presented by the system.

3.2.4 Results of Test 2

In Test 2, every participant input appropriate texts for each problem, confirming that our system can indeed acquire effective knowledge through interactions with experts in the target domain.

3.2.5 Discussion

In Test 1, it was found that undergraduates as users could not necessarily input appropriate problem texts. Thus, the system may acquire inappropriate knowledge through interactions with the undergraduates, so that it may generate inappropriate problems. In Test 1, the participants were assumed to be novices in the domain of mathematics. In contrast, every participant in Test 2 was considered to be an expert in the target domain. The expert participants gave appropriate problem texts to the system, indicating that our system needs interactions with a knowledgeable user.

The inappropriate problem texts input by the participants in Test 1 basically had solution structures different from those of problem data presented as *output*. For example, when the system required a text for a problem involving the "*purchase of pencils and pens*" whose equations of the solution were "$x+y$=W1, a6(1+a7)x+b6(1+b7)y=W5," one participant input the following text.

> Today, W1 pencils and pens were sold for W5 yen. The price of each pencil was a6 yen and that of each pen was b6 yen. The number of pencils sold today was a7 % more than that of yesterday, and the number of pens sold today was b7 % more than that of yesterday. How many pencils and pens were sold today?

The solution structure of the problem text above is given as follows.

Solution.

Let x denote the number of pencils sold today and y denote the number of pens sold today.

$x+y$=W1

a6x+b6y=W5

In this case, the numbers of pencils and pens yesterday, not today, should have been asked in order to provide an appropriate text for the *output*.

All participants in Test 1 succeeded in solving problems in the target domain, even though they were not knowledgeable in terms of problem generation. This may indicate that creating texts for word problems is a considerable difficult task, and that cognitive skills needed in solving and generating problems have different aspects.

4 Conclusions

In this study, we implemented a system that generates various mathematical word problems based on control of problem situations and their solutions. In implementing the system, we adopted the following approaches: (1) our system uses an episode as a case of problem generation, and (2) it acquires knowledge through interactions with a user. Our system generates new problems from initial problems stored in the system and expands their variety. We experimentally evaluated our system and verified that it can expand the variety of problems. We also found that it needs interactions with a knowledgeable user to create appropriately worded problems.

One of the most important future works is to propose learning support by our system. We believe that our system can support learners' problem generation by presenting various types of problems to them. Currently, we are integrating new functions into the system, while simultaneously investigating the effectiveness of presenting problems as examples for problem generation by learners.

References

1. Chi, M.T.H., Feltovich, P.J. and Glaser, R.: Categorization and Representation of Physics Problems by Experts and Novices, Cognitive Science, 5 (1981) 121-152.
2. English, L.D.: Children's reasoning processes in classifying and solving computational word problems, Mathematical reasoning: Analogies, metaphors, and images, Mahwah, NJ: Lawrence Erlbaum Associates (1997) 191-220.

3. Hirashima, T., Niitu, T., Hirose, K., Kashihara, A. and Toyoda, J.: An Indexing Framework for Adaptive Arrangement of Mechanics Problems for ITS, IEICE Transactions on Information and Systems, vol. E77-D, no.1 (1994) 19-26.
4. Hirashima, T., Umeda, T., and Takeuchi, A.: An Intelligent Problem-Sharing Environment on Web, Proceedings of ICCE2001 (2001) 449-456.
5. Kanenish, K., Hayashi, K., Mitsuhara, H., and Yano. Y.: Drill Exercise Generation in WBT from Knowledge Based Components, Transactions of Japanese Society for Information and Systems in Education, vol.20, no.2, (2003) 71-82.
6. Kojima, K. and Miwa, K.: A System that Generates Word Problems Using Problem Generation Episodes, In Proceedings of ICCE 2005 (2005) 193-200.
7. Martin, B. and Mitrovic, A.: Automatic Problem Generation in Constraint-Based Tutors, Proceedings of the Sixth International Conference on Intelligent Tutoring Systems (2002) 388-398.
8. Novick, L.R.: Analogical Transfer, Problem Silmilarity, and Expertise, Journal of Experimental Psychology: Learning, Memory, and Cognition, vol.14, No.3 (1988) 510-520.
9. Shirota, Y.: Web-Based Mathematical Learning Material Automatic Generation System for Economical Optimization Problems, Gakushuin Economics Papers, vol.41, no.1 (2004).
10. Takano, A., and Hashimoto, J.: Drill Exercise Generation based on the knowledge base, IPSJ Technical Report, NL-160 (2003) 23-28.

Time in the Adaptive Tutoring Process Model

Alke Martens

University of Rostock,
Department of Computer Science and Electrical Engineering,
Albert-Einstein-Str. 21
18059 Rostock, Germany
Tel.: +49 – (0)381 – 4987613
martens@informatik.uni-rostock.de

Abstract. Formal models can be found in different computer science domains - they have the advantage to be independent of application domains and of programming languages. ITSs development is usually not based on formal models. Based on automaton theory and on formal descriptions known from modelling and simulation, the formal tutoring process model (tpm) is a formal model for ITSs. The model exists as basic tpm and as adaptive tpm. The extension of the adaptive model is described in the paper. Extended with a temporal dimension, i.e. the 'counter', the static tpm can be used to realize another way of adaptation: the training case can be changed at runtime based on the counter values. This value can count the learner's steps in the training case, it can be interpreted as duration, or as validity of a state.

1 Introduction

Whereas Intelligent Tutoring Systems (ITSs) (also called intelligent teaching and training systems (ITTS)) can look back on a comparably long tradition, only few formal models can be found. This has led to a situation, where on one hand a quite homogeneous ITS architecture exists. On the other hand the interpretation of role and functionality of each of the parts of this architecture is heterogeneous. The ITS architecture has been described by several researchers, e.g. [3], [4], [12], [14], or [22]. The naming of the parts constituting the ITS architecture shows only slight variations. For example the learner model is often called student model, the pedagogical knowledge model can be found under the name tutoring module. But under the cover of the same or of similar terms, a complete different interpretation of role and functionality can be found. For example, in some ITSs, the expert knowledge module comes in the shape of an expert system. To integrate an expert system (a component which is a complete system itself) in a teaching and training system is a traditional approach (see e.g. [3] or [22]). In contrast to this, in the last years, several ITS developers have followed the direction of object oriented and component based software development (e.g. [11]). In these systems, execution functionality is encapsulated and separated from data, and system components communicate via interfaces.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 134–143, 2006.

The remains of former expert systems can be found in complex databases, which are based on an ontology of the application domain (see e.g. [13]). The execution functionality of former expert systems is now part of components responsible for execution of adaptation or selection processes. Summarizing, the ITS architecture can be interpreted in at least two ways, both of which have their advantages and disadvantages. One way is to see the ITS as a set of interacting subsystems, each with own data and execution functionality. The other way is to see the ITS as being constructed by a set of "passive" components and a central steering component. In the following paper, the second perspective will be taken.

If an ITS is constructed of "passive" components with centralized execution, the ITS architecture consists of an expert module, a learner module, a user interface and a kind of steering or tutoring module (see figure 1). The notion "module" is used to denote that the entity is neither a well defined and clearly structured model, nor an implemented component, but only a part of a system. The expert module can consist of an expert knowledge module and a pedagogical knowledge module. The central steering module is responsible for accessing the databases and for steering the interaction with the learner via the user interface. The steering module uses the learner module and the expert module for adaptation. The central steering module can be described by a formal model, which is called the tutoring process model. The tutoring process model has been developed to describe steering and adaptation in the case-based web-based ITS Docs 'n Drugs [8], [13] and in the case-based ITS protype TutMoSi-I [18]. Currently it is investigated in the "not-case-based" Adaptive Hypermedia system TRYPTRY.

The tutoring process model has three extensions. The first extension is the basic tutoring process model, which contains no learner model and can be used in simple teaching and training systems, where no adaptation to the learner takes place [16]. The second extension is the adaptive tutoring process model. This model contains an abstract learner model and can be used to describe the adaptation [16]. The third extension is the tutoring process model with a

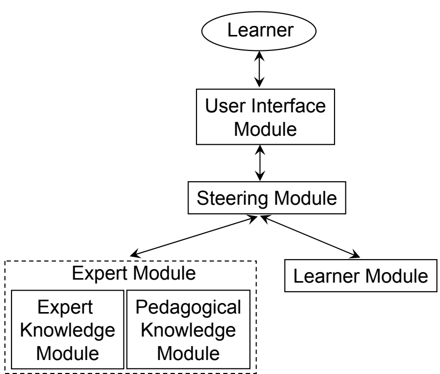


Fig. 1. ITS architecture

'time' as additional feature for adaptation. The timed tutoring process model is described in this paper.

The rest of the paper is organized as follows. In the next section, the role of time in a case-based ITS is discussed. The subsequent section describes the formal adaptive tutoring process model with adaptation to a 'time' counter. The paper closes with a discussion of the formal description in comparison to other approaches of ITS development.

2 Role of Time in Case-Based Training

Before explaining the formal model itself, a closer look at the different roles a temporal dimension can have in case-based training is necessary. In case-based training, the teaching material is often provided in a narrative manner. The learner takes over a certain role in the training scenario. For example, in Docs 'n Drugs [8], the learner acts as physician in a hospital. The training case is constructed based on a patient record and tells the story of a patient in an everyday clinical situation. It consists of multimedia descriptions, patient's utterances, and results of examinations. In TutMoSi-I, the learner is an expert in modelling and simulation. The learner has to interact with a client who describes part of a system which he wants to have modelled and simulated. In both ITSs, the learner has to decide about how to proceed in an appropriate way, given the situation at hand.

Allowing an adaptation to a value denoting time enriches case-based training. For example, in using Docs 'n Drugs in supervised courses at the University of Ulm, the unrealistic effect of ordering a laboratory examination and having the result available immediately has often been criticised by the students. However, embedding 'time' in a training case must take different 'interpretations' of time into consideration [1], e.g.:

- Temporal order (e.g. $t_1 \leq t_2$)
- Temporal interval interpreted as delay (e.g. $[t_1, t_2]$ with: t_1 information ordered, t_2 ordered information is available) or as duration
- Temporal interval interpreted as availability (e.g. $[t_1, t_2]$ with: t_1 information availability starts, t_2 information availability ends) or as validity (e.g. $[t_1, t_2]$ with: t_1 information is valid, t_2 state changes, information is invalid)

Integrating time in a training case can add to the realism of the training situation. But it also can be annoying. The system developer must decide what happens if the learner interrupts his interaction with the training case. Also, the system developer must take into account, what effects an instable internet connection might have if the learner works with a web-based system.

A case-based training case with temporal dimension be realized in three ways:

1. Static training case
2. Dynamic training case with internal clock
3. Dynamic training case with external clock

In the static training case changes of states are related to pre and post conditions (comparable to the adaptive tutoring process model, see [16]). Such conditions can be called 'time', although in the paper the term 'counter' is preferred. This counter can be associated with real numbers, but is neither related with an external nor with an internal clock. How the counter can be integrated in the static training case will be described in the subsequent section.

Dynamic comes into the training case by integrating a temporal dimension in form of a clock. The internal clock starts if the learner begins to work with the training case. It can be a function of real time, e.g. one hour in real time is related to one unit in training case time, or it can be another temporal unit, e.g. associating duration with steps in the training case, dependent on the duration of a step in real time. A time-counter function adds the amount of time associated with the next step chosen by the learner. In-between the learner's actions, no time passes.

Embedding an external clock as time trigger in the training case shifts case-based training in the direction of a simulation [19]. Same as the internal clock, the counting of time starts if the learner begins his interaction with the training case. The complete training case has duration and the states of the training case change 'dynamically' over time. In contrast to the internal clock, the external clock does not count the amount of time associated with steps in the training case. Instead, the external clock runs independent of the learner's interaction. Similar to the internal clock, passing of time in the training case might be a function of real time. The learner steers the training case's development, comparable to the 'human in the loop' simulation. The state changes might be realized in a discrete-event way (see e.g. [19]). Both so-called dynamic tutoring process models are described in [17].

3 Adaptive Tutoring Process Model

The adaptive tutoring process model consists of a training case, a learner model and the adaptation functions. Formally, the adaptive tutoring process model, which is an extension of the abstract tutoring process model (see e.g. [20], [16]), can be described as follows:

$$TPM_{adapt} = \langle C, LM, show, enable \rangle \tag{1}$$

with: C is the training case, LM is the learner model, *show* is the show state function, and *enable* ist the enable action function. These parts will be explained in the following.

Regarding a training case in an ITS, there are always two things available: the information displayed to the learner and the set of actions, the learner has available. This structural construction is independent of the implementation. Displayed information can be simple text or multi-media. Actions can be designed as simple buttons, as menues, as a selection tree or as complex multi-media tools. Given these two main constituents, the training case can formally be described as a structure consisting of a set of states Q and a set of actions A. Also, there must be at least one start state q_0 and a set of possible final states F. Each action must lead to a new state, i.e. to new information displayed to the learner. Vice versa, each state must have a set of actions associated, i.e. the set of actions which are possible, given the information currently displayed and the history of displayed information. Thus, two functions must be represented in the formal description of the training case: one for determining the state transition after the selection of an action, called δ, another for selecting the set of actions associated with a state, called *allow*. To allow for adaptation of the information displayed to the learner, another level of granularity in the model is required: states Q are constructed by bricks B. A state can be perceived as a wrapper, which consists at least of one brick. The information itself is encapsuled in the bricks. The function *select* determines the set of bricks which are related to a state. What has been described so far is the training case C in the formal tutoring process model, given above. The training case C is a structure itself:

$$C = \langle Q, A, q_0, F, B, \delta, select, allow \rangle \tag{2}$$

with: Q is the finite set of states, A is the finite set of actions, $q_0 \in Q$ is the start state, $F \subset Q$ is the finite set of final states, B is the finite set of bricks, δ is the state transition function, *select* is the select brick function, and *allow* is the select action function. The three functions are defined as follows. The state transition function is given as

$$\delta : Q \times A \longrightarrow Q \cup \{\perp\} \tag{3}$$

$\perp$ denotes that the actual implementation of the system should prevent this state transition to be executed, (e.g. final states have no state transition: $\forall q_f \in F, F \subset Q, a \in A : \delta(q_f, a) = \perp$). The select brick function is given as:

$$select : Q \longrightarrow 2^B \tag{4}$$

The select action function is defined as:

$$allow : Q \longrightarrow 2^A \tag{5}$$

For more details see e.g. [16]

In an interactive training case, adaptation at runtime takes place at two levels: adaptation to the learner's performance, his background knowledge, experience or his expertise, and adaptation to the coherence of the story line. To realize adaptation to the learner, a learner model is required. To realize adaptation to the development of the story line a record of the learner's actions is necessary. Both are part if a learner model. This learner model is a minimum definition of what a learner model should contain in an ITS: the information about the learner, the so-called learner profile LP, and a record of the learner's actions and the information given to the learner so far, the so-called learner knowledge LW. Thus, the learner model LM can be seen as a structure:

$$LM = \langle LP, LW \rangle \tag{6}$$

Both parts can be represented as structures. LP is a structure

$$LP = \langle id, expertise \rangle \tag{7}$$

with the components: id – the identification of the learner, and $expertise$ – the learner's expertise. LW is a structure

$$LW = \langle id, Lpath, Lresult, Lacq \rangle \tag{8}$$

with the components: id – the identification of the training case, $Lpath$ – the partial path, $Lresult$ – a set of results, and $Lacq$ – the set of facts.

To allow for adaptation, one further extension in the model is required. So far it has been described that states consist of bricks. Now, certain features must be embedded to allow for selection of bricks according to entries in the learner model. The same must take place regarding the actions. Thus, both, bricks as well as actions must be defined as structures. In the following, each brick $b \in B$ is a structure

$$b = \langle id, con, PRE, POST \rangle \tag{9}$$

with: id is the identification of the brick, con is the list of content elements, PRE is the set of pre conditions, and $POST$ is the set of post conditions. Accordingly, each action $a \in A$ is a structure:

$$a = \langle id, name, PRE \rangle \tag{10}$$

with: id is the identification of the action, $name$ is the (display) name, and PRE is the set of pre conditions.

The pre condition PRE determines, whether the according brick or action will be part of the actual display of the current learner. The evaluation of a pre condition takes into account the learner's profile LP as well as the learner's knowledge LW. The other way around, pre conditions can be constructed by the training case authors using as well profile information ('this brick should only be seen by experts') as facts in the training case ('if fact XY has been seen, the action Z should be active'). The post condition $POST$ is a set of facts (from the application domain). If a brick is selected for display, the learner has access to the facts. Thus, these facts form his potential knowledge – they are recorded in the set $Lacq$ of the learner model.

The functions *show* and *enable* of equation 1 have not been explained, yet. The show state function is defined as

$$show : 2^B \times LP \times LW \longrightarrow 2^B \tag{11}$$

with: $show(B_q, expertise, Lacq) = B_a$

B_q is the set of bricks determined by *select*, i.e. the bricks associated with the state Q, $expertise$ is the profile entry of the learner model, and $Lacq$ is the amount of facts the learner has acquired so far in the training case. The function *show* determines, which subset of bricks will actually be displayed, taking the learner model into account, with $B_a \neq \emptyset$, i.e. there should be at least one brick that can be shown to the learner. The set of pre conditions of a brick can be the empty set - this kind of brick will be displayed without condition to every type of learner. The enable action function is defined as:

$$enable : 2^A \times LP \times LW \longrightarrow 2^A \tag{12}$$

with: $enable(A_q, expertise, Lacq) = A_a$

Similar to the *show* function, the *enable* function takes the set of actions determined by *allow*, i.e. A_q, and derives, which actions will actually be available in this step, i.e. A_a with $A_a \neq \emptyset$. It uses the entries *expertise* and *Lacq* in the learner model. The set of temporarily available actions is adapted to the learner at run time. There must be at least one action that meets the pre conditions. The set of pre conditions PRE of an action $a \in A$ can also be the empty set - then, the action is always active.

When implementing the tutoring process model as described above, a certain sequence of steps must be taken into account (cited from: [20]). The sequence of steps is important, as the adaptation takes place at runtime - changes in the learner model must be recorded in the described temporal order to have the correct effects.

- Determine the next state with the δ function.
- Take this state and derive the set of associated bricks, using the *select* function.
- Take this set of bricks and the learner model and determine with *show*, which of the bricks should be shown to the current learner.
- Actualize the learner model with the bricks' post conditions (only of the bricks determined for display).
- Determine the amount of actions associated with the state, using the *allow* function.
- Take this set of actions and the actualized learner model and determine with *enable* the actions that should be available to the current learner.

The adaptive tutoring process model, as described above, can be used with small changes to realize the implementation of the static training case. To count the steps in the training case, a simple function can be used. This function adds 1 to *stepcount* for every action chosen by the learner. This can be directly embedded in the learner model LM, in the part LW. Additionally, a counter shall be embedded. The values of this counter can be used to reflect duration or delay, availability or validit. For each state, the counter has to add *counter* values. The *counter* values can be encoded as post conditions of bricks, i.e. they can be part of $POST$. Not all bricks necessarily have counter information. The counter information of a state is given by adding all counter values of bricks chosen for display. Thus, the counter information depends on the learner's performance and is changed at runtime.

Both counter values, *stepcount* and *counter*, have to be recorded in the learner model LM. As these values can be interpreted as part of the learner's knowledge, they are recorded in LW. It would be possible to collect the counter values in

Lacq. But to distinguish counter values from collected facts, a new list *Lcount* is embedded in the structure *LW* (see equation 8). *Lcount* contains all different types of counters. Each counter records the values. Currently, *Lcount* consists of the step counter *stepcount* and the post-condition value counter *counter*.

$$LW = \langle id, Lpath, Lresult, Lacq, Lcount \rangle \tag{13}$$

Both counter values can now be used for the adaptation process. Bricks as well as actions can contain a *stepcount* and/or *counter* related pre condition. For example, brick b_x might have a post condition, which sais: $counter := counter + 3$. Thus, if this brick has been displayed, the current *counter* value in *Lcount* has to be changed. The interpretation of this can be that the information displayed to the learner has the value of three temporal entities. The next brick might have a pre condition, e.g. $counter \leq 3$, thus it will not be selected for display. Similarly, a pre condition *stepcount* can be used – for example, an action contains a pre condition $stepcount \geq 5$, i.e. this action is only available, if the learner has made at least five steps in advance.

4 Discussion

In the last years, some new trends can be found in development of ITSs. One trend is to establish software engineering techniques in this field. The new perspective implies that software development is no longer perceived to be an art but a craft: software should be developed according to engineering principles as for example modularity, extensibility, and re-usability. Consequently, ITSs – regarded as software – should be developed in a modular, stepwise, and traceable manner. One method to apply software engineering techniques is to use existing patterns (or to use parts of pattern languages) at all stages of the development process. In ITSs approaches in this direction have been described e.g. by [6], [7], and [10]. Another trend is to develop standards and ontologies for ITSs and other e-learning systems. Standards provide the terminology. Ontologies give a hierarchy of concepts and relations between concepts. Standards are for example the Learning Technology System Architecture LTSA [9] and the Dublin Core Metadata [5]. Whereas standards predefine terms, which is a step towards re-usability, they lack a concrete description of how to realize the system's parts, which is a drawback. Thus, the most promising approach might be to combine software engineering techniques and usage of standards.

Software engineering techniques can be used to describe a system's purpose, its components and the component's interfaces and communication. The terms and concepts used in software engineering, e.g. patterns, are in most cases based on the UML (Unified Modelling Language) [2]. Another way to describe a system's functionality, which has its roots in theoretical computer science research and in modelling and simulation, is to use a formal (mathematical) description. In modelling and simulation, as well as in automaton theory, formal descriptions have been used for quite a long time to understand and describe how processes work. The advantage of a formal mathematical description is that it is independent of an application domain, even independent from software engineering. Software engineering uses models at different stages of the process of software development. Formal descriptions sketch the basic theoretical model, independent of how it is realized. Software engineering models can e.g. be based on formal models.

The tutoring process model is a formal model, which sketches the parts of a training case, a minimal learner model, and how adaptation of a training case can be realized. The model has bee used as basis for implementing the ITS Docs 'n Drugs [8], and the ITS prototype TutMoSi-I [18]. Docs 'n Drugs has been developed according to software engineering principles [11]. It was easy to integrate the tutoring process model in the software development process. Applying the tutoring process model in the process of ITSs development has proved the usefulness of the model, so far. Moreover, the tutoring process model with its several different extensions has become part of a pattern language. In this pattern language, the process steering component consists of an engine part and a specification part (see [10]). The functionality described by the model is part of the tutoring process specification.

In general, a formal description should be easy to communicate and easy to understand. A formal model like the one described in this paper comes with a certain complexity, which is not easy to grasp at first glance. Especially software engineers are more used to work with models described in UML than with automaton theory models. Thus, currently two main directions are pursued, one of which is the extension of the formal model, as described in this paper. The other is to transform the formally described tutoring process model in a more UML like description.

References

1. Allen, J.F.: Time and Time Again: The Many Ways to Represent Time. International Journal of Intelligent Systems, 6(4), (1991) 341–355
2. Booch, G., Rumbaugh, J. and Jacobson, I.: The Unified Modeling Language User Guide. Addison Wesley Longman, Massachusetts, US, (1999)
3. Clancey, W.J.: Methodology for Building an Intelligent Tutoring System. In: Kintsch, W., Miller, J.R., and Polson, P.G. (eds.): Methods and Tactics in Cognitive Sciences, Lawrence Erlbaum Associates, Hillsdale, New Jersey, London, (1984) 51–84
4. Corbett, A.T., Koedinger, K. R. and Anderson, J. R.: Intelligent Tutoring Systems. In: Helander, M., Landauer, T. K. and Prabhu, P. (eds): Handbook of Human-Computer Interaction, Elsevier Science B.V., (1997) 849–874
5. DCMI. Dublin Core Metadata Initiative, `http://dublincore.org`, (2002)
6. Devedzic, V.: Using Design Patterns in ITS Development. In: Lajoie, S.P. and Vivet, M. (eds): Proc. of the World Conference on Artificial Intelligence in Education AI–ED, Amsterdam, Netherlands, (1999) 657–659
7. Devedzic, V.: A Pattern Language for Architectures of Intelligent Tutors. In: Moore, J.D., Redfield, C. and Johnson, W.L. (eds): Proc. of the World Conference on Artificial Intelligence in Education AI–ED, San Antonio, TX, US, (2001) 542–544

8. Docs ‘n Drugs - The virtual hospital. `http://www.docs-n-drugs.de`, (2000)
9. Farance, F. and Tonkel, J.: LTSA Specification - Learning Technology Systems Architecture, Draft 8, http://www.edutool.com/ltsa, (2001)
10. Harrer, A. and Martens, A.: Towards a Pattern Language for Intelligent Teaching and Training Systems. Submitted to: Intelligent Tutoring Systems (2006)
11. Illmann, T., Weber, M., Martens, A. and Seitz, A.: A Pattern-Oriented Design of a Web-Based and Case-Oriented Multimedia Training System in Medicine. 4th World Conference on Integrated Design and Process Technology, Dallas, US (2000)
12. Lelouche, R.: Intelligent Tutoring Systems from Birth to Now. KI - Künstliche Intelligenz, Germany, 4, (1999) 5–11
13. Martens, A., Bernauer, J., Illmann, T. and Seitz, A.: “Docs ’n Drugs - The Virtual Polyclinic” An Intelligent Tutoring System for Web-Based and Case-Oriented Training in Medicine. In: Proc. of the American Medical Informatics Conference AMIA, (2001)
14. Martens, A.: Centralize the Tutoring Process in Intelligent Tutoring Systems. In: Proc. of the 5th Internat. Conf. New Educational Environments ICNEE, Lucerne, Switzerland, (2003)
15. Martens, A.: Case-based or Problem-based – An Investigation of Intelligent Tutoring Systems. Submitted to: Journal Methods of Information in Medicine, (2005)
16. Martens, A.: Modeling of Adaptive Tutoring Processes. In: Zongmin, M. (ed): Web-Based Intelligent e-Learning Systems: Technologies and Applications, Information Science Publishing, Idea Group Inc., Hershey, London, (2005) 193–215
17. Martens, A.: Towards a Dynamic Tutoring Process Model. Submitted to: International Conference on Advanced Learning Technologies, (2006)
18. Martens, A. and Himmelspach, J.: Combining Intelligent Tutoring and Simulation Systems. In: Proc. of the Internat. Conference on Simulation in Human Computer Interfaces, SIMCHI, Part of the Western Multi Conference WMC ’05, New Orleans, US, (2005)
19. Martens, A. and Uhrmacher, A. M.: Modeling Tutoring as a Dynamic Process - A Discrete Event Simulation Approach. In: Szczerbicka, H. (ed.): Proc. of the European Simulation Multiconference ESM, (1999) 111–119
20. Martens, A. and Uhrmacher, A. M.: A Formal Tutoring Process Model for Intelligent Tutoring Systems. In: Proc. of the European Artificial Intelligence Conference ECAI, (2004)
21. Merseth, K.: The Early History of Case-Based Instruction. Journal of Teacher Education, 42, 4, (1991) 243–249
22. Rickel, J.: Intelligent Computer Aided Instruction: A Survey Organized Around System Components. IEEE Transactions on Systems, Man, and Cybernetics, 19, 1, (1989) 40–57
23. Willms, J. and Möbus, C.: Evolution of the Hyothesis Testing Approach in Intelligent Problem Solving Environments. In: Gauthier, G., Frasson, C. and VanLehn, K.: Proc. of the 5th Int. Conf. on Intelligent Tutoring Systems ITS, Montréal, Canada, Lecture Notes in Computer Science 1839, Springer, (1998) 403–412

Coaching Within a Domain Independent Inquiry Environment

Toby Dragon, Beverly Park Woolf, David Marshall, and Tom Murray

Department of Computer Science, University of Massachusetts, Amherst, MA
140 Governor's Drive, Amherst, MA 01002
{dragon, bev, marshall, tmurray}@cs.umass.edu

Abstract. We describe a portable coaching environment used within a domain-independent inquiry-learning infrastructure. This coach reasons about a student's knowledge and offers pertinent, domain-specific feedback. It promotes good inquiry behavior by critiquing the student's hypotheses and supporting data and relationships among propositions. Four inquiry tutors in separate disciplines have been developed that use embedded expert knowledge bases and reusable domain-independent rules. We describe the functionality of the coach within an art history domain, discuss the implementation of the coach, and elaborate on the options given to domain authors for customization.

1 Introduction

This article describes a coaching environment that supports structured reasoning within an open-ended, inquiry-learning infrastructure. The infrastructure, called Rashi[1], invites students to reason about cases, posit theories and recognize when their data does or does not support their hypotheses [1, 2, 3, 4, 5]. The environment tracks students' investigations (e.g., hypotheses, inferences and data) and helps students articulate how evidence and theories are related. This infrastructure has been used to develop tutors in three domains, biology, geology and forestry. The Biology Tutor has been used with over 500 students and contains 6 distinct medical cases [3, 4, 5, 11].[2] This paper describes features of the coaching environment and gives specific examples within the context of a new domain, Art History. This section describes the basic elements of the system used across disciplines and how a student interacts with the system. Later sections describe the functionality of the coach within the art history domain, and the techniques used to provide this help. Finally, we discuss the alterations a domain expert can make to customize the coach for specific purposes.

The suite of Rashi tutors share a methodology, set of assumptions and tools that allow them to leverage the accomplishments and intuitions of each other. The overarching shared goal is to involve students in reasoning, critical thinking and hypothesis generation and thereby to generate more responsive and active learning. Compelling evidence exists that inquiry learning benefits skill and knowledge acquisition [13, 16]. When students manipulate artifacts themselves and think freely

[1] Rashi homepage is http://ccbit.cs.umass.edu/Rashihome/
[2] The Biology and Geology Tutors are located at http://ccbit.cs.umass.edu/Rashihome/projects/

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 144 – 153, 2006.

about problems, they become more actively involved and generally become more systematic and scientific in their discovery of laws [16]. The increased interactivity alone has been shown to increase learning [13].

Generic tools, common to all the inquiry tutors help students to frame hypotheses, gather evidence and construct arguments. Two types of tools are available: data collection and critical thinking tools. Data collection tools support identification of facts e.g. the *Image Explorer* allows students click "hotspots" to navigate to other images or collect data, the *Interview Tool* enables students to question sources, and the *Concept Library* provides a hyper-text repository for the subject at hand. Critical thinking tools include the *Inquiry Notebook*, where facts collected from data gathering tools are automatically entered, and the *Argument Editor*, where students create hypotheses and inferences. Facts from the *Inquiry Notebook* are dragged into the *Argument Editor* to support and refute arguments.

Different data collection methods create a broad and open-ended space for student exploration. Once data is collected, critical thinking tools provide both a central data repository and a place to form well-structured arguments.

2 Prior Research and Our Contributions

Other researchers have developed inquiry software that presents cases and simulation-based learning environments, along with tools for gathering, organizing, visualizing, and analyzing information during inquiry [6, 7, 8, 9, 10]. Some systems support authentic inquiry and knowledge sharing, and several track and analyze student data selections, providing students with space to explore subject matter from micro-economics [16] to medical diagnosis [14]. In some environments, students create hypotheses to explain a real–world phenomenon; they gather evidence and relate evidence to the hypotheses [14, 15, 16]. However, most systems do not evaluate a student's hypotheses and are restricted to one domain. Some have more of a data collecting than experimenting feel, are narrowly applicable and do not allow for more interactive tutoring. Limiting the domain allows the designer to facilitate specific types of experiments [13,16]. Some inquiry systems are built with the primary goal of teaching the inquiry process and not the domain [13,16,] while others teach only the specific domain and not the process.

The contributions of this work are to: 1) describe the functionality of the coaching component of Rashi within the context of a newly developed domain 2) provide details of the underlying structure of this coach that allow it to remain domain-independent 3) describe customizations available to authors that help customize the coach's behavior to specific domains or cases.

3 Coaching in Art History

This section describes the newly developed Art History Tutor and the functionality of the coaching component within it. Details are provided about functionality added to the system and how this drives the need for a coach built with and expert knowledge base. We then see an example of the coach offering feedback within this domain.

3.1 Support Critical Reading

The Art History Tutor is designed to model the teaching of critical reading skills, in which students read material to understand an artwork and then develop their own interpretation of that work based on readings. Students use the critical thinking tools to build a model of each argument presented in the reading. They carefully analyze arguments, decide which arguments hold up best to scrutiny, and state precisely why an argument may or may not be compelling. This creation gives them a strong sense of an author's assumptions, leaps of faith and unsubstantiated or specious claims. Through this teaching approach, students learn to recognize when an argument is well defended and cogent and to develop an appreciation of different methodological approaches to the interpretation of a work of art.

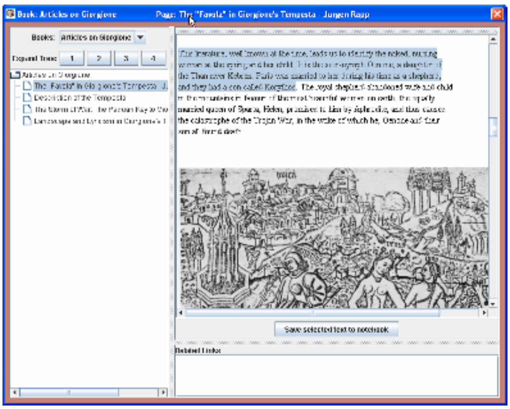


Fig. 1. (left) The Concept Library for Art History. (right) Giorgione, The Tempest, Gallerie dell' Accademia, Venice.

Developing an Art History Tutor required the definition of new technology as well as reusing existing features. The concept library was enhanced to include a text extraction button built to allow students to select a piece of text (Figure 1 left), for addition to their *Inquiry Notebook* and *Argument Editor* (seen later in Figure 2 top left).

The painting used for this case is *The Tempest* by the Renaissance painter Giorgione (1477-1510) (Figure 1 right). As students proceed through different papers and theories about *The Tempest*, they build hypotheses and supporting arguments in their *Argument Editor*, e.g., "The Tempest commemorates the battles for Padua". Students can also type in hypotheses of their own (Figure 2 top left). It is important for students to realize (and to model with relationships) that some statements made by one author refute claims made by other authors. It is also important for students to use information acquired through exploration to support/refute claims made in each argument. In the *Argument Editor*, Figure 2, the student supported the hypothesis "The Tempest commemorates the battles for Padua" with data such as "The buildings in the background represent the city of Padua." When the process of argument

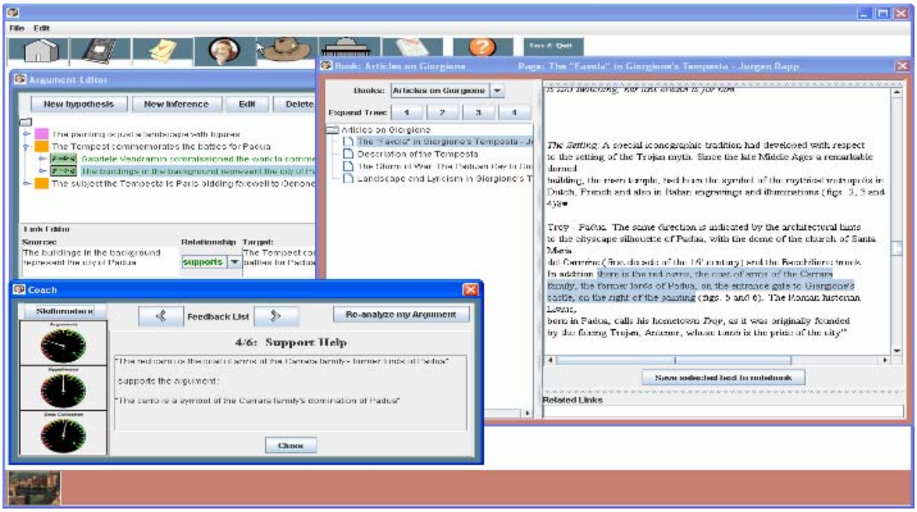


Fig. 2. The Rashi Coach in action, offering Support Feedback

modeling is concluded the student rates each hypothesis as "top", "possible" or "ruled-out." This is perhaps the most difficult part of the assignment.

Automated evaluation of student arguments is not a trivial process. While it is easy for a student to identify unsupported claims (represented as inferences with no supporting argument), it can be very difficult to decide if an inference is strongly defended. Often it is not the quantity of statements but instead the meaning of the statements that determines the validity of an argument. This is a compelling reason for using an expert knowledge base to evaluate student argument, rather than syntactic methods of argument evaluation.

3.2 Example Help from the Rashi Coach

While reading the literature and modeling a theory, students may overlook a piece of an argument. If help is requested, the tutor will evaluate the student's argument and call attention to sections of the text that further support the argument under construction. For example, if the student created the inference that "the buildings in the background represent the Italian city of Padua" but hasn't given any information to support it, the coach might call attention to the fact that the inference is not supported (Support Feedback). If the student requests further help, the coach might display a passage in an article that describes the meaning of a small insignia painted above the doorway in one of the background buildings as the coat of arms of the Carrara family – rulers of Padua (Figure 2, right window). Alternatively, the coach might choose to expose the insignia in one of the detail images of the painting and call the student's attention to it.

At this point the student has collected some data that supports the claim that the city is Padua and connected them directly to that inference in the *Argument Editor*

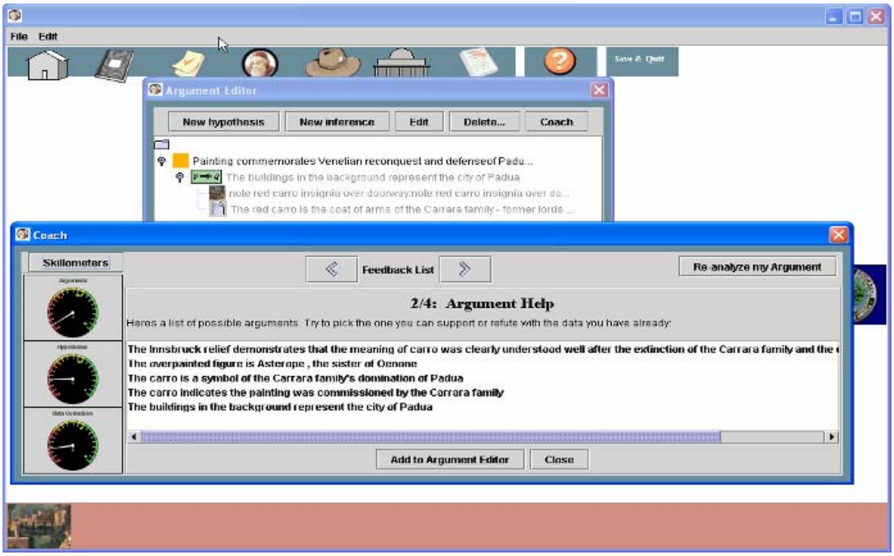


Fig. 3. An example of Argument Feedback

(Figure 3, top window). If the coach were again asked to assist, it would see that the student has information (the "red *carro*" and the fact that it is "the coat of arms" of the Paduan rulers) to support an inference that has not yet been expressed (Argument Feedback). If the student asks for help in forming the inference, the coach shows the student a list of possible arguments (Figure 3, bottom). Rather than just giving the correct answer, the coach provides a list containing the correct inference, and also includes other plausible inferences found "nearby" in the expert argument. This forces the student to make a decision rather than just receiving the correct inference.

In this simple example, we can see that the coach has the ability to find knowledge the student is missing and provide context-sensitive domain content to help support the student's inquiry learning process. Now we can consider the full range of the coach's functionality and how the coach accomplishes this.

4 Coach Functionality and Implementation

In order to offer help, a system must first reason about the student knowledge to provide feedback. Several questions need to be answered: What kind of help should be provided? When should help be offered and what actions should be taken? Different techniques have been shown to be effective [15]. However, inquiry is a process driven by students. Active students should not be disturbed or interrupted inappropriately. For this reason, the Rashi system does not currently intervene with the student, but instead provides help on demand.

The Rashi coach offers five feedback types that both provide domain knowledge and promote good inquiry behavior:

1. *Hypothesis Feedback* – The coach promotes considering multiple hypotheses and can offer a list of hypotheses from the expert knowledge base.
2. *Support and Refutation Feedback* – The coach encourages a top-down argument construction by urging students to supply supporting or refuting data for their arguments, Figure 2, left window. The system can bring them to a location where the most important data can be found, Figure 2 right window.
3. *Argument Feedback* –The coach encourages bottom-up argument construction by urging students to consider higher-level arguments. It offers an argument that the student might have missed, Figure 3, bottom window.
4. *Relationship Feedback* – The coach helps students identify relationships between propositions used in their arguments.
5. *Wrong Relationship Feedback* – The coach identifies contradictions between relationships in the student argument and the relationships in the expert knowledge base.

4.1 Representing and Using Student and Expert Knowledge

To create these types of feedback, we need a representation of both the student and the expert reasoning about the situation. We have an expert knowledge base to represent the domain knowledge and we track the student using an overlay of this expert knowledge, matching student input to expert knowledge. The student model is a matched subset of the expert argument. In this way, we can recognize what portion of the expert knowledge the student currently has, and encourage the student to discover more of the expert knowledge base.

After a student enters a proposition (hypothesis, inference or data), the tutor recognizes which portion of the expert knowledge base the student is discussing. The major challenge is to match student work to expert knowledge. The simplest method is to allow students to type in their own statements and use keyword searches to match the expert knowledge base. This has been used in several other tutors, e.g., [14, 16], yet it has obvious flaws; students may not phrase statements in the same way as the expert and so keywords may not match.

Rashi provides an additional way to understand the student's state of knowledge. Data collection happens within our own data gathering tools and thus the tutor recognizes the data as part of its expert knowledge base. This means we can identify which data has and has not been explored. This leads to a major gain in tutor understanding of student work. Even if the system is unable to match any student textual statements, it recognizes data collected.

Using the student argument and the expert knowledge base, Rashi employs a rule-based engine to identify problems with the student's argument and provide helpful feedback. Rules are generic in the sense that the same rules are applied to all four domains. The rules indicate how to identify inconsistencies by comparing a student and expert argument. These rules are directly related to the types of feedback available. For example, the rule that creates relationship feedback operates by identifying when two propositions are related in the expert knowledge base, yet unrelated in the student argument.

Using a rule-based engine is useful for several reasons. First, it allows the domain-independent system to be easily modified and responsive to an instructor's preference, e.g., the instructor can decide that the tutor should not intervene in the example above and can choose which rules will be applied within their domain. Since rules operate independently, they can be added and removed without interfering with each other. Also, since we are working with multiple representations in tandem (both student argument and expert knowledge base), using rules rather than complex graph traversal algorithms creates more readable and more easily modifiable code. Now that we have some understanding of how the coach is implemented, we can look deeper into the system to see how the coach can be altered to work more successfully with specific domains or cases.

4.2 Author Control over Coaching

The coaching system within Rashi offers some customization to allow for differences in author's preferences. The authors can choose which of the five types of feedback they would like their students to receive. More importantly though, the author can customize how the coach uses the expert knowledge base.

The coach could push students to say exactly the same thing contained in the expert knowledge base. Some tutors critique students until their input nearly matches that of the expert solution, e.g., [12]. In this way, the coach would help create an argument that encapsulates all of the knowledge within the expert system. However, we found this approach to be overly simplistic for our situation. Experts often include repetitive data in the knowledge base in order to create a complete representation, yet they do not want the student to necessarily be forced through the repetition as well.

To avoid this problem, Rashi takes a unique approach that pushes the student to encapsulate a critical mass of the expert argument without requiring the entire argument or specified subset. This is accomplished by adding extra knowledge to each proposition, providing the quantity of support or refutation required. The author specifies for each node in the expert knowledge base how many of its children are necessary to make a reasonable argument about that statement. Also, each proposition is labeled with an importance value, indicating its relative importance to the statements that it supports or refutes. This allows the coach to make decisions about which knowledge is most vital for a student. Using this extra information, Rashi urges students to supply a satisfactory argument composed of the most pertinent information without pushing them to recreate the entire expert knowledge base.

To discuss an example of this fine-tuning of the coach, we can return to the art history case previously discussed. Figure 4 shows a section of the knowledge base representing the paper's thesis. Once the knowledge base is developed, a faculty member author can adjust fine nuances of the coach's advice. For example, suppose the author decided that certain propositions are obvious and need not be mentioned by the coach. In the case described here, inference # 863 "Carro is a symbol of Carrera" stands between the data the student collected and the inference that began this exploration, # 784 "The city in the background is Padua." The student found evidence to refute this statement, # 782 "Carro is a coat of arms of the Carrara family" and # 800 "Note Carro." Suppose the author wanted the coach to omit stating the intermediate inferences. To let the coach know this, the author could mark inference

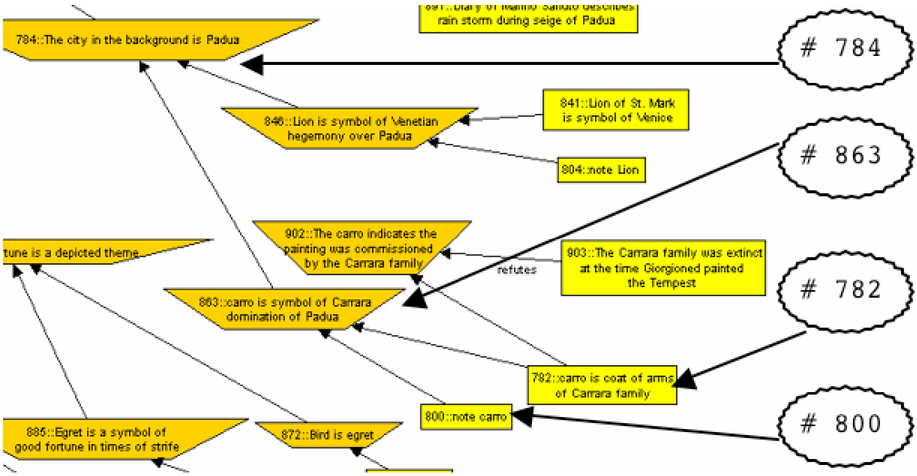


Fig. 4. A small piece of the Expert Knowledge Base for Art History

#863 as *visible=false*. Had the faculty done this, the coach would not have asked the student to form the argument #863 from the data (#782 and #800) after it was collected, but instead would have pushed them directly for #784. In this particular case, the author left the inference as *visible=true* because a coat of arms in a painting often depicts the painting's *patron* (which is an incorrect assertion in *the Tempest* because the Carrara family was extinct at the time Giorgione painted the picture - #903) so the author wanted the student to make the distinction about its use.

Faculty can adjust the knowledge base so that more important information is provided first. In this case, it makes sense for students to recognize the insignia in the painting before talking about the coat of arms of the Carrara family. The author assigns nodes with an *importance* rating that guides the coach to select a first choice from among the children of a node that require further support. In this case, node # 800 is given a higher importance rating than # 782. Importance ratings can be used at any level in the knowledge base to guide the coach toward a certain proposition first.

5 Conclusions and Future Work

This paper described the coaching environment within Rashi, a Web-based infrastructure offering critical thinking learning environments in four domains. An expert knowledge base and domain-independent rules help reason about student work and offer relevant domain-specific feedback within each domain to support students to refine their arguments and develop inquiry skills. We presented the types of feedback the coach can provide, and a general view of how this type of coaching can be accomplished in a domain-independent manner. Examples were presented of the knowledge base and coaching feedback in the newly developed art history domain.

Formative and summative evaluations have been conducted with more than 500 students using the biology, geology and forestry tutors. Empirical studies in large and small classes from several colleges and universities have shown improved domain knowledge, improved confidence in inquiry skills, and positive responses on usability of the interfaces. We expect to evaluate the Art History Tutor with UMass students in the near future.

We also plan to make certain improvements to the coach. We are exploring issues about how to balance a top-down vs. bottom-up approach to argument development. Currently the coach has a top-down approach that brings the user straight to observable data, and takes a bottom-up approach from those data. This can be problematic when considering long chains of inference. Data offered might not seem related to the topic at hand when not considering the intermediate inferences.

Another pressing issue is when to change topic and how to balance a breadth-first vs. a depth-first approach to argument construction. On one hand, the coach should not change its topic often, thus causing the student to lose focus. On the other hand, the coach should not press a student for every tiny detail about a topic before encouraging him/her to consider other hypotheses. We are exploring new approaches for choosing when to change topic.

Acknowledgements

Research on Rashi was funded in part by the U.S. Department of Education, " Expanding a General Model of Inquiry Learning", Fund for the Improvement of Post Secondary Education, Comprehensive Program, #P116B010483, B. Woolf, P.I., and by the National Science Foundation under grant DUE-0127183, "Inquiry Tools for Case-based Courses in Human Biology," M. Bruno, PI and Woolf, Co-PI, and NSF, CCLI #0340864, "On-line Inquiry Learning in Geology," D. Murray, P.I., B. Woolf co-PI.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

References

1. Woolf, B. P., Marshall, D., Mattingly, M., Lewis, J., Wright, S., Jellison, M & Murray, T. (2003). Tracking student propositions in an inquiry system. In U. Hoppe, F. Berdeho & J. Kay, (Eds.) Artificial Intelligence in Education, Proceedings of AIED 2003, World Conference, IOS Press, pp. 21-28.
2. Woolf, B. P., Reid, J., Stillings, N., Bruno, M., Murray, D., Reese, P., Peterfreund, A. & Rath, K. (2002) A General Platform for Inquiry Learning, Proceedings of the 6th Int'l Conference on Intelligent Tutoring Systems, Lecture Notes in Computer Science 2363, 681-697, France.
3. Murray, T., Woolf, B. & Marshall, D. (2004). Lessons Learned from Authoring for Inquiry Learning: A tale of three authoring tools. ITS 2004, Brazil.
4. Bruno, M. (2000). Student-active learning in a large classroom. Presented at Project Kaleidoscope 2000 Summer Institute. Keystone, Colorado. http://carbon.hampshire.edu~mbruno/PKAL2000.html
5. Bruno, M.S. & Jarvis, C. D. (2001). It's Fun, But Is It Science? Goals and Strategies in a Problem-Based Learning Course. *The Journal of Mathematics and Science: Collaborative Explorations, 4*(1): 25-42.

6. Aleven, V. & Ashley, K. D. (1997). Teaching Case-Based Argumentation Through a Model and Examples: Empirical Evaluation of an Intelligent Learning Environment. In B. du Boulay & R. Mizoguchi (Eds.), Artificial Intelligence in Education, Proceedings of AI-ED 97 World Conference, 87-94. Amsterdam: IOS Press.
7. Krajcik, J., Blumfeld, P., Marx, R., Bass, K., Fredricks, J. and Soloway, E. (1998). Inquiry in project-based science classrooms: Initial attempts by middle school students. The Journal of the Learning Sciences, 7 (3and4), 313-350.
8. White, B., Shimoda, T., Frederiksen, J. (1999). Enabling students to construct theories of collaborative inquiry and reflective learning: computer support for metacognitive development. *International J. of Artificial Intelligence in Education, 10*, 151-182.
9. Suthers, D., Toth, E. & Weiner, A. (1997). An integrated approach to implementing collaborative inquiry in the classroom, *Proceedings of the 2nd Int'l Conference on Computer Supported Collaborative Learning*.
10. Alloway, G., Bos, N., Hamel, K., Hammerman, T., Klann, E., Krakcik, J., Lyons, D., Madden, T., Margerum-Leys, J., Reed, J., Scala, N., Soloway, E., Vekiri, I., & Wallace, R. (1996). Creating an Inquiry-Learning Environment Using the World Wide Web. *Proceedings of the Int'l Conference of Learning Sciences.*
11. Murray, T., et. al. (2005). Evaluating Inquiry Learning through Recognition Based Tasks, International Conference on AIED, Amsterdam.
12. Aleven V., Koedinger, K. R., & Popescu, O. (2003). A Tutorial Dialog System to Support Self-Explanation: Evaluation and Open Questions. In U. Hoppe, F. Verdejo, & J. Kay (Eds.), *Proceedings of the 11th International Conference on AIED, AI-ED 2003* (pp. 39-46). Amsterdam: IOS Press.
13. White, B., Frederickson, J. Developing Metacognitive Knowledge and Processes: The Key to Making Scientific Inquiry and Modeling Accessible to All Students. *An Overview of the ThinkerTools Inquiry Project*. 1995.
14. Suebnukarn, S., Haddawy, P. "A Collaborative Intelligent Tutoring System for Medical Problem-Based Learning." *International Conference on Intelligent User Interfaces.* (Jan. 3-16). 2004.
15. Paolucci, M., Suthers, D., Weiner A. "Automated Advice-Giving Strategies for Scientific Inquiry." *Proceedings of ITS* (June 12-14, 1996) 372-381.
16. Shute, V., Glaser R. "A large-Scale Evaluation of an Intelligent Discovery World: Smithtown." *Interactive Learning Environments 1*. 1990. pp. 51-57.

How "Consciousness" Allows a Cognitive Tutoring Agent Make Good Diagnosis During Astronauts' Training

Daniel Dubois, Roger Nkambou, and Patrick Hohmeyer

GDAC Laboratory, University of Quebec in Montreal
P.O. Box 8888, Centre-ville Station, Montreal, Quebec, Canada
dubois.daniel.2@courrier.uqam.ca, nkambou.roger@uqam.ca,
hohmeyer.patrick@courrier.uqam.ca

Abstract. Striving in the real world is more and more what artificial agents are required to do, and it is not a simple task. Interacting with humans in general, and with students in specific, requires an awful lot of subtlety if one is to be perceived as a great tutor and a pleasant fellow. Similarly, the more various types of information an artificial agent senses, the more apt it may be. But then comes the need to process all this stuff, and that can overwhelm even the most powerful computer. «Consciousness» mechanisms can help and sustain an apt tutor, allowing it to consider various sources of information in diagnosing and guiding learners. We show in the present paper how they effectively support theses processes in the specific context of astronauts training on the manipulation of the Space Station Robotic Manipulation System, Canadarm2.

1 Introduction

In the training of astronauts, simulators turn out to be the only way for some tasks. In such a training context, free roaming may suit some, but being guided by an experienced coach has been shown to help significantly the learner [1]. But it never is an easy task, with lots of aspects to look after simultaneously, amounting to a good testbed for a tutoring agent's architecture.

A virtual simulator of the International Space Station (ISS) and its Robotic Manipulator System, Canadarm2, has been developed in our lab [2]. Designed to give mostly reactive feedback that helps enhance spatial and situational awareness [3], its tutoring capabilities are being augmented by those of our "conscious" cognitive tutoring agent. In this paper, we demonstrate how consciousness may allow an agent to cope with a complex environment, and diagnose learner errors to provide remediation in real-time. The paper is organized as follows: first, we clarify the concept of consciousness. Then we present Baars' theory after which our consciousness model is built, and the "conscious" tutor's architecture deriving from it. In a fourth section, we illustrate the training domain on which the "conscious" tutor is applied, and finally, we show how a diagnostic process unfolds through the cooperative work of the various components of the architecture.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 154 – 163, 2006.

2 What Consciousness Is About

Brackman and Minsky [4] said that the word consciousness/conscience is a suitcase concept. It melts together a nasty gang of different ideas, just as do intelligence, learning, memory, and intuition. Aside from more popular notions, Block’s individuation of various forms of consciousness [5] shows that it may also be thought of as including the internal mechanisms that realize consciousness. They allow us to represent and make the content of the present experience available to the rest of our internal, unconscious processes. The specific phenomenon he calls *access* consciousness is how we gain access to otherwise unreachable resources. Another aspect of the word aims at those faculties that keep us informed about the activities of our senses; he calls it monitoring consciousness. Block naturally also mentions consciousness as including the idea of self-consciousness (being aware of existing as an entity distinct from the rest of the world) and the dreaded *phenomenal consciousness*. This last flavor, and the self-consciousness, are those that most instinctively make us react negatively about the “consciousness” that machines might have.

3 Baars Global Workspace Theory

Baars [6] has proposed a theory that unifies many previous efforts in describing and modeling consciousness. In his view, consciousness accomplishes nine functions among which we find: Adaptation and learning, Contextualizing, Prioritizing and access control, Recruitment, Decision-making and Self-monitoring. All these functions, and all brain operations are carried by a multitude of globally distributed, unconscious specialized processors. Each has a limited ability, and a limited range of knowledge processing, but is very efficient. When one of them cannot complete its operation, it will try to make this situation published, to make it known to all other processors, in other words have the whole system become conscious (aware) of the situation. Processes that recognize it and know what to do about it, or how to take over from this step, will grab a copy of the information and process it, without any central coordination mechanism. What part of the system will effectively respond is shaped by the context: current goal and plan, current mood and emotions, complementary information brought back by other systems such as memories or emotional systems, processes currently in the forefront, etc.

The situation brought to consciousness is described by coalitions of processors presenting various aspects. Many such coalitions may try to have their information broadcast, but only one can have access to the broadcasting facility at a time, as only one situation can occupy the conscious "space" (one is conscious of only one idea, aspect, situation at a time). So, there is "competition" to gain access to this global workspace.

In summary, the various types and levels of consciousness filter and select information, glue the whole system together, and are the means for holding deliberations on deciding how to modify a plan or a concept, and about how to adapt the action to the specific situation; we often do these deliberations by "talking to ourselves". Operations that remain unconscious bring swiftness to the processing.

4 Our Implementation of Baars' Theory

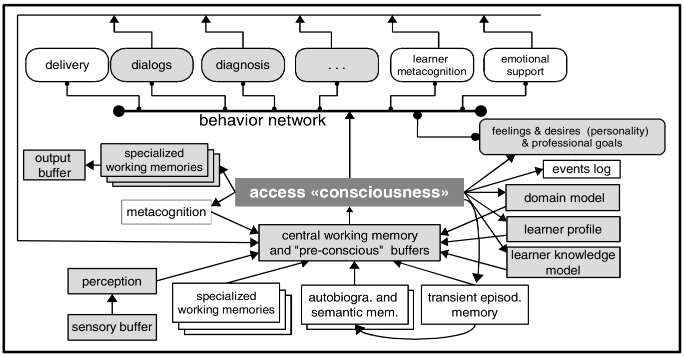


Fig. 1. Conceptual architecture of the "conscious" cognitive tutor. Grayed boxes indicate which functions are in the process of being implemented.

Our architecture is essentially rooted in professor Franklin's IDA architecture [7] and owes much to it, but brings some domain-specific extensions (such as learner modeling) and some modifications to the implementation.

The conceptual architecture (Fig.1) covers every major aspects of cognition, with many parallels to the physiology of the brain. Other functions that are internal to some entity, or result from the collaboration of many, remain hidden: action selection, deliberation, learning, automatization, and how the agent's feelings, desires, emotions and inhibitions influence all of these. At the center of the diagram, creating a connection between all of the entities, is the *access "consciousness"*. In the coming paragraphs, we will briefly describe various entities of the architecture; our goal here is to give a general understanding of how they work and why they react. This will allow us to illustrate their interactions in the section "Coaching Astronauts".

Senses and Perception. Incoming messages from any source land into the agent's sensory buffer. Every dynamic aspect of the "environment" (see Fig.4) is documented there: Canadarm2 configuration (rotation angle of the seven joints), position of the payload, camera selected on each of the three monitors along with its dynamic attributes (zoom, pitch and pan angles), visibility of objects. If the event was not manipulation related, other types of information would be supplied, such as exercise type and specifications. The perceptual *codelets* (to be explained in Behavior Network section below) scan the buffer and activate nodes in the Perceptual Network (PN), to which they transfer their data. These nodes represent the information and give it semantic meaning (concepts the agent can recognize: **"Canadarm2 manipulation", "user answer", etc.**). They also grant it importance on a semantic basis. A language has been developed to implement the communication between the Simulator and the sensory buffer.

A *Distributed* Learner Model. The learner model is distributed throughout the architecture. Transient Episodic memory (TEM) records a quite detailed account of perceived events, many related to the learner's actions. In its static part, the Learner Profile (LP) contains psychological information, including learner's learning style; its dynamic part tracks learner's mood and emotional state. Learner Knowledge Model (LKM) holds facts and the learning history, infers knowledge and trends, and computes statistics. It also volunteers information when it deems appropriate, eventually priming some *“feeling”* in the agent (described in the Personality section below). The LKM is the main mechanism in establishing the causes of the learner’s difficulties. It is implemented as a bayesian network coupled to codelets for its outputs. Factual knowledge nodes record evidences found in the access consciousness broadcasts.

The Behavior Network (BN) and the Codelets. Based on an idea from Maes [8] and modified by Negatu and Franklin [9]), the Behavior Network holds the repertoire of the agent’s know-how (subject matter delivery, message building, dialog, diagnosis, etc.) and offers means to decide upon which to activate. The Behavior Network (BN) is a hierarchical network of streams of behaviors. Each behavior node specifies its necessary preconditions and indicates the effects it should have on the environment. This network accomplishes high-level planning and selects the most appropriate action through the bidirectional flow of activation “energy” from node to node through the links that connect effects and preconditions.

Negatu and Franklin also modified Maes model so that each behavior is realized by a collection of *codelets* (simple unintelligent agents) that connect the BN to the rest of the agent. Codelets are the functional reproductions of Baars’ processes. They do not appear explicitly on the diagram, yet are essential to the architecture. They accomplish a major part of the operations in the agent, they render effective many functionalities, and they connect all of the entities to the access “consciousness”. For instance, *information codelets* carry information from one place to another, *expectation codelets* make sure expected results are met. Most eventually meet in the central Working Memory, where coalitions are formed.

The BN serves as the coordinator for the agent’s external actions, and generally counts on other functionalities to render a service. As an example, when all the conditions are met, the Diagnosis sub-network is set into motion and acts as the conductor for that operation. But for the operation to unfold, other agent’s functionalities have to respond, such as the LKM supplying information when it recognizes some request coming from the BN.

The Personality of the Agent. "Feelings", "desires" and "emotions", in our architecture, are the basic mechanisms forming the agent’s personality. Feelings and desires are the motivational mechanisms that feed the Behavior Network with activation energy and so orient action selection in line with the agent’s goals; emotions intervene elsewhere, at other times, during memorization and deliberations for instance [7]. When specific states arise in the agent, among other things they stimulate its feelings, and so call for a reaction that respects the agent’s “personality”. Such states may appear when the Learner Knowledge Model signals some important flaw in the learner’s knowledge, or they may appear after the broadcast of a perceived external situation, such as the possibility of a collision while the user is manipulating Canadarm2.

Access "Consciousness". This mechanism selects the most *important* coalition of codelets and broadcasts its information, allowing all other systems to become aware of the situation. This is crucial for the collaboration of the parts, for instance in reaching a diagnosis. For a better readability of the text, we will drop the quotation marks around the words *conscious* and *consciousness*; this is not intended as a statement about the truthfulness of our implementation.

4.1 Some Functional Aspects of the Architecture

Coaching is about subtlety, about considering lots of aspects, about cutting through noisy information, about finding patterns. Processing various sources and aspects is very natural for our conscious architecture: every functionality is made aware of important elements and can contribute information to the description of the situation, towards a decision, or towards action. For example, when the tutor perceives a lack of action from the learner, it will try to determine the nature of this apparent inaction. Its episodic memory will (try to) recall somewhat similar situations and bring them back into Working Memory so that they can be analyzed for commonalities. If still uncertain, the tutor may *feel* the need to interact with the user to clarify the case, and "advertise internally" its intention of doing so. Upon the eventual publication of this intention, the Learner Psychological Profile will respond with a recommendation about the style to adopt, based on the original learner's specification of his preferences but modified by the tutor's experience with him. At the same time, another aspect of the Learner Profile may send the information stating that the learner wants minimal interventions, effectively supplying to the debate some inhibition that will modulate the behavior that the tutor finally exhibits. Should the delay last much longer, the feeling about the need to intervene will grow stronger and may overcome the inhibition not to do so. Such internal debating offer great flexibility in the behavior adaptation. They can take place thanks to the loose coupling brought about by the access consciousness, and they can become as rich as designers wish them to become. Since the architecture can accommodate any number of functionalities towards finer behavioral decisions, diagnosis, and the subtle "human touch".

Every action taken by the user and every event in the environment is copied to the cognitive tutor. These events may not spur an immediate response, but even their simple absorption by the various memories creates a historical context, modifies statistics about the session and about the learner, and prepares future reactions.

5 Coaching Astronauts

The cognitive conscious tutoring agent (henceforth called the cognitive tutor, or more simply the tutor) uses the International Space Station (ISS) simulator as its major input and output channels, or more cognitively speaking, its senses and effectors. The three simulated monitors, mimicking the workstation aboard the ISS, allow the astronaut to see the Station and Canadarm2 through a choice of three among the dozen or so available cameras. It is like driving a car with just half the windshield not covered by mud, and no side windows. That explains the need for special training on spatial and situational awareness.

As any human coach will testify, coaching is not a trivial task. Many aspects must be looked after, often at the same time, and crucial information must be prioritized. The coach must be able to see trends, anticipate consequences, know when to intervene, and even adopt the proper interaction style! This turmoil constitutes an excellent test-bed for our architecture's ability to successfully process lots of varied information in real-time. We will demonstrate in the next section how some of the various parts of the tutor collaborate, and how they participate in deliberations to reach a diagnosis just as humans do with "internal debates" [6]. However, we will restrict the number of functionalities involved, keeping silent the agent’s “emotions”, the various memories, and much of the agent’s “personality”.

5.1 An Inspirational Collaboration to Reach a Diagnosis

In this section, we use the initial steps of some scenario where the astronaut has to move a payload. It illustrates how the lower levels of consciousness recognize (and eventually prioritize) information, and how the access consciousness takes over to select the most important one to make the whole agent aware of.

Let say the astronaut selects, in the Task menu, the “Move a payload” task. The arrival of a message about this fact in the sensory buffer stimulates the lower *awareness* levels of the agent. Without going into the details of the process, let’s assume that this request reaches the Working Memory (WM) of the agent. At this point, the agent is not yet “conscious” of that request. It has to be *broadcast* inside the agent so that every functionalities become aware of it and may decide to respond. In the WM, the coalition of codelets that contain the details of the request (date-time-menu Selection=Move_a_payload) becomes in competition with other coalitions that may already be there, all hoping to be selected for broadcasting by the access consciousness. As one may be conscious of only one idea, one situation at a time [6], only the most important information is broadcast (elected either for its recency, its urgency, its intensity, or because it was awaited for). Let’s assume here that the coalition concerning the menu selection is selected and broadcast. The *drives* mechanisms (*feeling* and *desires*) react and make the agent “feel” the need to respond to that request. The feeling for the need to select an appropriate exercise is stimulated and starts feeding energy into the “Select an exercise” stream of behaviors. The first behavior node of the activated stream sends its information codelets to the working memory to advertise the need for a suggestion of exercise. Supposing this request is selected and broadcast, the Learner Knowledge Model reacts sends to the WM an information codelet containing the competency level of the astronaut. It could also supply other information if appropriate, such as observed deficiencies in this field, or aspects that have not been covered yet with previous trainings. These are the information the Domain Model (DM) needs to respond with its recommendation of exercise.

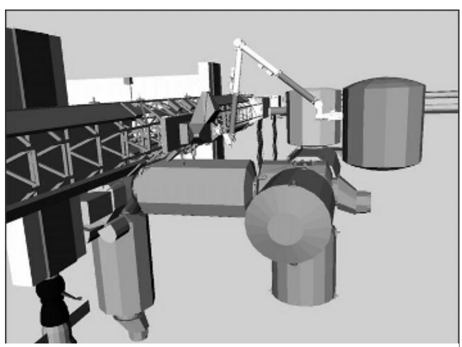


Fig. 2. Initial position and configuration of Canadarm2 for the Payload moving task submitted to the astronaut

We will from now on forego the “coalition selection-broadcasting” part of every cycle. Just remember that a coalition of information is always subject to competition with other coalitions in the WM, and is never certain to get selected and published. Any other source may submit information, at any time, that has more importance than the information coming from the process under way. This permits a high reactivity and a total flexibility for the system.

The Simulator takes over, shows the initial configuration of Canadarm2 and position of the payload (Fig.2), then shows the destination for the payload after the astronaut acknowledged the image. When the astronaut acknowledges the destination image, the cognitive tutor is informed of that fact and starts timekeeper codelets. Thirty seconds elapse, and no action is recorded. This is "announced" by a timekeeper codelet, and its broadcast prompts a reaction from the BN. The Diagnosis stream's first behavior, through its information codelets, goes on advertising the need for the probable causes of this inactivity period. Indeed, the stream has no Primary Diagnosis node for inactivity. LKM responds to this broadcast with the most likely diagnosis: inexperience with that kind of task and with Canadarm2, since this is the astronaut's first session on the Simulator, and we are at the beginning of the session (so, fatigue or distraction are not likely causes).

Hearing this information, the *feeling* of the need for remediation gets stimulated and starts feeding energy into the Remediation stream, while the energy feed into the Diagnosis stream slows down. The first behavior in that stream asks for remediation suggestions, and the Domain Model responds with the proper remediation for the cause to the diagnosed difficulty: "offer assistance". The next behavior in the Remediation stream sends the advertisement of its need for a message asking whether the user might like assistance. So, in the next few time steps, a message is deposited in the output buffer, which opens a window on the screen to this end, with choices for an answer by the user. The astronaut accepts the help offer. This reply very likely gets to be broadcast. The LKM contributes the low experience level of the user. So, a review of the initial steps (a check-list) seems in order, and that's what the DM will suggest (as always, in the form of an information codelet that goes into the WM). Then, the next Remediation behavior will serve to issue a request for the tutorial, to which another stream in the BN will respond, the Tutorial stream.

We won't go any further in this scenario, as our goal is to illustrate the way diagnosis is conducted in our architecture. It very much resembles human "internal debates" where "answers" come into our mind after we have voluntarily put some words or idea into our Working Memory, or how an idea "pops-up" when our Long Term Memory brings back an element of information somehow associated with the situation (we did not illustrate this aspect here).

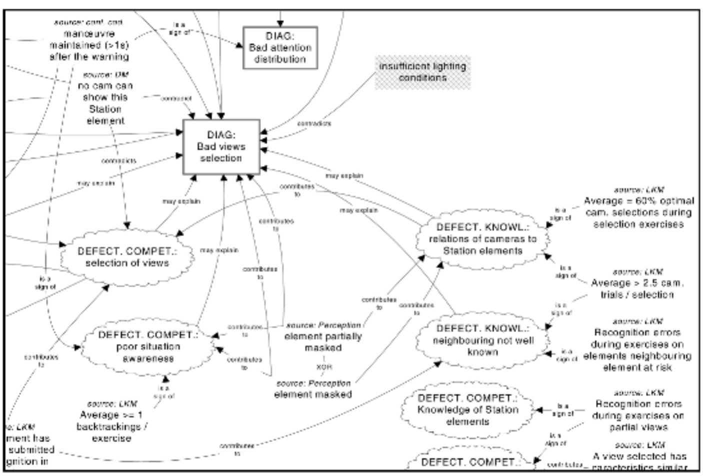


Fig. 3. Portion of the causal network after which the LKM and the Diagnosis stream of the BN are modelled

Let's now take another example of a diagnosis that happens further into a payload moving task, when a collision becomes imminent between Ca-nadarm2 and a component of the Space Station. To see more of a diagnosis taking place, let's now assume the astronaut has some experience with the Simulator and that some statistics have accumulated in the LKM. In the coming scenario, we will limit the complexity of the deliberations to the intervention of the BN and the LKM.

5.2 Reacting to a Collision Risk

When a close proximity or a collision risk happens, the tutor has to decide what to do with the situation. In Fig.4, we see the Simulator's user interface and how the views have been set-up on the three monitors. Monitor 1 (left) shows part of the JEM modules. This is sufficient to insure clearance of the payload, but not enough to show all of Canadarm2. On Monitor 3, both Canadarm2 and the JEM modules most at risk are completely visible. That is quite fine, unless the astronaut becomes so focused on Monitor 1 that he won't even notice the collision risk warnings. The astronaut continues with his "descent" manoeuvre, not having noticed the proximity warning; this continuation is noted by the attention codelet and makes it elevate its activation level, increasing its likeliness of being selected for the broadcasting of its information. In the likely event, the LKM will make good note of that information. Still unaware of the problem, the astronaut pursues the manoeuvre, and a "collision risk" warning comes on, on a red background. Again, this event is copied to the tutor, and the same processing happens. If an *opposition* codelet was previously sent by the LP to oppose intervening when a collision risk warning came on, this time, no opposition codelet is issued by the LP: intervention is in order! As the tutor has accumulated little evidence about the learner's knowledge (this is only a second session), the DM's suggestion for a first-level (immediate) remediation is to indicate the main cause of the collision. The Remediation stream advertises the ne-ed for a message exposing the situation. The Di-alog stream lights up and its codelets assemble a first message that appears on screen with the help of the Output Buffer:

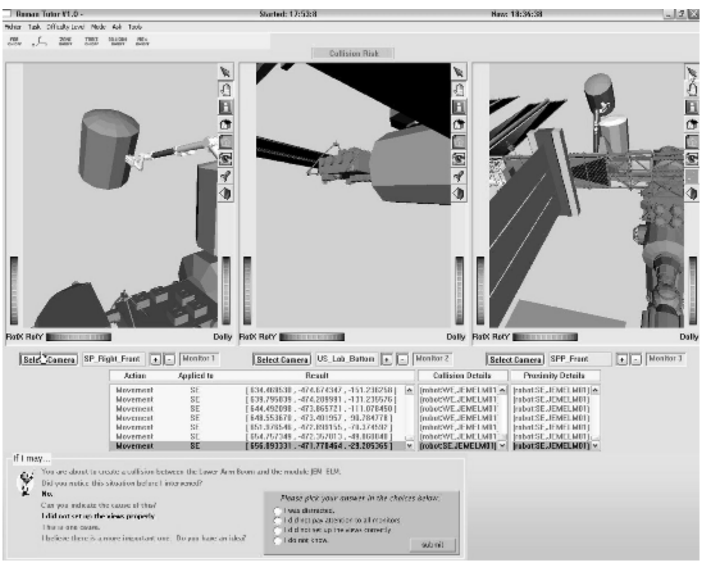


Fig. 4. The Simulator's User Interface emulates the three monitors of the workstation embarked on the ISS. Here, the three views selected provide an imperfect but sufficient information source.

- You are about to create a colli-sion between the Lower Arm Boom and the module JEM-ELM. Did you notice this situation before I intervened?

- "No", replies the astronaut with a click of the mouse on a choice button. A "Yes" could have meant that he was trying a very risky manoeuvre; this possibility would have needed investigation. A "No" confirms the diagnosis and strengthens the diagnosis node in the LKM about the user's attention having been focused on a single monitor. It also serves as a precondition in the BN Remediation stream to pursue the dialog in the same vein:

- "Can you indicate the cause of this?"

- "I did not set up the views properly". Upon the broadcasting of this answer, the LKM responds with two codelets containing the activation levels of this cause and of the main one. The activation level of the cause indicated by the astronaut is inferior to that of the main one, as compared by the expectation codelet. Now, that fact needs to be published, along with the fact that the astronaut did not identify the right cause. Let's assume that they do get published. Then the Remediation stream has to do a subtle loop to keep helping the learner uncover the major cause. :

- "This is one cause. I believe there is a more important one."

- "I do not know." is the reply selected by the astronaut. This user answer, recognized as ignorance, then taken in charge by an expectation codelet, is submitted for broadcasting. At this point, the Remediation stream states that a hint is needed and advertises this need in the Working Memory. LKM is apt at processing that request; it responds to this broadcast with codelets containing the facts supporting the diagnosis. The expectation codelet waiting for this information creates a coalition of the kind "MessageToBuild-Hint-Fact = the two colliding elements are apparent on the same monitor-Monitor 3-Fact = no alteration of manipulation after collision risk warning". The Dialog stream uses that information to build the next message:

- "Hint: the two colliding elements are apparent on the same monitor (Monitor 3). Also, you did not react to the collision risk warning. Now, can you see the major cause I am thinking about?" With two clicks on the list offered, the astronauts indicates "I did not pay attention to every monitor." and "I did not pay attention to the collision risk warning." After submitting his answer, the cognitive tutor processes it and concludes:

- "Right! In future manipulations, make sure you check every information source."

6 Conclusion

Implementing coaching in computers aims at allowing fine, personalized, just-in-time support to every learner. But coaching is complex, requiring adaptation to unpredictable combinations of task, learner's competences, personality, and mood. Humans are able to learn to do it well; hence they should be the reference model for a tutoring agent's architecture. Our agent architecture has the potential of reproducing the performance of human coaches because it reproduces the human mind functionalities. Its multifaceted processing is rendered possible thanks to the loose coupling of the faculties through the *access consciousness*. Our conscious agent's architecture being apt at taking into consideration a great number of factors, shows the potential of adapting the human-machine interactions with a nice human touch.

The next steps are concerned with elaborating the remediation part of the agent, for one thing, and implementing a flexible way to build the textual outputs. We will also look into creating the agent’s metacognition, so that it can improve itself. Developing a methodology and tools to put new agents within the reach of non computer-literate designers would also be a major step towards using he agent for all sort of cognitive experimentations. And this is part of the venture we are pursuing.

Acknowledgments. Our thanks go to the Canadian Space Agency and the Natural Sciences and Engineering Research Council (NSERC) for their logistic and financial support. Special thanks to professor Franklin and to the University of Memphis, who graciously granted us access to their technology (IDA). We also thank professors Pierre Poirier and Froduald Kabanza, and Khaled Belghith for their help.

References

1. Conati, C., VanLehn, K.: Toward Computer-Based Support of Meta-Cognitive Skills: a Computational Framework to Coach Self-Explanation. International Journal of Artificial Intelligence in Education 11 (2000) 389-415.
2. Kabanza, F., Nkambou, R., Belghith, K., Hartman, L.: Path-Planning for Autonomous Training on Robot Manipulators in Space. Proceedings of IJCAI’2005, (2005) 1729-1732.
3. Nkambou, R., Belghith, K., Kabanza, F., Khan, M.: Supporting Training on Canadarm Simulator using a Flexible Path Planner. Artificial Intelligence in Education. Amsterdam:IOS Press (2005) 953-955.
4. Brackman, J., Minsky, M.: Consciousness is a Big Suitcase. Edge (Feb. 27, 1998). [on-line].
5. Block, N.: Some Concepts of Consciousness. Philosophy of Mind: Classical and Contemporary Readings. D. Chalmers ed. Oxford University Press (2002).
6. Baars, B.J.: A cognitive theory of consciousness. NY:Cambridge University Press (1988)
7. Franklin S., Baars, B.J., Ramamurthy, U., Ventura, M.: The Role of Consciousness. In: Memory. Brains, Minds and Media. 1(bmm150) (2005) 1-38.
8. Maes, P.: How to Do the Right Thing. In: Connection Science Journal. 1(3) (1989) 291-323.
9. Negatu, A.S., Franklin, S.: An Action Selection Mechanism for "Conscious" Software Agents. Cognitive Science Quarterly 2 (2002) 363-386.

Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement

Hao Cen, Kenneth Koedinger, and Brian Junker

Carnegie Mellon University
5000 Forbes, Pittsburgh, PA, USA
hcen@andrew.cmu.edu, koedinger@cmu.edu, brian@stat.cmu.edu

Abstract. A cognitive model is a set of production rules or skills encoded in intelligent tutors to model how students solve problems. It is usually generated by brainstorming and iterative refinement between subject experts, cognitive scientists and programmers. In this paper we propose a semi-automated method for improving a cognitive model called Learning Factors Analysis that combines a statistical model, human expertise and a combinatorial search. We use this method to evaluate an existing cognitive model and to generate and evaluate alternative models. We present improved cognitive models and make suggestions for improving the intelligent tutor based on those models.

1 Introduction

A cognitive model is a set of production rules or skills encoded in intelligent tutors to model how students solve problems. (Production, skill, and rule are used interchangeably in this paper.) Productions embody the knowledge that students are trying to acquire, and allows the tutor to estimate each student's learning of each skill as the student works through the exercises [4].

A good cognitive model captures the fine knowledge components in a curriculum, provides tailored feedback and hints, select problem with difficulty level and learning pace matched to individual students, and eventually, improves student learning. However, initial models are usually generated by brainstorming and iterative refinement between subject experts, cognitive scientists and programmers. These first pass models are best guesses and our experience is that such models can be improved.

In this paper, we propose a method called Learning Factors Analysis (LFA) and use it to answer three questions relevant to the field of intelligent tutoring systems.

1. How can we describe learning behavior in terms of an existing cognitive model? We need to identify the initial difficulty level of each production and how fast can a student learn each rule (i.e., what is the learning rate). We can then provide parameters that indicate student performance on this set of rules and how that performance improves with practice and instruction on those rules.

2. How can we evaluate and improve a cognitive model in an inexpensive way? We need to identify the causes of the deviation from the deterministic cognitive model, define the measures of a model's complexity and fit, and mine the student-tutor log data.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 164 – 175, 2006.

3. How can we use the information from LFA to improve the tutor and the curriculum? We need to identify over-taught or under-taught rules, and even “hidden” knowledge components within them. As a result, we can adjust their contribution to curriculum length without compromising student performance.

2 Literature Review

One measure of the performance of a cognitive model is how the data fit the model. Newell and Rosenbloom found a power relationship between the error rate of performance and the amount of practice [13]. Depicted by equation (1), the relationship shows that the error rate decreases according to a power function as the amount of practice increase. The curve for the equation is called a “learning curve”.

$$Y = aX^{b}. \tag{1}$$

where
Y = the error rate
X = the number of opportunities to practice a skill
a = the error rate on the first trial, reflecting the intrinsic difficulty of a skill
b = the learning rate, reflecting how easy a skill is to learn

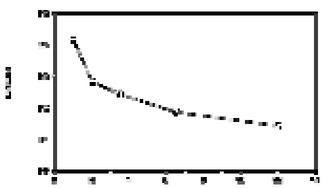


Fig. 1. A power law learning curve

The learning curve model has been used to visually identify non-obvious or “hidden” knowledge components. Corbett and Anderson observed that the power relationship might not be readily apparent in some complex skills, which have blips in their learning curves [5], as shown in figure 2. They also found the power relationship holds if the complex skill can be decomposed into subskills, each of which exhibits a smoother learning curve.

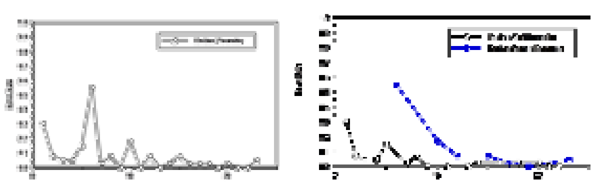


Fig. 2. A learning curve with blips (*left*) split into two smoother learning curves (*right*)

As seen in the graphs above, the single production Declare-Parameter produces a learning curve with several blips. However by breaking it into two more specific productions, Declare-First-Parameter and Declare-Second-Parameter, the model becomes more fine-tuned and recognizes that the skills are different. The knowledge decomposition (considering parameter position) that was non-obvious from the original model became revealed on closer inspection of learning curve data.

Other approaches to model refinement include having a simulated student to find incorrect rules and to learn new rules via human tutor intervention [16], using theory refinement to introduce errors to models incorrect student behaviors [1] and using Q-matrix to discover knowledge structure from student response data [15,2]. Compared with the simulated student approach, our method does not require building a simulated student. The theory refinement approach starts with an initial knowledge base and keeps correcting errors in the knowledge base from error examples until the knowledge base is consistent with the examples. It may lead to overfit the examples. The Q-matrix approach was used to automatically extract features in the problem set. The model found by this approach may be similar to the model adding or merging difficulty factors in our method.

3 The Cognitive Model and Its Data Under Investigation

We illustrate the LFA methodology using data obtained from the Area Unit of the Geometry Cognitive Tutor (see http://www.carnegielearning.com). The initial cognitive model implemented in the Tutor had 15 skills that correspond to productions or, in some cases, groups of productions. The productions are

Circle-area – Given the radius , find the area of a circle
Circle-circumference – Given the diameter, find the circumference of a circle.
Circle-diameter -- Given the radius or circumference, find the diameter of a circle.
Circle-radius -- Find the radius given the area, circumference, or diameter.
Compose-by-addition – In a+b=c, given any two of a, b, or c, find the third.
Compose-by-multiplication – In a*b=c, given any two of a, b, or c, find the third.
Parallelogram-area – Given the base and height, find the area of a parallelogram.
Parallelogram-side – Given the area and height (or base), find the base (or height).
Pentagon-area – Given a side and the apothem, find the area of a pentagon.
Pentagon-side – Given area and apothem, find the side (or apothem).
Trapezoid-area – Given the height and both bases, find the area of a trapezoid.
Trapezoid-base – Given area and height, find the base of a trapezoid.
Trapezoid-height – Given the area and the base, find the height of a trapezoid.
Triangle-area – Given the base and height, find the area of a triangle.
Triangle-side – Given the base and side, find the height of a triangle.

Our data consist of 4102 data points involving 24 students, and 115 problem steps. Each data point is a correct or incorrect student action corresponding to a single production execution. Table 1 displays typical student action records in this data set. It has five columns – student, success, step, skill, and opportunities. Student is the names of the students. Success is whether the student did that step correctly or not in the fist attempt. 1 means success and 0, failure. Step is the particular step in a tutor problem the students are involved in. "p1s1" stands for problem 1 step 1. Skill is the production rule used in that step. Opportunities mean the number of previous times to use a particular skill. It increments every time the skill is used by the same student, and can be computed from the first and fourth columns.

Table 1. The sample data

Student	Success	Step	Skill	Opportunities
A	0	p1s1	Circle-area	1
A	1	p2s1	Circle-area	2
A	1	p3s1	Circle-area	3

4 Learning Factor Analysis

LFA has three components: a statistical model that quantifies the skills, the difficulty factors that may affect student performance in the tutor curriculum, and a combinatorial search that does model selection.

4.1 The Statistical Model

The power law model applies to individual skills and does not typically include student effects. Because the existing cognitive model has multiple rules, and the data contains multiple students, we made four assumptions about student learning to extend the power law model.

1. Different students may initially know more or less. Thus, we use an *intercept* parameter for each student.

2. Students learn at the same rate. Thus, *slope* parameters do not depend on student. This is a simplifying assumption to reduce the number of parameters in equation 2. We chose this simplification, following Draney, Wilson and Pirolli [7], because we are focused on refining the cognitive model rather than evaluating student knowledge growth.

3. Some productions are more likely to be known than others. Thus, we use a *intercept* parameter for each production.

4. Some productions are easier to learn than others. Thus, we need a *slope* parameter for each production.

Based on the assumptions, we developed a multiple logistic regression model.

$$\ln\left(\frac{p}{1-p}\right) = \sum \alpha_i X_i + \sum \beta_j Y_j + \sum \gamma_j Y_j T_j \,. \tag{2}$$

Where
p = the probability to get an item right
X = the covariates for students
Y = the covariates for skills
T = the covariates for the number of opportunities practiced on the skills
Y T = the covariates for interaction between skill and the number of practice opportunities for that skill

α = the coefficient for each student, i.e. the student intercept
β = the coefficient for each rule, i.e. the production intercept
γ = the coefficient for the interaction between a production and its opportunities, i.e. the production slope.

4.2 Difficulty Factors

A difficulty factor refers specifically to a property of the problem that causes student difficulties (e.g., first vs. second parameter in figure 3). By assessing the performance difference on pairs of problems that vary by one factor at a time, we can identify the hidden knowledge component(s) that can be used to improve a cognitive model [9]. Difficulty factors have been used to empirically evaluate a small number of alternative models [6, 10, 11].

In our study, subject experts identified four multi-valued factors for the Area Unit of the Geometry Tutor. Table 2 lists their names and values.

Table 2. Factors for the Area Unit and their values

Factor Names	Factor Values
Embed	alone, embed
Backward	forward, backward
Repeat	initial, repeat
FigurePart	area, area-difference, area-combination, diameter, circumference, radius, side, segment, base, height, apothem

"Embed" indicates whether a shape is embedded in another shape. Consider two tutor problems requiring the same production rule CIRCLE-AREA at some step in the problem. In one of the problems, the circle is embedded in a square; while in the other one, the circle is presented alone. Students may find it harder to find the area of circle when it is embedded in another figure because extra effort is necessary to find the circle and its radius. "Backward" means whether the production rule to be used is in its backward form of a taught formula, or its forward form. The forward form of Compose-by-addition is S = S1 + S2, and its backward forma is S1 = S - S2. "Repeat" indicates whether the production rule has been used previously in the same problem. "FigurePart" indicates the part of the figure in the geometry shape to be computed.

4.3 Combinatorial Search

The goal of the combinatorial search is to do model selection within the logistic regression model space [8]. Difficulty factors are incorporated into an existing cognitive model through a model operator called Binary Split, which splits a skill a skill with a factor value, and a skill without the factor value. For example, splitting production Circle-area by factor Embed with value alone leads to two productions: Circle-area with the factor value alone (called Circle-area*alone), and Circle-area with the factor value embed (Circle-area*embed). Table 3 shows the data before and after a split with Embed.

Table 3. The data before and after split. Factors are incorporated in column Skill (after split). The opportunities (after split) change accordingly.

Student	Step	Skill	OPT	Factor	Skill (after split)	OPT
A	p1s1	Circle area	1	alone	CA-alone	1
A	p2s1	Circle area	2	embed	CA-enbed	1
A	P3s1	Circle area	3	alone	CA-alone	2

A* search is the combinatorial search algorithm [14] in LFA. It starts from an initial node, iteratively creates new adjoining nodes, explores them to reach a goal node. To limit the search space, it employs a heuristic to rank each node and visits the nodes in order of this heuristic estimate.

In our study, the initial node is the existing cognitive model. Its adjoining nodes are the new models created by splitting the model on the difficulty factors. We do not specify a model to be the goal state because the structure of the best model is unknown. We do specify the stopping criterion by setting the upper bound of the number of node expansions, for this paper to 50 node expansions per search.

The heuristic guiding the search is one of the two scoring functions for regression models – AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion) Each search is run twice, guided by a different heuristic each time. A good model captures sufficient variation in the data but is not overly complicated by balancing between model fit and complexity minimizing prediction risk [17]. AIC and BIC are two estimators for prediction risk, and hence used as heuristics in the search.

$$\text{AIC} = -2^*\text{log-likelihood} + 2^*\text{number of parameters}. \tag{3}$$

$$\text{BIC} = -2^*\text{log-likelihood} + \text{number of parameters} * \text{number of observations}. \tag{4}$$

where log-likelihood measures the fit, and the number of parameters, which is the number of covariates in equation 2**,** measures the complexity. Based on these two formulas, the lower the AIC or BIC, the better the balance between model fit and complexity. BIC puts a more severe penalty for complexity, leading to a smaller model than other methods.

A more interpretable metric for fit is Mean Absolute Deviance (MAD) -- the average of the absolute values of the differences between observed values and predicted values. We do not use it as a heuristic because it leads to over fitting. We include it as a measure of the improvement in the model fit.

Figure 4 illustrates A* search with AIC as the heuristic. The original model is evaluated and AIC is computed. The model is then split into a few new models by incorporating the factors. AICs are computed from each of the new model. A* selects the best one (the shaded node with value 5301) for the next model generation. A* does not always go down. It may go up to select a model (the shaded node with value 5312) to expand if all the new models have worse heuristic scores than a previous model has. After several expansions, it finds a best node with the lowest AIC value.

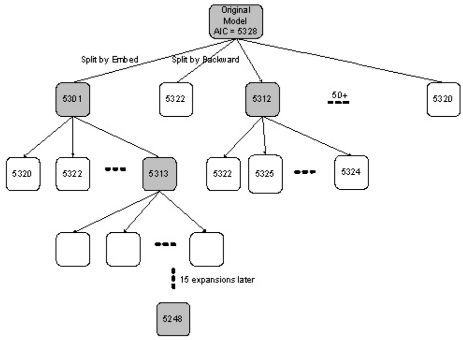


Fig. 3. Using A* algorithm search through the model space

5 Experiments and Results

5.1 Experiment 1

This experiment addresses the question -- How can we describe learning behavior in terms of an existing cognitive model? Specifically, we want to find out the learning rate and initial difficulty level of each rule, and the initial performance of students, given the data. The question is answered by fitting the logistic regression model in equation 2 and getting the coefficients. The coefficient estimates for the skills and students, and the overall model statistics are summarized in table 4.

Table 4. Statistics for a partial list of the skills, students and the overall model. Intercept for skill is the initial difficulty level for each skill. Slope is the learning rate. Avg Practice Opportunties is the average amount of practice per skill across all students. Initial Probabltiy is the estimated probability of getting a problem correct in the first opportunity to use a skill accross all students. Avg Probability and Final Probability are the success probability to use a skill at the average amount of opportunities and the last opportunity, respectively.

Skill	Intercept	Slope	Avg Opportunties	Initial Probability	Avg Probability	Final Probability
Parallelogram-area	2.14	-0.01	14.9	0.95	0.94	0.93
Pentagon-area	-2.16	0.45	4.3	0.2	0.63	0.84

Student	Intercept
student0	1.18
student1	0.82
student2	0.21

Model Statistics	
AIC	3,950
BIC	4,285
MAD	0.083

The higher the intercept of the each skill, the lower the initial difficulty the skill has. The higher the slope of the each skill, the faster students learned the skill. Pentagon-area is the hardest skill with the intercept of -2.16. Parallelogram-area is the easiest skill with the intercept of 2.14. Three skills have small slopes close to zero -- Compose-by-addition (-.04) and Parallelogram-area (-.01), Triangle-area (.03). Parallelogram-area was already mastered with an initial success probability .95. It appears that more practice on those skills does not lead to much learning gain. Interestingly, although PENTAGON-AREA is the hardest skill among all, it has the highest learning rate .45, leading to bigger improvement with more practice.

The coefficients for students measure each student's overall performance. The higher the number, the better the student performed. The AIC, BIC and MAD statistics provide a baseline for evaluating alternative models discussed below.

5.2 Experiment 2

This experiment addresses the question -- How can we improve a cognitive model? The question is answered by running LFA on the data including the factors, and searching through the model space. The improved models by LFA with BIC are summarized in table 5. The improved models by LFA with AIC is summarized in the interpretation.

Table 5. Top three improved models found by LFA with BIC as the heuristic. The table shows the history of splits and model statistics.

Model 1	Model 2	Model 3
Number of Splits:3	Number of Splits:3	Number of Splits:2
1. Binary split compose-by-multiplication by figurepart segment 2. Binary split circle-radius by repeat repeat 3. Binary split compose-by-addition by backward backward	1. Binary split compose-by-multiplication by figurepart segment 2. Binary split circle-radius by repeat repeat 3. Binary split compose-by-addition by figurepart area-difference	1. Binary split compose-by-multiplication by figurepart segment 2. Binary split circle-radius by repeat repeat
Number of Skills: 18	Number of Skills: 18	Number of Skills: 17
AIC: 3,888.67 BIC: 4,248.86 MAD: 0.071	AIC: 3,888.67 BIC: 4,248.86 MAD: 0.071	AIC: 3,897.20 BIC: 4,251.07 MAD: 0.075

LFA suggests better models, which make finer distinctions on some skills in the original model and identify which difficulty factors the subject experts thought would turn out to be psychologically important. All the better models found by AIC and BIC have better (i.e. lower) statistical scores than those of the original. For the best BIC model, its BIC is reduced by 37, and AIC by 62. The fit of the new model, as measured by MAD, is reduced by .012. The best AIC model reduces AIC by an even larger amount of 83, and increases BIC by 18. Its MAD is reduced by .02.

The improved skills common to most of the better models are Compose-by-multiplication, Compose-by-addition, Circle-area, and Triangle-area. We will discuss a few examples here.

All the new models suggest splitting Compose-by-multiplication into two skills – Cmarea and CMsegment, making a distinction of the geometric quantity being multiplied. By examining the positions of these problems in the curriculum, CMarea at the 43rd step and CMsegement at the 90th. As seen in table 6, although the final probability of CMarea is high .96, the initial probability of CMsegment is low .32. This sudden drop in the success probability at later steps corresponds to a significant blip in the learning curve as illustrated in figure 2. The distinction between different geometric quantities suggests treating the original skill differently. LFA successfully identified the blip without the need of visually inspecting learning curves.

Table 6. Success probabilities of CMarea and CMsegment

	Initial Probability	Avg Probability	Final Probability
CM*area-combination	.64	.89	.96
CM*segment	.32	.54	.60

The subject experts thought embedding a shape into another shape would increase the difficulty of a skill and identified a factor “Embed”, hoping LFA could make a distinction on it. LFA split these two skills by Embed in all the top AIC models. The three probabilities of CAalone and CAembed are shown in table 7. Does Embed make find the circle area harder? Note that problems with CAembed are introduced later in the curriculum after students have had significant practice with CAalone, about the time CAalone has reached the average probability of .81. At this point, CAembed has an initial probability of .71, indicating an increase in difficulty.

Table 7. Success probabilities of CAalone and CAembed

	Initial Probability	Avg Probability	Final Probability
CA*alone	.42	.81	.93
CA*embed	.71	.89	.92

5.3 Experiment 3

In experiment 2, LFA improved the original model by splitting skills. Experiment 3 addresses model improvement even further -- Will some skills be better merged than if they are separate skills? Can LFA recover some elements of truth if we search from a merged model, given difficulty factors?

We merged some skills in the original model to remove some of the distinctions, which are represented as the difficulty factors. Circle-area and Circle-radius are merged into one skill Circle; Circle-circumference and Circle-diameter into Circle-CD; Parallelogram-area and Parallelogram-side into Parallelogram; Pentagon-area, and Pentagon-side into Pentagon; Trapezoid-area, Trapezoid-base, Trapezoid-height into Trapezoid. The new merged model has 8 skills -- Circle, Crcle-CD, Compose-by-addition, Compose-by-multiplication, Parallelogram, Pentagon, Trapezoid, Triangle.

Then we substituted the original skill names with the new skill name in the data, ran LFA including the factors, and had the A* algorithm search through the model space. The improved models by LFA with BIC are summarized in table 8. The improved models by LFA with AIC are summarized in the interpretation.

Table 8. Top three improved models found by LFA with BIC as the heuristic

Model 1	Model 2	Model 3
Number of Splits: 4	Number of Splits: 3	Number of Splits: 4
Number of skills: 12	Number of skills: 11	Number of skills: 12
Circle *area Circle *radius*initial Circle *radius*repeat Compose-by-addition Compose-by-addition*area-difference Compose-by-multiplication*area-combination Compose-by-multiplication*segment	All skills are the same as those in model 1 except that 1. Circle is split into Circle *backward*initial, Circle *backward*repeat, Circle*forward, 2. Compose-by-addition is not split	All skills are the same as those in model 1 except that 1. Circle is split into Circle *backward*initial, Circle *backward*repeat, Circle *forward, 2. Compose-by-addition is split into Compose-by-addition and Compose-by-addition*segment
AIC: 3,884.95 BIC: 4,169.315 MAD: 0.075	AIC: 3,893.477 BIC: 4,171.523 MAD: 0.079	AIC: 3,887.42 BIC: 4,171.786 MAD: 0.077

LFA fully recovered three skills (Circle, Parallelogram, Triangle), suggesting the distinctions made in the original model are necessary. LFA partially recovered two skills (Triangle, Trapezoid), suggesting the some original distinctions are necessary and some are not. LFA did not recover one skill (Circle-CD), suggesting that the original distinctions might not be necessary. LFA recovered one skill (Pentagon) in a different way, suggesting the original distinction may not be as significant as the distinction caused by another factor. We discuss a few examples here.

In BIC model 1, Circle is split into Circle*area, and Circle*radius. The other two BIC models and all the AIC models split it into Circle*backward, and Circle*forward, which are equivalent to Circle-AR*area, and Circle-AR*radius because of the one-to-one relationship between forward and area and between backward and radius. Thus, LFA fully recovers the Circle skills.

None of the models recovered Circle-CD. This suggests that it may not be necessary to have two separate skills for Circle-circumference and Circle-Diameter. It appears that once students learn the formula circumference = π*diameter, they can fairly easily apply it in the forward or backward direction.

In one of the top AIC models, Pentagon is split into Pentagon*initial and Pentagon*repeat, instead of Pentagon*area and Pentagon*side. This suggests that the distinction between the first use of a Pentagon skill in a problem and later uses of that skill in the same problem may be more significant than the distinction between the area and the side. Usually repeated use of a skill in the same problem is easier than the original use. For instance, once a student makes the *initial* relatively difficult determination that the Pentagon formula is relevant to a problem and recalls it, he need only use it again and perform easier arithmetic in *repeated* opportunities in that same problem.

5.4 Combining the Results from Experiment 1, 2, 3

By combining the results from the three experiments, we can address question 3 -- How can we use LFA to improve the tutor and the curriculum by identifying over-taught or under-taught rules, and adjusting their contribution to curriculum length without compromising student performance?

Parallelogram-side has a high intercept (2.06) and a low slope (-.01). Its initial success probability is .94 and the average number of practices per student is 14.9. Much practice spent on an easy skill is not a good use of student time. Reducing the amount of practice for this skill should save student time without compromising their performance. Trapezoid-height has a low intercept (-1.55), and a positive slope (.27). Its initial success probability is .29 and the average number of practices per student is 4.2. The final success probability is .69, far away from the level of mastery. More practice on this skill is needed for students to reach mastery.

The advantage of LFA goes even further. An original rule may have two split rules, each of which need decidedly different amounts of practice, because they have different initial difficulty and learning rates. However, students who have appeared to master the original rule in the curriculum before even reading the second split rule might not get enough practice on the second split rule. Compose-by-multiplication is such a case, as seen in table 9.

Table 9. Statistics of Compose-by- Multiplication before and after split

	Intercept	slope	Avg Practice Opportunties	Initial Probability	Avg Probability	Final Probability
CM	-.15	.1	10.2	.65	.84	.92
CMarea	-.009	.17	9	.64	.86	.96
CMsegment	-1.42	.48	1.9	.32	.54	.60

With final probability .92 students seem to have mastered Compose-by-multimplication. However, the decomposition of the skill shows a different picture. CMarea does well with final probability .96. But CMsegment has final probability only .60 and an average amount of practice less than 2. The knowledge-tracing algorithm in the tutor may let the student go after he reaches the mastery on Compose-by-addition in the original model. But with the model found by LFA, the knowledge-tracing algorithm will be able to catch the weakness of students in acquiring CMsegment.

6 Conclusions and Future Work

Learning Factors Analysis is a way to combine statistics, human expertise and combinatorial search to evaluate and improve a cognitive model. The system we have developed is implemented in Java and is able to evaluate a model in seconds and conduct a search evaluating hundreds of models in 4-5 hours. The statistics for each model are meaningful, and the new improved models have better statistical scores and are interpretable. We are planning to use the method for datasets from other tutors to discover its potential for model and tutor improvement.

Acknowledgements

This research is sponsored by a National Science Foundation grant to the Pittsburgh Science of Learning Center. We thank Joseph Beck, Albert Colbert, and Ruth Wylie for their comments on earlier versions of this paper.

References

1. Baffes P., Mooney R J.: A Novel Application of Theory Refinement to Student Modeling. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96). Portland OR (1996) 403-408
2. Barnes T.: The Q-matrix Method: Mining Student Response Data for Knowledge. In: Proceedings of AAAI 2005 Educational Data Mining Workshop. (2005)
3. Cen, H, Koedinger, K, Junker B.: Automating Cognitive Model Improvement by A*Search and Logistic Regression. In: Proceedings of AAAI 2005 Educational Data Mining Workshop. (2005)
4. Corbett A.T., Anderson, J.R., O'Brien A.T.: Student Modelling in the ACT Programming Tutor. In: Cognitively Diagnostic Assessment. Erlbaum, Hillsdale, NJ (1995)
5. Corbett, A.T. and Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction, 4. (1995) 253-278.
6. Croteau E.A., Heffernan N. T., Koedinger K.R.: Why are Algebra Word Problem Difficult? Using Tutoring Log Files and the Power Law of Learning to Select the Best Fitting Cognitive Model. In: Proceedings of Intelligent Tutoring Systems. (2004)
7. Draney, K., Pirolli, P., & Wilson, M.: A Measurement Model for a Complex Cognitive Skill. In: Cognitively Diagnostic Assessment. Erlbaum, Hillsdale, NJ (1995)
8. Junker B.W., Koedinger, K., Trottini M.: Finding Improvements in Student Models for Intelligent Tutoring Systems via Variable Selection for a Linear Logistic Test Model., presented at annual meeting of Psychometric Society Vancouver BC (2000)
9. Koedinger K. R.: http://pact.cs.cmu.edu/koedinger/koedingerResearch.html. (2000)
10. Koedinger K. R., Mathan S.: Distinguishing Qualitatively Different Kinds of Leaning Using Log Files and Learning Curves. In: Proc. of Intelligent Tutoring Systems (2004)
11. Koedinger, K. R., Nathan, M. J.: The Real Story Behind Story Problems: Effects Of Representations On Quantitative Reasoning. The Journal of the Learning Sciences, 13 (2). (2004) 129-164.
12. Koedinger K.R., Anderson, J.R., Hadley W. H., Mark M. A.: Intelligent Tutoring Goes to School in the Big City. In: Proceedings of the 7th World Conference on Art, Intelligence and Education, AACE. (1995)
13. Newell, A., Rosenbloom, P.: Mechanisms of Skill Acquisition and the Law of Practice. In Anderson J. (ed.): Cognitive Skills and Their Acquisition, Erlbaum Hillsdale NJ (1981)
14. Russell S., Norvig P., Artificial Intelligence, 2nd edn. Prentice Hall (2003)
15. Tatsuoka, K.: Rule space: An approach for dealing with misconceptions based on item response theory. Journal of Educational Measurement. 20(4) (1983) 345-354.
16. Ur, S., VanLehn, K.: STEPS: A Simulated, Tutorable Physics Student. Journal of Artificial Intelligence in Education. 6(4) (1995) 405-437
17. Wasserman, L.: All of Statistics: A Concise Course in Statistical Inference. Springer (2004)

A Constraint-Based Collaborative Environment for Learning UML Class Diagrams

Nilufar Baghaei and Antonija Mitrovic

Intelligent Computer Tutoring Group
Department of Computer Science and Software Engineering
University of Canterbury, Private Bag 4800, New Zealand
{n.baghaei, tanja}@cosc.canterbury.ac.nz

Abstract. COLLECT-UML is a constraint-based ITS that teaches object-oriented design using Unified Modelling Language (UML). UML is easily the most popular object-oriented modelling technology in current practice. We started by developing a single-user ITS that supported students in learning UML class diagrams. The system was evaluated in a real classroom, and the results show that students' performance increased significantly. In this paper, we present our experiences in extending the system to provide support for collaboration. We present the architecture, interface and support for collaboration in the new, multi-user system. A full evaluation study has been planned, the goal of which is to evaluate the effect of using the system on students' learning and collaboration.

1 Introduction

E-learning is becoming an increasingly popular educational paradigm as more individuals who are working or are geographically isolated seek higher education. As such students do not meet face to face with their peers and teachers, the support for collaboration becomes extremely important [8]. Effective collaborative learning includes both learning to effectively collaborate, and collaborate effectively to learn, and therefore a collaborative system must be able to address collaboration issues as well as task-oriented issues [17].

In the last decade, many researchers have contributed to the development of CSCL and advantages of collaborative learning over individualised learning have been identified [14]. Some particular benefits of collaborative problem-solving include: encouraging students to verbalise their thinking; encouraging students to work together, ask questions, explain and justify their opinions; increasing students' responsibility for their own learning; increasing the possibility of students solving or examining problems in a variety of ways; and encouraging them to articulate their reasoning, and elaborate and reflect upon their knowledge [24, 27]. These benefits, however, are only achieved by active and well-functioning learning teams [15]. Numerous systems for collaborative learning have been developed; however, the concept of supporting peer-to-peer interaction in CSCL systems is still in its infancy. Various strategies for computationally supporting online collaborative learning have been proposed and used, while more studies are needed that test the utility of these techniques [17].

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 176 – 186, 2006.

This paper describes an Intelligent Tutoring System (ITS) that uses Constraint-Based Modeling (CBM) approach to support both problem-solving and collaborative learning. CBM has been used successfully in several tutors supporting individual learning [20]. We have developed COLLECT-*UML* [2, 3], a single-user version of a constraint-based ITS, that teaches UML class diagrams. In this paper, we describe extensions to this tutor, which support multiple students solving problems collaboratively. We start with a brief overview of related work in Section 2. Section 3 then presents COLLECT-*UML* and the evaluation study conducted with second-year university students taking a course in Introduction to Software Engineering. Section 4 describes the design and implementation of the collaborative interface as well as the system's architecture. Section 5 presents the collaborative model, which has been implemented as a set of meta-constraints. Conclusions are given in the last section.

2 Related Work

Three categories of CSCL systems can be distinguished in the context of the collaboration support [1, 17]. The first category includes systems that reflect actions; the basic level of support a system may offer involves making the students aware of the participants' actions. The systems in the second category monitor the state of interactions; some of them aggregate the interaction data into a set of high-level indicators, and display them to the participants (e.g. Sharlock II [21]), while others internally compare the current state of interaction to a model of ideal interaction, but do not reveal this information to the users (e.g. EPSILON [25]). In the latter case, this information is either intended to be used later by a coaching agent, or analysed by researchers in order to understand the interaction [17]. Finally, the third class of systems offer advice on collaboration. The coach in these systems plays a role similar to that of a teacher in a collaborative learning classroom. The systems can be distinguished by the nature of the information in their models, and whether they provide feedback on strictly collaboration issues or both social and task-oriented issues. Examples of the systems focusing on the social aspects include Group Leader Tutor [19] and DEGREE [6], and an example of the systems addressing both social and task-oriented aspects of group learning is COLER [7].

Although many tutorials, textbooks and other resources on UML are available, we are not aware of any attempt at developing a CSCL environment for UML modelling. However, there has been an attempt [25] at developing a collaborative learning environment for OO design problems using Object Modeling Technique (OMT) – a precursor of UML. The system monitors group members' communication patterns and problem solving actions in order to identify situations in which students effectively share new knowledge with their peers while solving OO design problems. The system first logs data describing the students' speech acts (e.g. *Request Opinion, Suggest*, and *Apologise*) and actions (e.g. *Student 3 created a new class*). It then collects examples of effective and ineffective knowledge sharing, and constructs two Hidden Markov Models which describe the students' interaction in these two cases. A knowledge sharing example is considered effective if one or more students learn the newly shared knowledge (as shown by a difference in pre-post test performance), and ineffective otherwise. The system dynamically assesses a group's interaction in the context of the constructed models, and determines when and why the students are having trouble learning new concepts they share with each other. The system does not evaluate the OMT diagrams and an instructor or intelligent coach's assistance is needed in mediating group knowledge sharing activities. In this regard, even though the system is effective as a collaboration tool, it would probably not be an effective teaching system for a group of novices with the same level of expertise, as it could be common for a group of students to agree on the same flawed argument.

CBM has been used successfully in several tutors supporting individual learning. The main contribution of this research is the use of CBM technique to support collaborative learning. The system provides feedback on both collaboration issues (using the collaboration model, represented as a set of meta-constraints) and task-oriented issues (using the domain model, represented as a set of syntax and semantic constraints). CBM is also used to model student and group knowledge.

3 COLLECT-*UML*: Single-User Version

COLLECT-*UML* is a problem-solving environment, in which students construct UML class diagrams that satisfy a given set of requirements. It assists students during problem-solving, and guides them towards a correct solution by providing feedback. The feedback is tailored towards each student depending on his/her knowledge. COLLECT-*UML* is designed as a complement to classroom teaching and when providing assistance, it assumes that the students are already familiar with the fundamentals of UML. For details on system's architecture, functionality and the interface refer to [2, 3]; here we present only the basic features of the system.

At the beginning of interaction, a student is required to enter his/her name, which is necessary in order to establish a session. The session manager requires the student modeller to retrieve the model for the student, if there is one, or to create a new model for a new student. Each action a student performs is sent to the session manager, as it has to link it to the appropriate session and store it in the student's log. Then, the action is sent to the pedagogical module. If the submitted action is a solution to the current problem, the student modeller diagnoses the solution, updates the student model, and sends the result of the diagnosis back to the pedagogical module, which generates appropriate feedback.

COLLECT-*UML* contains an ideal solution for each problem, which is compared to the student's solution according to the system's domain model, represented as a set of constraints [22]. The system's domain model contains 133 constraints that describe the basic principles of the domain. In order to develop constraints, we studied material in textbooks, such as [12], and also used our own experience in teaching UML and OO analysis and design.

Figure 1 illustrates a constraint from the UML domain. The relevance condition identifies a relationship of type aggregation in the ideal solution, and then checks whether the student's solution contains the same type of relationship, or a relationship of a different kind with the same name. The student's solution is correct if the satisfaction condition is met, when the matching relationship is of the same type (i.e. aggregation). The constraint also contains a message which would be given to the student if the constraint is violated. The last two elements of the constraint specify that it

```
(78
 "Check the type of your relationships. You need to use aggregations between some of
your classes."
   (and (match IS RELATIONSHIPS (?* "@" ?rel_tag "aggregation" ?c1_tag ?c2_tag ?*))
     (or-p (match SS RELATIONSHIPS (?* "@" ?rel_tag ?type ?c1_tag ?c2_tag ?*))
        (match SS RELATIONSHIPS (?* "@" ?rel_tag ?type ?c2_tag ?c1_tag ?*))))
 (test SS ("aggregation" ?type))
 "relationships"
 (?rel_tag ?c1_tag ?c2_tag))
```

Fig. 1. An example constraint

covers some aspects of relationships, and also identifies the relationship and the classes to which the constraint was applied.

We performed an evaluation study [3] in May 2005 with 38 students enrolled in a Software Engineering course. The students learnt UML modelling concepts during two weeks of lectures/tutorials. The study was conducted in two streams of two-hour laboratory sessions. Each participant sat a pre-test, interacted with the system, and then sat a post-test and filled a user questionnaire. The pre-test and post-test each contained four multiple-choice questions, followed by a question where the students were asked to design a simple UML class diagram. Table 1 presents some general statistics about the study. The average mark on the post-test was significantly higher than the pre-test mark ($t = 2.71$, $p = 4.33E\text{-}08$). The students spent on average 90 minutes interacting with the system.

Table 1. Some statistics about the study

	Average	s. d.
Attempted problems	5.71	2.59
Solved problems	47%	33%
Attempts per problem	7.42	4.76
Pre-test	52%	21%
Post-test	76%	17%

We also analyzed the log files, in order to identify how students learn the underlying domain concepts. Figure 2 illustrates the probability of violating a constraint plotted against the occasion number for which it was relevant, averaged over all constraints and all participants. The data points show a regular decrease, which is approximated by a power curve with a close fit of 0.93, thus showing that students do learn constraints over time. The probability of violating a constraint on the first occasion of application is halved by the tenth occasion, showing the effects of learning.

Students were offered individualised feedback on their solutions upon submission. The mean rating for the usefulness of feedback was 2.8. 67% of the participants had indicated that they would have liked to see more details in the feedback messages.

The comments we received on open questions pointed out several features of the system, which can be improved.

The results showed that COLLECT-*UML* is an effective learning environment. The participants achieved significantly higher scores on the post-test, suggesting that they acquired more knowledge in UML modelling. The learning curves also prove that students do learn constraints during problem solving. Subjective evaluation shows that most of the students felt spending more time with the system would have resulted in more learning and that they found the system to be easy to use.

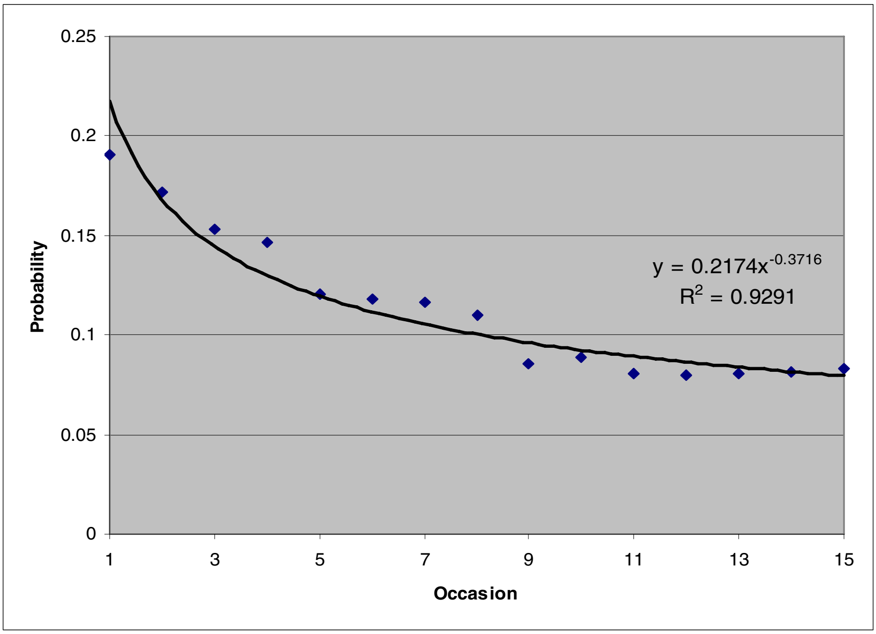


Fig. 2. Probability of constraint violation

4 COLLECT-*UML*: Multi-user Version

The collaborative version of COLLECT-*UML* is designed for sessions in which students first solve problems individually and then join into small groups to create group solutions. The system's architecture is illustrated in Figure 3. The application server consists of a session manager that manages sessions and student logs, a student modeller that creates and maintains student models for individual users, a domain model (i.e. the constraint set), a pedagogical module and a group modeller. The system is implemented in Allegro Common Lisp.

The interface is shown in Figure 4. The problem description pane presents a design problem. Students construct their individual solutions in the private workspace (right), and use the shared workspace (left) to collaborate while communicating via the chat

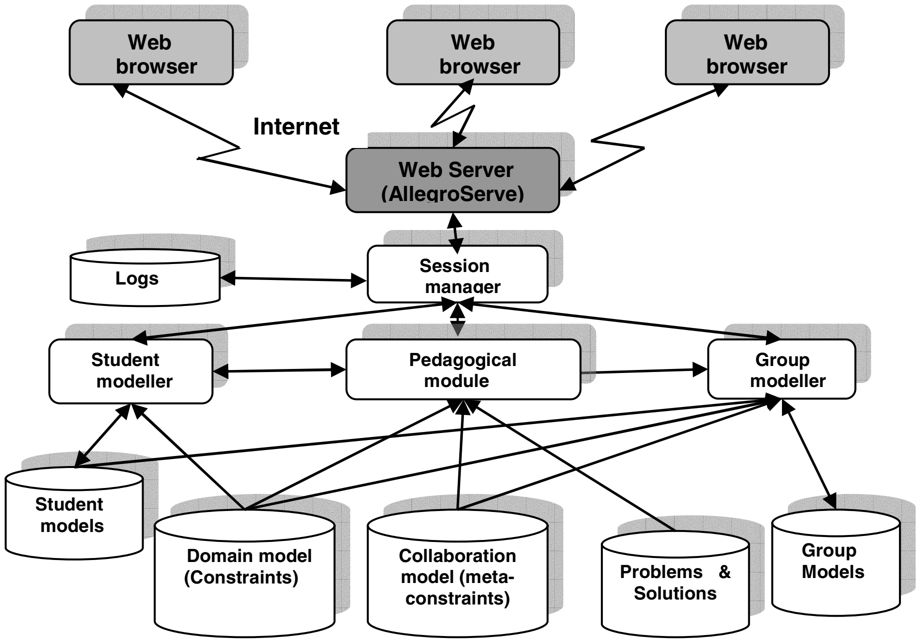


Fig. 3. The architecture of COLLECT-UML

window (bottom). The private workspace enables students to try their own solutions and think about the problem before start discussing it in the group. The group area is initially disabled. When all of the students indicate readiness to work in the group by clicking on *Join the Group* button, the shared workspace is activated. The students select the components' names from the problem text. The *Group Members* panel shows the team-mates already connected. Only one student, the one who has the pen, can update the shared workspace at a given time. Additionally, this panel shows the name of the student who has the control of this area and the students waiting for a turn.

A recent study [23] defines relevant characteristics of good collaboration and the authors have considered turn-taking as one of those characteristics. According to their results, explicitly handing over a turn can be a good way of compensating for the limited communication channel. An implication of providing such protocol is that deadlocks can be created in cases where one partner cannot proceed with problem-solving alone and at the same time refuses to pass the key over to the other partners. The advantage, however, is that it maintains clear semantics of a participant's actions and roles in the shared workspace [10]. The lack of providing turn-taking protocol in most of computer-mediated collaboration tools is considered to be one of the limitations of such tools [11].

The chat area enables students to express their opinions using sentence openers. The student needs to select one of the sentence openers before being able to express his/her opinion. The contents of selected sentence openers are displayed in the chat area along with any optional justifications. Sentence openers structure students' conversation and eliminate off-task discussions. A structured chat interface with specific sentence openers can promote more focus on reflection and the fundamental concepts at stake [5]. Although this kind of dialogue requires more effort from the student than using plain chat or email, as the student needs to categorize their own contributions, research shows that the quality of the dialogue can be higher [16]. In addition, structuring the dialogue makes it easier to analyze computationally [10].

Sentence openers provide a natural way for users to identify the intention of their conversational contribution without fully understanding the significance of the underlying communicative acts [19]. Results from various projects indicate that structured dialogues support students to stay on task and increase reflection [13]. However, requiring learners to select a sentence opener before typing the remainder of their contribution may tempt them to change the meaning of the contribution to fit one of the sentence openers, thus changing the nature of the collaborative interaction. According to Lazonder et al. [18], sentence openers should be derived from naturally occurring online text-based free dialogues, while Soller [24] states that it is critical to provide the widest and most appropriate range of sentence openers. Some experiments [4] show that in interfaces containing both structured and free chat tools, the former are used more frequently.

The group moderator can submit the solution, by clicking on the *Submit Answer* button on the shared workspace. The system gives collaboration-based advice based on the content of the chat area, students' participation on the shared diagram and the differences between students' individual solutions and the group solution being constructed. The task-based advice is given to the whole group based on the quality of the shared diagram.

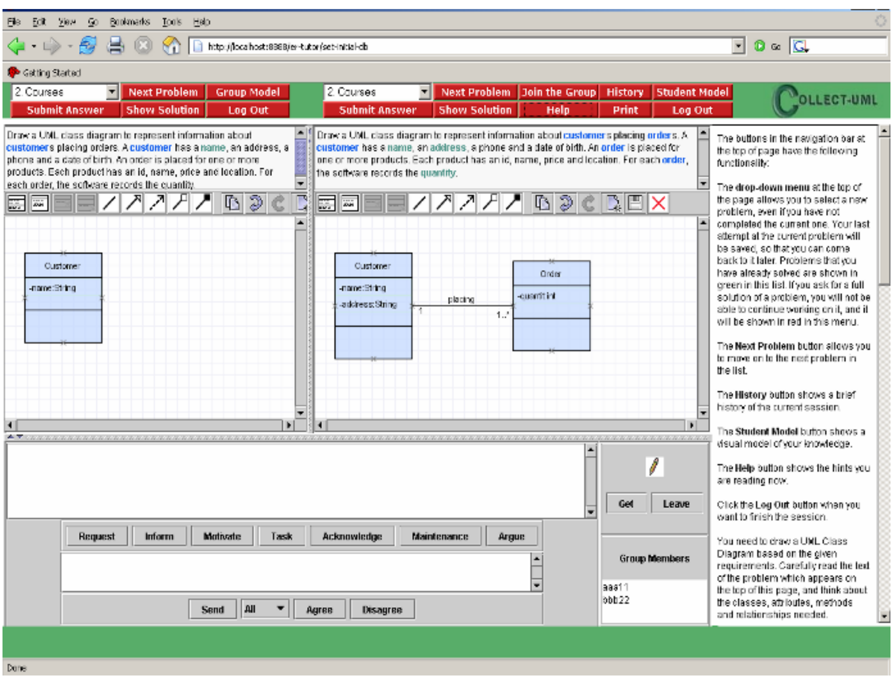


Fig. 4. COLLECT-UML interface

The *Next Problem*, *Submit Answer*, *Show Solution* and *Log Out* buttons at the top of the shared diagram are controlled by the group moderator only, while the *Group Model* button can be accessed by all the members. The students can use the *Help* button (at the top of the individual workspace) to get information about UML Modeling, *Submit Answer* to get feedback on their individual solutions and *Next problem* to move on to a new problem (regardless of the problem the group is working on at that point). The students cannot view full solutions in the individual workspaces (that option is only available under the shared workspace). Viewing the full solution by individual members of the group might stop them from thinking about the problem and/or collaborating with the rest of the group members.

5 Modeling Collaboration

The ultimate goal of COLLECT-*UML* is to support collaboration by modelling collaborative skills. The system is able to promote effective interaction by diagnosing students' actions in the chat area and group diagram using a set of 22 meta-constraints, which represent an ideal model of collaboration. These constraints have the same structure as domain constraint, each containing a relevance condition, a satisfaction condition and a feedback message. The feedback message is presented when the constraint is violated. In order to develop meta-constraints, we studied

```
(221
  "Some relationship types (associations) in your individual solution are missing from the
group diagram. You may wish to share your work by adding those association(s)/discuss it
with other members."
  (and (match SS RELATIONSHIPS (?* "@" ?rel_tag "association" ?c1_tag ?c2_tag ?*))
      (match GS CLASSES (?* "@" ?c1_tag ?*))
      (match GS CLASSES (?* "@" ?c2_tag ?*)))
  (or-p (match GS RELATIONSHIPS (?* "@" ?rel_tag "association" ?c1_tag ?c2_tag ?*))
       (match GS RELATIONSHIPS (?* "@" ?rel_tag "association" ?c2_tag ?c1_tag ?*)))
  "relationships"
  (?rel_tag ?c1_tag ?c2_tag))

 (237
  "You may wish to explain to other members why you agree or disagree  with a solution."
  (and (match SC DESC (?* "@" ?tag ?text ?*))
      (or-p (test SC ("agree" ?tag))
           (test SC ("disagree" ?tag))))
  (not-p (test SC ("" ?text)))
  "descriptions"
  nil)
```

Fig. 5. Examples of meta-constraints

existing literature on characteristics of an effective collaboration, such as [9, 23, 24, 26]. Figure 5 illustrates two examples of meta-constraints. Constraint 221 encourages student participation in problem-solving. This constraint makes sure that the student contributes associations from his/her individual solution to the group solution. On the other hand, there are constraints that check whether the student participates in the dialogue. Constraint 237 checks whether the student has specified any justification for their agreement/disagreement with the group solution.

A history of all contributions made by each user to the shared diagram as well as the messages posted to the chat area is maintained on the server, and the meta-constraints are evaluated against this history. Feedback is given on contributions which involve adding/deleting/updating components in the shared diagram, as well as contributions made to the chat area.

6 Conclusions and Future Work

This paper presented the single-user version of COLLECT-*UML*, and the results of the evaluation study performed. The results of both subjective and objective analysis proved that COLLECT-*UML* is an effective educational tool. The participants performed significantly better on a post-test after short sessions with the system, and reported that the system was relatively easy to use.

We then presented the multi-user version of the same intelligent tutoring system. We have extended COLLECT-*UML*' interface, and developed meta-constraints, which provide feedback on collaborative activities. The goal of future work is to complete the implementation of the multi-user version and conduct a full evaluation study with second-year University students enrolled in an undergraduate software engineering course. The study is planned for April 2006. Participants will be divided into three groups. The experimental condition will receive feedback on the domain model as well as their collaborative activities. The students will also be provided with a script addressing the characteristics of a good collaboration and the phases they are expected to go through, at the beginning of the session. The second group will receive feedback on their solutions only. These students will be provided with the same script at the beginning of the session, but will not receive feedback on collaboration. The control group will only receive feedback on the domain level. There will not be any type of support on the collaboration process available to this group. Our hypothesis is that all groups will increase their problem-solving skills, but that only the experimental group will improve collaboration skills. All participants will be assessed on their understanding of what characterises good collaboration at the end of the session by answering questions in the post-test. Their interaction in the shared diagram and chat area will also be analysed.

CBM has been used to effectively represent domain knowledge in several ITSs supporting individual learning. The contribution of the project presented in this paper is the use of CBM to model collaboration skills, not only domain knowledge. Comprehensive evaluation of the multi-user version of COLLECT-*UML* will provide a measure of the effectiveness of using the CBM technique in intelligent computer-supported collaborative learning environments.

References

1. Baghaei, N. and Mitrovic, A. (2005) *COLLECT-UML: Supporting individual and collaborative learning of UML class diagrams in a constraint-based tutor.* In Khosla, R., Howlett, R. and Jain L. (eds.) KES 2005, pp.458-464.
2. Baghaei, N., Mitrovic, A. and Irwin, W. (2005) *A Constraint-Based Tutor for Learning Object-Oriented Analysis and Design using UML.* In Looi, C., Jonassen, D. and Ikeda M. (eds.) ICCE 2005, pp.11-18.
3. Baghaei, N., Mitrovic, A. and Irwin, W. (2006) *Problem-Solving Support in a Constraint-based Tutor for UML Class Diagrams*, Technology, Instruction, Cognition and Learning Journal, 4(1-2) (in print).
4. Baker, M. J. and Lund, K. (1997) *Promoting reflective interactions in a computer- supported collaborative learning environment.* Journal of Computer Assisted Learning, 13(3), 175-193.
5. Baker, M., de Vries, E., Lund, K. and Quignard, M. (2001) *Computer-mediated epistemic interactions for co-constructing scientific notions: Lessons learned form a five-year research program.* In Dillenbourg, P., Eurelings, A. and Hakkarainnen, K. (eds.) European Perspectives on CSCL (CSCL 2001), pp.89-96.
6. Barros, B. and Verdejo, M.F. (2000) *Analysing student interaction processes in order to improve collaboration:The DEGREE approach.* Artificial Intelligence in Education, 11, 221-241.
7. Constantino-Gonzalez, M., and Suthers, D. (2000) *A coached collaborative learning environment for Entity-Relationship modelling*. In Gauthier, G., Frasson, C. and VanLehn, K. (eds.) 5th Int. Conf. Intelligent Tutoring Systems, pp.324-333.
8. 8. Constantino-Gonzalez, M., and Suthers, D. (2002) *Coaching Collaboration in a Computer Mediated Learning Environment*. In Stahl, G. (eds.) CSCL 2002, pp.583-584.
9. Constantino-Gonzalez, M. A., Suthers, D. and Escamilla de los Santos, J. (2003) *Coaching web-based collaborative learning based on problem solution differences and participation*. Artificial Intelligence in Education, 13 (2-4), 261-297.
10. Dimitracopoulou, A. (2005) *Designing Collaborative Learning Systems: Current Trends & Future Research Agenda.* In Koschmann, T., Suthers, D. and Chan T.W. (eds.) 6th Int. Computer Supported Collaborative Learning Conf. CSCL 2005.
11. Feidas, C., Komis, V. and Avouris, N. (2001) *Design of collaboration-support tools for group problem solving*. In Avouris, N. and Fakotakis, N. (eds.) Advances in Human-Computer Interaction, pp.263-268.
12. Fowler, M. (2004) *UML Distilled: a Brief Guide to the Standard Object Modelling Language*. Reading: Addison-Wesley, 3rd edition.
13. Gogoulou, A., Gouli, E., Grigoriadou, M. and Samarakou, M. (2005) *ACT: A Web – based Adaptive Communication Tool.* In Koschmann, T., Suthers, D. and Chan T.W. (eds.) 6th Int. Computer Supported Collaborative Learning Conf., pp.180-189.
14. Inaba, A. and Mizoguchi, R. (2004) *Learners' Roles and Predictable Educational Benefits in Collaborative Learning; An Ontological Approach to Support Design and Analysis of CSCL.* In Lester, J., Vicari, R. M. and Paraguacu, F. (eds.) 7th Int. Conf. Intelligent Tutoring Systems, pp.285–294.
15. Jarboe, S. (1996) *Procedures for enhancing group decision making*. In Hirokawa B. and Poole M. (eds.) Communication and Group Decision Making, pp.345-383.
16. Jermann, P., Soller, A. and Lesgold, A. (2004) *Computer software support for CSCL.* In Dillenbourg P., Strijbos J.W., Kirschner, P.A. and Martens R.L. (eds.) Computer-supported collaborative learning: Vol 3. What we know about CSCL ... and implementing it in higher education, pp.141-166.

17. Jerman, P., Soller, A. and Muhlenbrock, M. (2001) *From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning*. In Dillenbourg, P., Eurelings, A and Hakkarainen, K. (eds.) European Perspectives on CSCL (CSCL 2001), pp.324-331.
18. Lazonder, A., Wilhelm, P. and Ootes S. (2003) *Using sentence openers to foster student interaction in computer-mediated learning environments*. Computers & Education, 41, 291-308.
19. McManus, M. and Aiken, R. (1995) *Monitoring computer-based problem solving*. Artificial Intelligence in Education, 6(4), 307-336.
20. Mitrovic, A., Mayo, M., Suraweera, P. and Martin, B. (2001) *Constraint-based Tutors: a Success Story*. 14th Int. Conf. Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, pp.931-940.
21. Ogata, H., Matsuura, K., and Yano, Y. (2000) *Active Knowledge Awareness Map: Visualizing learners activities in a Web based CSCL environment*. Int. Workshop on New Technologies in Collaborative Learning, pp.89-97.
22. Ohlsson, S. (1994) *Constraint-based Student Modelling.* Student Modelling: the Key to Individualized Knowledge-based Instruction, Springer-Verlag, pp.167-189.
23. Rummel, N. and Spada, H. (2005) *Learning to collaborate: An instructional approach to promoting collaborative problem-solving in computer-mediated settings*. Journal of the Learning Sciences, 14(2), 201-241.
24. Soller, A. (2001) *Supporting Social Interaction in an Intelligent Collaborative Learning System.* Artificial Intelligence in Education, 12, 40-62.
25. Soller, A. and Lesgold, *A*. (2000) *Knowledge acquisition for adaptive collaborative learning environments.* AAAI Fall Symposium: Learning How to Do Things, Cape Cod, MA.
26. Vizcaino, A. (2005) *A Simulated Student Can Improve Collaborative Learning.* Artificial Intelligence in Education, 15, 3-40.
27. Webb, N. M., Troper, J. D. and Fall, R. (1995) *Constructive activity and learning in collaborative small groups*. Journal of Educational Psychology, 87, 406-423.

A Collaborative Learning Design Environment to Integrate Practice and Learning Based on Collaborative Space Ontology and Patterns

Masataka Takeuchi[1], Yusuke Hayashi[1], Mitsuru Ikeda[2], and Riichiro Mizoguchi[1]

[1] The Institute of Scientific and Industrial Research, Osaka University
{takeuchi, hayashi, miz}@ei.sanken.osaka-u.ac.jp
[2] Japan Advanced Institute of Science and Technology
ikeda@jaist.ac.jp

Abstract. The integration of practice and learning is a key to cultivation of organizational capability for creating or inheriting intellect. In this paper, firstly we address the critical research issues for collaborative space design to integrate practice and learning. Following the discussion, we have built an ontology which specifies the structure of the collaborative learning, described patterns of a collaborative space by reference to the learning theories and developed an intelligent function to support a collaborative space design on ontology-based KM environment, *Kfarm*, as a foundation for supporting collaborative space design.

1 Introduction

Organizational activity studies (e.g. knowledge management study [1,2]) discuss what a good knowledge is and what an ideal process is. For instance, the famous knowledge management model called "SECI model [3]" suggests us an ideal process of organizational knowledge creation and inheritance.

Our long-term objective is to develop a framework, named *Kfarm*[4], which totally supports creating/inheriting organizational intellect (not only knowledge but also skill or competency). For realizing such a rich support framework, we have constructed a basic information model, named "Dual Loop Model (DLM)", following the ontology engineering approach. The ontology engineering approach [5,6] is the way to provide a basis of the information model for system developers, system users, and a system to share concepts and relations of the target world. Concepts and relations in an ontology are well clarified to make both of computers and humans intelligible. We have adopted such an ontology as a core knowledge processing framework for *Kfarm* to support organizational members' intellectual activities.

While building DLM and designing *Kfarm*, as suggested by many KM theories, we have noticed that the integration of practice and learning is a key to cultivate organizational capability for creating or inheriting intellect, and found that the integration can be made reasonably in the case of learning the domain-general competency while doing practical activities with domain-specific intellect. To realize such rational integration, we have made the distinction between the following two types of intellect in DLM.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 187 – 196, 2006.

Intellect of type A: *knowledge or skill required for performing daily jobs.* Intellect which organizational members directly use, create and inherit for performing daily jobs in organizational activities, e.g. domain knowledge to make a plan for performing daily jobs or skill to execute the plan.

Intellect of type B: *competency to create and inherit intellect of type A.* A fundamental and implicit intellect which derives the organizational members' activities to create and inherit intellect of type A, e.g. a competency to create new intellect, competency to lead discussion, or competency to acquire intellect by observing others' behavior.

Our research goal includes to establish a rational design framework of collaborative space to integrate organizational activities to solve practical problem with intellect of type A, and learning intellect of type B to create or inherit intellect of type A. We have discussed a unified model for both type A and type B in our previous paper[4]. This paper focuses on the distinction between these two types. In this paper, firstly we address the critical research issues for collaborative space design to integrate practice and learning. Following the discussion, in Sect. 3, we will describe an intelligent function to support a collaborative space design we have developed.

2 Collaborative Learning Space Design Based on Ontology

To increase competitive power of an organization, organizational members should have a good *circulation* of the organizational intellect in the DLM. To achieve this we emphasize the need to support organizational members enhancing the *circulation* capability, which means learning an intellect of type B.

As mentioned earlier, intellect of type B mainly includes a competency to create and inherit organizational intellect of type A. This assures us that organizational members should always learn intellect of type B through group practice to create and inherit organizational intellect. Thus, it is important to design an integrated collaborative learning space for learning and practice.

Such collaborative learning space has been designed by an expert designer. However, such a design is often highly abstract and implicit, since intellect of type A and type B have similar characteristic of intellect and could not be clearly distinguished. Additionally, at such a design a designer may be ignorant of whether conflicts between practical goal and learning goal occurred or not.

A key to overcoming such a difficulty is to establish consistent and rational design based on a common conceptual foundation. That is collaborative learning space design based on ontology. Encouraging a designer to be compliant with the ontology, the design intention of collaborative space is expected to be explicit and the designer notices how to avoid and prevent conflicts at the design time.

In this study, consistent and rational design of collaborative learning space is typically achieved in the following three steps: **Step. 1)** Find a right opportunity to initiate collaborative learning in a practical space provided by *Kfarm*. **Step. 2)** Design collaborative learning matching with a state of the practical space. **Step. 3)** Provide an environment for the participants to run the collaborative learning in the physical space such as a classroom or cyber space such as chat or bulletin board.

Step 2 consists of the following three sub-steps: **Step. 2-1) Abstract design:** Design collaborative learning configuration including learning goals, tasks, and participants' roles. **Step. 2-2) Concrete design:** Embody the abstract design by assigning organizational members to the roles, preparing the materials for the task, and so on. **Step. 2-3) Negotiation:** Explain the design rationale to participants.

In this section, we discuss the role of collaborative space ontology and collaborative space patterns in the design process.

2.1 Conflict Between Learning Goal and Practical Goal

As we have discussed thus far, the goal of this study is to draw up design guideline of building a rationally integrated space for practice and learning. However, the combination is not simple because designers often face the conflict between learning goal and practical goal. Typical examples of the conflicts are shown below:

Conflict at the abstract design: Conflicts concerning goals, tasks, or roles sometimes occur at the abstract design. Assume that a designer wants to set up a collaborative learning for a novice participant to learn how to create a new idea by observing experts' behavior when they are involved in a brain storming session. From a practical viewpoint (to create intellect of type A), all the participants are expected to express their ideas freely. On the other hand, from a learning viewpoint (to learn intellect of type B), the expert participants are expected to explain how they come up with their ideas and the novice participants are expected to concentrate on observing experts' behavior. The two different viewpoints may cause a kind of undesirable conflicting feeling of participants.

Conflict at the concrete design: Conflicts often occur when embodying the abstract design. For example, from a practical viewpoint, a competent person is expected to be involved in the collaborative activity to improve the quality of the product and shorten the work time. However, in general, it is always hard to get competent persons involved in others' learning activity.

Ontology and patterns play an important role for a designer to avoid and prevent these conflicts. At consistent and rational design, ontology provides sharable and clearly defined concepts of collaborative learning space, and patterns offer guidelines to build up the integrated space for learning and practice based on the ontology.

2.2 Collaborative Space Ontology

We have built an ontology which specifies the structure of the collaborative learning. In addition, we have described the pattern and its frame which are expected to avoid and prevent the conflict by reference to the learning theories.

In this study, ontology is a fundamental conceptual framework to represent the design rationale or the designer's intention and to maintain the consistency among different viewpoints. A design based on ontology enables a designer to specify design rationale explicitly. The designer may notice hidden relations among different viewpoints. As a result, consistent design free from conflicts would be obtained. Additionally in some learning theory, the ways to avoid and prevent conflict are founded. In order to reuse design experiences, it is important to describe design know-how as patterns and store them. Ontology is also useful to produce consistent patterns because it provides a common vocabulary, data structure, and so on.

Table 1. The major concepts in the Collaborative Space Ontology

concept [isa] sub-concepts	Description
goal	desirable change of intellect
organizational goal	desirable change of organizational intellect
practical goal	creation or inheritance organizational intellect of type A (e.g. sharing an idea)
learning goal	learning organizational intellect of type B (e.g. learning creativity)
interaction goal	desirable change from personal intellect to organizational intellect (e.g. publishing an idea)
individual goal	desirable change of personal intellect (e.g. acquiring an idea)
scene	a situation in which participants carry on a work, a job, or a task to achieve the goal
practical scene	a situation in which participants carry on a practical task to improve quality of the task outcome and the task performance
learning scene	a situation in which participants carry on a simplified task comparable to the practical task to scaffold learning the intellect
configuration	interactions among roles of participants to achieve the goal
practical configuration	interactions among roles of participants who have the competency in order to achieve the practical goal with high quality and high performance
learning configuration	interactions designed with learning intention to give high priority to achieve the learning goal
space-time	the when and the where that collaborative activity expected to be carried out. (e.g. a classroom, chat or bulletin board)

Collaborative space ontology is constructed from the effective concepts which are described in the learning theory. We have done comprehensive studies on learning theories to build a "Collaborative Learning Ontology[8]". Following that study we aim at expanding the ontology. The ontology in this paper is constructed from both practical viewpoint and learning viewpoint. Practical viewpoint is creation and inheritance of intellect of type A, and learning viewpoint is learning of intellect of type B.

This ontology inherits the basic concepts relevant to organizational intellect creation and inheritance (e.g. person, intellect, vehicle, and activity). These concepts are defined in DLM. By sharing these concepts a design support system can communicate information about organizational intellect with *Kfarm*.

In order to constitute variety forms of collaborative space at a practical setting or an learning setting, the collaborative space ontology contains the concepts from both practical viewpoint and learning viewpoint; the goals, the task, the scenes, the configurations, the interactions, and so on (Table 1). Our ontology has been defined in an ontology editor of an environment for building/using ontology, Hozo[9], in which concepts are represented as frames with slots and the is-a relations among concepts.

2.3 Collaborative Space Pattern

Collaborative space pattern represents domain-neutral knowledge to prevent the conflict. All the concepts appearing in the patterns are defined in the collaborative space ontology and most of the patterns are supported by learning theories. In that point this pattern and other pattern (e.g. Organizational pattern[10]) differ greatly.

In this section we explain two types of patterns; a learning group configuration pattern to prevent the conflict at the abstract design, and an assignment tuning pattern to resolve the conflict at the concrete design.

The learning group configuration pattern represents an ideal collaborative learning configuration to achieve both the practical goal (to create or inherit intellect) and the learning goal (to learn intellect of type B); they describe rational relations among collaborative learning goals and participants' roles.

For example, the configuration pattern called "Share an idea for practice" (Table 2) has two goals such as a practical goal of sharing a good idea (which is intellect of type A) and a learning goal of developing creativity (which is intellect of type B) through practice. This pattern contains four roles such as a PRESENTER, ADVISER, APPRENTICE, and GUIDE. To achieve the practical goal, a PRESENTER explains her idea, and an ADVISER advises the PRESENTER to refine her idea. Through the interaction between ADVISER and PRESENTER, efficiency and quality of the practice will be improved. To achieve the learning goal, on the other hand, a PRESENTER also plays the role of a GUIDE of APPRENTICE. A GUIDE supports an APPRENTICE to develop creativity by explaining her thinking process.

Table 2. Learning group configuration patterns

Pattern name	Practical goal / Learning goal	Scene	Configuration: Roles	Configuration: Practical interaction / Learning interaction: Role	Role	Learning theory
Share an idea for practice	Sharing a good idea	practical scene (e.g. create business plan)	•AD. •P. & G. •AP.	<u>ADVISER(AD.)</u> refines P.'s idea by advising the P.	<u>PRESENTER(P.)</u> creates a good idea by explaining her idea to the AD.	[11] [12]
	Develop-ing creativ-ity			<u>GUIDE(G.)</u> supports AP. by explaining the thinking process	<u>APPRENTICE(AP)</u> develops creativity by asking G. for advice	[13]
Share an idea by observa-tion	Sharing a good idea	practical scene (e.g. create business plan)	•AD. •P. •O.	<u>ADVISER(AD.)</u> refines P.'s idea by advising the P.	<u>PRESENTER(P.)</u> creates a good idea by explaining her idea to the AD.	
	Develop-ing creativ-ity			<u>OBSERVER(O.)</u> develops creativity by observing the interaction between AD. and P.		[14]
Refine intellect through debate	Refining intellect	practical scene (e.g. refine plan)	•R.& D. (all partici-pant assigned as this role)	<u>REFINER(R.)</u> conceptualizes intellect by pointing out implicit part of intellect		[15]
	Develop-ing debate competency			<u>DEBATER(D.)</u> develops debate competency by explaining herself		[16] [17]

Table 3. Tuning patterns

<table>
<tr><th rowspan="3">Pattern name</th><th>Abstract design</th><th rowspan="3">Conflict at concrete design</th><th colspan="2">Tuning</th><th rowspan="3">Learning theory</th></tr>
<tr><th>Role</th><th colspan="2">Tuning steps</th></tr>
<tr><th>Scene</th><th>Tuning merit</th><th>Tuning demerit</th></tr>
<tr><td rowspan="2">Relax a role constraint & Add a support role</td><td>PRACTITONER role constraint: superior intellect</td><td rowspan="2">No candidate assigned to the expert role (hard to get competent person)</td><td colspan="2">First Step: relax the constraint of intellectual level required for the PRACTITIONER who is expected to create intellect of type A
Second Step: to add the SUPPORTER role</td><td rowspan="2">[13]</td></tr>
<tr><td>practical scene</td><td>Introducing a new learning goal for a PRACTITIONER to learn intellect of type B from a SUPPORTER</td><td>Lower efficiency of the task. Heavy burden to SUPPORTER</td></tr>
<tr><td rowspan="2">Divide a role constraint</td><td>PRACTITONER role constraint: several domain intellect (domain A, domain B)</td><td rowspan="2">No candidate assigned to the expert role (hard to get competent person)</td><td colspan="2">First Step: divide role constraint of PRACTITIONER into each domain expert constraint (P1, P2)
P1: domain A expert
P2: domain B expert
Second Step: assign appropriate participant to both P1 and P2</td><td rowspan="2">[18]
[19]</td></tr>
<tr><td>practical scene</td><td>Participant a (or b) inherits domain B (or A) intellect and clarifies her thinking process through discussion</td><td>No integration of several viewpoints. Less discussion</td></tr>
<tr><td rowspan="2">Shift a role constraint & Impose stronger constraint</td><td>PRACTITONER role constraint: superior intellect

OBSERVER</td><td rowspan="2">No candidate assigned to the expert role (hard to get competent person)</td><td colspan="2">First Step: relax the constraint of intellectual level required for the PRACTITIONER who is expected to create intellect of type A
Second Step: impose stronger constraint on the OBSERVER because OBSERVER is well required to have competency of observation and analysis</td><td rowspan="2">[14]</td></tr>
<tr><td>learning scene</td><td>PRACTITIONER gets better understanding and teaching skill by presentation to OBSERVER who develops practical competency.</td><td>Lower efficiency and quality of the task</td></tr>
</table>

A tuning pattern represents a pattern to adjust the abstract design to the real organization situation. For example, the tuning pattern named "Relax a role constraint & Add a support role (top of Table 3)" solves the problem that the designer cannot assign an appropriate participant to the expert role. This pattern has two steps to solve the problem; The first step is to relax the constraint of intellectual level required for the PRACTITIONER role; this role holder creates intellect of type A. The second step is to add the SUPPORTER role to compensate a drop of expert's intellectual level caused by the first step. Moreover, the pattern introduces a new learning goal for a PRACTITIONER to develop creativity (intellect of type B) from a SUPPORTER.

3 Design Support System for Collaborative Learning

3.1 Overview of the Design Support System

Fig. 1 illustrates the support process and the basic components of the system. The support consists of the following which correspond to the three design processes mentioned in the beginning of Sect. 2.

Abstract design support: Provide collaborative space configurations appropriate to the collaborative activity goals specified by designer using the learning group configuration pattern(Fig.1(A)). Designer may select one from the configurations and adjust it to her own design intention. In the case there is no candidate that fits with her design intention, she may design it from scratch.

Concrete design support: Provide the candidate resources suitable for the role in the collaborative space configuration(Fig.1(B1)). The candidates are selected based on the matching between the role definition and the resource property provided by Intellectual Genealogy Graph[7] which affords a good foundation for intelligent support for organizational activities(Fig.1(B2)). In the case that no candidate is found, the tuning patterns will be applied to the conflicts between the required constraint in the abstract configuration and the real situation and then the possible constraint relaxations are recommended(Fig.1(B3)).

Negotiation support: Provide an invitation message for all the participants based on the collaborative space design(Fig.1(C)). The message is automatically generated for each participant to explain the expected behavior in collaborative learning. Designer may modify the message to fit with her design intention. This message plays an especially important role to establish a better common understanding of collaborative learning among the participants.

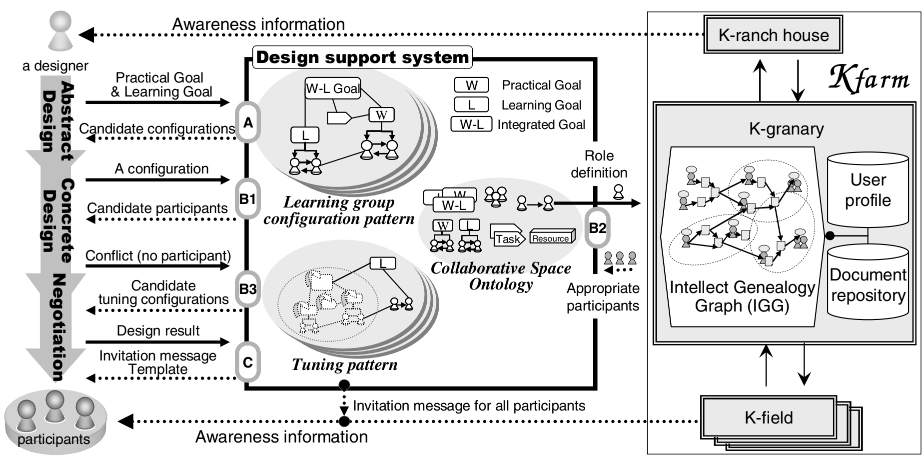


Fig. 1. Overview of the design support system for collaborative learning in *Kfarm*

3.2 An Example of the Design Support

In this section, we will see an illustrative example of support functions along a typical design process.

Assume that an organizational member released an idea memo through K-field to get feedback from other members. While monitoring the status of organizational intellect with K-ranch house, the designer found the memo is important for the development of organizational intellect of type A. The designer tries to refine the idea in a collaborative learning and utilize the space to develop novices' competency of creativity (intellect of type B). Then the designer examines the current status of the intellect on the memo by browsing Intellectual Genealogy Graph through K-ranch house and knows that the current state of the intellect is the "personal idea" that means it is not well refereed in the organization and is not represented in systemic way (i.e., implicit intellect).

The designer selects the practical goal of "Sharing a good idea" and the learning goal of "Developing creativity" from the goals recommended by *Kfarm*(Fig.2(A)). "Sharing a good idea" means to change the intellect status from personal to sympathetic, while "Developing creativity" means internalize the organizational competency of creativity to organizational members.

Kfarm provides the collaborative space configurations suitable for the two goals based on the collaborative group configuration pattern. In this case, *Kfarm* finds two patterns, "Share an idea for practice" and "Share an idea by observation" shown in Table 2. Assume that the designer selects "Share an idea for practice" and use it without any modification.

The next design phase is the concrete design where the designer embodies the abstract design by assigning real resources to the roles appearing in the abstract design(Fig.2(B1). The designer is carrying out the assignment of PRESENTER & GUIDE in the "Share an idea for practice" pattern. This role holder is expected to present her idea nicely and to exemplify her creativity in order for the novices to easily learn the roles of creating idea.

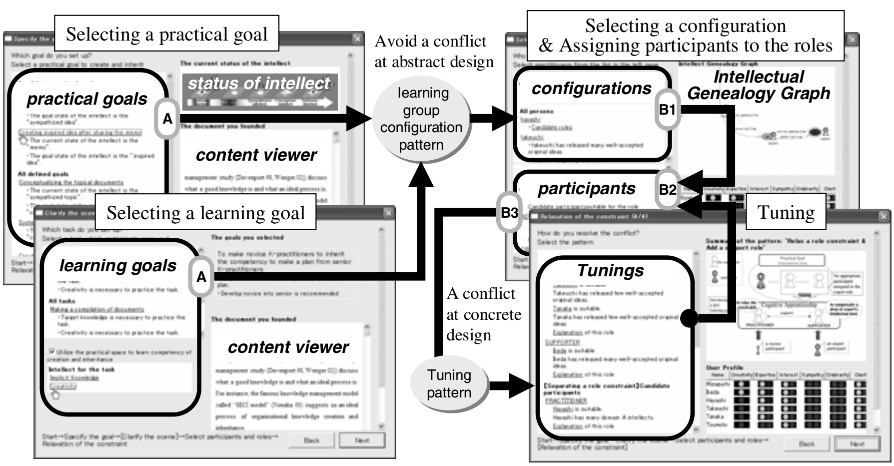


Fig. 2. A screenshot of the design support system

So the designer wants to find a participant who is highly creative (practical viewpoint; intellect creation of type A) and good at assisting others' learning (learning viewpoint; intellect learning of type B). The constraints on the role are specified in the collaborative space ontology and *Kfarm* finds candidates who satisfy the constraints with the aid of IGG in K-granary(Fig.2(B2)). K-granary, for example, interprets "highly creative person" as the person who has released many original ideas that are well accepted as meaningful systemic intellect for the organization.

If the realization of the abstract design is difficult because of conflicts(Fig.2(B3)), *Kfarm* suggests relaxation of the constraint based on the tuning pattern (Table 3). In our case, the pattern; "Relax a role constraint & Add a support role" is applicable. By reducing the intellect level required for the PRESENTER & GUIDE to enable an assignment to the role and at the same time introducing the SUPPORT role to keep the efficiency and the quality from practical viewpoint.

Once the concrete design is completed, *Kfarm* generates an invitation message for each participant.

4 Conclusions

We have discussed issues of design support for collaborative learning based on the collaborative space ontology and patterns, which will be able to make it easier to design an effective collaborative learning in practical environments. With the ontology engineering approach, it is possible to avoid and prevent conflicts in the design process which would have occurred in the design process or design result.

We have been developing a design support system based on the framework, which we call *Kfarm*. The relationship between this system and *Kfarm* enabled to show the candidate of the appropriate participants to the role of the collaborative learning.

We have implemented the prototype of the design support system except some details. It is difficult to make truly convincing evaluation of systems and tools in education because we need considerable long time to properly measure learning effect. Furthermore, in our case, it is also difficult to make comparative evaluation at the conceptual level because, as far as the authors' knowledge, there exists no similar system which supports integration of practice and learning for collaborative space design. However, collaborative space ontology we developed has been designed based on learning theories which are already shown to be valid and useful. For this reason we believe that the system based on the ontology supports the designer to design effective collaborative space. The implemented prototype system has shown the feasibility of our idea which strongly encourages us to go to the full evaluation after the completion of the full-scale implementation of the system.

References

1. Davenport, T., Prusaki, L.: Working Knowledge. Harvard Business School Press (1998)
2. Wenger, E., McDermott, R., M. Snyder, W.: Cultivating Communities of Practice. Harvard Business School Press (2002)
3. Nonaka, I., Toyama, R., Konno, N.: Seci, ba, leadership: a unified model of dynamic knowledge creation. Long Range Planning Vol. 4-5/1 (2000) 1-31

4. Hayashi, Y., Tsumoto, H., Ikeda, M., Mizoguchi, R.: *Kfarm*: An ontology-aware support environment for learning-oriented knowledge management. The Journal of Information and Systems in Education 1(1) (2003) 80-89
5. Mizoguchi, R., Bourdeau, J.: Using ontological engineering to overcome ai-ed problems. Int. J. of Artificial Intelligence in Education 11(2) (2000) 107-121
6. Ikeda, M., Seta, K., Mizoguchi, R.: Task ontology makes it easier to use authoring tools. In: Proc. of IJCAI'97, Nagoya, Japan. (1997) 342-347
7. Hayashi, Y., Tsumoto, H., Ikeda, M., Mizoguchi, R.: An intellectual genealogy graph affording a fine prospect of organizational learning. In Cerri, S.A., Gouard`eres, G., Paragua□cu, F., eds.: Intelligent Tutoring Systems. Volume 2363 of Lecture Notes in Computer Science., Springer (2002) 10-20
8. Inaba, A., Supnithi, T., Ikeda, M., Mizoguchi, R., Toyoda, J.: How can we form effective collaborative learning groups? In Gauthier, G., Frasson, C., VanLehn, K., eds.: Intelligent Tutoring Systems. Volume 1839 of Lecture Notes in Computer Science., Springer (2000) 282-291
9. Kozaki, K., Kitamura, Y., Ikeda, M., Mizoguchi, R.: Hozo: An environment for building/using ontologies based on a fundamental consideration of "role" and "relationship". In G´omez-P´erez, A., Benjamins, V.R., eds.: EKAW. Volume 2473 of Lecture Notes in Computer Science., Springer (2002) 213-218
10. Coplien, J.O.: A development process generative pattern language. In Coplien, J.O., Schmidt, D., eds.: Pattern Languages of Program Design. Addison Wesley (1995) 183-237
11. Vygotsky, L.S.: The problem of cultural development of the child. II. Journal of Genetic Psychology 36 (1929) 414-434
12. Vygotsky, L.S.: Mind in Society: The development of higher psychological processes. Harvard University Press, Cambridge, MA (1930, Re-published 1978)
13. Collins, A.: Cognitive apprenticeship and instructional technology. In Idol, L., Jones, B.F., eds.: Educational values and cognitive instruction: Implications for reform, Hillsdale, NJ, Lawrence Erlbaum Associates (1991) 121-138
14. Bandura, A.: Social Learning Theory. General Learning Press (1971)
15. Papert, S.: Mindstorms: Children, Computers, and Powerful Ideas. Harvester Wheatsheaf (1980)
16. Lave, J.: Cognition in practice: Mind, mathematics and culture in everyday life. Cambridge University Press (1988)
17. Lave, J.,Wenger, E.: Situated Learning: Legitimate peripheral participation. Cambridge University Press (1991)
18. Resnick, L.B.: Shared cognition: Thinking as social practice. In Resnick, L., Levine, J., S.Teasley, eds.: Perspectives on Socially Shared Cognition, Hyattsville, MD, American Psychological Association (1991) 1-22
19. Salomon, G.: What does the design of effective cscl require and how do we study its effects? In: Proc. of 2nd ACM Conference on Computer Supported Collaborative Learning. Volume 21(3)., ACM Press (1992)

The Big Five and Visualisations of Team Work Activity

Judy Kay[1], Nicolas Maisonneuve[1], Kalina Yacef[1], and Peter Reimann[2]

[1] School of Information Technologies, University of Sydney, Sydney, NSW 2006, Australia
{Judy, Nicolas, Kalina}@it.usyd.edu.au
[2] Research Centre for Computer-supported Learning and Cognition, University of Sydney, Faculty of Education, Sydney, NSW 2006, Australia
p.reimann@edfac.usyd.edu.au

Abstract. We have created a set of novel visualisations of group activity: they mirror activity of individuals and their interactions, based upon readily available authentic data. We evaluated these visualisations in the context of a semester long software development project course. We give a theoretical analysis of the design of our visualizations using the framework from the "Big 5" theory of team work as well as a qualitative study of the visualisations and the students' reflective reports. We conclude that these visualisations provide a powerful and valuable mirroring role with potential, when well used, to help groups learn to improve their effectiveness.

1 Introduction

Recent studies on computer-supported collaborative learning (CSCL) show that the expected beneficial outcomes of teamwork (such as high motivation, deep involvement in learning, and substantial knowledge gains) often do not materialise. An increasing number of studies and observations is reporting low participation rates, low levels of communication and collaboration (both in terms of quantity and quality of contributions), small knowledge gains, and little satisfaction with the group learning situation (e.g. [1], [2]). Kreijns, Kirschner and Jochems [3] have identified the tendency to assume that social interaction will occur automatically once the environments makes it possible, and the tendency to forget the social and psychological dimension of social interaction as two major pitfalls in designing and deploying collaborative learning systems.

We take the position that a group of students, in order to learn to work collaboratively, need to put effort not only in task-work, but also in *teamwork*. Teamwork can be defined as "…a set of interrelated thoughts, actions, and feelings of each team member that are needed to function as a team and that combine to facilitate coordinated, adaptive performance and task objectives resulting in value-added outcomes." ([4], p. 562). It is teamwork that ensures the success of teams at the workplace, and there is no reason to believe that this would be different for teams whose focus is on learning. The question of what processes and components comprise teamwork and how teamwork contributes to team effectiveness has received much attention in social psychology. A recent review of this body of research resulted in the identification of the "Big Five" components of teamwork [4]. The elements that make up teamwork, independent of the task a team has to perform, are [4]:

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 197 – 206, 2006.

- Team Leadership: ability to direct and coordinate other team members' activities, assess team performance, assign tasks, develop team knowledge and skills, motivate team members, plan and organise, and establish a positive atmosphere.
- Mutual performance monitoring: ability to develop common understandings of the team environment and apply appropriate task strategies to accurately monitor teammate performance.
- Backup behavior: ability to anticipate other team members' needs through accurate knowledge about their responsibilities. Includes the ability to shift workload among members to achieve balance during high periods of workload or pressure.
- Adaptability: ability to adjust strategies based on information gathered from the environment through the use of backup behavior and reallocation of intra-team resources. Altering a course of action or team repertoire in response to changing conditions (internal or external).
- Team orientation: propensity to take others' behavior into account during group interaction and belief in importance of team goal over individual members' goals.

Teams that enact these five competencies will enjoy improved performance. However, in order to fully realise this performance improvement potential, research shows that three *coordinating mechanisms* need to be in place in addition [4]:

- Shared mental models: An organising knowledge structure of the relationships among the tasks the team is engaged in and how the team members will interact.
- Mutual trust: The shared belief that team members will perform their roles and protect the interests of their teammates.
- Closed-loop communication: The exchange of information between a sender and a receiver irrespective of the medium.

In our approach to support online (learning) teams, we trace students' interaction behavior along these dimensions and provide visualisations which are mirrored back to the groups. It is important to note that at this stage in time we are not attempting to provide groups with *feedback* in the strong sense that we could identify and visualise differences between optimal performance and a group's actual performance; rather, we *mirror* information pertaining to the components of teamwork for the groups (see also [5]). We believe that groups will profit from 'only' mirroring information, provided that this information speaks to the appropriate points. Building on the theoretically and empirically well grounded "Big Five" framework, we hope to have identified the appropriate points.

We have created a set of novel visualisations of group activity: they have been designed to mirror the activity of the individuals and their interactions, based upon readily available authentic data from the groups. We have evaluated these visualisations in the context of a semester long software development project course. The students worked in teams of 5 to 7 and were assessed on the demonstrated quality of the software product and the effectiveness of the software and group processes in achieving that product. Students were required to make use of the version control system Subversion (SVN, http://subversion.tigris.org/) to maintain the versions of their software and Trac (http://www.edgewall.com/trac/) to support group communication via a Wiki as well as a Ticket system which supports allocation of tasks and tracing them against milestones. Data from these was used to build visualisations of the activity of each person in each group.

The two main questions addressed in this paper are: How well do the visualisations capture information relevant in the context of the Big Five framework? Is there a relation between patterns identifiable through these visualisations and group performance outcomes? In the next section, the visualisations are introduced. Next, we analyse to what extent these visualisations allow one to assess the Big Five components. Finally, we describe relations between patterns as revealed by the visualisations and group performance. We conclude with a comparison to similar approaches.

2 Overview of Visualisations

Activity Radar. As shown in Figure 1, an Activity Radar (inspired by [6]) consists of a circle, representing the range of participation, and colored dots, each representing an entity for which we want to compare participation levels: often a dot is a team member but it could also be a classroom or a group. Each dot is placed on a radius (always the same one) and moves to the centre as the member's level of participation increases: a person whose dot is in the centre has the highest level of participation whilst a person whose dot is on the perimeter has the lowest level of participation. The inner, darker circle perimeter represents the average level of participation. The unit of the participation depends on the medium (which explains why we have three different graphs associated with the three media, mixing different units in a single graph would not be meaningful). For the SVN and for the Wiki media, the amount of participation is the number of added lines. For the Ticket medium, it is the number of ticket events performed by the member. The highest participation (the center) and the average (the perimeter of the dark inner circle) values can be defined separately, thus changing the scale. Therefore the scale can be relative to the group only or to the whole class.

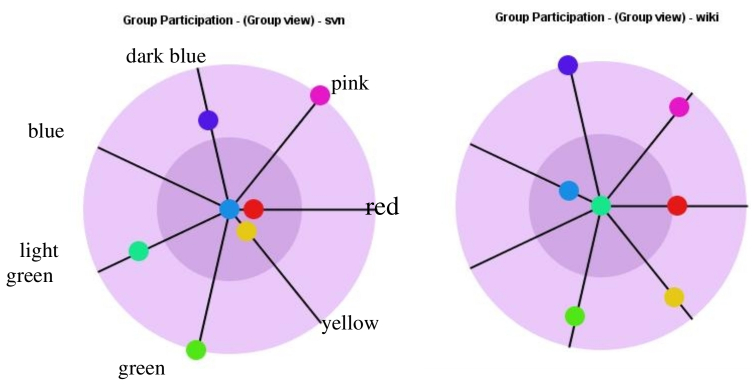


Fig. 1. Activity Radar for SVN and Wiki (colors will not show on black and white print)

For instance Figure 1 shows that the bulk of the SVN activity (on the left) is done by three students (blue, yellow and red). Whilst the blue dot was still very active on the Wiki medium (on the right), the red and yellow dots were much less engaged. The most active person on the Wiki (light green dot) was fairly inactive on the SVN.

Interaction Network. This representation is based on Social Network Analysis [7], which is concerned with capturing relationships and flows between entities. It assumes that these relationships reveal some important features of the group. The network is modeled as a unidirectional graph (although we have introduced some direction for one medium), consisting of a set of nodes and edges, where each node represents a user and an edge represents an interaction between the two corresponding users. In our context, we defined the notion of interaction between two members when they modify the same resource (in a specific interval of time or not). The width of the edge is proportional to the number of interactions between them.

Figure 2 shows an (ideal) case of a group where all team members interacted a lot with each other over the Wiki medium. In contrast, Figure 3 for SVN shows a different pattern of participation for a group where only three team members interacted.

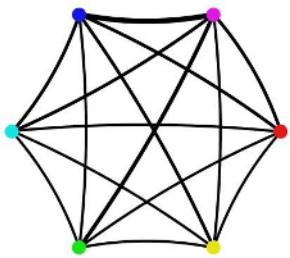


Fig. 2. Interaction Network for Wiki

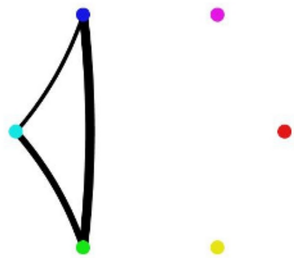


Fig. 3. Interaction Network for SVN

Wattle Tree. We have created the Wattle Tree[1] (see Figure 4), a novel graphical representation, where each user's activity is shown in a climbing vertical "tree" (timeline). The tree starts when the user first performs an action in any of the three media considered. The left axis indicates the day.

Wiki-related activity is represented by yellow "flowers", which, like the flowers on a wattle tree, look like yellow circles, appearing on the left of the trees. SVN-related activity is similarly represented, as (light and dark) orange flowers on the right of the tree. The size of the flower indicates the size of the contribution. Tickets-related actions are represented by leaves (lines in our current representation): a dark green (left) leaf indicates a ticket was open by the user whilst a light green (right) leaf indicates a closed ticket. The length of the left leaf is proportional to the time it remained opened. Those still open are shown at a standard, maximal size, as in the case of the bottom ones of Figure 4. A well-organised, efficient group should have many leaves, of small to medium length, on either side, with lots of activity (Wiki, SVN) in between. A small number of left leaves, especially if they are of maximal length, indicates that users work on very chunky and large tasks, and do not use the ticket system to good effect (eg. forget to close their tickets).

The group represented in Figure 4 shows that the team members used the Ticket system moderately well. Early in the project they used the Wiki often, whilst the SVN

[1] We use the analogy with the wattle tree (acacia), an Australian native tree with green leaves and grapes of round, fluffy yellow flowers.

activity kicked in a third of the way into the project. The overall activity is quite well spread in time, except for the clearly visible semester break. The distribution of workload shows a greater burden taken by the two left-most members but all members sustained a quite steady load. This is consistent with a normal, well involved group.

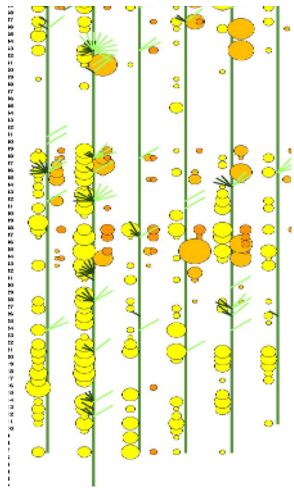


Fig. 4. Wattle Tree of a well-functioning group

3 How the Visualisations Can Relate to Each Big5 Factor

We seek here to investigate how the five important factors of successful team work and their respective behavioral makers can be assessed through our visualisations. We address each "Big 5" element in turn.

Team Leadership. This factor is easily observed in the Ticket actions: the team leader typically assigns many tickets (seen on AR/T[2]), to each team member (reflected on IN/T). S/he also interacts with all team members, through ticket activity and Wiki medium. We expect a good team leader to be close to the centre in AR/T within the average circle in AR/W, and to see thick connections from the leader to all other team members in the IN/T. The IN/W may also show important connections involving the team leader, but not exclusively. We also expect to see in the Wattle Tree a continuous activity from the team leader throughout the project, and a rise in ticket and Wiki activities before the deadlines.

Mutual Performance Monitoring. This factor can be partially assessed by the INs. A very low level of interactions between team members on all media may indicate that they do not monitor each other nor pick up mistakes and lapses. The Wattle Tree can also reveal important information, if we observe the time elapsed between team members' actions. Members of a monitoring team are quicker to respond to each other, and tickets do not remain open for a long time without reassignments.

[2] AR stands for Activity Radar, T for Ticket. Similarly we denote AR/W and AR/S, as well as IN/W, IN/T, IN/S for the three Interaction Networks.

Table 1. Team Leadership

Behavioral Markers	Activity Radar			Interaction Network			Wattle Tree[3]
	T	W	S	T	W	S	
Facilitate team problem solving	+	+		+	+		Continuity in time
Provide performance expectations and acceptable interaction patterns	+	+		+	+		Continuity in time
Synchronize and combine individual team member contributions	+	+		+	+		Continuity in time, esp. before deadlines
T = Ticket; W = Wiki; S = Subversion							

Table 2. Mutual performance monitoring

Behavioral Markers	Activity Radar			Interaction Network			Wattle Tree
	T	W	S	T	W	S	
Identifying mistakes and lapses in other team members' actions				+	+	+	Time elapsed between interactions, opening time of tickets.
T = Ticket; W = Wiki; S = Subversion							

Backup Behavior. A main aspect of backup behavior is the ability to shift workload amongst team members to achieve balance during periods of high workload and pressure. The IN/T would give an idea of how well tickets were distributed by the team leader. Activity Radars, on all media, are an indicator of how much each member participated on average. In particular the assignment and re-assignment of tickets gives a good indication of how tasks are distributed. However the most information is given by the Wattle Tree, as it shows, at any given time, the amount of activity for each team member. Whilst the actions captured may not precisely reflect the amount of work done by the participant (e.g. a small Wiki entry may in fact be the result of several hours work), there is a good indication of how workload is distributed. A week before an important deadline for instance, where there is usually a burst of activity, a team that practices backup behavior would shift tasks at that time to the less busy members. So we would expect to see an even workload during these periods of pressure, even if they are preceded by a short, uneven period.

Table 3. Backup behavior

Behavioral Markers	Activity Radar			Interaction Network			Wattle Tree
	T	W	S	T	W	S	
Recognition of workload distribution problem in the team.	+	+	+	+			Distribution of workload, re-balancing of workload
Shifting work to underutilised team members	+	+	+	+			
T = Ticket; W = Wiki; S = Subversion							

[3] In each table, we indicate with a '+' whether the visualisation addresses the marker. For the Wattle Tree, more complex, we give a textual clarification of what element addresses it.

Adaptability. This factor is difficult to assess as its absence does not imply that the team is not successful. For instance the task, team and resources may be problem-free, hence the team does not have the opportunity to show its adaptability. However if we know of a problem or a change then we can observe how the team reacted. One team, for instance, had an inactive "team leader" so one member informally took the lead and the team managed to complete the task. Whilst all the visualisations may show a problem (e.g. an inactive team member), the reaction of the team may not be always observable on the visualisations. When the time of these changes is known, then we can gain cues from the Wattle Tree since it is time-based. For example, we could see when the other team member took informal leadership of his team and how long it took the others to respond to his actions. Importantly, the team members, having deep knowledge of the dynamics and situation may recognise evidence of such shifts.

Table 4. Adaptability

Behavioral Markers	Activity Radar			Interaction Network			Wattle Tree
	T	W	S	T	W	S	
Identify cues of change, assign meaning to it, and develop a new plan to deal with it	+	+	+	+	+	+	See actions related to critical times. Time between actions
T = Ticket; W = Wiki; S = Subversion							

Team Orientation. The Wattle Tree provides a nice picture of the degree of involvement of each individual during periods of high pressure, such as the completion of a project milestone. AR and IN diagrams give an indication of how much the members participate overall, how much they interact and in which direction. Thick, even-colored links between team members show that they interact a lot, on average in a symmetric two-way fashion. Reassignment of tickets, tickets closed by other members are also evidence that other team members participate in a task.

Table 5. Team orientation

Behavioral Markers	Activity Radar			Interaction Network			Wattle Tree
	T	W	S	T	W	S	
Increased task involvement, information sharing, strategising, and goal setting	+	+	+	+	+	+	Coinciding actions in time
T = Ticket; W = Wiki; S = Subversion							

4 Relation to Group Performance

Here we only describe relationships between the visualisations and observed team performance, in particular relationships with the quality of outcomes (as reflected in grades for group management). Our first, and most important, source for gaining an understanding of student perceptions, was the final reports. Each group was required to write a 1-2 pages reflective statement about its achievements, limitations and what had been learnt. In addition, students reported their contributions and made a reflective statement. Nine of the ten groups had access to the diagrams to use for their reports and six did so. The second main source of evidence was the information in the Subversion and Trac repositories: whenever we wished to understand more about a group, we could examine these in detail. Indeed, it is this huge collection of information that our visualisations are intended to summarise and overview. The following analysis is based around the elements of the Big Five model and reports our observations of the role of our visualisations, in relation to them.

Team Leadership

Facilitate team problem solving: In effective groups, the characteristic pattern of Wiki activity by the team leader appeared as many yellow flowers on their Wattle Tree. The Interaction Network clearly distinguished successful leaders because it showed that they interacted with all group members. For less successful groups, there are clear indications of problems: for example, a nominal group leader had absolutely no interactions with anyone on any medium. The Wattle Tree pattern of SVN submissions and Wiki activities for each member gave a gross indication of people performing assigned tasks. These were very valuable when combined with scrutiny of the details in the system.

Provide performance expectations and acceptable interaction patterns: In the final reports, it was striking that, in all 5 groups which referred to the diagrams, the team leader was one of the people who did so. They referred to them in relation to just these aspects of performance and interaction. One of them noted that they were unreliable in relation to one team member who did not make their own SVN submissions, but clearly showed that this aspect of the visualisations presented a very clear pattern.

Synchronise and combine individual team member contributions see and evaluate information that affects team functioning: For some groups, this was visible in the SVN interactions which seemed to occur when individual code was integrated. Our visualisations are complemented by the information that is presented by Trac/SVN on the history of each document.

Mutual Performance Monitoring

Identifying mistakes and lapses in other team members' actions: In the case of extreme social loafing, the Wattle Tree (and, to a lesser degree, the AR diagrams) made this very clear. The students tended to avoid criticising each other in their reports but two leaders pointed to the pattern in these diagrams as clear indicators of failure to contribute. The individuals involved made no comment on them (even though they should have read the whole report, including the leader's comments critical of them), perhaps because they had nothing to offer to refute this. In the case mentioned above where the group leader did the SVN commits for another person, that person mentioned this in their report and pointed to comments in the commit which indicated this. This case clearly shows that the displays made lapses evident.

Student reflections included several mistakes, such as doing the wrong job, but these were not visible in our displays. However, there were cases where students reported a task being taken over because the person initially allocated it had problems and, somewhat surprisingly, this was reflected in a change in the Wattle Tree with a shift from SVN to Wiki activity.

Backup Behaviour
Recognition of a workload distribution problem in the team. The size of the flowers and their frequency are an indication of this. Although we were only able to make the diagrams available at the end of the course, they would have made an excellent basis for a group discussion earlier. For example, where a person was not active, this could be explored by the group.

Shifting work to underutilised members: As described above, this was sometimes visible in the change in pattern of SVN/Wiki activities of the individuals involved as well as interactions on the SVN for code from tasks taken over.

Adaptability
Identify cues of change, assign meaning to it, develop new plan to deal with it: This is similar to the issues of back up described above under backup.

Team Orientation
Increase task involvement, information sharing, strategising and goal setting. In successful groups, this seems to have been reflected in high Wiki activity early in the semester. It is also reflected in the Interaction Network diagrams with the most successful groups having quite rich interaction on the Wiki, with all members having some interactions with all others. There was also one group where the division of work meant that half the group had high interaction on the Wiki and the other half had high interaction on SVN. This matched their individual reflections.

5 Conclusion

We have presented our visualisations that externalise the long term activity of groups. The Activity Radar derives from Erickson [8], but we focus on asynchronous contributions over a long period rather than current chat actions. The Interaction Network is somewhat like Social Network Analysis diagrams. The most novel of our visualisations, the Wattle Tree, was inspired by Donath et al. [9] which summarises activity on a single medium by diffuse communities. The importance of this work is reflected in the large body of work in the CSCW community on various aspects of awareness [10], much of it informing the facilities available in Trac/SVN. We have not found any work that has captured the long term, high level, graphic summary of individual activity within teams. The Wattle Tree has proved particularly useful because it gives the right mix of detail and high level summary over a long period.

We have motivated our approach to visualising team interactions with the Big5 model of teamwork [4], and demonstrated that the visualisations can express various aspects of the components of teamwork. A qualitative analysis revealed a number of relations between patterns observable in the visualisations and team performance. As mentioned in the introduction, the visualisations do not communicate normative information; they do not show how a group or a team member ought to perform. Current research fails to offer optimal values for the five components. It is also unlikely that a general optimal combination of values can be identified: they are affected by the situational demands, as well as the history of the group (e.g. how well the group members know each other) [8]. One way around this problem is to work purely inductively and base feedback on what worked for teams in similar situations [11]. This, however, requires analysis of a great number of similar situations. In our approach, the interpretations are left to the team members themselves or to those who are very familiar with the specifics of teams, such as their managers or, in instructional settings, tutors. This avoids the possibility of ill-founded feedback and advice; it can also be argued that leaving the normative decisions to groups themselves has positive motivational effects and has the potential to eventually lead to more stable and satisfied groups [12].

Acknowledgements. This work is funded by the Australian Research Council.

References

1. Heath, E.F., *Two cheers and a pint of worry: An on-line course in political and social philosophy.* Journal of Asynchronous Learning Networks, 1998. **2**: p. 15-33.
2. Zumbach, J., Reimann, P., Schoenemann, J. *Effects of resource distribution and feedback on computer-mediated collaboration in dyads. Towards sustainable and scalable educational innovations informed by the learning sciences*. 2005: IOS Press: Amsterdam.
3. Kreijns, K., Kirschner, P., Jochems, W., *Identifying the pitfalls for social interaction in computer-supported collaborative learning environments: a review of the research.* Computers in Human Behavior, 2003. **19**(3): p. 335-353.
4. Salas, E., Sims, D.E., Burke, C.S., *Is there a "Big Five" in teamwork?* Small Group Research, 2005. **36**(5): p. 555-599.
5. Reimann, P., Kay, J. *Adaptive visualisation of user models to support group coordination processes. 2nd joint workshop of cognition and learning through media-communication for advanced e-learning*. 2005. Tokyo, Japan.
6. Erickson, T. *Designing Online Collaborative Environments: Social Visualisations as Shared Resources. 9th International Working Conference on the Language-Action Perspective on Communication Modelling (LAP2004)*. 2004. New Brunswick, NJ.
7. Scott, J., *Social network analysis: A handbook*. 1991, London: Sage.
8. Erickson, T., Kellog, W.A., *Social translucence: An approach to designing systems that mesh with social processes.* ACM transactions on computer-human interaction, 2000. **7**(1).
9. Donath, J.S., *A semantic approach to visualizing online conversations.* Communications of ACM, 2002. **45**(4): p. 45-49.
10. Cadiz, J.J., Venolia, G., Jancke, G., Gupta, A. *Designing and deploying an information awareness interface. ACM CSCW*. 2002.
11. Soller, A., Wiebe, J., Lesgold, A. *A machine learning approach to assessing knowledge sharing during collaborative learning activities. Computer Supported Collaborative Learning (CSCL'02)*. 2002.
12. Kildare, R., Williams, R.N., Hartnett, J. *An online tool for learning collaboration and learning while collaborating. Australasian Computing Education*. 2006. Hobart, Australia.

Cognitive Tutors as Research Platforms: Extending an Established Tutoring System for Collaborative and Metacognitive Experimentation

Erin Walker[1], Kenneth Koedinger[1], Bruce McLaren[1], and Nikol Rummel[2]

[1] Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA
{erinwalk, koedinger, bmclaren}@cs.cmu.edu
[2] Department of Psychology, Albert-Ludwigs-Universitat Freiburg, Germany
rummel@psychologie.uni-freiburg.de

Abstract. Cognitive tutors have been shown to increase student learning in long-term classroom studies but would become even more effective if they provided collaborative support and metacognitive tutoring. Reconceptualizing an established tutoring system as a research platform to test different collaborative and metacognitive interventions would lead to gains in learning research. In this paper, we define a component-based architecture for such a platform, drawing from previous theoretical frameworks for tutoring systems. We then describe two practical implementation challenges not typically addressed by these frameworks. We detail our efforts to extend a cognitive tutor and evaluate our progress in terms of flexibility, control, and practicality.

1 Introduction

A cognitive tutor is an intelligent tutor that compares student action during problem-solving to a model of correct action, and provides context-sensitive hints and error feedback. These tutors have been shown to be effective at increasing student learning in long-term classroom studies by as much as one standard deviation over traditional instruction [1,2]. In general, cognitive tutors have focused on instruction to increase domain knowledge but have lacked support for collaborative activities or metacognitive tutoring. However, cognitive tutors could become even more effective if used in combination with collaborative and metacognitive interventions.

There is a need for research on which interventions are most effective. Collaboration can increase student mastery of domain knowledge, reasoning strategies, and social skills, but it is only effective when designed to encourage particular behaviors [3]. Although cognitive tutors like the Cognitive Tutor Algebra I (CTAI) are used in conjunction with collaborative classroom activities, it is difficult to control whether these activities occur as intended and difficult to measure their effectiveness. It is necessary to determine which activities produce desired learning effects and to control their execution. Similarly, preliminary research on supporting metacognition in intelligent tutoring systems has yielded encouraging results [4]. More research is needed to evaluate different interventions in a classroom context.

Tutoring systems are ideal environments for experimentation with collaborative and metacognitive interventions. They are useful settings for the implementation of

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 207 – 216, 2006.

collaborative activities: The controlled environment adds structure, and actions can be tutored so that desired behaviours are exhibited. Further, implementing metacognitive instruction within the context of an intelligent tutoring system can provide monitoring and support mechanisms for metacognition that may increase learning. Trying many interventions on the same system allows researchers to examine varying effects and explore how the interventions might complement each other. There are additional benefits to using an *established* tutoring system: savings in development time, preexisting relationships with classrooms that are using the tutor, and a proven baseline of effectiveness. For example, the CTAI is an integral part of the Pittsburgh Science of Learning Center (PSLC) which facilitates experimental studies in real-world contexts by providing access to schools and programmers.

An established tutoring system is a solid basis for a research platform. A research platform should be *flexible* in terms of the number of experiments that can be conducted, *controlled* in terms of the number of factors that can be compared in an experiment, and *practical* in terms of its ability to facilitate interventions that are used in a real classroom. The idea of a cognitive tutor as a research platform has been exploited before; existing tutoring systems like Project LISTEN have been used for large-scale data collection and analysis and have been extended to test hypotheses about tutoring cognitive skills [5]. However, these extensions have not contributed to a flexible framework for experimentation with more complex interventions. Preexisting tutoring systems have been extended to test collaborative and metacognitive hypotheses [6], but these projects have not created a platform that researchers can use to compare scenarios. Building a collaborative or metacognitive tutoring system from scratch to test certain research hypotheses [7,8] can provide flexibility and control, but because these systems are not based on an established tutor, they may not be practical for classroom use and would require much development to make them so.

In this paper, we describe the extension of an established tutoring system into a research platform that provides support for *both* collaborative and metacognitive interventions. We define a theoretical architecture for such a platform, discuss practical challenges in implementing the architecture, and evaluate our specific efforts in extending the CTAI into a platform for experimentation.

2 Component-Based Architecture

A research platform must allow flexibility in terms of the number of tutoring experiments that can be run and control within an experiment to facilitate ablation studies. These requirements can be met using a component-based architecture that emphasizes the development of independent, reusable components [9]. An ideal implementation of this approach yields a situation where components can be created for use in a variety of situations and can be added or removed without much effort.

Component-based architectures have been proposed as a way to enhance the effectiveness of intelligent tutoring systems, but it is often difficult to integrate components created for different purposes [10]. Standards for curriculum representation like the Scaleable Content Object Reference Model have been proposed to resolve this problem [11], and distributed architectures like KnowledgeTree [12] and multi-agent approaches such as I-help [13] have been developed to integrate individual web-based services more effectively. Architectures designed for collaborative activities have also been developed, with a focus on synchronizing objects between components to create shared activity spaces [14]. We model our approach after these approaches, focusing on developing reusable components and interaction standards.

We have developed a component-based architecture (Figure 1) for a research platform for collaborative and metacognitive tutoring, based on Ritter and Koedinger's architecture for plug-in tutoring agents [15]. We focus on students solving problems in a single application, unlike the above research, which focuses on the integration of multiple applications. We intend the components we discuss to be situated in a larger web-based system for the delivery of multiple applications (e.g., [12]), and to be capable of receiving curriculum information independently.

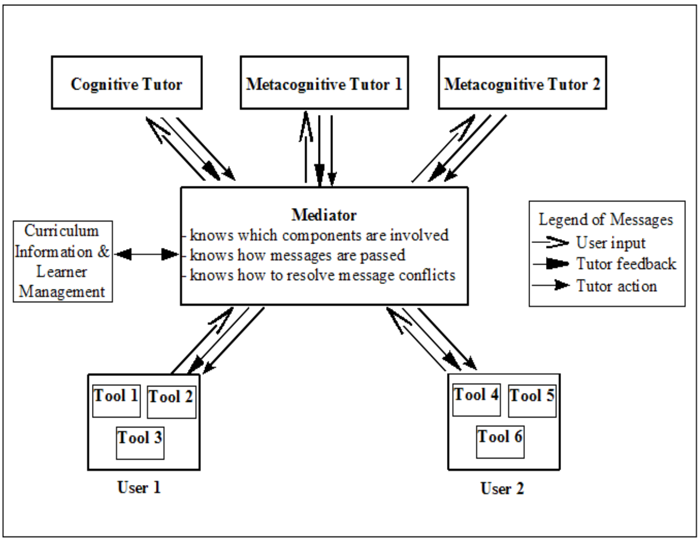


Fig. 1. Our component-based architecture. The figure depicts 3 tutors, 6 tools, and 2 users, but in theory, it could have any number of tools, tutors, and users.

We separate the components in our architecture into tools and tutors. Tool components are the part of the application that the user sees, while tutor components are the part of application that evaluates and offers advice to the student [15]. A spreadsheet is a tool, while the software module that gives feedback to a student on completing the spreadsheet is a tutor. Adding multiple tool components to a given application can facilitate collaboration; each collaborator might use a different tool on a different computer. Adding multiple tutor components to a given application can facilitate metacognitive tutoring. A tool that incorporates self-explanation could have two corresponding tutors, one that is responsible for the cognitive elements of the task, and one that is responsible for the self-explanation elements of the task [6]. To facilitate reuse, tools should be built to be compatible with any corresponding tutor, and tutors should be built to be usable with any corresponding tool.

Establishing standards for messages exchanged by components during a tutoring session, based on the protocol in [15], can also enable component reuse. In general, tools send messages that communicate information about user actions, while tutors send messages that provide feedback about user actions. A proposed set of messages is shown in Table 1. To facilitate reuse, any message that can be sent by a tool should be understood by a tutor, and any message that can be sent by a tutor should be understood by a tool. Although this protocol was originally designed to support cognitive tutoring, it can be applied to metacognitive and collaborative tutoring, where message content may be different but message structure would be the same.

Table 1. Message types for tool and tutor components

Component	***Message Type***	***Message***	***Meaning***
Tool	User action	Note input	Student uses an interface element
		Note create	Student creates an element
		Note delete	Student deletes an element
		Note hint request	Student asks for a hint
		Note done	Student indicates problem complete
Tutor	Tutor feedback	Approve	Answer is correct
		Flag	Answer is incorrect
		Point to	Points to an element
		Send message	Gives feedback
		Update assessment	Changes student assessment
	Tutor action	Send input	Performs user action
		Undo input	Undoes user action
		Send create	Creates element
		Send delete	Deletes element

Following Ritter [16], we define a remotely located *mediator* to control the interaction between components. The mediator is aware of which components are involved in a collaborative session, and during tutoring, receives messages and passes them to the appropriate components. The mediator also holds a set of rules for dealing with message conflicts (sample rules are described in [16]). Because knowledge for how to deal with interacting components is located within the mediator, only the mediator has to be changed when adding or removing components.

3 Practical Challenges

Although theoretical frameworks can provide guidelines for the design of independent, reusable components, there is a question as to whether these architectures are practical for development and classroom use. It is not always feasible to design for reuse because usability can be in conflict with reusability, and it can be impractical to develop reusable components on a realistic schedule. Architectures must be more specific about expectations for component reuse.

3.1 Feasibility of Designing for Reuse

There is a conflict between designing for reuse and designing for usability. Different tools can be easier to use together if their relation is represented in the interface. When a student has to input values in a spreadsheet and then plot the values on a graph, it is easier to have each point plotting widget located next to the appropriate spreadsheet cell, so that the connection between the two tasks is clear. However, tools then become dependent on each other, making it more difficult for developers to reuse them. Theoretical architectures need to account for compromises that must be made between usability and reuse. Developers could consider linked tools as a single tool, making the larger tool reusable.

Further, developing for reuse is not always an attainable goal. On a tight development schedule, it is unlikely that programmers can give tools the higher level of functionality required for reuse. When a tool was not originally designed for collaborative use, adding functionality for tutor actions is not a high priority. It also may be unreasonable to modularize components to the level required for reuse. For small tutoring interventions, it may be simpler to modify the primary tutor rather than to add an adjunct tutor. Frameworks should prioritize reuse requirements so that developers can implement critical functionality for reuse but code flexibly enough to facilitate the future implementation of other requirements. Explicit implementation guidelines for usability and reuse would aid in design decisions.

3.2 Separation Between Tool and Tutor Components

Taking usability and a practical development schedule into account can create difficult decisions when attempting to implement a theoretical framework that does not specify tool and tutor responsibilities. We attempt to define tool responsibilities to maximize the flexibility and experimental control provided by a multi-agent architecture. Tools should act as more than an interface to the tutor. For instance, in an equation solver, when a student subtracts both sides of an equation by five the result needs to be displayed by the tool. If the tool behaves as an interface, some agent would be needed to compute the result of the student action, which means that the tool is dependent on that agent for updating its state. One might argue that it is not practical to emphasize independence for tools that do not mimic anything outside a tutor. However, one should design with future extensions in mind. In an architecture where components can be easily added and removed, it is necessary that the tool hold the logic for calculating the result of student actions.

A more difficult question is whether the tool should require permission from a tutor to update its state based on student actions. When a student requests to create a point on a graph, does the tool wait for permission from a tutor to approve the request before creating the point (synchronous operation), or create the point, allow the student to continue, and then deliver tutor feedback (asynchronous operation)? Waiting for permission means that the tool cannot function independently of a particular tutor, but providing asynchronous feedback might decrease usability. One solution could be to increase tool functionality so that it can always undo the previous action. However, this solution is problematic for usability, as it may be confusing to the student to see their action mysteriously reversed (e.g., creating a point and then immediately deleting it). The best solution may be to sacrifice reuse and have certain actions, like user "create" actions, require an immediate response from a tutor.

The final decision in component separation is whether to give the tool knowledge about tutoring. When an entry in a spreadsheet is flagged, does the tool or the tutor know to display the error by turning the text red? If the knowledge is located in the tool, tutors will not need to know about the specifics of each tool in order to send approve and flag messages. However, having the tool know about the tutoring might be unreasonable in situations where tools are pre-designed; one would not expect Microsoft Excel to be developed with a tutoring framework in mind! Developers have to be prepared both to use pre-designed tools and to design tools specifically for tutoring. When using an off-the-shelf tool in a tutoring scenario, a translator component can be built to transform a flag message into a more specific series of messages for that particular tool. When designing or modifying a tool for use in a component-based tutoring framework, it should have feedback behaviors built in. Although theoretical architectures specify that tool and tutor components should be separated, examining the issue yields a set of guidelines on where that separation should occur.

4 Evaluation of Progress

We now evaluate our efforts in extending the CTAI into a research platform. We wished to retain the strengths of the CTAI while adding collaborative functionality. To this end, we expanded the CTAI to incorporate a peer tutoring script (PTS). We have implemented the architecture from Section 2, while negotiating the challenges discussed in Section 3. Our changes form the beginning of a flexible, controlled, and practical framework for future extensions.

4.1 Proposed Extension to the Cognitive Tutor Algebra-1

The CTAI focuses on "the mathematical analysis of real-world situations and the use of computational tools" [1]. Students read a word problem and use various tools to solve the problem. As students work, the cognitive tutor monitors their progress based on a model of performance and gives immediate feedback. When students make an error, the cognitive tutor will immediately "flag" it (e.g., by turning input text red) and, for common errors, output a message that explains the misconception. At any time, the student can request help, and the cognitive tutor will provide hints. The tutor keeps an estimate of student mastery of skills. Skill levels are displayed so that students are aware of their progress, and problems are chosen based on skills that students have not yet mastered.

We are integrating collaboration and metacognitive tutoring into the CTAI using a peer tutoring script. Instead of the computer tutoring students on math, students tutor each other, while the computer provides collaborative support. The human peer tutors prepare by solving the problem that they will be teaching with the help of the computer tutor. While tutoring, peer tutors mark tutees' answers as wrong or right, provide hints and feedback in a chat window, and assess the tutees' skill mastery. Peer tutees can also engage the tutor in discussion in the chat window. See Figure 2 for a screenshot of the peer tutor's interface. Peer tutoring scripts have improved learning, particularly when peer tutors prepare ahead of time, peer tutors provide elaborated explanations which peer tutees use constructively, and students set goals for the tutoring and monitor skills being acquired [17]. We believe that the PTS encourages these elements and will enhance the effectiveness of the CTAI.

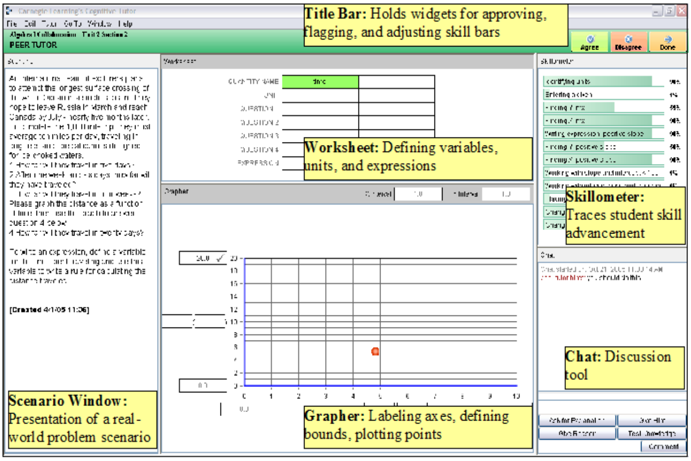


Fig. 2. Screenshot of peer tutor's interface in the peer tutoring script (PTS)

4.2 Implementation of Extensions

To implement PTS, we changed the CTAI so that the tool and tutor components function independently of each other, are remotely located, and communicate through a mediator, as illustrated in Figure 3. Although the CTAI was designed to follow the architecture described in [15], development constraints lead its current state to evolve from this ideal. Components were dependent on each other for launching, and some tool functionality was located within the tutor. The tool and tutor were capable of communicating remotely using the message protocol defined in Section 1, but there was no mediator in place. We created central classes that could function remotely to control functions such as beginning a session, launching the tutor, and shutting down the tutor. We added a mediator to intercept remote messages between the tool and tutor.

We then created some new tutor components, which required further negotiation of the tool/tutor separation. We built an echoing module to echo input from one user's screen onto the other. The echoing tutor receives user action messages, and transforms them into the appropriate tutor feedback and action messages. For example, a “note input” message on a given widget would be changed into a parallel “send input” message. To make the echoing tutor effective and reusable, we implemented most of the tool-side functionality detailed in Section 3.2, and improved tool response to tutor action messages, which were rarely used in the CTAI. We tested the echoing tutor in a configuration with the original cognitive tutor, and two original tools (see Figure 3). We also implemented a prototype metacognitive tutor to listen for certain messages and provide metacognitive instruction when appropriate.

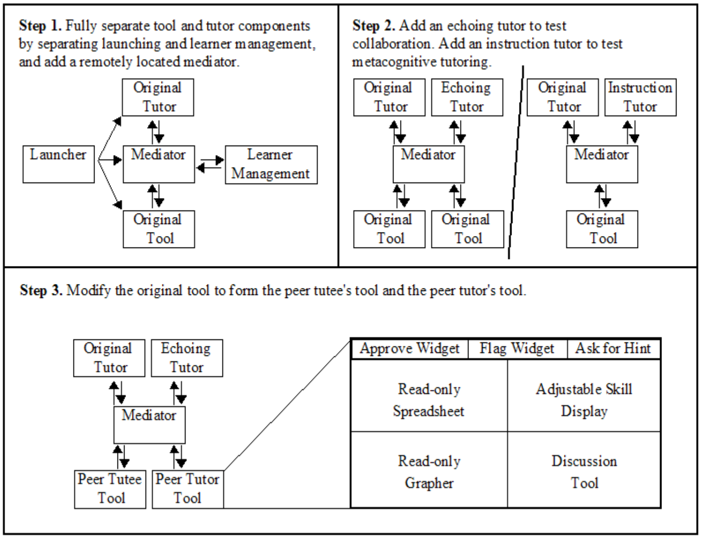


Fig. 3. Three steps in extending the CTAI to implement the peer tutoring script

Finally, we created the tool components required for the PTS, attempting to plan for both usability and reuse. We added a chat tool for the peer tutor and tutee. We also modified the peer tutor's tools, disabling widgets for inputting answers, and adding widgets for approving and flagging to the title bar. This setup is more usable because these new widgets are in one place, but reusable in that changing the title bar removes peer tutoring functions from the tools while retaining other characteristics.

4.3 Results and Discussion

We evaluate our implementation with respect to our criteria for a good research platform: flexibility, control, and practicality. The framework that we have developed is flexible in terms of the potential reuse of components that have been implemented and the variety of new components that could be integrated. In addition to the peer tutoring extension to the CTAI, developed components suggest other extensions. Combining the echoing agent with two regular tools yields a collaborative setup for solving cognitive tutor problems. Combining the echoing agent and the cognitive tutor agent with a regular tool and a modified peer tutor tool so that the peer tutor cannot input values or perform tutoring actions yields an actor/observer configuration, where one person solves the problem, the other watches.

A variety of new components can be integrated into this framework, illustrating the extensibility of the CTAI. The mediator potentially allows any components to connect to it, as long as they include a translator class to translate the messages into the protocol we have developed. For example, we intend to use the framework to implement another collaborative session type, called the *collaborative problem-solving script* (CPS), where two students alternate between working independently and together to solve problems [18]. Students collaborate at the same computer terminal. They require a tool modified for the requirements of the CPS, the cognitive tutor, an instruction tutor that listens for certain messages from the cognitive tutor and provides scripted instruction, and a collaborative help tutor that listens for certain messages from the tool and provides adaptive support. Adding components to the CTAI would not be possible without the implemented framework.

The framework can also ensure experimental control in research by facilitating ablation studies to examine the independent effects of components. Because components have no knowledge of each other but are connected through the mediator, specifying a different configuration in the mediator can remove the component. For example, it is simple to compare versions of the PTS with or without a specific tutoring agent, simply by changing the mediator to include or exclude that agent. One could also make comparisons between different versions of the tools; for example, comparing the differing effects of using the regular skill display to the effects of using the peer tutorable display. In the CPS, removing the instruction tutor and/or the collaborative help component allows different interventions to be compared. The implemented framework allows examination of why an intervention is effective.

Practicality is the final criteria for evaluating our implementation. We have attempted to design for usability in addition to reuse, and tried to keep the development demands for reuse to a minimum. We have extended the CTAI, a tutor that has been shown to be effective, which means we are comparing our interventions to a gold standard. The tutor is already used in roughly 2000 classrooms across the United States, and is an integral part of the Pittsburgh Science of Learning Center (PSLC), which is engaged in facilitating experimental research in real classrooms. Using the CTAI means there is institutionalized support with accessing teachers, schools, and CTAI developers. At this early stage, it appears that the extended CTAI is practical for development and classroom use.

We have developed a theoretical architecture for extending an established cognitive tutor into a platform for collaborative and metacognitive experimentation, discussed some practical challenges with the implementation, and evaluated our efforts to expand the CTAI using a peer tutoring script. We will soon be evaluating the PTS in the classroom, and using the framework with other scenarios such as the CPS, which should give us a clearer idea of the effectiveness of the CTAI as a research platform. Other established tutoring systems can be extended in a similar manner, using a multi-component architecture that is specific about component responsibilities and takes usability needs and development schedule into account. Reconceptualizing the cognitive tutor as a research platform is a powerful idea for furthering educational research and improving intelligent tutoring technology.

Acknowledgements

This research is funded by the National Science Foundation, award #0354220. I thank Jonathan Steinhart, Dale Walters, Steve Ritter, and Dejana Diziol for their help.

References

1. Koedinger, K.R.; Anderson, J.R.; Hadley, W.H.; and Mark, M.A.: Intelligent Tutoring Goes to School in the Big City. *Journal of Artificial Intelligence in Education, 8*. 30–43. (1997).
2. Morgan, P., & Ritter, S. (2002). An experimental study of the effects of Cognitive Tutor® Alegbra I on student knowledge and attitude. (Available from Carnegie Learning, Inc., www.carnegielearning.com/research/research_reports/morgan_ritter_2002.pdf).

3. Johnson, D. W. and Johnson, R. T. (1990). Cooperative learning and achievement. In S. Sharan (Ed.), *Cooperative learning: Theory and research* (pp. 23-37). NY: Praeger.
4. Aleven, V., McLaren, B. M., Roll, I. and Koedinger, K. R. (2004). Toward Tutoring Help Seeking: Applying Cognitive Modeling to Meta-Cognitive Skills; *Proceedings of the 7th International Conference on Intelligent Tutoring Systems* (ITS-2004).
5. Beck, J.E., Mostow, J., & Bey, J. (2004). Can automated questions scaffold children's reading comprehension? *The 7th International Conference on Intelligent Tutoring Systems.*
6. Aleven, V., Roll, I., McLaren, B. M., Ryu, E. J., & Koedinger, K. R. (2005). An architecture to combine meta-cognitive and cognitive tutoring: Pilot testing the Help Tutor. *Proceedings of 12th International Conference on Artificial Intelligence in Education*.
7. Weinberger, A., Reiserer, M., Ertl, B., Fischer, F. & Mandl, H. (2003). Facilitating collaborative knowledge construction in computer-mediated learning with structuring tools. In R. Bromme, F. Hesse & H. Spada (Eds.), *Barriers and Biases in network-based knowledge communication in groups*. Dordrecht: Kluwer.
8. McLaren, B. M., Bollen, L., Walker, E., Harrer, A., and Sewall, J (2005). Cognitive Tutoring of Collaboration: Developmental and Empirical Steps Toward Realization. *Proceedings of the Conference on Computer Supported Collaborative Learning*,.
9. Krueger, Charles W.. Software reuse. *Computing Surveys*, 24(2):131–183, June 1992.
10. Roschelle, J., Kaput, J., Stroup, W., & Kahn, T. M. (1998). Scaleable integration of educational software: Exploring the promise of component architectures. *Journal of Interactive Media in Education* (6).
11. ADL (2004a). *Sharable Content Object Reference Model 2004 2nd Edition Overview*..
12. Brusilovsky, P. KnowledgeTree: A distributed architecture for adaptive e-learning. In *Proc. of WWW2004 - The Thirteen International World Wide Web Conference*, 2004.
13. Vassileva, J., McCalla, G., and Greer J. (2003). "Multi-Agent Multi-User Modeling", *User Modeling and User-Adapted Interaction*, *13* (1-2), 179-210.
14. Muhlenbrock, M., Tewissen, F. and Hoppe, H. U.: A framework system for intelligent support in open distributed learning environments, *International Journal of Artificial Intelligence in Education,* 9, 256-274.
15. Ritter, S. and Koedinger, K.R. An architecture for plug-in tutor agents. Journal of Artificial Intelligence in Education, 7, 3/4 (1996), 315-347.
16. Ritter, S. (1997). Communication, cooperation and competition among multiple tutor agents. In du Boulay, B. and Mizoguchi, R. (Eds). *Artificial Intelligence in Education: Knowledge and media in learning systems* (pp. 31-38). Amsterdam: IOS Press.
17. Walker, E. (2005). Mutual peer tutoring: A Collaborative Addition to the Cognitive Tutor Algebra I. Accepted as a Young Researcher's Track paper at the *International Conference on Artificial Intelligence and Education*.
18. McLaren, B. M., Walker, E., Koedinger, K., Rummel, N., Spada, H., and Kalchman, M. (2005). Improving Algebra Learning and Collaboration through Collaborative Extensions to the Algebra Cognitive Tutor., Poster Presented at CSCL-05, Taipei, Taiwan.

Forming Heterogeneous Groups for Intelligent Collaborative Learning Systems with Ant Colony Optimization*

Sabine Graf[1] and Rahel Bekele[2]

[1] Vienna University of Technology, Austria
Women's Postgraduate College for Internet Technologies
graf@wit.tuwien.ac.at
[2] Addis Ababa University, Ethiopia
Faculty of Informatics
rbekele@sisa.aau.edu.et

Abstract. Heterogeneity in learning groups is said to improve academic performance. But only few collaborative online systems consider the formation of heterogeneous groups. In this paper we propose a mathematical approach to form heterogeneous groups based on personality traits and the performance of students. We also present a tool that implements this mathematical approach, using an Ant Colony Optimization algorithm in order to maximize the heterogeneity of formed groups. Experiments show that the algorithm delivers stable solutions which are close to the optimum for different datasets of 100 students. An experiment with 512 students was also performed demonstrating the scalability of the algorithm.

1 Introduction

Cooperative learning is one of the many instructional techniques to enhance student performance described in the academic literature [4], [13], [19]. While the advantages of cooperative learning are very well documented [1], [11], [12], making it more efficient by creating heterogeneous groups has been given little attention. Researchers in the area of cooperative learning also claim that many of the unsuccessful outcomes of group work stem from the formation process (e.g., [14], [18]). Although group formation is said to play a critical role in terms of enhancing the success of cooperative learning ([12], [19]) and therefore increasing the learning progress of students, it is observed that there is only little research done that addresses the formation of groups in a heterogeneous way.

Moreover, the potentials of computer-based methods to assist in the group formation process have not been explored fully. Despite the popularity of computer-based tools to support collaborative learning [3], [8], [15], designers mainly

* This research has been funded partly by the Austrian Federal Ministry for Education, Science, and Culture, and the European Social Fund (ESF) under grant 31.963/46-VII/9/2002.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 217–226, 2006.

focus on collaborative interaction to address the techniques of sharing information and resources between students. Inaba [10] incorporated the grouping aspect and constructed a collaborative learning support system that detects appropriate situations for a learner to join in a learning group. Also Greer et al. [9] considered the formation of groups in tools that address the issue of peer-help.

While the above systems have proved to be appropriate in several contexts, they do not specifically reveal how the groups can be initially formed. It seems that considerations of the personality attributes are usually neglected in forming groups. The objective of this paper is, therefore, to address this limitation incorporating personality attributes as well as the performance level to form heterogeneous groups.

The research work has two main goals. The first one is the development of a mathematical model (Sect. 2) that addresses the group formation problem through the mapping of both performance and personality attributes into a student vector space. This serves as a foundation for the application of formal methods in the determination of heterogeneous groups. The second one is to provide a tool that implements the mathematical model. This tool can be used to supplement existing intelligent collaborative learning systems which do not consider the formation of heterogeneous groups so far. As a consequence, learners get more out of collaborative learning and their learning progress increases.

For maximizing the heterogeneity of the groups, the tool uses an Ant Colony Optimization algorithm described in Sect. 3. We describe in Sect. 4, how the algorithm is adopted in the group formation problem and experiments applying the algorithm with real-world data are presented in Sect. 5.

2 The Mathematical Approach of Group Formation

In this section, the conceptual framework for our mathematical model to form heterogenous groups of students is described.

2.1 The Student Space

For the definition of the student space, attributes whose values can be obtained from easily available indicators are selected based on expert opinion and discussion with colleagues. These attributes are *group work attitude, interest for the subject, achievement motivation, self-confidence, shyness, level of performance in the subject, and fluency in the language of instruction*. Each of these attributes has three possible values, where 1 indicates a low and 3 a high category value.

By applying the concepts of a vector space model, each student is represented in a multi-dimensional space by a vector whose components are made up of the values of personality and performance attributes. For instance, student S_1 may be represented by the vector $S_1(3, 1, 2, 1, 3, 3, 2)$, indicating that group work attitude is positive, interest of the subject is low and so on. For collecting the values of the attributes in order to apply the approach in real-world, a data collection instrument was designed and available at [2].

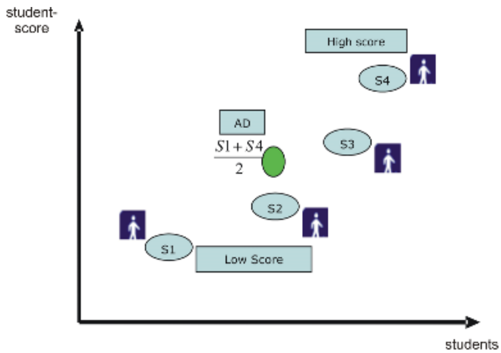


Fig. 1. Illustration of the measure of goodness of heterogeneity

The *student-score* for a particular student, used to measure heterogeneity, represents the total score of a student computed as the sum of all values of the student's attributes.

2.2 Heterogeneity of Students

In heterogeneous groups, it is important that students have different values of the attributes considered. This may be measured by the Euclidean distance (ED) between two students.

Let $ED(S_1, S_2)$ be defined as the distance between the vectors representing two students in space. Applying the Euclidean distance, this becomes

$$ED(S_1, S_2) = \sqrt{\sum_{i=1}^{n} (A_i(S_1) - A_i(S_2))^2}\ , \qquad (1)$$

where $A_i(S_j)$ represents the value for a particular attribute A_i for a student S_j and n represents the number of attributes.

2.3 Goodness of Heterogeneity in Groups

As shown in Fig. 1, a reasonably heterogeneous group refers to a group where student-scores reveal a combination of low, average and high student-scores. This is justified by the recommendation of Slavin [18] who proposes that students should work in small, mixed-ability groups of four members: one high achiever, two average achievers, and one low achiever. This idea is extended further and applied in student-scores.

The measure of goodness of heterogeneity (GH) is developed with the assumption that in a reasonably heterogeneous group, after taking the maximum and minimum student-score, the rest of the student-scores are expected to lie half way between the maximum and minimum score. In this case, the absolute difference of the average difference (AD) and the rest of the student-scores is minimal. Figure 1 illustrates the concept of the goodness of heterogeneity, assuming that each group has four members. In the following, we assume and also recommend a group size of four, as it is also suggested by Slavin [18]. Nevertheless, the group size can also be extended or reduced by increasing or decreasing the number of students with average score.

The measure of GH can be computed as follows. Let AD_i be the average of the maximum and the minimum student-score in the i-th group.

$$AD_i = \frac{\max scoreof(S_1, S_2, S_3, S_4) + \min scoreof(S_1, S_2, S_3, S_4)}{2} \ . \qquad (2)$$

The measure of goodness of heterogeneity is then defined as

$$GH_i = \frac{\max scoreof(S_1, S_2, S_3, S_4) - \min scoreof(S_1, S_2, S_3, S_4)}{1 + \sum_j |AD_i - scoreof(S_{j(i)})|} \ , \qquad (3)$$

where $S_{j(i)}$ is the student-score of the j-th student in group i, excluding the maximum and the minimum student-score.

Where a reasonable heterogeneity is experienced, the numerator in (3) should be greater than the denominator hence yielding a relatively high value of GH_i. It is trivial to show that $GH_i = 0$ when all students in a group have equal student-scores; $GH_i < 1$ when there is unreasonable heterogeneity in the group (meaning student-scores are at two extremes) and $GH_i > 1$ in reasonably heterogeneous groups. The greater GH_i, the better the heterogeneity.

2.4 Forming Heterogeneous Groups

An experiment by Bekele [2] shows that students who were grouped according to GH perform better than students grouped randomly or on a self-selection basis. But the GH deals on the basis of score values and does not distinguish between the individual characteristics. To address the limitation of GH, our approach additionally incorporates the Euclidean distance between the group members in the process of forming heterogeneous groups.

Considering the group building process as a whole, we have another aim regarding the goodness of heterogeneity. Aiming only at high GH values will result in some groups with very high GH and the remaining students will form groups with low GH. To form groups with a similar degree of heterogeneity, the deviation of GH values need to be considered additionally.

Thus, the objective of building heterogeneous groups can be formulated as follows:

$$F = w_{GH} \cdot GH + w_{CV} \cdot CV + w_{ED} \cdot ED \rightarrow max \ , \qquad (4)$$

where GH is the sum of the goodness of heterogeneity values, as defined in (3), of all groups. CV is the coefficient of variation based on all GH values and ED is the Euclidean distance of all groups, whereby the Euclidean distance of one group can be calculated by summing up the Euclidean distance between all combinations of group members according to (1). Each of these terms is weighted by the corresponding w. Aiming at a high heterogeneity, the fitness F should be maximized.

As can be seen, forming heterogeneous groups is not trivial. In a former experiment by Bekele [2], an iterative algorithm was developed to build heterogeneous groups based on GH. Euclidean distance is considered by the restriction that ED between at least two students has to exceed a certain threshold. By extending the objectives of [2] and including the Euclidean distance and the coefficient of variation of GH values in the optimization process, the problem becomes even more complex. For this reason and also because the problem is an NP-hard problem, we developed a tool based on an artificial intelligence approach, namely Ant Colony Optimization. In the next section, Ant Colony Optimization is introduced.

3 Ant Colony Optimization

Ant Colony Optimization (ACO) [5] is a multi-agent meta-heuristic for solving NP-hard combinatorial optimization problems, e.g. the travelling salesman problem. In the following, a brief introduction into ACO as well as a description of the applied algorithm is provided.

3.1 Background

ACO algorithms are inspired by the collective foraging behaviour of specific ant species. When these species of ants are searching for food sources they follow a trail-laying trail-following behaviour. Trail-laying means that each ant drops a chemical substance called pheromone on its chosen path. Trail-following means that each ant senses its environment for existing pheromone trails and their strength. This information builds the basis for their decision which path to follow. If there is a high amount of pheromones on a path, the probability that the ant will choose it is also high. If there is a low amount of pheromones, the probability is low. The more often a path is chosen, the more pheromones are laid on it which increases the probability that it will be chosen again. Since the decision is based on probabilities, an ant does not always follow the way that has the highest pheromone concentration. Paths which are marked as poor are also chosen, but with lower probability. Pheromones evaporate over time, leading to the effect that rarely used trails will vanish. These strategies enable natural ants to build a map of pheromone trails which indicates the best paths to a food source.

Several ACO algorithms exist that model and exploit this behaviour for solving graph-based NP-hard combinatorial optimization problems. One of the biggest advantages of ACO algorithms is that they can be applied very well to different optimization problems. The only requirement is that the problem can be represented as a graph, where the ants optimize according to the best path through this graph.

3.2 Ant Colony System

Ant Colony System (ACS) [6] is one of the most successfully applied ACO algorithms. In [7], Dorigo and Gambardella compared ACS with other optimization algorithms, e.g., neural networks and genetic algorithms for different instances of the travelling salesman problem. As a result, it is shown that ACS is competitive to the other algorithms and sometimes even finds better solutions.

The procedure of ACS is as follows. The first step is to represent the problem as graph where the optimum solution is a certain - e.g. the shortest - way through this graph. After initializing each edge of the problem graph with a small amount of pheromones and defining each ant's starting node, a small number of ants (e.g., 10) runs for a certain number of iterations. For every iteration, each ant determines a path through the graph from its starting node to the destination node. It does this by applying a so-called random proportional transition rule at each decision point. This rule decides which of all possible next nodes l included in the list J to choose, based on (1) the specific edge's amount of pheromones, also called global information τ, and (2) local information η representing the costs or utility of choosing the node. Equation (5) describes how to calculate the probability p that ant k goes from node i to node j.

$$p_{ij}^{k} = \frac{[\tau_{ij}] \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in J_i^k} \left([\tau_{il}] \cdot [\eta_{il}]^{\beta} \right)} \quad . \tag{5}$$

The transition rule itself consists of two strategies. In the *exploring strategy* the ants act similar to natural ants by deciding according to the probabilities p_{ij}^{k}. In the *exploiting strategy* the already gathered knowledge about the problem is used straight forward, choosing the node that fits best according to its local and global information. Which strategy is used is decided randomly for each transition whereby the parameter $q0$ determines the probability.

When the ant arrives at the destination node, the fitness of the newly found solution is calculated. In case the newly found solution outperforms the existing solutions, it is saved to memory as the currently best one. Additionally, to avoid that succeeding ants chose the same path, a local pheromone trail update rule is applied, decreasing the amount of pheromones on the found path slightly.

After all ants have found a solution, the ant which found the best one so far spreads pheromones according to the pheromone trail update rule. Furthermore, the amount of pheromones on each edge is reduced by the evaporation factor ρ.

ACS can be improved by additionally combining it with a local search method. This local search method can be embedded in different ways. The most usual way is to apply it to each found solution [6].

4 Forming Groups with Ants

In the following, we describe how we applied the ACS algorithm to the group forming problem and the necessary modifications to solve the problem with ACS.

4.1 Representing the Group Forming Problem as Graph

As already mentioned, the only requirement to use ACO algorithms is to represent the problem as graph. The representation form we used is based on the idea

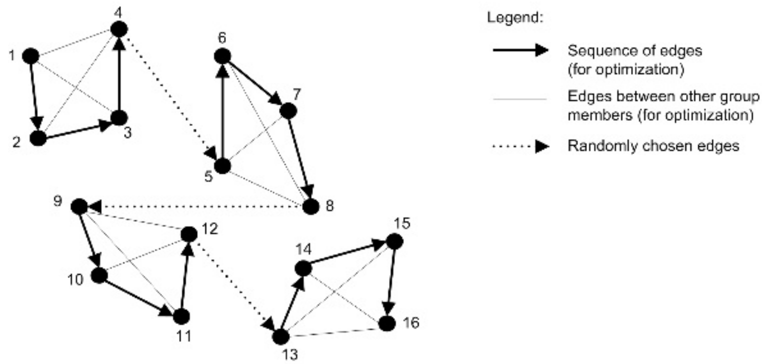


Fig. 2. Representation of the grouping problem as graph (group size = 4)

of ordering students comparable to the travelling salesman problem. The first m students belong to the first group, the second m student to the second group and so on, whereby m is the maximum number of students per group. Figure 2 shows this representation for a group size of four students, whereby the order is indicated by arrows. Having in mind that edges are used for pheromones and therefore indicate how good this edge is, within a group each newly assigned group member is linked not only to the last assigned group member but also to all other members of the group (indicated by solid lines in Fig. 2). This is because the important information for optimization is not the order in which the students are assigned to a group but the fact that exactly these m students belong together. Therefore, also the decision which student starts a new group is performed randomly (see dotted arrows in Fig. 2).

4.2 Applying ACS

For applying ACS to our grouping problem, we need to decide how to measure the local information of an edge. In our case, local information means the benefit to add a specific student to a group to which some group members already are assigned. As described in Sect. 2, heterogeneity of a group depends on the Euclidean distance between all group members and the GH of the group. Regarding ED, the benefit of adding a specific student is the sum of the ED of the new student and all already assigned group members. Because GH can be only calculated if the group is completed, the benefit for adding a student is based on the difference between the scores of the students in the best possible group and the scores of the students in the current group incorporating the specific positions of each student (one high score, one low score, and two average scores). Both local information values are normalized so that a high value indicates a good result and all values are between 0 and 1. For calculating the overall local information of an edge, both information values are weighted and summed up.

The global information is mainly calculated according to ACS. The only modification which has to be done for the grouping problem is that updating pheromones, in both pheromone trail update rules, is done for the edges between the newly assigned student and all other group members rather than only for the edge between the newly assigned student and the student which is assigned last. The amount of pheromones is for each of these edges equal.

The measurement of the quality of a solution is calculated according to the objective function described in (4). The objective of the algorithm is to maximize the heterogeneity of all groups based on the GH value of all groups, the coefficient of variation of these GH values, and the overall Euclidean distance. To improve the performance of ACS, a local search method called 2-opt [16] is applied to each solution an ant found.

5 Experiments and Results

This section demonstrates that the group formation based on ACS works effectively using real-world data. Based on 512 student data records we created five randomly chosen datasets of 100 records to demonstrate that the proposed algorithm works not only for a specific dataset but also for randomly chosen real-world data. Additionally, we show one experiment with all 512 records to show the scalability of our approach.

Each experiment consists of 20 runs. The parameter for ACS are assumed according to literature [6] or based on experiments. We assumed β=1, ρ=0.1, $q0$=0.9, and the number of ants=10. The weights are decided as follows: w_{GH}= 0.35, w_{CV}=0.15, and w_{ED}=0.5. Because the coefficient of variation impacts the GH values, we assumed w_{GH} and w_{CV} together as important as w_{ED}.

A run with 100 students stops after at least 100 iterations and only when the solution has not changed over the last $t * 2/3$ iterations where t is the number of already calculated iterations. In all experiments, the GH values and the CV values were stable, indicating that the best values were already found, and the values of ED varied only slightly per run. Looking at Tab. 1, this can also be seen by the small CV values of the fitness. These values show that the solutions of each run are similar and indicate that the algorithm finds solutions which are stable and close to the optimum for all datasets with 100 students.

Because of the NP-hard nature of the problem, some modifications for running the experiments with 512 students were necessary. The main issue of scalability is the local search method. Therefore, we modified it by applying 2-opt not for all students but only for 20 % of the students which were randomly selected

Table 1. Results of different datasets

Dataset	No. of students	Average GH	Average CV	Average ED	Average Fitness	SD Fitness	CV Fitness
A	100	129.813	39.223	363.936	52.141	0.033	0.064
B	100	117.200	35.182	377.415	51.558	0.029	0.057
C	100	114.234	41.906	374.147	49.422	0.033	0.067
D	100	132.176	31.344	354.588	52.584	0.027	0.050
E	100	131.958	31.437	372.214	54.870	0.046	0.084
F	512	537.595	45.552	1915.024	46.704	0.370	0.793

for each solution. This approach is also used successfully by Lo et al. [17]. Furthermore, the general goal changed from looking for a solution which is close to the optimum to finding a good solution. Therefore, the termination condition changed to stopping after 200 iterations. As can be seen in Tab. 1 the CV value of the fitness is higher than for the experiments with 100 students but it is still less than 1. This indicates that the found solutions are stable, good solutions but not that close at the optimum than for the experiments with 100 students.

Comparing the result of the experiment with 512 students with the result of the iterative algorithm in [2], aimed at finding heterogeneous groups according to the goodness of heterogeneity, it can be seen that the proposed algorithm delivers much better results. The iterative algorithm results in an average GH value of 1.6 per group while the proposed algorithm found an average GH value of 4.2. Regarding ED, the iterative algorithm considers only the maximum difference of two students in a group while the proposed algorithm includes the ED values of all combinations of group members. Nevertheless, the average ED values of the proposed algorithm are slightly higher which indicated a much better heterogeneity.

6 Conclusions and Future Work

In this paper we have presented a mathematical approach for forming heterogeneous groups of students based on their personality traits and performance. The approach is based on the different characteristics of the students, a general measure of the goodness of heterogeneity of the groups, and its coefficient of variation. The second aim of this paper was to present a tool that implements the proposed mathematical approach by using an Ant Colony Optimization algorithm. Experiments were performed, showing that the algorithm finds stable solutions close to the optimum for different datasets, each consisting of 100 students. An experiment with 512 students was performed demonstrating the scalability of the algorithm.

Because building heterogeneous groups improves the learning progress in collaborative learning, future work will deal with combining the tool with online learning systems, especially collaborative intelligent tutoring systems. We plan to develop a mediator agent that facilitates the group formation process, and to implement it in an already existing system. Another issue for future work is to provide the users with more options to adjust the algorithm, for example, to allow the user to determine a certain duration of running the algorithm or also a certain quality of solution.

References

1. Ames, C., and Ames, R. (eds.): Research on Motivation in Education. Academic Press Inc., Orlando, USA (1985)
2. Bekele, R.: Computer Assisted Learner Group Formation Based on Personality Traits. Ph.D Dissertation, University of Hamburg, Hamburg, Germany (2005). Retrieved 5 January, 2006, from http://www.sub.uni-hamburg.de/opus/volltexte/2006/2759

3. Collins, A., Brown, J.S.: The Computer As A Tool For Learning Through Reflection. In: Mandl, H., Lesgold, A. (eds.): Learning Issues For Intelligent Tutoring Systems, Springer Verlag, New York (1988) 1–18
4. Dansereau, D., Johnson, D.: Cooperative learning. In: Druckman, D., Bjork, R.A.: Learning, Remembering, Believing: Enhancing Human Performance, National Academy Press, Washington, D.C. (1994) 83–111
5. Dorigo, M., Di Caro, G.: The Ant Colony Optimization Meta-Heuristic. In: D. Corne, M. Dorigo, F. Glover (eds.), New Ideas in Optimization, McGraw-Hill, London, UK (1999) 11–32
6. Dorigo, M., Gambardella, L.M.: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1 (1997) 53–66
7. Dorigo,M and Gambardella, L.M.: Ant Colonies for the Traveling Salesman Problem, BioSystems, Vol. 43, No. 2 (1997) 73-81
8. Florea, A.M.: An Agent-Based Collaborative Learning System: Advanced Research In Computers and Communications in Education. In Proc. of the Int. Conf. on Computers in Education, IOS press (1999) 161–164
9. Greer, J., McCalla, G., Cooke, J., Collins, J., Kumar, V., Bishop, A., Vassileva, J.: The Intelligent Help Desk: Supporting Peer-Help in a University Course. In Proc. of the Int. Conf. on Intelligent Tutoring Systems, Springer-Verlag (1998) 494–503
10. Inaba, A., Supnithi, T., Ikeda, M., Mizoguchi, R., Toyoda, J.: How Can We Form Effective Collaborative Learning Groups? In Proc. of the Int. Conf. on Intelligent Tutoring Systems, Springer-Verlag (2000) 282–291
11. Jacobs, G.: Cooperative Goal Structure: A Way to Improve Group Activities. ELT Journal. Vol. 42, No. 2 (1988) 97–100.
12. Johnson, D.W., Johnson, R.T.: Cooperative Classrooms. In: Brubacher, M. (eds.): Perspectives on Small Group Learning: Theory And Practice, Rubican Publishing Ind., Ontario (1990)
13. Johnson, D.W., Johnson, R.T.: Leading the Cooperative School. Interaction Book, Edina, MN (1989)
14. Johnson, D.W., Johnson, R.T.: The Internal Dynamics of Cooperative Learning Groups. In: Slavin, R., et al. (eds.): Learning to Cooperate, Cooperating to Learn, Plenum, New York (1985) 103–124
15. Krejins, K., Kirschner, P.A., Jochems, W.: The Sociability of Computer-Supported Collaborative Learning Environments. Educational Technology and Society, Vol. 5, No. 1 (2002) 26–37
16. Lin, S.: Computer Solutions of the Traveling Salesman Problem, Bell Systems Journal, Vol. 44, (1965) 2245–2269
17. Lo, C.D., Srisa-an, W., Chang, J.M., Chern, J.C.: The Effect of 2-opt and Initial Population Generation on Solving the Traveling Salesman Problem unsing Genetic Algorithms. In Proc. of World Multiconference on Systemics, Cybernetics and Informatics (2001) 282–287
18. Slavin, R.E.: Developmental and Motivational Perspectives on Cooperative Learning: A Reconciliation. Child Development, Vol. 58, No. 5., Special Issue on Schools and Development (1987) 1161–1167
19. Slavin, R.E.: When Does Cooperative Learning Increase Achievement? Psychological Bulletin, Vol. 94 (1983) 429–445

Toward Legal Argument Instruction with Graph Grammars and Collaborative Filtering Techniques

Niels Pinkwart[1], Vincent Aleven[1], Kevin Ashley[2], and Collin Lynch[3]

[1] Carnegie Mellon University, HCI Institute, 5000 Forbes Avenue,
Pittsburgh PA 15213, USA
{nielsp, aleven}@cs.cmu.edu
[2] University of Pittsburgh, School of Law, 3900 Forbes Avenue,
Pittsburgh PA 15260, USA
ashley@pitt.edu
[3] University of Pittsburgh, Intelligent Systems Program,
Pittsburgh, PA 15260, USA
collinl@cs.pitt.edu

Abstract. This paper presents an approach for intelligent tutoring in the field of legal argumentation. In this approach, students study transcripts of US Supreme Court oral argument and create a graphical representation of argument flow as tests offered by attorneys being challenged by hypotheticals posed by Justices. The proposed system, which is based on the collaborative modeling framework Cool Modes, is capable of detecting three types of weaknesses in arguments; when it does, it presents the student with a self explanation prompt. This kind of feedback seems more appropriate than the "strong connective feedback" typically offered by model-tracing or constraint-based tutors. Structural and context weaknesses in arguments are handled by graph grammars, and the critical problem of detecting and dealing with content weaknesses in student contributions is addressed through a collaborative filtering approach, thereby avoiding the critical problem of natural language processing in legal argumentation. An early version of the system was pilot tested with two students.

1 Introduction

The field of law is an established and interesting application area for AI. (e.g. Aleven, 2003; Ashley 1990; Bench-Capon et al., 1998; Walton 2002). Argument is central to the practice of law, and therefore training in the skills of argument and advocacy are essential parts of legal education. Although there is a variety of law-related intelligent tutoring systems (e.g. Munjewerff and Breuker 2001), there are still only few intelligent tutoring systems specifically designed for assisting students in the construction of legal arguments. Exceptions include CATO (Aleven 2003) and ArguMed (Verheij 2003). CATO takes an example-based approach to teach students to make arguments based on past cases; ArguMed focuses more on structural aspects and provides assistants that support users in creating visual representations for defeasible arguments.

To some extent, the small number of tutoring systems for legal argumentation can be explained by that fact that the underlying domain is ill-structured. Legal argumentation is a kind of natural language discourse that focuses on interpreting the meaning

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 227 – 236, 2006.

of general legal concepts in light of specific facts. In contrast to well-structured domains like mathematics, for most tasks in legal argumentation there is no unambiguously defined "correct" solution which could be used as a basis for an ITS.

The ITS approach described in this paper aims at supporting students in studying examples of legal argumentation drawn from US Supreme Court transcripts of oral arguments. The goal is to help students understand the dialectic in which advocates propose and modify tests (i.e. decision rules) for a case and the Justices pose hypothetical fact situations to assess the merits of the proposed tests. The texts involved in this task are rather unstructured and involve a wide range of (legal and world) knowledge. Thus, they are not well accessible for an ITS without applying natural language processing (NLP) techniques, which would be very error-prone in the interpretive field of legal argumentation.

In the next sections, we first discuss the importance of tests and hypotheticals for understanding legal arguments, and propose a structured representation format together with a corresponding visualization. We then describe a novel approach for intelligent tutoring based on these argument structures, in which the system is capable of detecting several types of weaknesses (not restricted to purely structural ones) in the student's conceptions of legal arguments, while retaining a partially textual representation format but without involving NLP. The paper concludes with a description of studies planned with the ITS.

2 Diagrams to Visualize Court Argument as Hypothesis Testing

In US Supreme Court oral arguments, contending attorneys each formulate a hypothesis about how the problem should be decided with respect to a set of issues. They may propose a test and identify key points of the facts at hand on which the issue should turn. The Justices test those hypotheses by posing hypothetical scenarios. These scenarios are designed to challenge the hypotheses' consistency with past decisions and with the purposes and principles underlying the relevant legal rules. These oral arguments provide interesting material for legal educators. They are concentrated examples of many conceptual and reasoning tasks that occur in Socratic law school classrooms. As discussed in Rissland (1989) and Ashley (1990), the oral arguments illustrate important processes of concept formation and testing in the legal domain. As such, studying the transcripts of these arguments and the contained process of test and hypothetical proposition and modification is a valuable task for beginning law students. Yet, this task is quite difficult for them due to the complexity of the argument.

One idea to overcome these problems is to augment the textual documents with structured graphical representations that express the argument structure explicitly, thereby providing data usable by an underlying intelligent support system. In general, the use of graphical representations to support argumentation is not a new approach. Suthers and Hundhausen (2003) have shown that using graph structures can support group argumentation processes (e.g., argument graphs invite relating parts to each other), and Van Gelder (2002) shows that reasoning with graphical representations can indeed be effective in strengthening critical thinking skills (measured by pre/post gains compared to traditional teaching methods). In the legal domain, Carr (2003) has used Toulmin schemas (Toulmin 1958) for collaborative legal argumentation, and the

Araucaria system (Reed and Rowe 2004) employs visual premise/conclusion structures. ArguMed (Verheij 2003) provides intelligent feedback through an argumentation "assistant" that analyzes structural relations between contributions in diagrams. Out of the three, only Carr (2003) conducted an empirical evaluation, but does not report whether his system caused learning gains. In summary, though a lot of promising general approaches for graphically supporting argumentation exist, current literature does not show much evidence for the educational effectiveness in the domain of legal argumentation.

In contrast to the approaches and systems referred to before, we recommend a novel special-purpose argument representation geared toward a particular kind of argumentation process in which a normative rule (or "test") is proposed, tested, and "debugged," primarily by means of hypotheticals. The main ontological categories in our argument representation are, simply, tests and hypotheticals. The representation can be used to track how attorneys modify their proposed tests to handle the hypothetical cases presented by the justices. It does not have the wide applicability of a general representation (such as Toulmin schemas), but its more specific ontological categories may help students interpreting argumentation processes, e.g. by focusing their attention on the relevant information. Similar to the approach adapted in Araucaria (Reed and Rowe 2004), we allow the student to explicitly relate argument structures to the textual transcript of the oral argument using simple markup techniques. There is evidence that students indeed make use of such markup functions if the system makes it easy to do so (Farzan and Brusilovsky 2005).

3 Three Mechanisms for Intelligent Support

Figure 1 shows a screenshot of the prototype system we implemented using the Cool Modes framework (Pinkwart 2005). The left side of the figure contains the transcript of the oral argument in a case called Lynch vs. Donnelly, 465 U.S. 668 (1983). At the bottom left, there is a palette with the elements (tests, hypotheticals, current fact situation) and relations (test modification, distinction of hypothetical, hypothetical leading to test change, general relation) that the student can use to construct via drag and drop mechanisms a graphical representation of the argumentation in the transcript. The workspace on the right side of the figure contains the argument representation. Figure 1 shows the result of a third year student's system usage within an exploratory study. The diagram records a variety of hypothetical cases presented by the Justices (five in total) and also contains the attorney's responses to these hypotheticals, in which he distinguished them from the facts at hand or by formulating tests that should cover the hypothetical and the problem.

As argued, rules which are guaranteed to detect errors in the student's argument graphs are virtually impossible, as there are no "ideal solutions" in the ill-structured domain of legal argumentation. However, the student's conception of the argument may have *weaknesses* (in the sense of *potential* problems) that can be classified into several types. Using the authentic student solution of the exploratory study as an example, the next subsections describe these different types of weaknesses, their detection within argument graphs, and how the intelligent tutoring system we have implemented responds to these detected "weak spots".

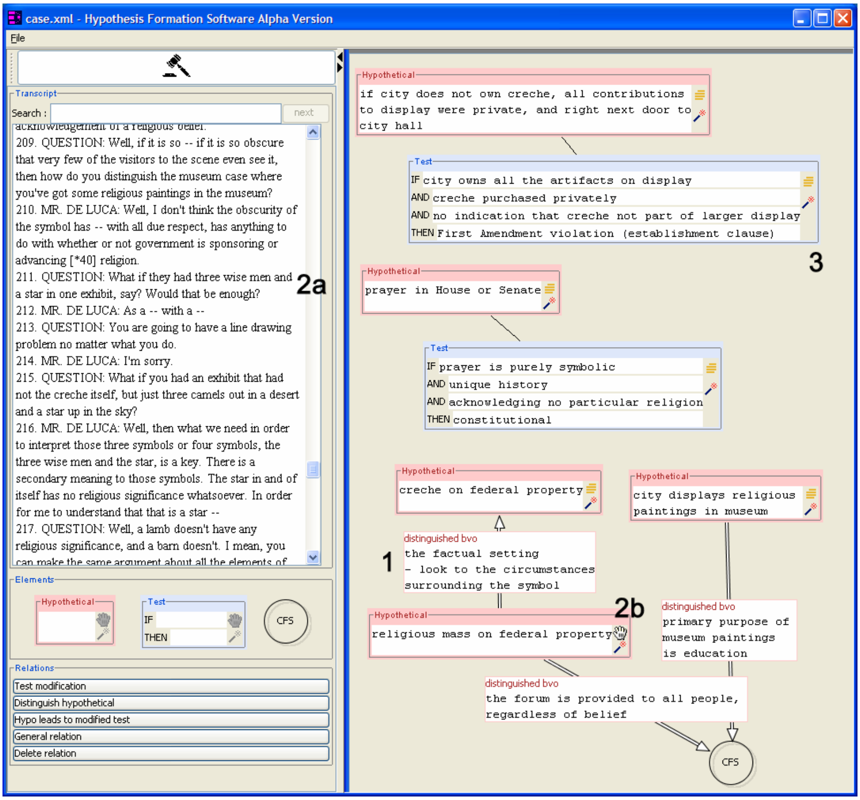


Fig. 1. Example of a graphical argument model

3.1 Structural Weaknesses

First, the argument representation created by a student can have *structural* weaknesses. Examples of weaknesses of this type are isolated elements in the argument graph, empty text fields, or the absence of any test element in the workspace. Figure 1 illustrates two more advanced structural weaknesses: only two out of the five hypotheticals are explicitly related to a test, the other three are not. Since attorney's often respond to hypotheticals by posing a new test, or a modification of an existing test, it may be that the student has overlooked (or misunderstood) a formulation of a test that appears in the transcript. In addition, the location marked (1) in the figure shows that the student distinguished two hypotheticals from each other. Attorneys and judges might well do this in an argument, but typically a hypothetical should also be related to the current facts in some way, e.g. through distinction. Since the student did not add links to the diagram to represent these relations, this part of the graph is a good candidate for system feedback to the student. Section 3.4 discusses how our ITS comments on these weaknesses, taking into account the remaining uncertainty that is based on weaknesses being just indications of *possible* errors.

Since structural weaknesses are related to the abstract structure of the argument only, they can be detected by logical formalisms that ignore both the content of the diagram text boxes and the markups of the transcript. Approaches for intelligent tutoring based on graphical argument structures are not new – early work in this field has been done by Paolucci, Suthers and Weiner (1996), who made use of syntax rules and an expert knowledge base to check argument graphs. Our proposed approach avoids "expert solutions" and model tracing, and relies on a graph grammar (Rekers and Schürr 1998) to analyze argument structures. The grammar consists of a set of terminal symbols T, a set of nonterminals N, a start symbol S, and a set of production rules. Both terminal and nonterminal symbols can have attributes, and a production rule is a tuple (L,R) of graphs over $T \cup N$, which can be applied to a graph G that contains a subgraph M_L which matches L. The result of the rule application is a graph $G' = G \cup R \setminus M_L$. Thus, a rule application replaces the subgraph that matches the left side of the rule with the graph in the right side.

We use the grammar to check the diagrams created by students for properties that represent structural argument weaknesses. Compared to other formalisms for "attribute value checking" which underlie many constraint-based tutors, the grammar based approach we propose is much better adapted to the graph structures we employ. A further advantage of the formalism is its declarative character: rules can easily be specified (cf. examples below) and applied in the system as parameters of the generic parsing algorithm. This avoids the need of programming a complex graph algorithm for each single property of the diagram that one wants to check.

The grammar contains two types of rules: first, "construction oriented" rules model the process of building argument graphs. The following rule illustrates this and covers the situation that a "test" element is added to an empty workspace (in this case, the test gets assigned the value "unchanged" for the "version" attribute):

```
L = < {S},∅ >
R = < {TEST},∅ >
TEST.version = "unchanged"
```

In addition, "feedback oriented" rules directly express a specific weakness and thus enable the system to produce well-defined feedback. The following is an example of such a "feedback oriented" production rule, which can detect the structural weakness of "hypothetical distinction without relation to the facts" that was discussed in relation to Figure 1:

```
L = < {HYPO1,HYPO2,W},{Distinguish_from(HYPO1,HYPO2)} >
R = < {HYPO1,HYPO2},{Distinguish_from(HYPO1,HYPO2)} >
HYPO2.connection = "false"
W.type = "isolated_hypo_distinguished_from_hypo"
W.locations = {HYPO1,HYPO2}
```

The right side of the rule matches the student solution of Figure 1 by identifying HYPO1 with the "religious mass on federal property" hypothetical and HYPO2 with the "crèche on federal property" element (note that the "connection" attribute of HYPO2 is used to express its lack of relation to the facts). The nonterminal node W in the left side of the rule represents the detected weakness.

3.2 Context Weaknesses

The second weakness type can be characterized as *context* weaknesses. These essentially deal with the relation between the argument graph and the transcript. Even if an argument diagram has no structural weaknesses, the relation between the elements in the diagram to the source material (i.e., the transcript) as expressed through the markups can reveal problems. Examples of context weaknesses are a lack of evidence for the tests/hypotheticals in the diagram (missing nodes or links), important passages of the transcript that are referenced in the diagram but with (apparently) the wrong element type (e.g., a test being marked up as a hypothetical), or seemingly irrelevant parts of the transcript being marked up. Figure 1 illustrates two of these weaknesses. In (2a), the transcript lines "What if they had three wise men and a star in one exhibit, say? Would that be enough?" contain a hypothetical posed by a judge, but the student's solution does not refer to it in any way. Also, the hypothetical (2b) is not linked to the transcript (visible through the hand symbol in the right corner of the element).

The same graph grammar formalism that is used to detect structural weaknesses (described above) is also applicable for detecting context weaknesses, which obviously is an advantage on the technical level. We make use of node and edge attributes in the grammar rules to represent constraints on the links that are created between the transcript and the elements in the graph. A context rule for the most important context weakness (missing link to important part of transcript), can be specified as follows:

```
L = < {S},Ø >
R = < {HYPO},Ø >
HYPO.link = 211
```

This rule is comparable to the start rule in the "structural weaknesses" part, differing only in that it requires specific elements to be present in the argument graph. This rule requires a student to mark up line 211 of the transcript and link it to a hypothetical. Similar rules can be formulated to explicitly declare "irrelevant" parts of the transcript that should not be marked up.

Taken together, structural and context rules allow a teacher to specify in detail a particular test/hypothetical structure linked to well-defined parts of the transcript. However, we are not advocating an approach in which the student's graph is compared against an expert solution. Due to the ill-structuredness of the legal domain, it is not possible to define a small set of "correct" solutions. Instead, we use the graph grammar formalism to partially specify solutions (e.g., only the two most important test versions and six central hypotheticals are required to be marked up by students).

3.3 Content Weaknesses

Finally, the *content* of the textual elements created by the student can be inappropriate, even if the overall argument structure is good and related to the transcript in a reasonable way. Students may well have difficulties in understanding, e.g., the essence of a proposed test, as evidenced by a poor paraphrase in the corresponding test node they add to the graph. Obviously, this type of weakness is hardest to check, since it involves interpretation of legal argument in textual form. In addition, due to the ill-structured domain, student answers will not be simply either right or wrong, but instead have a certain quality in terms of a number of criteria. For instance, location (3) in Figure 1 contains the student's description of the test proposed in the argument. For a general reader (and also for an ITS), it is hard to tell if this is an adequate summary of the test or not. The graph structure and peers working on the same task can help.

Our two-step approach to address the problem is NLP-free and involves a technique known as "collaborative filtering" (Konstan and Riedl 2002) – we make the assumption that a larger group of students works on the same task, either individually or in small groups. In our variant of the collaborative filtering method, students are asked to rate samples of other's work relative to their own work. The system then combines the ratings into an overall score and thus "filters" for quality. More specifically, for selected parts of the transcript (i.e., parts where a test is mentioned), after a student has created a corresponding element in the graph, he is first presented with a small number of alternative answers (given by peers) and asked to select all those he considers *similar* to his own answer. Then, the student gets a second selection of answers (some known to be good, some known to be of poor quality, some given by peers) and is asked to select all those he considers *at least as good as* his own. The system then uses a combination of similarity and recommendation ratings to compute a heuristics of the quality of student answers.

A base rating b_x of an answer given by student x can be calculated based on the recommendations given by x. If the student had n answers to choose from, and the ones he selected as being at least as good as his own had a quality measure $q_1, \ldots, q_k$ (0 for very bad, 1 for very good, see below for the calculation of quality measures for peer answers), while those he did not recommend had quality measures $q_{k+1}, \ldots, q_n$, then b_x can be calculated as

$$b_x = \frac{1}{n}\left(\sum_{i=1}^{k} q_i + \sum_{i=k+1}^{n} (1 - q_i)\right)$$

Figure 2 shows a window presented to the student for the base rating calculation. In the figure, three answers can be selected as "at least as good as his own" by the student. If the three available answers have quality ratings of 1, 0.8, and 0.3 (i.e., two good ones and two bad ones), and the first two have been selected like the figure shows, then the base rating for the student's answer is $b_x = 0.33*(1+0.8+0.7) = 0.825$. Considering the good quality of the test description provided by the student, this base rating is acceptable as an initial value.

The base rating b_x measures in how far a student can recognize good answers and thus serves as a heuristic of his own answer's quality, but does not rate the answer the student has actually typed in himself. In our approach, the base rating is therefore supplemented by two other ratings which measure the quality of what the student has actually typed in. First, the *similarity estimations* given by the students are used. If students with good own ratings rate another solution as similar, this raises the rating of the peer solution (the peer rating will also be reduced based on similarity with poor solutions). Second, the *recommendations* that a student's answer receives by his peers are used, with recommendations by good students having a higher impact. The overall quality q_x of a student answer is then calculated as the weighted average of b_x and the other two measures. The weights of the peer-dependent ratings are based on the number of selection options that peers had. This takes into account the importance of peer's opinions while at the same time eliminating the cold start problem (how to handle the first users of the system, before peer ratings are available?) through the inclusion of relations to known correct/incorrect solutions, which feed into b_x.

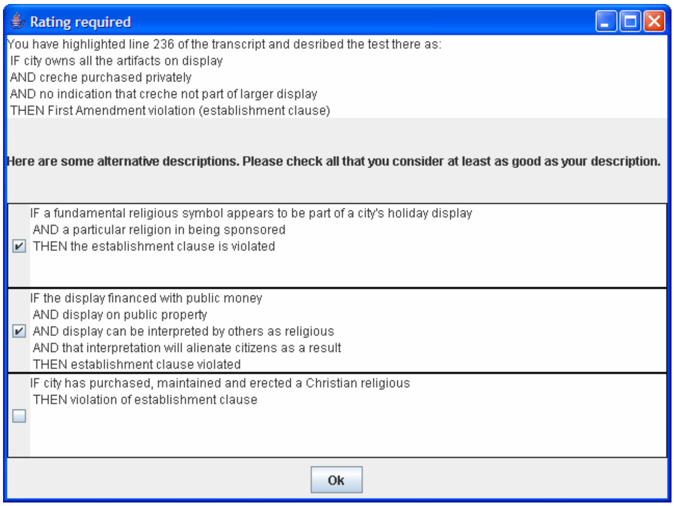


Fig. 2. Similarity rating dialog

3.4 Tutor Feedback

The previous sections described how different kinds of weaknesses can be detected in the student's argument graphs. Having detected a weakness, the question is how the system should react to it. Our notion of weakness includes the possibility of "false alarms": a student's solution can be of high quality and still cause a tutor intervention. This seems inevitable in an ill-structured domain, where correctness is hard to define even for human domain experts, for example: Does the fact that the student distinguishes two hypotheticals (see location 1 in Figure 1) without relating the hypotheticals to the current fact situation indicate that the student did not understand the role of hypotheticals in the argument, or was this just a wrong use of graphical elements?

Since these questions cannot be answered in a general way by an ITS, it does not make sense for an ITS to use most of the detected weaknesses as a basis for telling users directly that they were wrong in their answer. However, following the idea of weaknesses as the presumably weak parts of student's work, it makes sense to use them as tailored and personalized self-explanation prompts by inviting the student to re-think and explain these parts of his work. Self-explanation has been shown to be effective in many domains, including ill-structured ones (Schworm and Renkl 2002). Table 1 shows some of the weaknesses that were identified within this article together with related short versions of self-explanation prompts.

Table 1. Examples of self-explanation prompts

Weakness Description	Type	Self-explanation prompt (short version)
Isolated hypothetical distinguished from hypothetical	structural	In your solution, the hypotheticals H1 and H2 are distinguished from each other. Yet, hypothetical H2 is not related to any test or the current fact situation. Please explain why you did so, either in free text or by modifying the diagram.
Important part of transcript not marked	context	Please look at this part of the transcript (scroll to line L) and explain its role within the argument.
Low quality rating of contribution	content	Please reexamine what you marked here (scroll to line L) and explain it again.

4 Conclusions and Outlook

The approach as presented in this paper is designed to support first-year law students in learning legal argumentation skills by having them create graphical models of argument transcripts, and presenting them feedback on the weaknesses in their models. The ITS used to generate this feedback is based on two formalisms, which enable a heuristic check of student answers for different types of weaknesses: a graph grammar formalism and a collaborative filtering technique. It does not make use of NLP, which can be considered an advantage in the highly interpretive and ill-structured domain of legal argumentation, but nevertheless is able to give content-related feedback. Furthermore, the approach requires only very minimal system-side knowledge about specific legal cases, which facilitates using the system with a new transcript.

The pilot studies we conducted essentially confirmed the suitability of the ontological categories and the graphical representation format. Based on these, further research will try to find empirical evidence for the effectiveness of the presented tutoring approach, both compared to control groups that make use of the diagram tool without feedback, and also to groups that work traditionally. In particular, we are interested in "fine tuning" the selection of feedback prompts and the collaborative filtering mechanism in terms of which peer answers are best to present to a student.

Acknowledgements

This research is sponsored by NSF Award IIS-0412830. The contents of the paper are solely the responsibility of the authors and do not necessarily represent the official views of the NSF.

References

1. Aleven, V. 2003. Using Background Knowledge in Case-Based Legal Reasoning: A Computational Model and an Intelligent Learning Environment. *Artificial Intelligence* 150: 183-238.
2. Ashley, K. 1990. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals*. Cambridge MA, MIT Press/Bradford Books.

3. Bench-Capon, T., Leng, P., and Staniford, G. 1998. A computer supported environment for the teaching of legal argument. *J. of Information, Law & Technology* 3
4. Carr, C. 2003. Using Computer Supported Argument Visualization to Teach Legal Argumentation. In *Visualizing Argumentation*, 75-96. London, Springer.
5. Farzan, R., and Brusilovski, P. 2005. Social Navigation Support through Annotation-Based Group Modeling. In Proc. of UM, 387-391. Berlin, Springer.
6. van Gelder, T. 2002. Argument Mapping with Reason!Able. *The American Philosophical Association Newsletter on Philosophy and Computers* 85-90.
7. Konstan, J., and Riedl, J. 2002. Collaborative Filtering: Supporting social navigation in large, crowded infospaces. In *Designing Information Spaces: The Social Navigation Approach*, 43-81. Berlin: Springer.
8. Muntjewerff, J., and Breuker, J. 2001. Evaluating PROSA, a system to train solving legal cases. In Proc. of AIED, 278–285. Amsterdam, IOS Press.
9. Paolocci, M., Suthers, D., and Weiner, A. 1996. Automated Advice-Giving Strategies for Scientific Inquiry. In Proc. of ITS, 372 - 381. Berlin, Springer.
10. Pinkwart, N. 2005. *Collaborative Modeling in Graph Based Environments*. Berlin, dissertation.de Verlag.
11. Reed, C., and Rowe, G. 2004 Araucaria: Software for Argument Analysis, Diagramming and Representation. *International Journal of AI Tools* 14:961-980.
12. Rekers, J., and Schürr, A. 1997. Defining and parsing visual languages with layered graph grammars. *Journal of Visual Languages and Computing* 8:27-55.
13. Rissland, E. 1989. Dimension-Based Analysis of Hypotheticals from Supreme Court Oral Argument. In Proc. of AI & Law, 111-120. New York, ACM Press.
14. Schworm, S., and Renkl, A. 2002. Learning by solved example problems: Instructional explanations reduce selfexplanation activity. In Proceedings of the 24th Annual Conference of the Cognitive Science Society, 816-821. Lawrence Erlbaum.
15. Suthers, D., and Hundhausen, C. 2003. An Experimental Study of the Effects of Representational Guidance on Collaborative Learning Processes. *Journal of the Learning Sciences* 12:183-218.
16. Toulmin, S. 1958. *The Uses of Argument*. Cambridge University Press.
17. Verheij, B. 2003. Artificial argument assistants for defeasible argumentation. *Artificial Intelligence* 150:291-324.
18. Walton, D. 2002. *Legal Argumentation and Evidence*. Penn State Press.

SPRITS: Secure Pedagogical Resources in Intelligent Tutoring Systems

Esma Aïmeur, Flavien Serge Mani Onana, and Anita Saleman

Université de Montréal, Département IRO
CP 6128, Succ. Centre-Ville, Montréal (Québec), H3C 3J7 Canada
{aimeur, manionaf, salemana}@iro.umontreal.ca

Abstract. During the training phase in an Intelligent Tutoring System, learners usually require help. However, it may happen that the tutor cannot provide such help, unless it has access to additional pedagogical resources. Moreover, in a collaborative but competitive learning environment in which each user could be both learner and expert, security problems may arise. For instance, the exchanges between users could require security services such as anonymity, confidentiality and integrity. In this paper, we introduce a system, called *SPRITS*, whose aim is to provide the tutor with mechanisms to capture, exploit, organize, deliver and evaluate learners knowledge, in a secure way, based on the learner-expert concept. Our main contribution is the introduction of security services in an ITS for the benefit of learners. This may be helpful to protect learners' privacy as well as communication contents and pedagogical resources in an artificial competitive peer environment, thus allowing the tutor to better evaluate learners.

1 Introduction

Much work has been done to improve learner knowledge acquisition, especially in Intelligent Tutoring Systems (ITS), which provide courses adapted to the learner's profile. Usually, knowledge is not known or not accessible although it may exist inside the organization in which the learner is. In this case, using traditional ITSs alone may not be sufficient since the curriculum may not contain all the knowledge required or there may be a lack of pedagogical resources. Therefore, one interesting way to improve a learner's knowledge is to help him meet the expertise he needs to resolve his current problems. This can be achieved, for instance, through, a new component capable of managing knowledge, as well as those to whom this knowledge belongs.

Managing knowledge is related to handling and organizing information about this knowledge, and making it available to all the members from a community such as a group of learners or employees. However, such management may incur some difficulties:

– Localization of expertise: the tutor does not always know which pedagogical resource or which expert is able to help, nor where to find the information. Therefore, a learner may waste much time in looking for expertise to complete his training and evaluation.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 237–247, 2006.

– Motivation of experts: experts are not often motivated because they have to answer the same questions in a repetitive way, or they receive questions that are not relevant to their field of expertise [5].
– Lack of security: some malicious learners may try to access and/or monitor the learning process of their peers (questions asked, evaluation of answers, feedback about helpers, etc.). This may happen in any competitive learning environment, such as artificial peers [7, 11, 2], in which some learners may try to modify other learners' evaluations made by the tutor. In such an environment, due to shyness or privacy concerns, learners may need to send questions and/or answers anonymously. There may also be a need for keeping confidential the data exchanged between the tutor and the learners, or ensuring the integrity of these data to avoid their falsification by malicious learners, since everyone is involved in a competition. The security management in SPRITS is presented in Section 3.4.

Numerous systems [5, 3, etc.] have been developed to solve some of these problems. In particular, most of these systems consider only the learner requests and not his pedagogical competency, meaning that the concept of learner-expert is not taken into account as it is in our case. A major exception is *I-Help*, which has been developed [11, 20] to support peer-help. There are similarities between I-Help and our system, SPRITS (Section 3). However, the introduction of the security aspects and the use of a recommender system in the context of SPRITS largely differs from I-Help, as discussed in Section 4.

Our approach consists in providing an environment of collaborative assistance, in which each learner can access the expertise of other learners, while being able to provide them with his own expertise. This approach consists in two main parts. In the first part, we want to avoid the expert having always to answer the same questions. For this purpose, we use Case-Based Reasoning (CBR) to integrate a module of questions and answers, which allows finding and re-using the answers related to questions similar to the current one. If no answer is found or if the learner is not satisfied with any of the retrieved answers, the second part is used. Here, the system searches in the user profile database someone with the potential expertise to be recommended to the learner. SPRITS makes it possible to find users who can provide expertise in a particular field. In SPRITS, each user can be seen at the same time as a learner and an expert. This is interesting since, on the one hand, it has been demonstrated that having learners teach each others may increase their performance at various academic levels [9] and, on the other hand, there is a need to help the helpers [13]. By recommending experts to learners seeking assistance or information retrieval in a precise field, SPRITS enables not only the sharing and the re-use of knowledge, but also the connection of individuals working within the same organization.

2 Preliminaries

In this section, we review the concepts of Recommender Systems and Case-Based Reasoning. We also review the general notion of Public Key Cryptography. All these notions are used in SPRITS.

Recommender Systems. *Recommender systems* [4, 1] are well known in the context of information retrieval. They are used in *e-commerce*, where they allow entities providing products (goods and services) to guide the choices made by potential *customers*. The recommender system principle can easily be adapted to hold in the context of e-learning, considering pedagogical resources (exercises, answers, experts, etc.), learners and the tutor as being, respectively, products, customers and merchants. Collaborative Filtering (CF) is the best-known of the recommendation techniques. It was first introduced by the developers of Tapestry [10], a filter-based electronic mail system designed for the intranet at the Xerox Parc Palo Alto Research Center. CF techniques are becoming increasingly important in any domain in which there is an unprecedented growth in the number of users, such as education, e-commerce, etc. The main characteristic of CF is that recommendations for a given user are based on the behaviour and the evaluations of the other users on objects of the system [18]. Usually, the behaviour and the evaluations are represented in the form of ratings given on the objects proposed by the system [12]. In this paper, we use CF to recommend experts to learners who need help in an ITS.

Case-Based Reasoning. *Case-Based Reasoning* (CBR) is used to solve problems by matching the problem description to a previously solved case, using the past solution to solve the new problem [14]. CBR is defined as a four-step cycle that consists in retrieving items similar to the new one in the case base (retrieval phase), adapting the solutions of retrieved cases to the new problem (reuse phase), testing the adapted solutions and if necessary repair them (revision phase) and, finally, adding the new learned case in the case base (maintenance phase) [14]. In SPRITS, we use a textual Case-Based Reasoning technique [15] to collect expertise in the form of Question & Answer (Q&A) pairs and reuse them by retrieving the questions most similar to the current user's request.

Cryptographic Tools. The notion of *Public Key Cryptosystems* (PKCs) was introduced by Diffie and Hellman [8] to allow two people to exchange confidential information over insecure channels even if they don't share a secret key ahead of time. (Unbeknownst to them, the related notion of Public-Key *Distribution* was invented earlier by Merkle, but published later [16].) Formally, a PKC consists of three efficient algorithms: a *Key-Generation Algorithm*, which generates pairs of Private Key and Public Key; an *Encryption Algorithm*, which computes the ciphertext for a message, given the public key; and a *Decryption Algorithm*, which computes the cleartext message back from the ciphertext, given the secret key. The public key can be made available to all so that anyone can encipher messages, but only the legitimate receiver, who keeps secret the corresponding private key, can decipher them. Diffie and Hellman [8] also introduced the related notion of *digital signatures*, by which the party receiving a message can ascertain the identity of the sending party as well as the integrity of the message. This process works with two keys as well: A secret *signing key* is used on the message to generate its signature, but anyone can use the corresponding *signature* *verification key* to make sure that the message is legitimate and that it has not been modified in transit.

3 The SPRITS System

3.1 Architecture and Components

In addition to the standard components (curriculum, tutor, pedagogical strategies and learner model), Figure 1 illustrates the *Recommender System* module. This module consists of two recommenders: the *Q&A Recommender* (`Q&A Rec`), which allows the retrieval of answers to requests similar to the one made by the learner, and the *Expert Recommender* (`Expert Rec`), which recommends experts according to the learner's request and/or search criteria. The techniques used are Collaborative Filtering (CF) and Case-Based Reasoning (CBR). These techniques require taking into account experts that have been selected and evaluated positively (ratings usually greater than 3 in a one-to-five scale) and to retrieve similar experts in the expert case base.

Moreover, each user in SPRITS is considered to be a learner and an expert. Therefore, the learner model is also called the *learner-expert model*. Data contained in this model are stored in the *learner-expert profile database* (Figure 1) and include the learner's demographic data, his level knowledge about the subject matter being taught, his evaluations and scores, as well as his *learning base* and *expertise base*. The learning base contains pairs of questions/answers consisting of questions that the learner asked and the associated answers. As for the expertise base, it consists of pairs of questions/answers containing questions addressed to the expert and the answers he provided. Each learner is identified by a secret pseudonym created by the tutor. This pseudonym is stored in the learner-expert profile database and is not communicated to the learner for security purposes (Section 3.4).

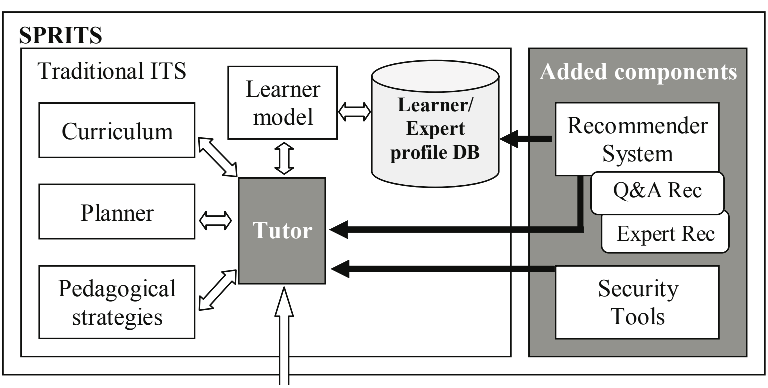


Fig. 1. Architecture of SPRITS

3.2 Pedagogy in SPRITS

In this section, we explain the pedagogical method introduced in SPRITS. Let us imagine a company, with hundreds of employees that have access to an ITS for their lifelong training. Suppose that the big boss of the company relies on an ITS to evaluate his employees. After the training phase, a summative evaluation takes place. One of the objectives of this exercise is to enable learners (employees) to improve the ITS thanks to their expertise, in order to make it available to other novice learners. Furthermore, learners are evaluated depending on the quality and the quantity of expertise they provide to the ITS. They ask questions and receive answers from experts (who are other learners) by way of the tutor. During his training, when a learner receives an answer from an expert, he has two possibilities: either he accepts the answer because it solves his problem well, or he can reject it if he judges that the answer is not appropriate for him. In a learning context, we think that a learner who wants a specific-oriented answer to one of his questions needs to quickly understand the answer, and for this reason it is necessary for him to have an answer corresponding to his way of learning. One learner will better understand if he gets examples, another one will need to know the subject perfectly and thus will prefer a list of references to allow him to get a complete overview of the subject whereas a third one will simply need a short and clear answer, etc. Therefore, it is important to take into account not only the technical skills of the expert, but also his pedagogical competency. For us, these two types of capabilities (technical and teaching) are the ones that influence the choice of an expert. Now the question is: how can the learner determine the pedagogical competency of an expert? The approach we have chosen is based on ratings: each learner who receives an answer can evaluate the expert according to some criteria (see below). The pedagogical competency of an expert is therefore determined by the community of learners through the ratings and comments they provide about experts. The ratings and comments determine the characteristics of the expert from the learner's point of view, as well as the perception of his pedagogy. They are used to improve future recommendations.

SPRITS also enables the user to find experts according to search criteria, such as "looking for an expert whose expertise area is Java security". In particular, the search criteria can be based on statistics such as "experts having provided answers to more than 60% of questions addressed to him". More generally, statistics are computed not only from the interactions between the users themselves (number of questions asked, number of answers sent or received, etc.), but also from the ratings given by learners to experts. The ratings are based on how much the expert *answers quickly*, *is easy to understand*, *answers coherently*, *answers clearly*, *clarifies the significant points*, *gives some examples*, *shows some interest* and *refers to other experts*. These criteria are evaluated according to a 1-to-5 scale. Moreover, there is a general evaluation in which the learner uses the same 1-to-5 scale to answer the question: *Would you recommend this expert?*

3.3 The Recommendation Process

The recommendation process consists of two recommenders, the Q&A Recommender and the Expert Recommender. They operate in a *cascade* [4]: the former followed by the later.

Q&A Recommender. The learner's request is sent to the query processing module (part of the Q&A Rec) for the extraction of keywords. The retrieval and maintenance components of the Case Base manager module enable, respectively, the retrieval of cases similar to the learner's request and the addition of new cases in the case base when an expert answers a request. The method used is textual case-based reasoning. The vector of weighted terms is derived from the processing carried out as follows: the description of the question is first filtered and the words known to be usual ("the", "like", etc.)_ are removed since they are not useful as indexes. Then, the stemming consists in transforming the words to keep only their root. We use Porter's algorithm [17] for this purpose. The remaining words can be regarded as indexes; they receive a weighting measuring their importance [19]. The similarity between each case from the case base and the target case (the learner's request analysed in the query processing module) is calculated using the cosine method by the retrieval phase of Case Base Manager. This phase is crucial for the Q&A Recommender because it is the one that will allow recommending answers to cases similar to the current request and therefore to give solutions that will help solve the learner's problem. Whereas the case where the retrieval phase gives no result or the similar cases found are rejected by the learner, the request is sent to the Expert Recommender module.

Expert Recommender. The Expert Recommender is used when no similar case is found, or when none among those found has proved satisfactory for the learner. In that case, the system searches for appropriate experts to answer the question. The appropriate experts are users who not only have the technical competency to answer the learner's request, but are also capable of making their answers understandable (pedagogical aspect). We use a hybrid technique [4] for recommendation, based on collaborative filtering and case-based reasoning. Collaborative filtering is based on the ratings given by the users of the system: the target user (for whom we want to make recommendations) is compared to all the users or a set of users of the system using the ratings each one has given. In other terms, we look for correlations between users relatively to the evaluations made by users (Pearson correlation). For this purpose, we only consider results to the rating for the question: "Would you recommend this expert?". Moreover, the system makes it possible for the user to find experts according to several selection criteria (Section 3.2). From these criteria, we apply collaborative filtering to the corresponding ratings. In the case of a recommendation based on case-based reasoning, knowing that a user appreciated one or more experts in the past, the system looks for experts similar to them according to his criteria and recommends them to him. What are these criteria? They correspond to all the data that characterize a user as an expert: on the one hand the votes he got, on the other hand all the other data, such as the number of questions received and answered, the number of questions transferred to other experts, etc.

3.4 Security in SPRITS

Coming back to the company introduced in Section 3.2, learners are in a competitive environment, yet they have to conduct collaborative learning. Nowadays, there exist several techniques aiming at intercepting communication contents between two entities [6]. Therefore, a malicious learner may decide to tamper with all or part of the help requested by other learners or answers provided by experts. He may also refuse to answer questions asked by some targeted learners, with the purpose of limiting their knowledge. Furthermore, he may falsify the evaluation of targeted learners, so that their summative evaluations would decrease. Finally, no learner should have access to the contents of messages[1] addressed to others, as well as those meant for the tutor.

To overcome these problems, security services (anonymity, confidentiality and integrity) are offered by SPRITS. First of all, the tutor generates a pair of public/private cryptographic keys for himself as well as a pair of signing and signature-verification keys. He makes his public encipherment and signature-verification keys available to all in a way that cannot be spoofed. Then, the tutor generates similar sets of four keys for each learner—this can be done dynamically if new learners enter the system—and distributes them to corresponding learners, after encipherment with its private key and authentication with its signing key.[2]

From here, all messages are encrypted and signed. This will enable subsequent secure two-way communication between the tutor and each learner, both from the confidentiality and integrity viewpoints. When a message must be sent anonymously from an expert to a learner, the message is sent to the tutor, who verifies the signature and deciphers it with the expert's public keys before re-enciphering it and re-signing it with its own secret keys for transmission to the learner.

To illustrate the process, let us imagine that the following question, Q, comes to Eric's mind in the course of his training on Java: "How to connect an application with a database?" Furthermore, suppose that, based on the tutor's knowledge on her expertise, Nadia is the most competent expert about connection between applications and databases (in the context of Java), and that Marc can also help. In order to ask his question securely, Eric connects to SPRITS and uses the tutor's public enciphering key and his own signing key to transmit his request. Upon reception, the tutor first verifies the validity of the signature. In case of success, he deciphers the encrypted Q and obtains its contents. The tutor answers the question using its own pedagogical resources, as well as those coming from the recommendation process (Section 3.3). Whenever they come from the Q&A Recommender, these pedagogical resources are forwarded unchanged

[1] In SPRITS, a message could be a question, an answer, etc.

[2] It is fine that the tutor knows the learners' secret keys because we have assumed that it can be fully trusted. To prevent inadvertent leaks, however, it is better if the tutor erases those keys from its memory after distributing them to the learners.

but enciphered with Eric's public key and signed with the tutor's signing key, as they are just a recommendation of answers from similar cases. However, if the pedagogical resources come from the Expert Recommender, the tutor selects the most appropriate experts (top-n list is used in our case) for the learner. For Eric's request, for instance, Nadia and Marc are respectively number one and number two of the top-n list.

The tutor enciphers Q—nothing on Q gives information about the identity of the learner—with Nadia's public key (expert number one), signs it and sends it to her. Nadia uses the tutor's public key to provide her (signed) answer, R. Finally, the tutor forwards the answers to the learner with its own signature (*instead of Nadia's*). This answer is of the form (c, R), where c is a *one-time index* attributed by the tutor to Nadia. The index c, is later used by the learner to send his feedback about the answer R. When the tutor receives this feedback, it makes the mapping between c and the pseudonym (Section 3.1) of the expert, so that it is able to update this expert's profile adequately.

Continuing with our example, Eric could declare that the answer provided by c (not knowing who is behind this c) solved his problem. He may also rate the answer as 5 in a one-to-five scale. But it could also happen that Eric is not satisfied with answer R. In that case, he may express his discontent to the tutor with a rating of 1 (poorest possible), which would be used by the tutor to downgrade Nadia's profile. If Eric is not satisfied, the tutor would provide him with Marc's anonymized answer, as he came second in the top-n list. The whole procedure would then start all over again. The process just described preserves the anonymity of both learners and experts, provided they trust the tutor, meaning that no learner/expert will collude with it. This assumption holds in the context of an ITS, since the tutor is assumed to be honest towards each learner. This process also ensures the confidentiality of exchanged data since learners/experts and the tutor encipher the contents of their communications. It also ensures integrity of exchanged data thanks to digital signatures. The above security services enable the protection of learners/experts' privacy during their training, since nobody has direct access to the profile of any other.

3.5 Implementation and Validation

The implementation of SPRITS is two-fold. On the one hand, we have a *cleartext mode* in which we look at how the recommender system operates. On the other hand, we define the *secure mode*, which aims at securing the communications between learners/experts and the tutor. (We do not put the emphasis on the direct communication between just learners/experts, since SPRITS does not offer a discussion forum). We have implemented and validated the use of our recommender system in an ITS, using the cleartext mode. We are currently working on the implementation of the secure mode. When the two modes will be operational, switching between them will be allowed, depending on the *desiderata* of each learner/expert. Figure 2 presents screenshots from SPRITS.

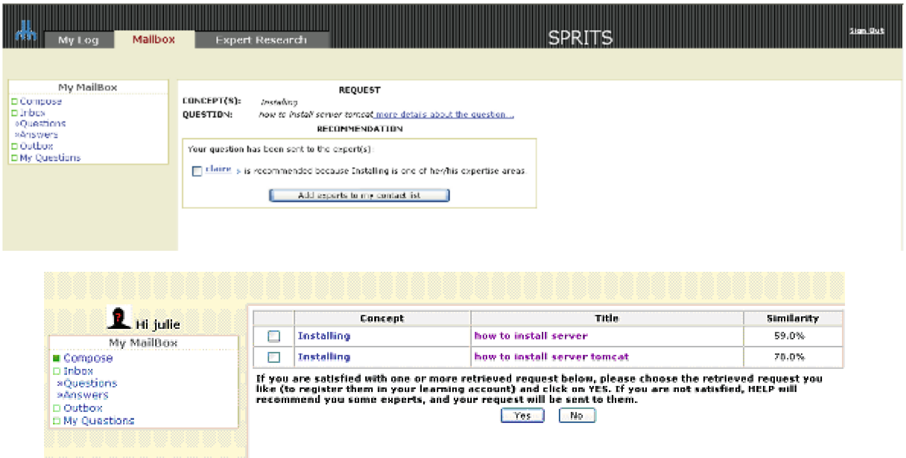


Fig. 2. Screenshots of SPRITS

4 Comparison with Related Work

In this section, we discuss the I-Help system, because of its aspects in common with our system. *I-Help* is a peer system initially developed by Greer *et al.* [11], under the name *PHelpS*, to help learners as they learn, and to deploy learners themselves in both helper and helpee mode. The I-Help system consists of two components: the *I-Help Pub* and the *I-Help 1-on-1*, which are respectively public and private components. The public component operates as discussion forums in which learners are allowed to send questions, responses or comments, to be shared with other members of the forums or peers. As for the private component, a private discussion is allowed only between the learner who has requested help and a single peer, who can help him as an expert. Globally, our system has the same objectives as the private component of I-Help. However, in contrast with I-Help, we introduce security aspects in SPRITS to guarantee confidentiality and integrity of exchanged data, as well as privacy and anonymity for learners and experts. These security aspects also aim at overcoming shyness from learners since questions, answers, evaluations, etc., can be provided anonymously (Section 3.4). Furthermore, in the private component of I-Help, each learner, A, is associated with an agent whose role is to negotiate with the agents of other learners, for the localisation of potential helpers [20]. More precisely, the negotiation produces with the top-n helpers. In our case, SPRITS uses a *recommender system* to output a list of learners-experts who can potentially help.

5 Discussion and Conclusions

Information security has become an important topic about which one cannot afford to be unaware, even in the context of an ITS. In this paper, we have presented SPRITS, a system offering security services (confidentiality and integrity of exchanged data, anonymity of learners/experts), as well as additional pedagogical resources, to an ITS. The example of a company that uses an ITS to ensure lifelong training and learning to its employees is relevant in the sense that, sometimes, the competitive spirit may override that of collaboration. Here, employees are in a competitive situation within the company, as they want to prove their competency skills and/or they need a promotion. Therefore, some malicious employees may decide to cheat, for instance by faking the votes or evaluations they provide in the ITS. This illustrates the necessity to equip the ITS with security mechanisms. Employees are involved in collaborative learning since they have to make the ITS benefit from their expertise, in order to make it available to other novice learners of the company. This collaborative learning aspect is thus important because it provides the tutor with pedagogical resources it can use for its teaching purposes. Therefore, we have introduced a recommender system as a component that outputs such pedagogical resources. We have implemented SPRITS successfully and a validation has been carried out to test the recommender system. As for the security aspects, we are currently running the test and validation processes. In addition, in contrast with the use of pairs of public/private keys by the learners/experts and the tutor, we are analysing the possibility to introduce a global secure environment in an ITS. One way to do that is to use the SSL protocol, as this protocol enjoys great success in the context of e-commerce, in which security problems occur with more vehemence. This issue was left for future work.

References

1. Aïmeur E., Burke R. and Mouaddib A., "Workshop on Multi-Agent Information Retrieval and Recommender Systems", *IJCAI-05*, Edinburgh, Scotland, 2005
2. Aïmeur E., Frasson C. and Dufort H. "Co-operative Learning Strategies for Intelligent Tutoring Systems", *Applied Artificial Intelligence. An International Journal*, 14(5):465–490, 2000.
3. Becks A., Reichling T. and Wulf V., "Supporting collaborative learning by matching actors", *Proceedings of the 36th Annual Hawaii Conference on System Science (HICSS'03)*, Big Island, Hawaii, January 2003.
4. Burke R., "Hybrid recommender systems: Survey and experiments", *User Modeling and User-Adapted Interaction*, 4(12):331–370, Hingham, MA, 2002. Kluwer Academic Publishers.
5. Budzik J. and Hammond K.J., "Q&A: a system for the capture, organization and reuse of expertise", *Proceedings of the 62nd Annual Conference of the American Society for Information Science (ASIS)*, Medford, October/November 1999.
6. Campbell S., "How to think about security failures", *Communication of ACM*, 49(1):37–39, 2006.
7. Chan T.W. and Baskin A.B., "Learning companion systems", *Intelligent Tutoring Systems: At the crossroads of Artificial Intelligence and Education*, Frasson & Gauthier (Eds.), New Jersey: Ablex, 1990.
8. Diffie W. and Hellman M.E., "New directions in cryptography", *IEEE Transactions on Information Theory*, 22(6):644–654, October 1976.
9. Falchikov N., *Learning together: Peer tutoring in higher education*, Routledge-Falmer, 2001.

10. Goldberg D., Nichols D., Oki B.M. and Terry D., "Using collaborative filtering to weave and information tapestry", *Communications of the ACM*, 35(12):61–70, December 1992.
11. Greer J., McCalla G., Cooke J., Collins J., Kumar V., Bishop A. and Vassileva J., "The Intelligent Helpdesk: Supporting Peer-Help in a University Course", *Proceedings of ITS 1998: Fourth International Conference on Intelligent Tutoring Systems*, pp. 494–503, August, San Antonio, Texas, 1998.
12. Herlocker J.L., Konstan J.A. and Riedl J., "Explaining collaborative filtering recommendations", In *Proceedings of the 2000 Conference on Computer Supported Cooperative Work (CSCW00)*, pp. 241–250, Philadelphia, Pennsylvania, 2000.
13. Kochakornjarupong D., Brna P. and Vickers P., "Who helps the helpers? A situated scaffolding system for supporting less experienced feedback givers", *Artificial Intelligence in Education*, Looi (Eds.), pp. 851–853, IOP Press, 2005.
14. Kolodner J.L. and Leake D.B., "Chapter 2: A tutorial Introduction to Case-Based Reasoning", In *Case-Based Reasoning: Experiences, Lessons & Future Directions*, Leake D. (Editor), pp. 81–92, 1996.
15. Lenz M., "Defining knowledge layers for textual case-based reasoning", In *Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning (EWCBR-98)*, Dublin, Ireland, pp. 298–309, September 1998.
16. Merkle R.C., "Secure communications over insecure channels", *Communications of the ACM*, 21:294–299, 1978.
17. Porter M.F., "An algorithm for suffix stripping", *Program*, 14(3):130–137, 1980.
18. Resnick P., Iacovou N., Sushak M., Bergstrom P. and Riedl J., "Grouplens: An open architecture for collaborative filtering of netnews", In *Proceedings of the 1994 Computer Supported Collaborative Work Conference*, pages 175–186, Chapel Hill, North Carolina, 1994.
19. Salton G. and McGill M.J., *Introduction to Modern Information Retrieval*, McGraw Hill, 1983.
20. Vassileva J., McCalla G. and Greer J. "Multi-Agent Multi-User Modelling in I-Help", *User Modelling and User Adapted Interaction*, 13(1):179–210, 2003.

The Pyramid Collaborative Filtering Method: Toward an Efficient E-Course

Sofiane A. Kiared[1], Mohammed A. Razek[2], and Claude Frasson[1]

[1] Département d'informatique et de recherche opérationnelle
Université de Montréal C.P. 6128, Succ. Centre-ville Montréal,
Québec Canada H3C 3J7
{kiaredas, frasson}@iro.umontreal.ca
http://www.iro.umontreal.ca/~frasson
[2] Faculty of Science, Math. and Computer Science Department
Azhar UniversityCairo, Egypt
abdelram@sci-azhar.edu.eg
http://www.sci-azhar.edu.eg/dep/math/abdelram.html

Abstract. Web-based applications with very diverse learners fail because they fail to satisfy various needs. Some people use collaborative filtering methods to analyze learners' profiles and provide recommendation to a new learners, but this methods provides false recommendations from beginners. We present a new method, which provides recommendations that depend on the credibility rather than the number of learners. We have designed, implemented, and tested what we call the Intelligent E-Course Agent (IECA). Our evaluation experiment shows that our approach greatly improves learners' knowledge and therefore presents a course that is more closely related to their needs.

1 Introduction

Recently, research teams have investigated ways of solving problems for learners who use hypermedia systems and have proposed several ways to overcome them. A recommendation system might work. It would try to personalize students' needs by building up information about their likes and dislikes, what succeeds and what fails [9]. It would rely on two techniques: content-based filtering (CB) and collaborative filtering (CF) [1, 7]. Collaborative filtering techniques information that is based on the opinions of those whose needs and preferences are like those of the learner. But this method does not take into account their credibility.

To clarify our point of view, consider a web-based educational system called Annaba. Every day, more than 100 learners use it. Each chooses topics to study. Suppose that units U_i, and U_j of topic C_k were selected 1000 times and 500 times respectively. The existing collaborative filtering method would recommend U_i, but U_j would be better. We suggest that many learners were still beginners and therefore choose the wrong unit. In this paper, we suggest a new selection method, one that depends on the credibility of the learners. Suppose now that the credibility of learners who select U_i, and Uj are 0.3 and 0.7 respectively. The selection value for U_i, and U_j would become 1000*0.3 and 500*0.7, which would equal 300 and 350 respectively. In that case, Annaba would recommend the correct unit: U_j.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 248 – 257, 2006.

In this paper, we will modify a Pyramid Collaborative Filtering Model (PCFA) [14] for filtering and recommending a unit rather than a learner. PCFA has four levels.

Moving from one to another depends on three filtering techniques: domain-model filtering, user-model filtering, and credibility-model filtering.

Our underlying hypothesis is that we can return a unit that has been selected by a learner who satisfies the following conditions: (1) help from someone with extensive knowledge of the concept being taught, (2) someone whose behavior and learning style are like those of the learner, and (3) someone whose credibility guarantees his choice (i.e., we can depend on his unit). To satisfy the third condition, we follow use a PCFA. We need answer two questions: Is this a unit that the learner needs? Does it contain information that he can understand? Based on what we learned by asking those three questions, we have designed, implemented, and tested what we call an Intelligent E-Course Agent (IECA).

We have organized the rest of this paper as follows. Section 2 briefly describes some related work; section 3 shows how the IECA method works; section 4 presents an overview of our system and IECA architecture; section 5 presents some experimental results; and section 6 suggests future projects.

2 Related Work

Before proposing a new architecture, I find it necessary to discuss some available ones. Adaptive technologies will probably new ways to offer efficient web-based education systems [2]. But the main ones, which still use the learner-style concept, fall into the following three categories [4, 10]: Adaptive presentation and curriculum sequencing; adaptive navigation support; and adaptive collaboration support.

2.1 Adaptive Presentation & and Curriculum Sequencing

These technologies aim at adapting learning style. Adaptive-presentation technologies adapt the content of user interfaces to the user's goals, knowledge, and other information that the user model stores. Curriculum-sequencing technology offers to a learner the most suitable individually planned unit through the learning materials (the optimal path) [4, 10]. Examples in this category are CITS [12], Arthur and CS383 [10], ACE and INSPIRE [10].

2.2 Adaptive-Navigation Support

The goal of adaptive-navigation-support technology is to help learners navigate in hyperspace by changing the appearance of each visible link. In other words, it helps learners to find their paths by adapting link presentation to their goals, knowledge, and other characteristics. The most popular of these techniques are direct guidance, sorting, hiding, annotating, and generating (which is the newest and most popular technology in the context of e-commerce) [3, 4, 8].

2.3 Adaptive-Collaboration Support

Adaptive-collaboration support is a new way to enhance the quality of web-based education. It involves communication between several learners (social interaction) and potentially collaboration. The aim is to use a system's knowledge of many users (stored in user models) in forming a matching group of learners [10, 4].

3 IECA Methodology

Although the content-based approach [9] studies the contents of recommended items, collaborative filtering [1] treats learners as a community. Using content-based and collaborative-filtering techniques to recommend units is an interesting and challenging application, but they consider neither the credibility of learners nor their learning styles. We have applied the PCFA to the collection of books. This paper focuses on only two levels of PCFA: domain-model filtering and user-model filtering.

3.1 Problem Statement

Following [13], a course could contain several concepts. Each would consist of units: background, definition, problems, examples, and exercises. Let C be an online course. Suppose that C contains k chapters and that each chapter can be divided into n units:

$$C = \{U_{11}, U_{12}, \ldots U_{1n}, U_{21}, \ldots U_{2n}, \ldots, U_{i1}, \ldots, U_{ij}, \ldots, U_{in}, \ldots, U_{k1}, \ldots, U_{kn}\}$$

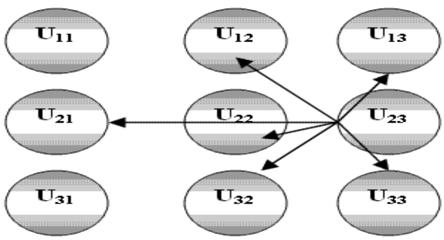


Fig. 1. Interdependence of units

If each unit has its own exercises, learner L must pass them before moving on to the next unit. Suppose that L fails to pass U_{23}, for some reason. Maybe he cannot understand the preceding or following units, as shown here:

Our system would switch him to a suitable unit, which would help him understand U_{23}. In this sense, we would apply the PCF technique in order to

- find units that share the same concept,
- find a subset of these units that meets L's learning style, and
- choose one unit of this subset, one that has a greater credibility, which means that we must calculate the amount of credibility for each unit.

We would return the unit of a learner, therefore, who satisfies three conditions: he knows a lot about the concept being taught is like L_n in behavior and learning style and we can depend on his unit because of its credibility). To satisfy the later, we would use PCF [14]. This method suggests three questions. Does the learner need this unit? Can the learner understand it ? And can we guarantee that this unit will meet his needs? To answer that question, we would need to find best learner and recommend his unit. So, we would apply PCF on a units set.

3.2 Identifying the Suitable Unit for a Learner

In this section, we will discuss all stages in the process of identifying the suitable unit algorithm (as shown in figure 2). To some extent, the algorithm will identify the helper (who is also a learner) who has almost the same characteristics (i.e. learning style) of the learner and followed at least a part of the course. The first step shows how to find a helper with information that L_n needs.

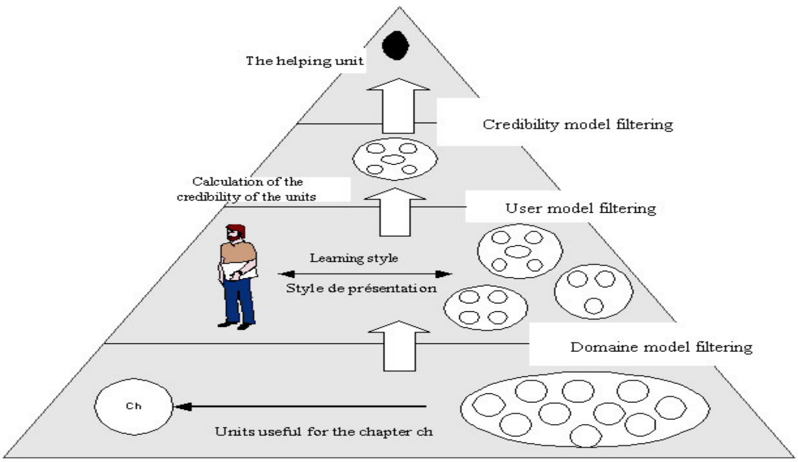


Fig. 2. The pyramidal approach of the IECA

Step 1: Dominant meaning filtering

- For each learner $L_u \in L$, $L_u \neq L_n$, compute the dominant meaning similarity $S(Q, C_h)$ between the new learner's concept Q and others concepts C_h that L_u has visited.
    - Suppose that a set of the C_h concept's dominant meanings is $\{w_1, ..., w_m\}$ and that the dominant meaning set of the query Q is $\{q_1, ..., q_s\}$. We can evaluate the dominant meaning similarity $S(Q, C_h)$:

$$S(Q, C_h) = \frac{1}{s}\sum_{i=1}^{s} \frac{1}{m}\left[\sum_{j=1}^{m} \Theta(w_i, q_j)\right], \qquad (1)$$

where $$\Theta(w_i, q_j) = \begin{cases} 1 & w_i = v_j \\ 0 & w_i \neq v_j \end{cases}$$

- List learners for whom the dominant meaning similarity value is higher than the threshold fixed for the concept Q .
- Let us call this $List_q$.

Step 2: User model filtering

- For each learner $L_u \in List_q$, compute the similarity of user behaviors $B^q_{u,v}$ as follows :

After participating, we could have a visiting vector for each user. Visiting vector $V_q = (v_i^q)_{i=1}^l$ represents the atomic units in concept Q, which have been visited by user v . Component v_i^q is equal to zero if v does not visit it and equal to one if he or she does. Therefore, we can compute the similarity between users v and u as follows:

$$B(u,v) = \tfrac{1}{D}\sum_{k=1}^{D} B^k_{v,u} \quad , \tag{2}$$

Where D is the number of concepts, and

$$B^k_{v,u} = \frac{\sum_{i=1}^{l} v_i^k u_i^k}{\sqrt{\sum_{i=1}^{l} (v_i^k)^2 \sum_{i=1}^{l} (u_i^k)^2}}$$

- For each learner $L_u \in List_q$, compute the learning style similarity $LS_{u,v}$.

Following [11], the GA distinguishes several learning styles (LS): visual V ; auditory A ; kinesthetic K ; visual-kinesthetic VK ; visual-auditory VA ; and visual-auditory-kinesthetic VAK .

$$LS_{u,v} = \begin{cases} 1 & LS_u = LS_v \\ 1/2 & LS_u \in \{V,A,K\} \,\&\, LS_v \in \{VK,VA,KA\} \\ 2/3 & LS_u \in \{VK,VA,KA\} \,\&\, LS_v \in \{VKA\} \\ 1/3 & LS_u \in \{V,A,K\} \,\&\, LS_v \in \{VKA\} \end{cases} \tag{3}$$

- For each learner $L_u \in List_q$, compute the similarity degree between user's modeling, $\mathrm{UMD(u,v)} = \frac{1}{2}\left(B^q_{u,v} + LS_{u,v}\right)$.

- Keep the list of learners for whom the dominant-meaning-similarity value is higher than the threshold fixed for concept Q. Let us call this $List_{q2}$.

Step 3: Credibility filtering

- For each learner $L_u \in List_{q2}$, compute the credibility of learner Ω_u. The credibility Ω_u [14] is the dependability degree of learners on the information presented by helpers during a learning session.
- Thus, the best helper for learner L_v is one who has the greatest value for Ω_u; let that be L_h.
- Therefore the algorithm will return the unit of the learner L_h.
- If learner L_u is a new participant in the system, then copy the entire unit of helper L_h to that of L_u. Otherwise, copy the file unit except parts of the course [that L_u has] already followed.

Note: the credibility here is for the unit, not the learner. Therefore, we suppose that the unit's credibility is associated with that of the person who has visited it. Suppose that N_{13} learners have visited unit U_{13}. Say N^s_{13} of them have succeeded and N^f_{13} failed. Therefore, the credibility of N_{13} is divided into two types: positive and negative.

Positive credibility C^s_{ij} is the average number of learners who have visited Uij and success times the summation of their credibilities. C^s_{ij} is computed as :

$$C^s_{ij} = \left(\frac{N^s_{ij}}{N_{ij}}\right) \sum_{k=1, L_k \in \{N^s_{ij}\}}^{N^s_{ij}} C(L_k)$$

Negative credibility C^f_{ij} is defined as the average number of learners who have visited U_{ij} and failed times the summation of their credibilities. C^f_{ij} is computed as follows:

$$C^f_{ij} = \left(\frac{N^f_{ij}}{N_{ij}}\right) \sum_{k=1, L_k \in \{N^f_{ij}\}}^{N^f_{ij}} C(L_k)$$

Therefore, compute the credibility of U_{ij} as follows:

$$C_{ij} = \begin{cases} C^s_{ij} - C^f_{ij} & C^s_{ij} > C^f_{ij} \\ 0 & C^s_{ij} \le C^f_{ij} \end{cases}$$

4 System Overview

This section shows how IECA has implemented, demonstrate IECA architecture, and present a scenario for a searching session.

4.1 IECA Architecture

The first time that learners use the system, it allows them to enter their interests (username, password, subjects that they prefer, and so on). Then, it asks questions to predict their learning styles. IECA architecture contains two sides, that of the client and that of the server, as shown in figure 3. The former presents user interfaces where learners can interact and use IECA's features. The latter contains the IECA, which interacts with user profiles, a dominant-meaning dictionary, and the course unit (which is stored as an xml file).

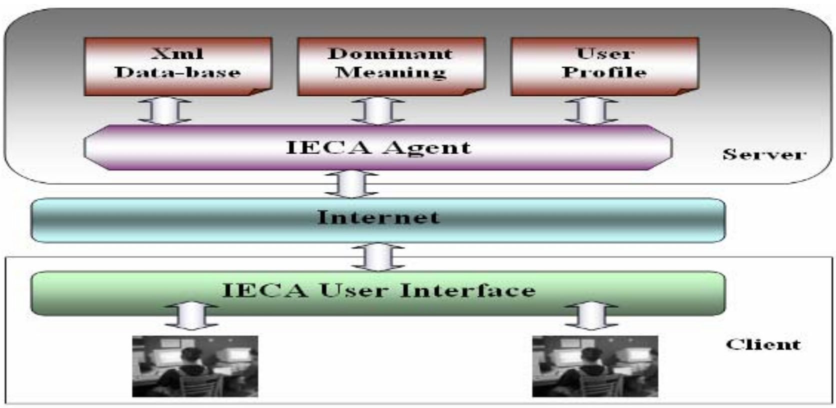


Fig. 3. The IECA architecture

4.2 IECA Scenario

To implement the system, we used ASP .NET technology -- Microsoft C# .NET framework 1.1, XML, and an Access database (for the collaborative part only) -- along with IIS as a web server. Conceptually, the general model has these components: sign-up and registration, learning-style test and the learning session.

Sign-Up and Registration
The new learner signs up by using the following registration form in order to create a personal profile. Each profile stores personal information: last name, first name, login (known as static information), and information about learning style, dominant-meaning words, and behavior (dynamic information). The learner may change this information at any time by editing it.

Learning-Style Test

Immediately after registration, the system opens a session. At this stage, the learner cannot begin the course. Only two operations are available: passing the learning-style test or disconnecting from the system. According to [5], the learning-style test consists of questions. The IECA uses the answers to identify a specific the learning style. There are seven distinct styles: visual, auditory, kinesthetic, visual-auditory, visual-kinesthetic, auditory-kinesthetic, and visual-auditory-kinesthetic.

Learning Session

Our application focuses on courses. Once they see the course summary (figure 4), the learners can begin a course by selecting either the last section completed before signing off or, for new learners, the first section. The example that we use for our application is a course on the Internet. It consists of seven chapters, each having one to three sections. Each section provides information on Internet. To monitor progress, we have included a multiple-choice test at the end of each section. A learner who passes with a grade of 50% will move on to the next section. Otherwise, the IECA will provide assistance with the helping unit.

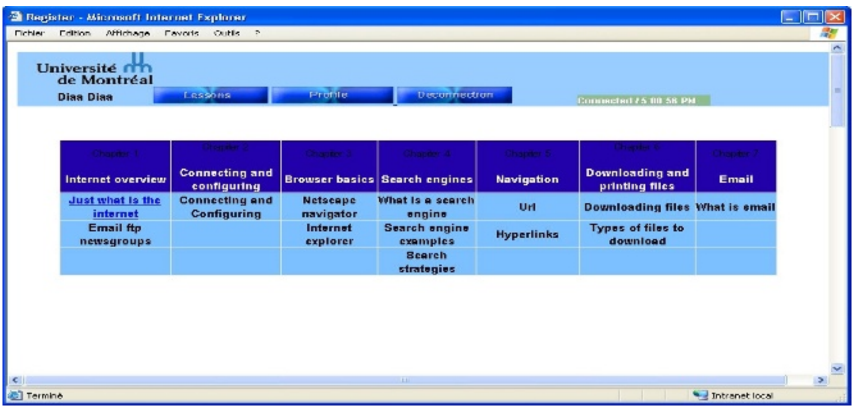


Fig. 4. The course synopsis

5 System Evalution

We come now to the results of our experiment. Our aim, of course, was to measure the effectiveness of our system. We selected two groups of 30 learners. Learners from the one called NASSG followed the course without the assistance of an IECA. Those from the one called ASSG did. Figure 5 presents the averages for each group by course unit.

The two histograms indicate that ASSG did better than NASSG. Our system offers learners a considerable advantage, but the two groups have the same contribution at units U_{101} and U_{401}.

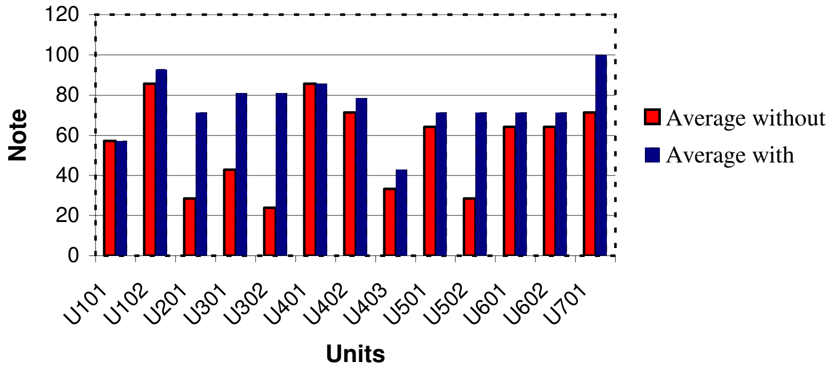


Fig. 5. Average of correct answers per unit

6 Conclusion and Future Work

We have described the use of PCF to provide learners with the best units available in hyperspace. We based our proposed intelligent-agent system on a three level PCF, which uses dominant-meaning filtering at the lower level, user-model filtering at the middle level and credibility filtering at the upper. This technique answers the following questions: Does the learner need this unit? Can he understand the information? Our experiments show that this method greatly improves e-course performance in terms of both efficiency and credibility. Our future work will focus on how to improve testing results by conducting more experiments with more learners.

References

1 Breese J., Heckerman D., C. K. (1998). Empirical analysis of predictive algorithms for collaborative filtering. Proceedings 14th Conference on Uncertainty in Artificial Intelligence, Madison WI: Morgan Kauffman.
2 Brusilovsky P. 2000. “Adaptive Hypermedia: From Intelligent Tutoring Systems to Web-Based Education Intelligent Tutoring Systems.” Lecture Notes in Computer Science, 1839 (2000): 1-7. Paper presented at the 5th International Conference on Intelligent Tutoring Systems, ITS 2000, 19-23 June in Montreal.
3 Brusilovsky, P.: 2003. “Adaptive Navigation Support in Educational Hypermedia: The Role of Student Knowledge Level and the Case for Meta-Adaptation.” British Journal of Educational Technology 34. 4 (2003): 487-497.
4 Brusilovsky, P. 1998. “Adaptive Educational Systems on the World-Wide-Web: A Review of Available Technologies.” Paper presented at the 4th International Conference on Intelligent Tutoring Systems (ITS '98), in San Antonio .
5 Yammine, K., M. Razek, E. Aïmeur, and C. Frasson. 2004. “Discovering Intelligent Agent: A Tool for Helping Student Searching a Library.” Paper presented at the 7th International Conference on Intelligent Tutoring Systems (ITS-04), September in Maceió, Brazil.
6 Brusilovsky P, J. Eklund, and E. Schwarz. 1998. “Web-Based Education for All: A Tool for Ddeveloping Adaptive Courseware.” Computer Networks and ISDN Systems (Proceeding of Seventh International World Wide Web Conference, 14-18 April. 30 (1-7), 291-300.

7 Herlocker, J.L., J.A. Konstan, and J. Riedl. 2000. “Explaining Collaborative Filtering Recommendations.” Paper presented at the ACM 2000 Conference on Computer Supported Cooperative Work.

8 Kayama, M. and T. Okamoto. 1999. “Hy-SOM: The Semantic Map Framework Applied on an Example Case of Navigation.” In Advanced Research in Computers and Communications in Education. Frontiers ub Artificial Intelligence and Applications, Vol. 2, Amsterdam: IOS Press, 252-259.

9 Lynch, C. 2001. “Personalization and Recommender Systems in the Larger Context: New Directions and Research Questions.” Paper presented at the Seconf DELOS Network of Excellence Workshop on Personalisation and Recommender Systems in Digital Libraries, Dublin, Ireland, 18-20 June.

10 Papanikolaou, K.A. and M. Grigoriadou. 2004. “Accommodating Learning Style Characteristics in Adaptive Educational Hypermedia Systems.” Paper presented at the 3rd International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, 23-26 August, at the Eindhoven University of Technology in Eindhoven.

11 Razek, M.A., C. Frasson, and M. Kaltenbach. 2002. “Using a Machine Learning Approach to Support an Intelligent Collaborative Multi-Agent System.” Paper presented at TICE 13-15 November, in Lyon.

12 Razek, M.A., C. Frasson, and M. Kaltenbach.2003. “A Context-Based Information Agent for Supporting Intelligent Distance Learning Environments.” Paper presented at the 12th International World Wide Web Conference, 20-24 May, in Budapest.

13 Razek, M.A., C. Frasson, and M. Kaltenbach. 2003. “Using Adaptability to Create an Automatically Adaptive Course Presentation on the Web.” Paper accepted for IEEE ICALT, 9-11 July, in Athens.

14 Razek, M.A., C. Frasson, and M. Kaltenbach. 2004. “A New Filtering Model towards an Intelligent Guide Agent.” Paper presented at the 17th International FLAIRS Conference, 17-19 May, in Miami Beach.

From Black-Box Learning Objects to Glass-Box Learning Objects

Philippe Fournier-Viger[1], Mehdi Najjar[2], André Mayers[2], and Roger Nkambou[1]

[1] Department of Computer Science, University of Quebec at Montreal
{fournier-viger.philippe, nkambou.roger}@uqam.ca
[2] Department of Computer Science, University of Sherbrooke
{mehdi.najjar, andre.mayers}@usherbrooke.ca

Abstract. In the field of e-learning, a popular solution to make teaching material reusable is to represent it as learning object (LO). However, building better adaptive educational software also takes an explicit model of the learner's cognitive process related to LOs. This paper presents a three layers model that explicitly connect the description of learners' cognitive processes to LOs. The first layer describes the knowledge from a logical perspective. The second describes cognitive processes. The third builds LOs upon the two first layers. The proposed model has been successfully implemented in an intelligent tutoring system for teaching Boolean reduction that provides highly tailored instruction thanks to the model.

1 Introduction

Teaching resources are the mean to teach domain knowledge within tutoring systems. In the field of e-learning, a popular solution to increase their reuse is to represent them as learning objects (LOs). The concept of LO is sustained by a set of principles and standards that facilitate their reuse and distribution. However, tutoring systems generally treat LOs as black boxes. i.e. presented as they are and without individualised feedback for each learner. Moreover, modelling the cognitive processes of learners is fundamental to build educational software that provides highly tailored instruction [2]. This article presents a model that unify some principles of the cognitive modelling theories, which attempts to model the human processes of knowledge acquisition, and standards related to the concept of LOs, which takes on the challenges of knowledge reuse and distribution. LOs described according to our approach are "glass-box LOs" because they include an explicit description of cognitive processes. The remainder of the article is organised as follows. First, the LO concept is defined. Second, the virtual learning environment (VLE) in which the model has been implemented is introduced. Then, the three next sections describe the three layers of our model. We then present a case study where the model has been implemented an evaluated. Finally, the last section announces further work and present conclusion.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 258 – 267, 2006.

2 The Learning Objects Concept

The concept of LO relies on the main idea of structuring learning materials into reusable units. Over the recent years, many definitions of LOs have been proposed. To summarise most of these definitions, one can state that a LO is an autonomous resource (digital or not) that is reusable in training activities [4]. To clarify their role and their nature, this paragraph describes the four steps of the LOs' lifecycle. The first step of a LO lifecycle consists in creating an information object (IO). i.e. an electronic document of any format. E-learning institutions usually opt for Web documents deliverable via Internet browsers. Among typical examples of IOs: a Web page explaining the game of chess or an e-book on linear algebra. Furthermore, authors should avoid creating IOs that include unnecessary references to external contexts, because IOs can be presented individually. IOs should be customizable, to facilitate their integration within particular contexts. Finally, one must determine IOs' granularity carefully, because a large granularity reduces the number of IOs that can be assembled together. For example, an e-book is less reusable than its chapters. The second step of the LOs lifecycle consists in adding metadata to the IOs. LOM (http://ltsc.ieee.org/wg12/20020612-Final-LOM-Draft.html[1]) is one of the most important metadata standards. It offers about 80 attributes to describe an IO. On one hand, metadata facilitate the localisation of IOs stored in repositories. On the other hand, they inform about how to use IOs (for example, with regard to copyrights or technology requirements). Moreover, they make possible the automatic selection of IOs by a computer. Metadata are also the element that distinguishes between IOs and LOs. More precisely, appending a learning objective transforms an IO into a LO, as it ensures that the IO is intended for teaching [4].The third step is optional in the lifecycle of a LO and consists in packaging several LOs to facilitate their distribution (LOs aggregation). For example, a professor can group together a set of objects for a teaching activity. Since a package is also an IO, if one adds the required metadata, the aggregate will be also considered as a LO. In the popular aggregation standard IMS-CP (http://www.imsglobal.org/content/packaging/), a package is a zip file which contains IOs or LOs and a single file which acts as a table of contents. The fourth step of the LOs lifecycle is LOs' delivery. VLEs usually treat LOs as black boxes. i.e. presented as they are without individualised feedback for each learner. The most significant adjustment usually consists in generating dynamic sequences of LOs, following results of questionnaires. The weak personalisation is partly explained by the fact that these tutoring systems often teach full courses and attach great importance to the participation of human teachers. Building better adaptive educational software also takes an explicit model of the learner's cognitive process related to LOs [2]. This article proposes a model for the creation of LOs that include cognitive processes description. Our model organise a domain's knowledge according to three layers, whose each one describes knowledge from a different angle. The first layer defines an ontological and logical representation. The second defines a cognitive representation. The third organise knowledge in LOs. The next section introduces the VLE for which the model was tested.

[1] All URLs mentioned in the paper were last accessed December 14, 2005.

3 The REDBOOL Boolean Reduction VLE

REDBOOL is a VLE for teaching Boolean reduction. Here, the subject-matter domain is the algebraic Boolean expressions and their simplification by means of reduction rules, which are generally taught to undergraduate students on first cycle of higher education. The tool's purpose is both to help student learn Boolean reduction techniques and to increase confidence with the software. Fig. 1-a illustrates REDBOOL's interface. Preliminary definitions and explanations are available to learners in the "Theory" tab of the VLE. A teaching session consists in solving a set of problems. For example, Fig. 1-a shows the problem to reduce the expression "(((a | F) & (T) | (~C))". Boolean expressions can be composed of truth constant "T" (true), truth constant "F" (false), proposals "a,b,c,d,e,f" conjunction operator "&", disjunction operator "|" and negation operator "~". The objective of an exercise consists in reducing an expression as much as possible by applying some of the 13 reduction rules, such as the disjunction rule of a proposal "a" with the truth constant "False" ((a | F) = (a)). A learner can select part of the current expression in the "Reduction" field and modify it by using the virtual keyboard proposed. The learner must click on the "Submit step" button to validate changes. In the bottom area of the window, the learner can see the last rules applied. The "Advices" section shows the system's feedback (hints, advices, etc.). The following sections detail each layer of our model with examples from REDBOOL.

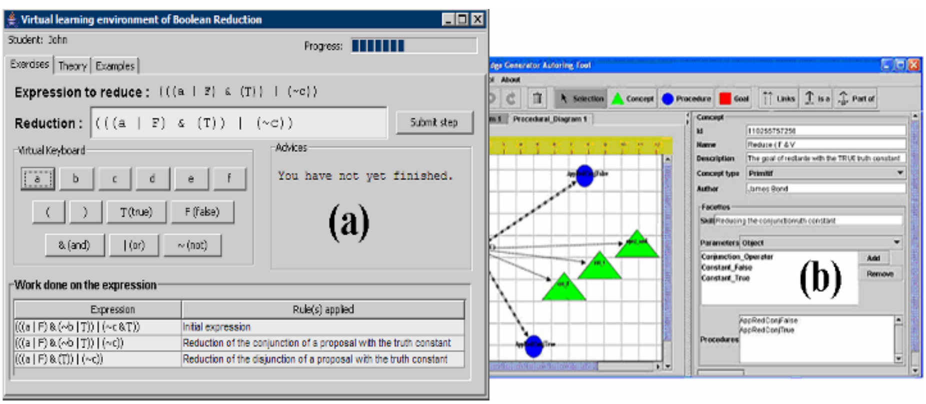


Fig. 1. The REDBOOL VLE (a) and the DOKGETT authoring tool (b)

4 Layer 1: Logical Representation of the Domain Knowledge

The first layer of our model contains a logical representation of the domain's concepts and their relationships. The formalism used is description logics (DL). We have chosen DL because they are well-studied and widely used logic based languages that offer reasoning algorithms whose complexity is often lower than those of first order logics. DL employ an ontological approach. i.e., to describe the instances of a domain, they require the definition of (1) general categories of instances and (2) the various types of logical relationships among categories and their instances. The ontological approach is appropriate since reasoning usually occurs at the level of categories. In the DL terminology, whereas a TBOX (terminological box) describes the general knowledge of a field, an ABOX (assertional box) describes a specific world. TBOX contains axioms which relate to *concepts* and *roles*. ABOX contains a set of assertions which describe *individuals* (instances of concept). Table 1 gives as example a part of the TBOX defined for REDBOOL.

Table 1. Part of layer 1 knowledge for REDBOOL

Concepts	Roles
TruthConstant $\equiv$ BooleanExpression $\sqcap$ $\neg$Variable	operator
TruthConstantF $\sqsubseteq$ TruthConstant	leftOperand
TruthConstantT $\sqsubseteq$ TruthConstant	rightOperand
Variable $\equiv$ BooleanExpression $\sqcap$ $\neg$DescribedExpression	
DescribedExpression $\equiv$ BooleanExpression $\sqcap$ ($\exists$operator.$\top$ $\sqcap$ $\forall$operator.Operator) $\sqcap$ $\neg$ TruthConstant	
DisjunctionExpression $\equiv$ $\forall$rightOperand.BooleanExpression $\sqcap$ $\exists$leftOperand.$\top$ $\sqcap$ $\forall$operator.OperatorDisjunction $\sqcap$ $\exists$rightOperand.$\top$ $\sqcap$ $\forall$leftOperand.BooleanExpression $\sqcap$	

Atomic concepts and atomic roles are the basic elements of a TBOX. Their names begin respectively with an uppercase and a lowercase letter. The atomic concepts and atomic roles can be combined with constructors to form *concept descriptions* and *role descriptions*. For example, the concept description "BooleanExpression $\sqcap$ Variable" results from the application of constructor $\sqcap$ to atomic concept *BooleanExpression* and *Variable*. To formally describe constructors' semantic, it is necessary to define *interpretations*. An interpretation I consist of an interpretation domain Δ^I and an interpretation function $\cdot^I$. The interpretation domain is a set of individuals. The interpretation function assigns to each atomic concept A a set of individual $A^I \mid A^I \subseteq \Delta^I$ and to each atomic role R; a binary relation $R^I \mid R^I \subseteq \Delta^I \times \Delta^I$. The concept constructors of a basic DL named $\mathcal{AL}$ [3] are $\neg$A, $\exists$R.$\top$ and $\forall$R.C, which are interpreted as $\Delta^I \setminus A^I$, $\{a^I \in \Delta^I \mid \exists b^I.(a^I,b^I) \in R^I\}$ and $\{a^I \in \Delta^I \mid \forall b^I.(a^I,b^I) \in R^I \Rightarrow b^I \in C^I\}$, respectively. The symbols a^I and b^I represent individuals that are members of Δ^I for an interpretation I. The letters A and B stand for atomic concepts. The letters C and D represent concepts descriptions. The letters R denote atomic roles. A TBOX contains terminological axioms of the form C $\equiv$ D or C $\sqsubseteq$ D (defined as $C^I \subseteq D^I$ and $C^I = D^I$, respectively). An ABOX contains assertions expressed in term of *nominals*. An interpretation I assigns to each nominal a, an individual a^I from Δ^I. An ABOX contains membership assertions (C(a)) and role assertions (R(a, b)), where a and b represent nominals. The assertions' semantics are $a^I \in C^I$ and $(a^I,b^I) \in R^I$, respectively. The primary purpose of DL is inference. From a DL knowledge base, it is possible to infer new facts, such as deducing nominals that are members of a concept, finding all concepts D that subsume a concept C (C $\sqsubseteq$ D), verifying disjointness of two concepts

C and D or checking that a concept C is satisfiable. Note that several free and comercial inference engines are available.

To the best of the authors' knowledge, Teege [11] first proposed to use DL to represent domain knowledge of VLEs. He stated three important originalities. One of them consists of using DL to represent the theory to be taught (TBOX) as an abstraction of natural language (ABOX). This abstraction is necessary to distinguish learners' answers and the VLE's explications/examples, from the domain concepts. In addition, a VLE could extract concepts from natural language answers to form a TBOX; and then, compare knowledge of the learners with those of the learning system. Teege demonstrates that inference engines are useful for various tasks such as finding concepts subsuming a misunderstood concept to better explain what characterises it or detecting modeling inconsistencies (e.g., unsatisfiable concepts).

The first of the three layers that constitutes our model represents concepts of a domain as DL concepts, as proposed by Teege [11]. In each defined TBOX, concepts symbolise categories of objects handled in a VLE, and roles represent relationships between these objects. Table 1 shows a part of the TBOX for REDBOOL. The first axioms state that truth constants, variables and described expressions are distinct types of Boolean expressions and specify that there are two types of truth constants ("true" and "false"). The last concept axiom asserts that a disjunction expression is a described expression that has a disjunction operator and Boolean expressions as its left and right operands. No ABOX is defined because it is the level of concrete answers and examples. When a learner interacts with our VLE, an ABOX is created dynamically by processes that will be explained afterwards. The layer 1 knowledge is stored into OWL files (http://www.w3.org/TR/owl-features), a popular file format for some DL. Numerous authoring tools and inference engines are available. OWL also offers mechanisms to increase reuse such as versioning and namespaces. Thus, authors can split up layer 1 knowledge in several files. As presented further, this facilitates the encoding of the knowledge in LOs.

5 Layer 2: Cognitive Representation of the Domain knowledge

Layer 1 allows the logical representation of the domain knowledge. However, building better adaptive educational software also takes an explicit model of the learner's cognitive process. This section presents the layer 2 of our model, which meets this purpose.

5.1 The Psychological Foundations

To structure, organise and represent the layer 2 knowledge, we have been inspired by cognitive psychology theories, which attempt to model the human process of knowledge acquisition. This knowledge is encoded according to the way in which these contents are handled and used. Although there is no consensus on the number of subsystems or on their organisation, the majority of the authors, in psychology, mentions – in some form or in another –semantic knowledge [10], procedural knowledge [1] and episodic knowledge [12]. In this paper, we do not discuss the episodic knowledge part of our model since it is the part that records lived episodes (a history of the use of the two other types of knowledge) for each learner.

The semantic memory contains descriptive knowledge. Our model regards semantic knowledge as concepts taken in the broad sense. According to recent researches [5], humans consider up to four concept occurrences simultaneously (four dimensions) in the achievement of a task. However, the human cognitive architecture has the capacity to group several concepts to handle them as one, in the form of a vector of concepts [5]. We call described concepts these syntactically decomposable concepts, in contrast with primitive concepts that are syntactically indecomposable. For example, in propositional calculus, "a | F" is a decomposable representation of proposal "a", a non-split represent-tation with the same semantic. The concept "a | F" represents a disjunction between proposal "a" and the truth constant "F" (false), two primitive concepts. The disjunction logical operator "|" is also a primitive concept. In this way, the semantic of a described concept is given by the semantics of its components.

The procedural memory is composed of procedures. i.e., means to handle semantic knowledge to achieve goals. Contrary to semantic knowledge, which can be expressed explicitly, procedural knowledge is represented by a succession of actions achieved automatically – following internal and/or external stimuli perception – to reach desirable states [1]. Procedures can be seen as a mean of achieving a goal to satisfy a need, without using the attention resources. For example, during the Boolean reduction process, substituting automatically "~T" by "F", making abstraction to the explicit call of the truth constant negation rule (~T = F), can be seen as procedural knowledge which was acquired by the repetitive doing. In our approach, we subdivide procedures in two main categories: primitive procedures and complex procedures. Executions of the first are seen as atomic actions. Those of the last can be done by sequence of actions, which satisfy scripts of goals. Each one of those actions results from a primitive procedure execution; and each one of those goals is perceived as an intention of the cognitive system.

We distinguish goals as a special type of semantic knowledge. Goals are intentions that humans have, such as the goal to solve a mathematical equation, to draw a triangle or to add two numbers [8]. Goals are achieved by means of procedural knowledge. In our model, a goal is described using a relation as follows: (R: X, A_1, A_2 … A_n). This relation allows specifying a goal "X" according to primitive or described concepts "A_1, A_2 … A_n" which characterise the initial state. In a teaching context, stress is often laid on methods that achieve the goal rather than the goal itself; since these methods are in general the object of training. Consequently, the term "goal" is used to refer to an intention to achieve the goal rather than meaning the goal itself. Thus, procedures become methods carrying out this intention.

5.2 The Computational Representation of the Psychological Model

Layer 2 of our model defines a computational representation of the cognitive model described above. This knowledge is stored in files named SPK, which describe knowledge entities according to sets of slots. Concepts are encoded according to six slots. The "Identifier" slot is a character string used as a unique reference to the concept. The "Metadata" slot provides general metadata about the concept (for example, authors' names and a textual description). The "Goals" slot contains a goals prototypes list. The latter provides information about goals that students could have and which use the concept. "Constructors" specifies the identifier of procedures that can create an instance of this concept. "Component" is only significant for described concepts. It indicates, for each concept component, its concept type. Finally, “Teaching” points to some didactic resources that generic teaching strategies of a VLE can employ to teach the concept. Goals have six slots. "Skill" specifies the necessary skill to accomplish the goal, “Identifier” is a unique name for the goal, “Metadata” describes the goal metadata, "Parameters" indicates the types of the goal parameters, "Procedures" contains a set of procedures that can be used to achieve the goal, and “Didactic-Strategies" suggests strategies to learn how to achieve that goal. Ten slots describe procedures. The “Metadata” and “Identifier” slots are the same as for concepts/goals. “Goal” indicates the goal for which the procedure was defined. “Parameters” specifies the concepts type of the arguments. For primitive procedures, “Method” points to a Java method that executes an atomic action. For complex procedures, “Script” indicates a list of goals to achieve. “Validity” is a pair of Boolean values. Whereas the first indicates if the procedure is valid and so it always gives the expected result, the second indicates if it always terminate. “Diagnosis-Solution” contains a list of pairs “[diagnosis, strategy]” that indicate for each diagnosis, the suitable teaching strategy to be adopted. Finally, “Didactic-Resources” points to additional resources (examples, exercises, tests, etc.) to teach the procedure.

We have developed an authoring tool that permits to model and to generate SPK files (Fig. 1-b). The left-hand side of the environment consists in a drawing pane where knowledge entities are represented by different shapes, and arrows represent relations between them. The right-hand side of the environment permits the author to specify detailed information about the selected knowledge entity in terms of the slots described above.

5.3 The Layer 2 Knowledge for REDBOOL

The authoring tool was used to represent the cognitive processes of learners for REDBOOL [9]. As an example, in a single SPK file, we encode the layer 2 knowledge of REDBOOL. The primitive concepts are truth constant “True”, truth constant “False”, conjunction operator, disjunction operator and negation operator. The main described concepts are conjunction expression, disjunction expression and negation expression. The file includes procedures and goals for the 13 Boolean reduction rules. It also contains definitions of goals and procedures to create concrete instances of concepts (because each concept’s occurrence must be created prior to being handled) and procedures for common errors. In REDBOOL, procedures are fired as a learner operates the graphical interface’s buttons (the button/procedure association is found in the “Method” slot of procedures), and the resolution trace is recorded. The VLE connects interactions with the interface to the layer 2 knowledge, and therefore the tutor embedded within the VLE can take decisions on the basis of the cognitive activity of each learner. The next section explains the link between layer 2 and layer 1, which allow a VLE tutor to make the correspondence between the user activity and the logical description of the domain knowledge found in layer 1.

5.4 Links Between Layer 1 and Layer 2

To establish links between the logical representation of layer 1 and the cognitive representation of layer 2, it is necessary to add additional slots to layer 2 concepts. For this purpose, each primitive concept has a "DLReference" slot that points towards a DL concept. This slot is useful during the instantiation of primitive concepts by procedures. To properly explain the instantiation process of primitive concepts, we will first consider the instantiation of the “F” truth constant. The “Constructors” slot of the “F” truth constant concept states that the procedure “P_CreateTruthConstant-False” can be used to instantiate the concept. This procedure has its action defined as such. To simulate the instantiation process, our tools adds in an ABOX a nominal associated to the DL concept mentioned in the "DLReference" slot of the concept to instantiate. The table 2 illustrates the resulting assertions added to an ABOX. The nominal “f1” represents a concept instance, and the “TruthConstantF(f1)" assertion declare that “f1” is an “F” truth constant. For each instances created, a different nominal is added to the ABOX. In the same vein, the example shows "t1" an instance of the primitive concept “T” truth constant, and “d1”, an instance of the disjunction operator primitive concept.

Table 2. ABOX assertions that represent a “(T & F)” Boolean expression

ABOX
TruthConstantF (f1), TruthConstantT(t1), DisjunctionExpression(e1) , leftOperand(e1,t1), rightOperand(e1, f1), DisjunctionOperator(d1), operator(e1, d1)

In addition to the “DLReference” slot, each described concept encompasses a slot named “Components”, which list one or more roles. Each role associates to a nominal that represent an instance of the described concept, a nominal that represent one of its parts. For example, the nominal “e1” in table 2 correspond to an instance of the described concept “T & F”. The “DisjunctionExpression(e1) ” assertion declares that “e1” is a disjunction expression. The “operator(e1, d1)”, “leftOperand(e1, t1)” and “rightOperand(e1, f1)” links the described concept represented by “e1” to nominals that represent its components. Furthermore, a learner can carry out a procedure that replaces a described concept’s component. For instance, when a learner substitute “~T” by “F” in the Boolean expression “a & (~T)”. In this case, the tools we have developed adapt the ABOX accordingly.

Because there is no direct link between layer 2 concepts, correspondence is achieved at the DL level. The absence of link between layer 2 concepts also facilitates the extension of the layer 2 knowledge. Indeed, an author can easily add concepts to any SPK file by associating logical descriptions that extends those of other concepts. Added concepts become automatically compatible with existing procedures and goals. Authors can also add new procedures for existing goals, since satisfaction links between a goal and a procedure is stored in procedures’ slots. As a result, authors can create new SPK files that extend existing SPK files without changes.

6 Layer 3: Encoding Knowledge as LOs

The third layer builds LOs upon the two first layers. The first step to obtain LOs is creating IOs. According to our model, an IO consists of SPK file(s), OWL file(s), and the VLE. The XML encoding of SPK and OWL files makes the files easily customisable. To package files together, we have recourse to IMS-CP, a standard commonly used for LOs (cf. section 2.3).

The second step of LOs' lifecycle consists in adding metadata to IOs. IMS-CP packages allow inclusion of metadata compatible with many standards. We use the RELOAD authoring tool (http://www.reload.ac.uk) to specify metadata according to the LOM standard. Moreover, creating a LO requires specifying the learning objectives that it can teach [4]. This addition indicates the pedagogical use of the IO. We consider learning objectives that relate (1) to the acquisition of a skill or (2) to the mastery of a semantic knowledge. First, to check the acquisition of a skill is equivalent to testing the ability to attain a goal. Here, the importance resides in learners' ability to realise the goal. The procedures employed are of no importance, since several correct procedures might achieve the same goal. If a learner accomplishes a goal many times with varied problems and without committing errors, one can conclude that the learner possess the corresponding skill. A concept becomes manifest only during a procedure execution which satisfy the goal using that concept. Consequently, a learner must be able to achieve several goals that used the concept in order to show that s/he acquired the concept. This definition of learning objective for the semantic knowledge covers the traditional one of researchers in pedagogy. For example, Klausmeier [7], which indicates that mastering a concept require understanding relationships that characterise it. The action of retrieving relationships can be encoded as procedures. In summary, the learning objectives are expressed in term of goal(s) to master. In this sense, our model follows the view of Anderson et al. [2] that tutoring systems should focus on teaching procedural knowledge. We propose three slots for learning objectives. The "Identifier" and "Metadata" slot have the same use as for concepts, goals and procedures. "NecessaryGoals" stipulate goals whose mastery is jointly required to meet the learning objective. Learning objectives are added in our SPK files.

7 Evaluation

A practical experimentation was performed to test the ability of our model to represent cognitive activities [9]. We asked ten (10) students in computer sciences and in mathematics who attend the course "MAT-113" or "MAT-114" (dedicated to discrete mathematics) to practice Boolean reduction with REDBOOL. An assisted training, aiming to familiarise them with the tool, was given; before leaving them practising. To compare the learners' behaviours, we forced the system to provide them common problems. Parameters of this experiment are 1(4), 2(4), 3(5), 4(6), 5(7) where $x(y)$ stands for y exercises of complexity x, for each student. Complexity ranges from simple (1) to complex (5). For each learner, the system noted the procedures used as well as the concepts' instances handled. Analysis of the collected data by a virtual tutor allows it to deduce goals (and subgoals) formulated during the reduction process. For complexity 1 to 5, the number of goals visited for a complete reduction was about 4, 7, 10, 21, and 40, and the number of concepts' instance manipulated was roughly 4, 14, 24, 35 and 52, respectively.

8 Conclusion and Further Work

We have proposed a model for creating reusable glass-box LOs that incorporates logical, cognitive, as well as didactic knowledge (cf. section 5.2). The model has been experimented successfully and authoring tools are available for each steps of the modelling process. The inclusion of a logical structure to describe domain knowledge facilitates the separation of the knowledge in multiple files, and provides a basis for logical reasoning. Moreover, using cognitive structures permit building tutors that presents LOs together with individualised feed-back. In a near future, our work will focus on representing knowledge for new domains. We are also investigating different ways to benefits from the knowledge encoded in our LOs.

References

1. Anderson, J.R. (1993). *Rules of the mind*. Hillsdale, NJ: Erlbaum.
2. Anderson, J.R, Corbett, A.T., Koedinger, K.R. & Pelletier, R. (1995). Cognitive Tutors: Lessons learned, *Learning Sciences*, *4*(2), 167-207.
3. Schmidt-Schauß, M. & Smolka, G. (1991). Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1), 1–26.
4. Duncan, C. (2003). Granularization. In Littlejohn, A. (Ed.), *Reusing Online Resources: a sustainable approach to e-learning*. London: Kogan Page.
5. Halford, G.S., Baker, R., McCredden, J.E. & Bain, J.D. (2005). How many variables can humans process?, *Psychological Science*, *16*(1), 70-76.
6. Klausmeier, H.J. (1990) Conceptualizing. In Jones, B.F. & Idol, L. (Eds.), *Dimensions of Thinking and Cognitive Instruction* (pp. 20-34), Erlbaum.
7. Mayers, A., Lefebvre, B. & Frasson, C. (2001). Miace, a Human Cognitive Architecture, *SIGCUE OUTLOOK*, 27 (2), 61-77.
8. Najjar, M. & Mayers, A. (2004). Using Human Memory Structures to Model Knowledge within Algebra Virtual Laboratory. In *Proc. of ITRE 04*. pp. 155-159.
9. Neely, J.H. (1989). Experimental dissociation and the episodic/semantic memory distinction. *Experimental Psychology: Human Learning and Memory*, 6, 441-466.
10. Teege, G. (1994). Using description logics in intelligent tutoring systems. In *Proceedings of the 2004 DL Workshop*. pp. 75-78.
11. Tulving, E. (1983). *Elements of Episodic Memory.* NY: Clarendon Press.

Adaptation in Educational Hypermedia Based on the Classification of the User Profile

Gisele Trentin da Silva[1] and Marta Costa Rosatelli[2]

[1] Departamento de Informática e Estatística, Universidade Federal Santa Catarina, INE-CTC, Cx.P. 476, Campus Universitário, Florianópolis-SC, 88040-900, Brazil
gisele@inf.ufsc.br

[2] Programa de Pós-Graduação em Informática, Universidade Católica de Santos
R. Dr. Carvalho de Mendonça, 144, Santos-SP, 11070-906, Brazil
rosatelli@unisantos.br

Abstract. This paper presents SEDHI - an adaptive hypermedia system for a web-based distance course. The system architecture includes three main modules: the Classification Module, the Student Module, and the Adaptation Module. SEDHI classifies the students according to profiles that were defined based on a statistical study of the course user and usage data, and adapts the navigation using the techniques of link hiding and link annotation. The results of an evaluation of the SEDHI prototype show the potential of the classification and adaptation approach.[1]

1 Introduction

In the context of distance education, web-based courses are frequently offered to a large number and broad range of students. Most times, the contents and presentation of didactic materials are exactly the same to all students. The objective of this work is to consider user and usage data to provide support to navigation. Among others, the aim is to facilitate the work of the human tutors who typically give support to students in distance courses.

This paper presents SEDHI (Educational Hypermedia System, in Portuguese), a system for web-based distance education. Based on a statistical study of user and usage data [8] of a particular course, a classifier with three student profiles was defined. SEDHI classifies the students according to these profiles and adapts the navigation using the techniques of link hiding and link annotation [1][2].

The paper is structured as follows: section 2 reports on related work; section 3 provides an overview of SEDHI; section 4 describes the system evaluation; section 5 presents the conclusions and directions for further work.

2 Related Work

Adaptive Hypermedia (AH) [1][2] makes possible the organization of hypermedia environments, the conduction of the user through desirable paths, the omission of

[1] This research is partially supported by FAPESP grants 03/08279-2 and 03/08776-6. The authors thank Ilog Tecnologia, IEA, and Sebrae Nacional.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 268–277, 2006.

links irrelevant to the objectives, preferences, and interests of the user, making the navigation in hyperspace more attractive and organized, according to the user profile and his or her representation in the User Model. Adaptation occurs typically in two levels: adaptive presentation, in which the contents are adapted; and adaptive navigation, in which the links are adapted.

A number of educational hypermedia systems that adapt the navigation have been developed, e.g. ISIS-TUTOR [3]; ELM-ART [5]; AHA [6]; INTERBOOK [4]; and KBS Hyperbook System [10], among others. Such systems support navigation using different techniques in order to improve learning outcomes.

In web-based educational systems that generate large amounts of student data, data mining algorithms [7] can provide input for adaptive navigation. Data mining is indicated for discovering pedagogical relevant data for providing feedback, supporting teachers and learners, and so on [11][12]. In addition, adaptation to students based on user and usage data is relevant in collaborative learning [8][9]. In this work, students are classified according to profiles that were defined based on a statistical analysis of a course user and usage data. While it might be argued that profiles are limited from the point of view of student modelling, some educational adaptive hypermedia systems adopt the profiling approach (see [14] for an example). Regarding classification algorithms, this approach allows predefining classes based on data attribute values and providing adaptation based on past student behaviour obtained from the web-based systems databases.

3 SEDHI: An Educational Hypermedia System

SEDHI design is based on a study of a web-based distance course on Entrepreneurship. The course objective is to teach the students how to write a Business Plan in a step-by-step fashion, train the student in organizing ideas and resources, and consider the relevant aspects for planning and starting a business. The course is offered since the year 2001 by a Brazilian national organization that provides support services to small businesses. So far, more than 250,000 students completed the course. Each edition of the course has around 70 classes of 200 students, summing 14,000 students in each course edition.

The course is designed to be completed in two months (60 days). The contents are divided into five Modules - Entrepreneur Profile, Identifying Business Opportunities, Market Analysis, Products and Services Development, and Financial Analysis - that are presented with hypermedia (text and figures). Additional resources include interactive games, activities, exercises, and hints. In each class an instructor is in charge of answering students' questions, and monitoring both the students' progress through the contents and participation in the learning activities. The learning environment allows the student to access to the contents, communication and collaboration tools (e.g., discussion list and chat), technical support, digital library, FAQ, and help. Also, there is an illustrative character that accompanies the student since his or her first interaction with the environment. The course learning management system handles other users: instructors, administrators, and the technical support team.

Besides presenting characteristics and tools common to most distance education courses, this particular course presents a set of features that allow making use of individual user and usage data to provide adaptation: the large database of the learning management system; the large number of students enrolled; the great heterogeneity in the students' background; and the availability of data on the interaction of the students with the system. Using this data SEDHI was designed and implemented to classify students according to pre-defined profiles. Based on this classification, the system adapts the navigation in the course contents using the techniques of link hiding and link annotation.

3.1 Architecture

Figure 1 shows SEDHI's architecture. The system main components (4, 5 and 7 in Figure 1) are detailed afterwards.

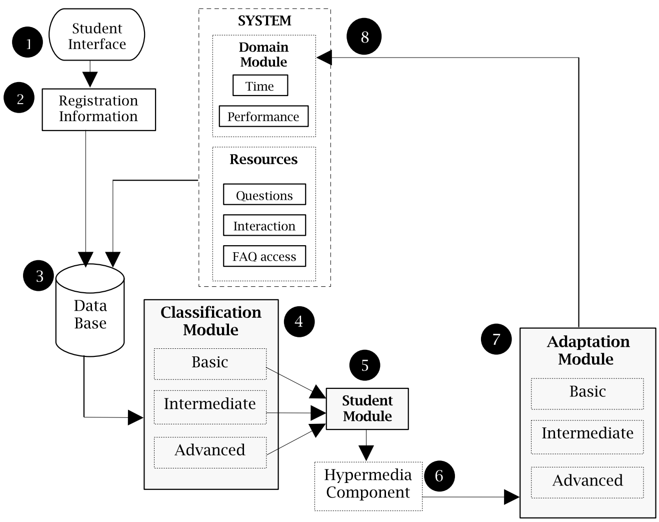


Fig. 1. SEDHI Architecture

3.2 Classification Module

Instance-Based Learning [13] and the 1-Nearest Neighbour (NN) method are used for classifying the students' profiles. The Euclidean distance, indicated to numeric attributes, is used as a measure of dissimilarity.

The Classification Module considers a set of attributes that were chosen based on the course design, environment available resources, and availability of data in the learning management system. These attributes are listed below:

1. **Time:** spent on navigation: corresponds to the time that the student spent on navigating through the Modules of the course contents;
2. **Questions:** asked to the tutor: possibly indicate that a student who asks many questions needs support;

3. **Interaction:** refers to number of contributions to the discussion list about topics of the course contents. A high number of contributions possibly indicate that the student understands the contents;
4. **FAQ access:** a high number of accesses to Frequently Asked Questions (FAQ) indicates that the student needs support;
5. **Performance:** on the exercises: refers to the student score on the exercises included at the end of each Module; and
6. **Experience:** on other web courses: indicates if the student is familiar with web browsing, hypertext, and resources usually available in distance courses.

Based on a statistical analysis of the user and usage data of a 200 students sample, the attributes values were used to define three classes. Regarding the **Time** attribute, the values in the sample were discretized and normalized in each Module of the course contents. The mean and standard deviation were calculated to define the limits of three time ranges: Short, Medium, and Long. The shortest time, the mean minus the standard deviation, the mean plus the standard deviation, and the longest time were considered the limits of the time ranges. Table 1 shows an example of the **Time** attribute: variables, corresponding ranges, and respective values for Module 3.

Table 1. Time spent in Module 3

Year	Time in Module 3 (minutes)	Value
Short	$T\_M3 < 19$	1
Medium	$19 \leq T\_M3 < 97$	2
Long	$T\_M3 \geq 97$	3

The values of the remaining attributes (**Questions**, **Interaction**, **FAQ access**, **Performance**, and **Experience**) were just normalized as they are discrete values. The mean and standard deviation were also used to define three ranges: Low, Medium, and High. Table 2 shows an example of the **FAQ access** attribute: variables, corresponding ranges, and respective values.

Table 3 presents the classifier obtained: attributes, respective values, and corresponding profiles.

The similarity between the current state of a particular student and the three examples of the classifier - Basic, Intermediate, and Advanced - is calculated by

Table 2. Number of accesses to FAQ

Year	FAQ access	Value
Low	$A\_FAQ < 4$	3
Medium	$4 \leq A\_FAQ < 26$	2
High	$A\_FAQ \geq 26$	1

Table 3. Classifier

ATTRIBUTES-VALUES						
Time	Questions	Interaction	FAQ access	Performance	Experience	**PROFILES**
3	3	3	3	3	3	Basic
2	2	2	2	2	2	Intermediate
1	1	1	1	1	1	Advanced

Table 4. Similarity calculation

Distance_Basic $d(x, b)$	Distance_Intermediate $d(x, i)$	Distance_Advanced $d(x, a)$
$d(x, b) = \sqrt{\sum_{r-1}^{n} (x_r - b_r)^2}$	$d(x, i) = \sqrt{\sum_{r-1}^{n} (x_r - i_r)^2}$	$d(x, a) = \sqrt{\sum_{r-1}^{n} (x_r - a_r)^2}$
where,	where,	where,
d = distance between x and b	d = distance between x and i	d = distance between x and a
x = student current state	x = student current state	x = student current state
b = example of basic profile	i = example of intermediate profile	a = example of advanced profile
r = individual attribute of x and b	r = individual attribute of x and i	r = individual attribute of x and a
n = number of attributes	n = number of attributes	n = number of attributes

means of the 1-NN method. The student is classified according to the shortest distance to one of the profiles defined (see Table 4): $d(x, b)$, $d(x, i)$ and $d(x, a)$.

3.3 Student Module

When the student logs on the system for the first time, the Student Model is initialized with the information provided by the student on registration and, regarding the remaining attributes, with the values of the Basic profile. From the first interaction on, as the student advances in the contents, the Classification Module is executed and the Student Module is updated, registering the student current status as Basic, Intermediate, or Advanced.

3.4 Adaptation Module

An AHS needs an internal representation of the user profile in order to adapt the navigation. In SEDHI, to each advance of the student in the contents, the Hypermedia Component informs the Adaptation Module about the student profile. The Adaptation Module adapts the content navigation according to the profile and to the Module that the student is currently working on.

Among the adaptive navigation techniques, the course instructional designers considered link hiding and link annotation [2] appropriate, according to the course design and environment available resources.

Link hiding consists of showing only the links that are important. In educational hypermedia it is usual to hide the links that are not ready to be learned. The instructional designers defined which links should be hidden in each of the three profiles according to the course design. The environment additional resources - *Glossary*, *Teacher Hints*, and *Deep Dive* - were used to implement link hiding. *Glossary* contains information about the meaning of particular words; *Teacher Hints* complement the contents with hints from an expert; and *Deep Dive* deepens the explanation about certain topics of the contents.

As all the available resources should be provided to the student classified as Basic, all the links and resources as originally modelled are shown. In the Intermediate profile, *Glossary* is disabled and *Teacher Hints* and *Deep Dive* are enabled. In the Advanced profile all additional resources are disabled, as it is considered that the student is familiar with the subject and does not need additional information to understand the contents.

The technique of link annotation has the objective to annotate a set of links in order to conduct the student from the current node to related nodes according to the profile contained in the Student Module. Educational hypermedia that use adaptive link annotation usually distinguish three knowledge levels of the user about a particular node: not-known, in study, and learned. In SEDHI, as the Student Model is updated, the adaptation of navigation is divided in three different levels of knowledge, according to the three profiles. As these levels are visually annotated, they can help the student in the content navigation. The three knowledge levels were defined according to the course domain experts, taking into account its contents and the level of difficulty of the 200 students of the sample analysed. According to the course final evaluation, 74.03% of the students considered Module 5 the most difficult and hard to learn. Module 4 is pointed out as the second more difficult by 14.29% of the students. Likewise, 7.79% of the students had problems in Module 3; 3.09% in Module 2; and no students had problems in Module 1. In the sample analysed, 65.50% of the students finished the course before the established time limit. Link annotation in SEDHI uses different colours [5], combined with textual information on the current state of the node [3], as described below.

- Basic Level: the initial links of the contents are annotated with the orange colour and with the textual information *ready-to-learn*. The remaining links are annotated with the red colour and with the textual information *not-ready-to-learn*. As the student progresses, the access to the contents that follows is liberated, and the links are annotated with the orange colour and with the textual information *ready-to-learn*. The links already studied are annotated with the green colour and with the *learned* indication.
- Intermediate Level: the links of Modules 1, 2, and 3 are annotated with the orange colour and with the textual information *ready-to-learn*. These contents can be accessed whether or not the student accessed the previous links. The first link of Module 4 is annotated with the orange colour and with the textual information *ready-to-learn*. The remaining links are annotated with the red colour and with the textual information *not-ready-to-learn*. All links in Module 5 are in red and annotated with the information *not-ready-to-learn*. The student can access this Module only when he or she concludes Module 4 or is classified as *Advanced*. For links that are annotated in red,

the system liberates the access (annotating them with the orange colour and with the textual information *ready-to-learn*) only if the student has accessed the previous links. As the student moves forward, the links already accessed are annotated with the green colour and with the *learned* indication.

– Advanced Level: the links of Modules 1, 2, and 3 are annotated with the orange colour and with the textual information *ready-to-learn*. These contents can be accessed whether or not the student accessed the previous links. The first link of Module 5 is annotated with the orange colour and with the textual information *ready-to-learn*. The remaining links are annotated with the red colour and with the textual information *not-ready-to-learn*. As the student moves forward, the links that follow are annotated with the orange colour and with the textual information *ready-to-learn* and the links already accessed are annotated with the green colour and with the *learned* indication.

4 Evaluation

Five students took part in a preliminary evaluation of the SEDHI prototype. The evaluation objective was to test whether the classification was correct and the system adaptive behaviour was appropriate. The students had a limited time to navigate in the course environment and contents. The time limit was

Table 5. Attributes and values

ATTRIBUTES-VALUES						
Time	Questions	Interaction	FAQ access	Performance	Experience	**PROFILES**
61m 02s	6	2	30	0	0	Basic
06m 27s	1	2	1	0	1	Intermediate
13m 05s	1	4	1	1	4	Advanced

Table 6. Attributes and normalized values

ATTRIBUTES-VALUES						
Time	Questions	Interaction	FAQ access	Performance	Experience	**PROFILES**
2	3	2	3	3	3	Basic
1	1	2	1	3	2	Intermediate
1	1	1	1	3	1	Advanced

```
SimBasico sqrt ( 2^2  + 2^2  + 1^2  + 2^2  + 0^2  + 1^2  )  = 3.7416573867739
SimIntermediário  sqrt ( 1^2  + 1^2  + 0^2  + 1^2  + 1^2  + 0^2  )  = 2
SimAvancedo  sqrt ( 0^2  + 0^2  + 1^2  + 0^2  + 2^2  + 1^2  )  = 2.4494897427832
2
```

Fig. 2. Classification module: Intermediate profile

considered as the end of the course. The user and usage data were discretized (when applicable) and normalized. The results of the classification of three out of the five students that participated in the evaluation (classified as Basic,

Fig. 3. Link hiding: Intermediate profile

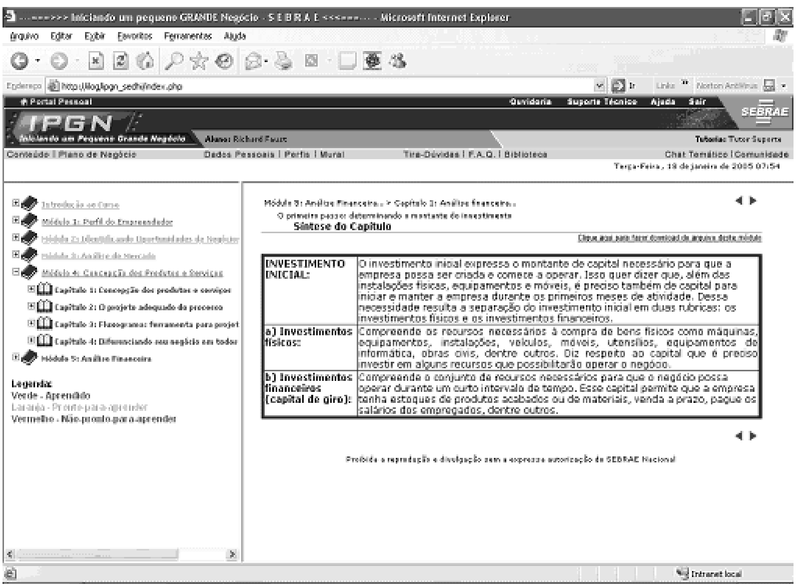


Fig. 4. Link annotation: Intermediate level

Intermediate, and Advanced) in Tables 5 and 6. Table 5 shows the variables and values obtained; Table 6 shows the variables and values after normalization.

Figures 2 to 4 illustrate the results obtained for a student that was classified as Intermediate. At the end of the course the student was in Module 5. Figure 2 shows the Classification Module with the attributes values and the similarity calculation results (value 2). Figure 3 shows the adaptation provided using link hiding. According to the Intermediate profile the *Glossary* link is hidden (e.g., the word that is encircled in Figure 3) and the links of *Teacher Hints* and *Deep Dive* are shown. Figure 4 illustrates the adaptation provided using link annotation, showing the user interface state at the end of the course. The links that were accessed are in green with the *learned* indication (Modules 1 and 4), the links in orange indicate *ready-to-learn* (Modules 2 and 3), and the links in red indicate *not-ready-to-learn* (subtopics of Modules 4 and 5).

4.1 Discussion

The classifier and the state of the user interface demonstrate that adaptation was carried out as expected from the point of view of implementation and as indicated by the course instructional designers. However, it is necessary to evaluate the effects of the adopted adaptation approach on learning outcomes. Parameters like the course conclusion, the completion of the Business Plan, and the feedback questionnaire that usually is filled out by the students at the end of the course can be used to evaluate this approach in new classes against the data of previous classes.

5 Conclusion and Further Work

This paper presented SEDHI - an adaptive hypermedia system for web-based distance courses that classifies the users and provides adaptation to navigation according to their profiles. The evaluation of the prototype allowed verifying the appropriateness of the system classifier and adaptation techniques. The following steps include evaluating the impact of the system adaptation approach on the students' performance. Nevertheless, the evaluation results show the potential of data mining techniques towards a more individualized adaptation in the context of web-based courses attended by a large number and broad range of students. In particular, using alternative data mining techniques like clustering would allow a refinement in the categories of student profiles.

References

1. Brusilovsky, P.: Methods and Techniques of Adaptive Hypermedia. User Modeling and User Adapted Interaction **6**:2-3 (1996) 2–129
2. Brusilovsky, P.: Adaptive Hypermedia. User Modeling and User-Adapted Interaction **11**:1-4 (2001) 87–110
3. Brusilovsky P.; Pesin L.: ISIS-Tutor: An Adaptive Hypertext Learning Environment. In: Proc. JCKBSE'94. Pereslavl-Zalesski, May (1994) 83-87

4. Brusilovsky, P., Schwarz, E., Weber, G.: A Tool for Developing Adaptive Electronic Textbooks on WWW. In: Proc. of WebNet'96. San Francisco (1996) 64-69
5. Brusilovsky, P., Schwarz, E., Weber, G.: ELM-ART: An Intelligent Tutoring System on World Wide Web. In: Frasson, C., Gauthier,G., Lesgold, A. (eds.): Intelligent Tutoring Systems. LNCS, Vol. 1086. Springer-Verlag, Berlin (1996) 261-269
6. De Bra, P.: Teaching Hypertext and Hypermedia Through the Web. In: Proc. of WebNet'96. San Francisco (1996) 130-135
7. Dunham, M.H.: Data Mining: Introductory and Advanced Topics. Pearson, New Jersey (2003)
8. Gaudioso, E., Boticario, J.G.: User Data Management and Usage Model Acquisition in an Adaptive Educational Collaborative Environment. In: P. De Bra, P. Brusilovsky, R. Conejo (eds.): Adaptive Hypermedia and Adaptive Web-based Systems. LNCS, Vol. 2347. Springer Verlag, Berlin (200)143-152
9. Gaudioso E., Boticario, J.G.: Towards Web-based Adaptive Learning Communities. In: U. Hoppe, F. Verdejo, J. Kay (eds.): Artificial Intelligence in Education. IOS Press, Amsterdam (2003) 237-244
10. Henze, N., Nejdl, W.: Adaptivity in the KBS Hyperbook System. In: 2nd Workshop on Adaptive Systems and User Modeling on the WWW. Toronto (1999)
11. Merceron, A., Yacef, K.: Educational Data Mining: A Case Study. In: C. Looi, G. McCalla, B. Bredeweg, J. Breuker (eds.): Artificial Intelligence in Education. IOS Press, Amsterdam (2005) 467-482
12. Merceron, A., Yacef, K. A.: Web-Based Tutoring Tool with Mining Facilities to Improve Learning and Teaching. In: U. Hoppe, F. Verdejo, J. Kay (eds.): Artificial Intelligence in Education. IOS Press, Amsterdam (2003) 201-208.
13. Mitchell, T.M. Machine Learning. McGraw-Hill, Boston (1997)
14. Raad, H., Causse, B.: Modeling of an Adaptive Hypermedia System based on Active Rules. In: S. Cerri, G. Gouarderes, F. Paraguau (eds.): Intelligent Tutoring Systems. LNCS, Vol. 2363. Springer Verlag, Berlin (2002) 149-157

Combining ITS and eLearning Technologies: Opportunities and Challenges

Christopher Brooks[1], Jim Greer[1], Erica Melis[2], and Carsten Ullrich[2]

[1] Advanced Research in Intelligent Educational Systems (ARIES)
University of Saskatchewan, Saskatoon, SK, Canada
cab938@mail.usask.ca, greer@cs.usask.ca

[2] German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
melis@dfki.de, cullrich@dfki.de

Abstract. The development of Intelligent Tutoring Systems (ITS) and eLearning systems has been progressing largely independently over the past several years. Both types of systems have strengths and weaknesses – ITSs are typically domain specific and rely on concise knowledge modeling and learner modeling, while eLearning systems are deployable in a wide range of circumstances and focus on connecting learners both to content and to one another. This paper provides possibilities for convergence of these two areas, and describes two of our experiences in providing an ITS-style approach to eLearning systems.

1 Introduction

The term eLearning brings to mind several core concepts; learning activities supported by Web-technologies including learning management systems (LMS) such as WebCT, Moodle, etc., conferencing and discussion systems, and rich multi-media content. ELearning applications fall into a broad qualitative spectrum, and critics have attacked these products for having a lack of pedagogical and psychological validity, as well as an absence of controlled evaluations.

By comparison, Intelligent Tutoring Systems (ITS) have mostly been focusing on supporting and scaffolding of problem solving in learning. Typically they have been built on specialized, rich knowledge representations, and use cognitive diagnosis and user modeling techniques to respond to the needs of the learners. Until quite recently, employing Web-technologies has not been a major focus of ITS. Only a few ITSs use technologies such as adaptive hypermedia [1] or metadata and knowledge management [2, 3].

This suggests that there should be gains from integration and collaboration between the two communities. Despite this, the cultural differences between the two communities has led to little cross-fertilization of ideas and technologies. This paper discusses this topic as follows: Section 2 examines the differences between common ITS and eLearning environments. This is followed in Section 3 by a discussion of our experiences with two different systems that aim to fill the gap between these communities, ActiveMath [4] and iHelp Courses [5].

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 278–287, 2006.

Section 4 concludes the work by identifying specific challenges that exist, and potential ways to address these challenges.

2 Differences

ITS has grown out of artificial intelligence, cognitive psychology, and education and has typically focused on the creation of specialized research systems which are domain dependant and mostly aimed at school education. As the area has been mostly one driven by research, implementations tend to be unique in the features they provide, contain hand-crafted ontologies developed by a small group of developers, and lack interoperability between one another. ELearning systems, on the other hand, are a mostly technology-driven enterprise so far mostly worked on by institutions aimed at higher education and workplace training. This community tends to be more risk-adverse, and its motivating factors are interoperability through standardization (for instance, the IMS specifications, see below) and wide-scale deployment. Thus the thrust of traditional eLearning research is the issue of reuse, interoperability of components, integration with organizational software, and authoring of content. Table 1 provides a coarse distinction of the main features of ITS and eLearning systems. It must be noted that many instances of exceptions to this classification are beginning to emerge as efforts (such as ours) are initiated to reduce the boundaries between ITS and eLearning.

Table 1. Features of typical ITS and eLearning applications

ITS	eLearning
aiming at improved learning	organizing learning & presenting material
restricted content	massive content
carefully crafted content	content crafted by normal authors
single author/designer	potentially collaborative authoring
fix abstract domain ontology	several ontologies, content-based
elaborate feedback	simple feedback
some feedback generated	pre-scripted feedback
tightly integrated components	service approach
few generalizable solutions	scalability and reuse important

2.1 Technologies in eLearning

LMSs contain functions for managing authors, instructors, administrators, and learners in courses (e.g., roles, passwords, etc.), connecting learners together (e.g., by discussion forums and chat systems), and providing and managing access to content (e.g., access rules, quizzes, etc.). These systems generally offer a very simple level of monitoring and feedback mechanisms – instructors can usually see only a coarse-grained view of what content students have accessed (or discussions students have engaged in) and students typically can obtain pre-scripted simple feedback or limited branching to alternate content resulting from instructor-created quizzes.

These systems use the Web and browsers for delivery and allow for learning which is independent of time, place, and pace. Both hypermedia and multimedia are used to help motivate learners, though it has been suggested that this may neither last nor result in deep learning as the grip of the new media evaporates. Still, psychological research suggests some value of multimedia for attracting attention and for grasping complex information through multiple sensory channels. In addition to the rich content provided by these systems, there is a strong potential to leverage Web-technologies for personalization and adaptation, and there is a growing awareness for its importance to eLearning (see, e.g., [6]).

Standardization of learning objects hold the promise to make reusability of learning material easier. The standards are aimed at solving a number of problems including the description of technical, administrative, and pedagogical aspects of content (e.g., IEEE LOM [7]), the interconnections between content and learning actors (e.g., IMS Learning Design [8]), the aggregation and ordering of content for deployment (e.g., IMS Content Packaging [9]), and how content should be sequenced for the the learner (e.g., IMS Simple Sequencing [10]).

The standardization process has been largely influenced and governed by commercial interests and tries to be completely comprehensive, which makes the standards simultaneously large and cumbersome yet failing to include specific needs. In particular, the metadata as well as much of the technology usage have not yet been targeting deep learning and are not much informed by empirical psychological results [11].

In addition to standards-compliant learning materials, a number of other Web technologies hold great promise. For instance, the use of XML, XSL, and the Resource Description Framework (RDF) serve the separation of structure, presentation, and semantics and they provide a rich and extensible layer.

Web services are clearly becoming the choice for system-to-system integration and could help specialized ITS components attain a higher level of interoperability. Web services typically require developers to provide strict definitions of the functionalities that can be requested from a stand-alone application living on its server. This is typically done in a blackbox manner, where a wrapper around domain tools (e.g., an equation solver or a computer algebra system) is generated and exposed to the world. Unlike the typical glassbox ITS system (such as the physics problem solver in Andes [12]), blackboxes are often more difficult to use when the goal is to generate feedback based on human problem solving spaces. For instance, computer algebra systems compute solutions in steps and by algorithms that typically are different from human problem solving behavior.

The lack of adaptivity to individual learners is the main shortcoming of traditional eLearning approaches. Customizing feedback or limiting learner options is based on fairly superficial knowledge, typically the answer given to a question in a quiz. Guidance for learners must be completely scripted by authors with no run-time inference or subtle adaptation based on individuals' actions. The task of selecting content for presentation to the learner is left to authors (or to the learners themselves). More successful forms of adaptivity have been in adjusting presentation style to specific devices or to dichotomic learning styles (e.g. verbal vs. visual learners such as in the INSPIRE and 3DE projects.

Despite strong adoption of eLearning in the marketplace (including the open source world), some eLearning developers have become aware of the downfalls of current technologies. However, the idea of 'diagnosis' does not belong yet to the common ground.

2.2 Intelligent Tutoring Systems

Typically the "intelligent" in ITS refers to (1) a problem solving system that can assist and help to produce feedback and hints to learners; (2) model tracing that predicts the learner's current mastery and likely next step in order to scaffold problem solving; (3) knowledge tracing that assesses the learner's abilities and concept-mastery in order to release new exercises or topics to learn; and (4) tutorial dialogues for scaffolding problem solving. Certainly, the literature reveals many more ideas that have been proposed in ITS-research such as tools for inquiry learning and for collaborative learning.

ITS-research has a long record of student modeling, of appropriate responses to students' problem solving activities, of collaborative learning techniques. It offers a range of techniques for macro- and micro-adaptation [13] which adapt both *what* is presented to the learner and *how* it is presented. Many ITSs realize (pedagogical) ideas and technologies that are informed by empirical results from cognitive and pedagogical psychology, e.g. on cognitive models, self-explanation, or the zone of proximal development. Moreover, controlled experiments belong to the arsenal of methods practiced in the ITS-community.

3 Filling the Gap

It is our claim that eLearning can be made more intelligent and ITS more open and reusable while preserving their useful existing features. In particular, Web-technologies can be employed to enhance adaptivity technologically, to reuse interoperable components, and to make systems more widely available and maintainable. ITS-techniques can be used to make adaptation truly beneficial for learning, to provide student modeling, tutorial dialogues and other useful ideas and tools developed over years.

As eLearning goes through an explosion of adoption, more and more systems are integrating ITS research into traditional eLearning environments. This section describes two deployed systems which seek to integrate some level of adaptation and personalization while recognizing the specifications and standards of the broader eLearning community.

3.1 iHelp Courses

The LORNET network investigates ontologies, artificial intelligence, data mining, and multi-media in an eLearning project. We are investigating how adaptive systems can embed intelligence into standardized learning objects, as well as connect users in real time and asynchronous virtual communities. iHelp Courses [5] is one project within LORNET in which a web-based and learning object-based content management system provides some learner personalization while maintaining standards compliance. It requires instructors to provide aggregations of content using the IMS Content Packages. Content can be in any format renderable by a browser (including Flash, Java Applets, etc.). The flexibility of the format allows an instructor to take existing learning materials and readily integrate them into a new course. Further, by supporting eLearning standards for content aggregation, instructors gain the benefits of traditional content management systems (i.e. portability of content).

Personalization is achieved through the use of a fine grained role structure. Learners are assigned to an arbitrary number of hierarchically arranged roles which govern the path(s) that they can take through the learning material. Instructors associate rules that determine the availability of a piece of content with content/role tuples. These rules follow the spirit of the IMS Simple Sequencing specification and can access both the structure of content, the knowledge-tracing model of the user, and various system functions. As a user can be put into an arbitrary number of roles, rules are compiled into Java Bytecode, loaded on-demand, and executed to provide high performance.

Rule functions outside of the scope of simple sequencing can be easily added to iHelp Courses, and provide increased user modeling functionality. Data we collect includes dwell time on each learning object, the path taken to get to a particular point, and a history of the responses the learner has provided. This modeling is deeper than the user tracing laid out by eLearning specifications, and integrates with knowledge representation and reasoning efforts in ITS and Semantic Web research.

iHelp Courses has been deployed at the University of Saskatchewan, and has been used with over 200 students in distance eLearning courses and over 2000 students in blended learning environments.

3.2 ActiveMath

The EU-project Language-Enhanced, User-Adaptive, Interactive eLearning for Mathematics (LeActiveMath) is carried out by 8 European partners from research institutes, universities and an educational publisher. It combines expertise in Web technologies, knowledge representation and services, in ITS including user modeling, in mathematics teaching and competency-based pedagogy, and in computational linguistics. LeActiveMath builds upon ActiveMath, a multilingual, Web-based, adaptive learning environment for mathematics. It combines several components and services in one application using a distributed architecture based on XML-RPC as well as an asynchronous event framework [14]. It employs computer algebra services and domain reasoner for generating feedback and assessment for exercises and develops tutorial dialogues. LeActiveMath provides advanced tools such as an open learner model, semantic search, interactive concept mapping, and assembling tool. LeActiveMath employs a semantic

XML-representation for mathematics, OMDoc [2], compliant with LOM. A presentation engine transforms the content into a variety of output formats (e.g., HTML, MathML, and PDF). ActiveMath realizes personalization through instructional planning [15]. For instance, the content may vary for different learning contexts and in detail and difficulty. ActiveMath adapts and personalizes content by assembling learning objects according to prerequisite and other relations and by modifying this according to preferences, learning scenario, goal-level and ability.

On the one hand, LeActiveMath relies on Web-fueled knowledge representation and Web-technologies such as XSLT, Web-server, brokerage of services, servlets, JSP, and Velocity templates. On the other hand, it also realizes typical AI-techniques such as student modeling, adaptive hypermedia, adaptive course generation, blackboard suggestion mechanism, feedback in problem solving based on back-engine computation and reasoning.

LeActiveMath is tested at different European universities and in German high schools in several small- and large scale evaluations.

4 Challenges and Opportunities

Opportunities are opened up by the growing quantity of learning material that has been tagged and annotated using standards and that is available via the Web. Intelligent tutoring systems can benefit from employing an extensible and reusable knowledge representation scheme that is accessible for other systems as well. This includes the formats but also the concrete metadata that characterize an instructional item or learning object. The Web as a knowledge base could and should be employed in inquiry learning, especially if strong (semantic) search facilities can be provided. This is how many students now learn anyway.

Although our work aims at filling the gap between ITSs and eLearning systems, it is clear that a number of challenges still exist.

Extensibility. The majority of (if not all) eLearning specifications use XML to aid interoperability. In theory, this allows vendors to extend specifications in a conforming fashion by introducing new elements and attributes within their own namespaces. In practice however, vendor extensions to XML documents tend to break when imported into other eLearning applications. This is actually a side effect of using XML in a structural fashion – the relationship of one element to another is based on a hierarchy called the Document Object Model (DOM). Introducing new elements to "wrap" existing elements reorders the hierarchy exposed by the DOM, and thus typical structural querying languages (such as XPath) tend to have problems. While this can be avoided (both in the design of the XML Schema, and in the implementation of XPath queries), the fact remains that vendor-based extensibility is quite low. The shift to more semantic-based representations (in particular semantic web technologies, such as RDF and OpenMath) increases extensibility without hampering the gains by interoperability. ActiveMath, for instance, uses an extensible implementation of OpenMath.

A further development of the XML-structures and of the metadata for instructional items (such as worked examples and exercises) is needed with the objective to include feedback-relevant data for problem solving activities as well as structures and metadata for the generation of tailored exercises as described in more detail in e.g. [16]. For instance, substructures in examples and exercises have to be identified and annotated with metadata in order to make them first-class citizens which can be diagnosed, faded and replaced adaptively, etc. In ActiveMath, we created such structures which carry more knowledge [16] and submitted them to standardization by IMS Math QTI.

Domain Ontologies. Domain ontologies have to be constructed and maintained. In a realistic setting which is that of collaborative authoring, this may be difficult: several ontologies may exist and these ontologies are in constant evolution and are gradually refined in collaboration with the domain experts and through experimentation with learners. Mapping is important in order to enable the reuse of domain-specific material that will be more and more available on the Web. In our experience (see [17]) and that of other ontology-builders, too, large content ontologies (e.g. 1,000 concept+) are extremely difficult to keep up to date and to maintain manually, hence automatic tools have to be developed and used. One avenue we follow in ActiveMath is the creation of domain maps from content and feeding the domain map back to the content authors for quality control and consistency checking. This way, we can keep relevant ontology information from the content (similar to T-Box and A-Box approach in description logics).

The management of ontologies and detection of inconsistencies is critical. ITS might contribute experiences from the development of ontologies from an instructional point of view, as well as from a technical point of view. The automatic creation and management of those - maybe multiple - domain ontologies poses even bigger challenges. Visualization alone is not sufficient here but quality management is needed that includes consistency checks and version control that builds on dependencies and semantics. Formal methods can contribute to discovering inconsistencies and version maintenance.

Contextualized Metadata. With the goal of reusing Learning Objects it is sometimes impossible to assign a single one-dimensional value for metadata. Consider, for example, the *difficulty* value of a learning object. Any exercise or example will be more difficult for an elementary school student than for a university student. Yet such simple contextual inferences may not be possible with current standards, where difficulty has a single qualitative value. One possible solution is the contextualization of metadata (providing multi-dimensional metadata) as currently done in ActiveMath. Such a need did not occur previously in ITSs adapting the difficulty of exercises or examples to the capability of the student because these ITSs are normally narrowly focused for a particular audience only.

Using the previous example, one can also imagine definitions relative to various kinds of assessment contexts (e.g., an author assesses the difficulty, a teacher, a student, statistical assessment of difficulty, etc.). While some work has gone into solving these problems within iHelp Courses (see [18]), validation of this approach is required. It is important to note that informal human-readable metadata, such as instructional designer reviews and opinions about a learning object and are rarely available in standardized form and therefore, they are essentially useless to the content management systems.

Creating more diverse sets of metadata for different purposes leads to another problem - who owns the metadata? Where should it be stored? Traditional eLearning standards store metadata along with a learning object, e.g., in a Content Package. Once multiple pieces of metadata can be created for a single piece of content one can envision distributed systems being employed to store data close to the creators. This opens another Pandora's box: collaborative authoring. With distributed databases the search becomes more expensive (which is bad for dynamic generation) and even worse, the maintenance of those knowledge bases becomes much more difficult. Here, again, quality maintenance must include semantic consistency checks.

Validation of Metadata. Misleading and incomplete metadata are rampant in actual deployed learning object systems, in particular learning object repositories [19]. ITSs typically specify explicitly how accurate a given piece of knowledge is (often through probabilistic models, such as Bayes nets). Hybridizing these approaches and providing a probabilistic overlay on contextualized metadata (both author generated and observed within the learning environment) is the next milestone for iHelp Courses.

Dynamic Content and Sequences. While IMS Simple Sequencing (IMS-SS) is appropriate for representing a completely generated course, it is still insufficient for true interactivity and reactivity that goes beyond the programmed instruction. In IMS-SS reactivity is defined only in terms of interaction of learner with content rather than including other dynamic aspects of the learner such as 'field of study'. Moreover, currently IMS-SS is not suited for adding or deleting content dynamically. Therefore, IMS Simple Sequencing should be extended to allow for more informed instructional planning. Ideally this should include the ability to re-plan in an ad-hoc manner based on changes in the learner model or learning environment, as well as be usable within different kinds of planning environments (classic, constraint-based, probabilistic, etc.) [20].

Distribution of Services. Currently most eLearning and ITS systems follow a centralized architecture. With the inclusion of ITS technologies, such as domain-specific reasoning engines (e.g. an algebraic problem solver) or specialized user modeling components, a distributed architecture must be considered.

Web-services seem to provide the correct amount of flexibility and generality to fit this architectural need. Brusilovsky [21] for instance, provides a description of how Web-services can be used to create a centralized user modeling server. Brooks et al. [22] provide a semantic Web-based architecture that can be used to collect and distribute user modeling information in eLearning environments. In ActiveMath with its central student model and action history, events are passed asynchronously and requests to/from services exchanged [14].

New Models of (Life-Long) Learning. Both, ITS and eLearning could develop learning scenarios that rely on self-organization models with little control for certain phases in learning. Such models are typical for the Web and have the potential to engage and motivate certain learners. Feedback and encouragement is useful but too much control can hinder self-guidance and construction of knowledge particularly for adult learners.

5 Conclusion

The aim of our research is to provide working systems that increase the efficiency of teaching and at the same time are effective for learning and pedagogically as well as cognitively sound. Therefore, they have to take advantage of ITS and Web-technologies. We – very briefly – described how this integration is pursued in ActiveMath and iHelp Courses.

Acknowledgement

This publication is partly a result of work in the context of the LeActiveMath project, funded under the 6th Framework Program of the European Community – (Contract IST-2003-507826). The authors are solely responsible for its content. The iHelp Courses project was funded in part by the University of Saskatchewan and the Government of Saskatchewan through the Technology Enhanced Learning program, as well as the Government of Canada through the LORNET NSERC research network.

References

1. Brusilovsky, P.: Adaptive hypermedia. User Modeling and User Adapted Interaction **11**(1/2) (2001) 87–110
2. Melis, E., Andrès, J.B.E., Frischauf, A., Goguadze, G., Libbrecht, P., Pollet, M., Ullrich, C.: Knowledge Representation and Management in ActiveMath. Annals of Mathematics and Artificial Intelligence **38**(1-3) (2003) 47–64.
3. Mohan, P., Greer, J.: E-learning specifications in the context of instructional planning. In Hoppe, U., Verdejo, F., Kay, J., eds.: AI in Education, AIED-2003, IOS Press (2003) 307–314
4. Melis, E., Andrès, E., Büdenbender, J., Frischauf, A., Goguadze, G., Libbrecht, P., Pollet, M., Ullrich, C.: ActiveMath: A Generic and Adaptive Web-Based Learning Environment. International Journal of Artificial Intelligence in Education **12**(4) (2001) 385–407
5. Brooks, C., Kettel, L., Hansen, C., Greer, J.: Building a learning object content management system. In: Proceedings of World Conference on E-Learning in Corporate, Healthcare, & Higher Education (E-Learn 2005), Vancouver, Canada. (2005)
6. Sampson, D., Karagiannidis, C., Kinshuk: Personalized learning: Educational, technological and standardization perspectives. Interactive Educational Multimedia (2002)
7. IEEE Learning Technology Standards Committee: 1484.12.1-2002 IEEE standard for Learning Object Metadata (2002)

8. IMS Global Learning Consortium: IMS learning design specification (2003)
9. IMS Global Learning Consortium: IMS content packaging information model (2003)
10. IMS Global Learning Consortium: IMS simple sequencing specification (2003)
11. und Manfred Schweres, K.U.L.: E-Learning braucht Kontinuität. Mehr nicht? Telepolis (2004)
12. Martin, J.D., VanLehn, K.: OLAE: Progress toward a multi-activity, bayesian student modeler. In Brna, P., Ohlsson, S., Pain, H., eds.: Proceedings of the World Conference on Artificial Intelligence in Education, Edinburgh, Scotland, AACE (1993) 410–417
13. Shute, V.: SMART: Student modeling approach for responsive tutoring. User Modeling and User-Adapted Interaction **5** (1995) 1–44
14. Melis, E., Goguadze, G., Homik, M., Libbrecht, P., Ullrich, C., Winterstein, S.: Semantic-Aware Components and Services of ActiveMath. British Journal of Educational Technology **37**(3) (2006) 405–423
15. Ullrich, C.: Course Generation Based on HTN Planning. In Jedlitschka, A., Brandherm, B., eds.: Proceedings of 13th Annual Workshop of the SIG Adaptivity and User Modeling in Interactive Systems, Saarbrücken, Germany (2005) 74–79
16. Goguadze, G., Palomo, A.G., Melis, E.: Interactivity of Exercises in ActiveMath. In: In Proceedings of the 13th International Conference on Computers in Education (ICCE 2005), Singapore (2005) 107–113
17. Winter, M., Brooks, C., Greer, J.: Towards best practices for semantic web student modeling. In Looi, C.K., McCalla, G., eds.: Proceedings of the 12th International Conference on Artificial Intelligence in Education AIED-2005, Amsterdam, The Netherlands, IOS Press (2005)
18. Brooks, C., McCalla, G.: Flexible learning object metadata. International Journal of Continuing Engineering Education and Life-Long Learning (IJCEELL) (2006) to appear.
19. Friesen, N.: Final report on the international lom survey. Technical Report 36C087, Canadian Avisory Commiittee for ISO/IEC JTC1/SC36 (2004)
20. Ullrich, C.: Tutorial Planning: Adapting Course Generation to Today's Needs. In Grandbastian, M., ed.: Young Researcher Track Proceedings of 12th International Conference on Artificial Intelligence in Education, Amsterdam, The Netherlands (2005) 155–160
21. Brusilovsky, P.: A component-based distributed architecture for adaptive web-based education. In Hoppe, U., Verdejo, F., Kay, J., eds.: AI in Education, AIED-2003, IOS Press (2003) 386–388
22. Brooks, C., Winter, M., Greer, J., McCalla, G.: The massive user modeling system. In: Proceedings of The 2nd International Workshop on Designing Computational Models of Collaborative Learning Interaction in conjunction with ITS 2004, Maceio, Brasil (2004)

From Learner Information Packages to Student Models: Which Continuum?

Lahcen Oubahssi[1] and Monique Grandbastien[2]

[1] Université René Descartes Laboratoire CRIP5 /AIDA
45 Rue Saints Pères 75270 PARIS Cedex 06
oubahssilahcen@voila.fr
[2] Université Henri Poincaré Nancy1 Lab. LORIA/AIDA
Bât. LORIA Campus scientifique BP 239 - 54506 VANDOEUVRE Cedex France
monique.grandbastien@loria.fr

Abstract. User adaptation has been a major issue of ITS research since the beginning. At first knowledge about individual learners was used to produce adaptive educational hypermedia. Then, the deployment of web based learning management systems put the need to exchange learner data between systems in the forefront. Nowadays, specifications for describing learner information in a standardized way are being developed. So far most existing systems have processed learner data. Yet the processed data are neither the same nor are they used for the same purposes, even if they are data about the learners. We argue that it is time to bridge the gap between the many sets of data about a learner processed within different learning environments. We propose a set of categories for describing learner information that goes beyond IMS LIP and could bring a link with ITS learner models.

1 Introduction

Since computers began to be used in education and training it has been argued that one of the main benefits for learners would be the delivery of training material and experiences adapted to their personal needs. Adapting one's behavior to a user implies that data about this user are available to the system. Indeed user adaptation has been a major issue in many prototypes and the popular ITS « four module architecture » includes a learner model. Then research groups tried to use knowledge about individual learners included in learner models in order to produce adaptive educational hypermedia. Finally, the deployment of web based learning management systems put the need to exchange learner data between systems in the forefront to exchange learner data between systems. In this context specifications for describing learner information in a standardized way are currently being developed.

In this paper we argue that it is time to bridge the gap between the many sets of data about a learner processed within the many categories of learning environments that may be run. We propose a set of categories for describing learner information that goes beyond IMS LIP and could bring a link with ITS learner models.

The paper is organized as follows: After a brief introduction we review the main motivations and achievements of learner models in ITS and AH (Adaptive Hypermedia)

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 288 – 297, 2006.

research (section 2) and the set of specifications currently proposed by several consortia working for the e-learning standardization field (section 3). In section 4, we first present the SERPOLET LMS from which we draw our set of categories, then we present the model itself, as an extension of the LIP and we show how the newly introduced categories could allow a better adaptation to users' needs and especially a connection between this extended LIP model and a learner model built for an ITS or an AH system. We conclude with our views about data exchanges between learner models and learner information packages used in LMS.

2 Learner Models in ITS and AH Systems

W. Chen and R. Mizoguchi [1] proposed a learner model ontology which is worth considering to answer the question "what is to be found in a learner model?" The higher levels are learner assessment and learner model information. In the lower levels, we find information about the learner, divided into static and dynamic information. With a slightly more restrictive view, G. McCalla [2] pointed out that learner models roughly include two parts. The first part is related to characteristics (similar to "static information") that transcend any given activity of the learner. The second part called episodic part (similar to "dynamic information") includes data about the learner's experience during a given learning event. Its content may differ depending on the level of granularity of the current learning activity. The important point to keep in mind is that ITS and AH learner models are mostly about this second part. Indeed they were not built in the context of many on line learners and they aimed at a fine grained scaffolding of the learner.

In his review of 24 ITS authoring systems, Murray [3] mentioned that most student models are in fact overlay models, which means that student knowledge is related to domain concepts or procedures. Moreover, in many systems student knowledge is strongly linked to instructional expertise as student data are used for making instructional decisions such as which kind of help to provide, or which next exercise to select. However, in addition to this simple overlay view, some systems include sets of misconceptions, pre-existing knowledge, and learner preferences. Learner data are mostly derived from the analysis of learner machine interactions during the session. They are completed by pre-test results and answers to questionnaires. They range from simple historical records to sophisticated sets of fine-grained labeled concepts. Finally, even if the concept of open learner model was proposed by J. Kay [5] and others, the ITS models were mainly built to be processed by machines and not by humans. This is another key point to keep in mind.

The same techniques have been used to build learner models in Adaptive Hypermedia Systems as summarized by Brusilovsky in several papers [4]. Learner data are used to adapt subject matter presentation and to personalize navigation.

Recent approaches [2], [6] are using data mining and clustering methods in order to enrich the previously mentioned techniques for building more dynamic learner models.

To summarize the learner models used so far in ITS and AH systems are mostly based on overlay models linked to subject domain models. The data included in the models are mostly derived from the analysis of the learner's interactions with the system and they are mostly intended to allow the system to adapt its short term current behavior. From a different viewpoint, more permanent data are recorded in other contexts as described in the next section.

3 Existing Learner Metadata Models

3.1 IEEE PAPI Model

PAPI *(Public And Private Information for Learners)*[1] is a standard developed within the IEEE P1484.2 Learner Model Working Group. Its objective is to specify the semantics and the syntax of a 'Learner Model', which characterizes a learner and his knowledge. It includes elements such as knowledge (from coarse to fine-grained), skills, abilities, learning styles, records, and personal information. The standard allows these elements to be represented in multiple levels of granularity, from a coarse overview, down to the smallest conceivable sub-element. The standard allows different views of the Learner Model (learner, teacher, parent, school, employer, etc.) and will substantially address issues of privacy and security. PAPI Learner was initially developed for learning technology applications but may be easily extended to other types of human-related information such as medical and financial applications.

PAPI distinguishes personal, relations, security, preference, performance, and portfolio information. The personal category contains information about a user's name, contacts and address. The relations category serves to specify user's relationships (e.g. classmate, teacherIs, teacherOf, instructorIs, instructorOf, belongsTo, belongsWith). Security aims to provide slots for credentials and access rights. Preference indicates the types of devices and objects the user is able to recognize. Performance is for storing information about a user's measured performance through learning material (i.e. what does a user know). Portfolio is for accessing a user's previous experience [7].

PAPI represents one of the first proposals which offered a framework that organizes learner data. There are many learner data that this standard does not take into account [8], and which can be exchangeable between various e-learning systems. This explains why this proposal has been extended by IMS in its new standard IMS LIP.

3.2 IMS LIP Model

IMS Learner Information Package[2] is based on a data model that describes those characteristics of a learner needed for the general purposes of: Recording and managing learning-related history, goals, and accomplishments; Engaging a learner in a learning experience; Discovering learning opportunities for learners.

The specification supports the exchange of learner information among learning management systems, human resource systems, student information systems, enterprise e-learning systems, knowledge management systems, and other systems used in the learning process. In this paper such systems will be called learner information systems regardless of any other functionality they may possess or roles they may

[1] http://edutool.com/papi/.
[2] http://www.imsproject.org/profiles/index.cfm

fulfill. The IMS LIP specification does not address requests for learner information or exchange transaction mechanisms.

IMS LIP v1.0 defines a user data model as a set of 11 categories to be imported or exported between systems. Its categories are: Identification, Accessibility, QCL, Activity, Goal, Competency, Interest, Transcript, Affiliation, Security-key and Relationship. We note that IMS LIP provided valuable extensions compared to the PAPI model, but again it does not meet all the systems' needs in terms of learner data exchange, which explains its adaptation within application profiles.

3.3 Other Models and Conclusion

There are other proposals for a standardization of the learner data, but they not enter the objective of this contribution. Let us note in particular two standards AICC[3] and SCORM[4]. The standard SCORM offers a data model for managing all the learning productions. This model comprises a set of fields in order to allow a standardized exchange of data between a runnable training unit and the platforms.

The learner's follow-up was one of the principal concerns of AICC. In this model, the data exchanges between the learning system (CMI for Computer-Managed Instruction) and a given training module (indicated by CBT for Computer Based Training) are done via files. This approach allows the division of data between several modules constituting the training.

To conclude, we note that the two models presented above (PAPI & IMS LIP) form one of the first bases of structuring user data which can be exchanged between e-learning systems. We consider that its elements are not sufficiently complete to cover all the learner data which can be exchanged between e-learning systems. The elements of this model remain a general learner description. However, we note in the IMS LIP model a first attempt to model educational learner data. Indeed, in the category "Activity" we find the educational activities related to a learner.

Our analysis of the data in the Global Open and Distance Learning Cycle (GODLC) [9] [10] enabled us to identify other learner data which should be exchanged between e-learning systems, and which can improve the content of existing learner models.

4 A New Learner Data Model

4.1 Presentation of SERPOLET Learning Management System

To carry out this metadata model, we profited from a field experiment, which enabled us to understand the LMS users' real needs and to find the best adaptations to meet their expectations. We also benefited from a previously done reengineering work, in which we had defined a core of reusable functional modules to meet the specific needs of the SERPOLET[5] system customers. In short, the SERPOLET system originates from a research project conducted from the end of the eighties to 1995 by G.

---

[3] http://www.aicc.org
[4] http://www.adlnet.org/
[5] http://www.a6.fr

Claës. During the past decade, it was partially reshaped, several times recoded and substantially completed and it gave birth to several commercial products designed to fulfill customers' needs.

Within the A6 Company, SERPOLET acts as a kernel set of services that are completed by additional modules in order to customize learning delivery platforms to users' needs. One of the characteristics of this system is its authoring tool which offers: course management and learners' training plans, the management of the learner's educational path and the management of the learner's booklets.

In the next section, we present our proposal and highlight its new elements.

4.2 General Characteristics

Within the company, the development of this model was made to answer the questions of the learner data interoperability between several e-learning systems.

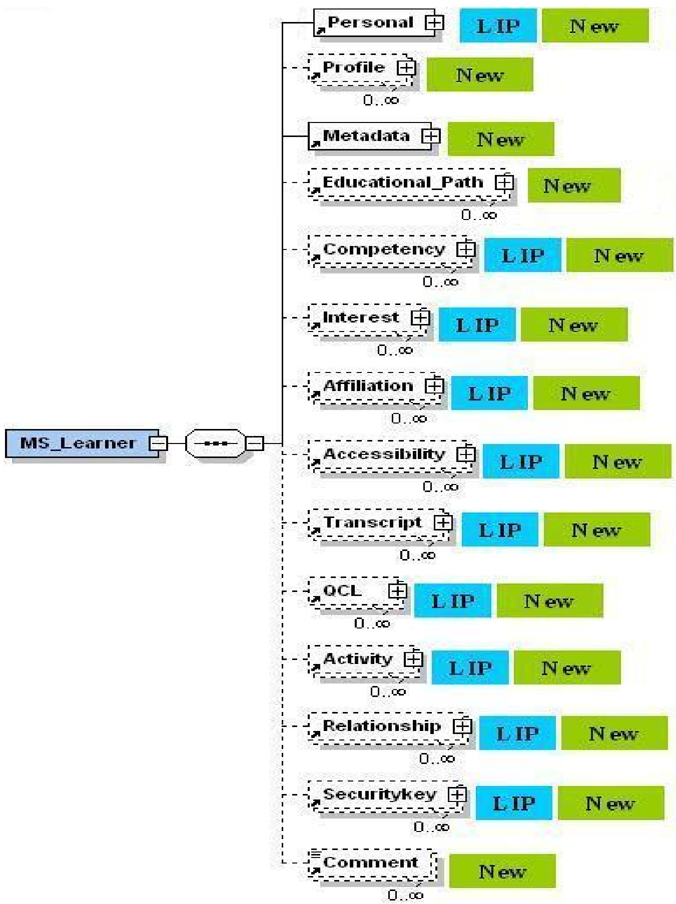


Fig. 1. SERPOLET learner data model

To elaborate this model, we first analyzed the principal existing models, in particular the models presented above (PAPI, IMS LIP) as well as the RDCEO[6] (Reusable Definition of Competency or Educational Objective) standard. We studied the elements described in the various categories included in these models.

Second, we analyzed the various learner data exploited in the SERPOLET e-learning system. Our objective was to identify the learner data candidates for interoperability between e-learning systems.

While constructing our diagram we kept in mind the following objectives: To distinguish the learner data which are specific to the system from those which can be exchanged with other external systems; To determine the level of detail which we can take into account in our model, if we want to describe all the learner data in the training life cycle; To remain LOM compliant as requested by our clients.

In this new model, we propose 14 categories in which we gather the different learner data which can be exchanged between e-learning systems. We took into account the elements of the analyzed models as well as the needs specified in the Global Open and Distance Learning Cycle. Moreover the new elements brought in this model arise from the analysis of existing data and the needs expressed for a better interoperability of learner data between e-learning systems.

Figure 1 gives an overview of the categories which constitute our data model.

The *LIP label* means that the category has been borrowed from IMS LIP without any change. *LIP label* and *New label* together mean that new fields are proposed for an existing IMS LIP category whereas *New label* alone shows a new category.

The main drawbacks mentioned about PAPI and IMS LIP are the very restricted set of data related to the learning process. In our model, we tried to develop the educational learner data; We also improved the elements of certain existing categories, in particular in the IMS LIP model.

The main innovations of our proposal concern mostly four categories: "**profile**", "**educational path**", "**metadata**", and "**comment**".

In the "**profile**" category, we define a learner profile, as a set of information gathered as the output of a learning unit. It is one of the new introduced categories which distinguishes our model from the IMS LIP model. Indeed, it is necessary to provide the teachers with a whole set of learner knowledge which enables them to define the learner's educational path. The principal elements which constitute this category are as follows: The *profile identifier*, the *profile description*, the *profile Ref-course*.

```
<Profile>
        <Identifier>Profile001-Inf12_Paris4</ Identifier>
        <Description>
                Control of the technologies of information and communication
        </Description>
        <Ref-course>Courses of computer and Internet certificate </Ref- course >
        <Comment>All the students of LMD must have this profile at the end of their
        teaching course </Comment>
</Profile>
```

Fig. 2. Example of profile

[6] http://www.registry.ed.ac.uk/transcripts/EDSGuide.htm

In the "***Metadata***" category, new also, we regroup the data that make it possible to describe learner data. It represents a fundamental element for a system of data structured with metadata. In this category we describe the specific data concerning the recording of these very data. The elements described in this category are as follows: The identifier of the data, the person contributing to the data, the diagram of the data used the language of the data, and comments.

In the "***Educational path***" category, we describe the educational steps carried out by a learner during his training. We note that the educational path is not present in any of the existing models. We define the learner educational path as the set of the steps, the activities and the choices which characterize his/her training. The educational path differs according to learners. Thus, it can be marked by interruptions or reorientations according to the rhythm and aspirations of each one. They are organized according to a progression making it possible to achieve educational goals with identified courses. The elements of this category are: The educational path identifier, the learner's objective achieved at the end of his/her course, the educational path description, the courses followed by the learner during his/her training, the learner's assets during and at the end of the course, and the learner's course comments.

The other categories (*Affiliation*, *Accessibility*, *Transcript*, *Relationship*, and *Securitykey*) are similar to those existing in the IMS LIP model

In this model, we tried to give a great importance to learner educational data. That led us to define a new category which describes the learner's educational path. It is one of the main characteristics of this model.

As for every model, we had to choose which categories are mandatory. In this model, five elements are mandatory, namely: the learner's identifier, his/her first name, middle name, contact and the data diagram used. All the other elements are optional. This choice is related to the technical constraints required by any e-learning system. Among these constraints, let us note the following ones.

In each e-learning system, we need to have a single identifier for each learner indexed on the system server. This element makes it possible to identify the learner in a single way, and to manage the learners' rights in the system.

The learner's first name, his/her middle name, and contact represent a minimum of information to identify the learner in an e-learning system.

Lastly, the data diagram makes it possible to identify the diagram used to exchange learner data among systems.

About the vocabulary used, i.e. the recommended lists of suitable values it is to be noted that certain vocabularies used in the IMS LIP are too general and inappropriate to correspond to certain e-learning system realities. If we take the example of the element "Activity" (IMS LIP 6.1), which can take as value: work, service, education, training, and the military, considering that certain adaptations of our system relate to vocational training, this information loses its relevance. Then the solution is to define our own vocabularies and to define their equivalence in the IMS LIP model although such a correspondence is not always possible, and can lead to a loss of information or a redundancy. The definition of new elements also compelled us to choose new vocabularies such as:

- In the learner's educational path: The type of contents used in the courses.
- In the learner's competences: The learner's interest.

- In the learner's diploma: Its name, the level associated to it and the name of the organization which delivered the diploma.
- In the activity: The type of activity and its statute.

Lastly, we chose to define at the end of each category, an element "Comment". This element makes it possible to supplement the metadata carried in the category. It is not mandatory. It is used if one wants to add information or a remark to complete the category elements.

In the following section, we show how this diagram is implemented and used in e-learning systems as well as how useful it is to transfer learner data from one system to another.

4.3 Utility and Usability of the Model

4.3.1 The Use of the Model

The way of implementing and using our model in e-learning systems is simple. To explain it, we will take the example of the SERPOLET system which seeks to exchange its learner data with another external system.

To use our model, it is enough to define an API (Application Programming Interface) which has a double role: the first consists in generating the XML file which respects this data model starting from the various learner data which are stored in various data bases of the SERPOLET system; The second role consists in making the reciprocal way, i.e. from the XML file provided by an external system and which respects our model, the API reads the data and dispatches them in the SERPOLET system data base.

This principle is not new. We already used it in the implementation of the SCORM standard in the SERPOLET system. Indeed, it makes it possible to read the XML files which index teaching resources, as well as to read and to generate the file "imsmanifest.xml" which indexes the organization of these resources in a teaching SCORM module. It is also used to ensure the transfer of learner follow-up data between a SCORM module and the system data base.

4.3.2 Usefulness of the Model

To show the usefulness and the importance of this model for the exchange of learner data between e-learning systems, we present an example which shows a real case that we have met lately with the users of the SERPOLET platform. The example is as follows: An organization that uses the SERPOLET system wishes that a group of learners, registered in the SERPOLET system, take part in collaboration sessions in another LMS. The question is how to give learners access to the second system?

Without a common model, the solution consists in defining all the technical parameters necessary for a very good data exchange. But this type of solution solves the problem only partially, and is both expensive and time-consuming because every time we need collaborations of this type with other systems it is necessary to define and develop a new exchange protocol.

The use of a common data model is more economical, indeed. It makes it possible to have a common base of data exchange in similar situations. No development is required. There will be only the generation of the learner data in XML format by the first system. These XML file data will be exploited by the second system to dispatch them and to record them in its data base.

Let us note that there will always be a reason for one learner to follow or to supplement his/her training with another e-learning system. Thus, the adoption of such a model makes it possible to save time and money, and it is well-known that price and quality factors take an increasing place in the choice and use of e-learning systems.

5 Conclusion and Further Trends

In this paper, we detailed a proposal for a learner data model used in e-learning systems. We tried to offer solutions to the issue of learner data interoperability between e-learning systems.

The development of this proposal is the result of real uses and practices of the LMS. Indeed, the use of the SERPOLET system by public or private organizations, led to seek a whole set of adaptations and improvements in response to clients' requirements, in particular as regards interoperability at various levels.

We consider that this model is open; It can be adapted for other external systems' needs and for other types of users and not only for learners. It could also be re-used by the existing organizations of standardization to improve the existing standards, in particular IMS LIP [11].

About communication between learner data which are part of the general characteristics of the model and more fine grained and evolutionary learner data, our view is as follows: As knowledge engineering shifted from a myriad of knowledge representation formats to a small set of W3C recommendations, the outputs of ITS learner models should shift from a one-system-one-learner-model paradigm to a learner metadata standard based representation. In such a perspective, the fine-grained topic overlay models used in ITS or AH should at least result in some data to update for instance a competency category or a learning goal category in a platform learner model, otherwise they are lost for further learning experiences and we always start from scratch. We also need more discussion and agreement on several other topics such as:

Which data have to be exchanged between learning systems and for what purpose, which data are "learning unit dependant", which may be useful outside the unit, which data are domain dependant and which are not, which data are recorded for humans and machines, which data are recorded only for machine processing (this point is of growing importance for distance learning instructors who do not see their students and who need data about them in order to react efficiently to their requests), which data can be inferred, which data must be obtained from external sources (learner records, questionnaires, etc.).

Our proposal is grounded on the analysis of the data exchanged within the SERPOLET implementations, and it should be compared with other similar analyses

References

1. Chen, W. Mizoguchi, R.: Communication Content Ontology for Learner Model Agent in Multi-agent Architecture, Proceedings of AIED99 Workshop on Ontologies for Intelligent Educational Systems, 1999.
2. Mc Calla, G.: The Ecological Approach to the design of E-learning Environments : Purpose-based Capture and Use of Information About Learners, JIME, (7) 23 p., 2004
3. Murray, T.: Authoring Intelligent Tutoring Systems: An Analysis of the state of the art, IJAIED, (10), 34 p., 1999

4. Brusilovsky, P.: Adaptive navigation support in educational hypermedia: The role of student knowledge level and the case for meta-adaptation. BJET, 34 (4), p. 487-497. September 2003.
5. Kay, J. Lum, A.: Exploiting Readily Available Web Data for Reflective Student Models, Proceedings of AIED 2005, IOS Press, Amsterdam, Netherlands, p. 338-345.
6. Esposito, F. Licchelli, O. Semeraro, G.: Discovering Student Models in e-learning Systems, Journal of Universal Computer Science, (10), n°1, p. 47-57, 2004
7. Rosemary, R.: Metadata standards for the description of portal users: a review Version 1.0., 2003. PORTAL WP6. http://www.fair-portal.hull.ac.uk/downloads/Metadata.pdf
8. Paramythis, A. Loidl-Reisinger, S. Kepler, J.: Adaptive Learning Environments and e-Learning Standards. Electronic Journal on e-Learning Vol. 2 n. 1. p. 181-194, 2004.
9. Grandbastien, M. Oubahssi, L. Claës, G.: A process oriented approach for modelling on line Learning Environments. Supplemental proceedings AIED2003. vol.4. Sydney. p. 140-152.
10. Oubahssi, L. Grandbastien, M. Ngomo, M. Claës, G.: The Activity at the Center of the Global Open and Distance Learning Process. Proceedings of AIED2005, IOS Press, Amsterdam, Netherlands, p. 386-394.
11. Oubahssi, L. : Conception de plates-formes logicielles pour la formation à distance, présentant des propriétés d'adaptabilité à différentes catégories d'usagers et d'interopérabilité avec d'autres environnements logiciels. Thèse de Doctorat en informatique, Université René Descartes Paris V, 2005.

Towards a Pattern Language for Intelligent Teaching and Training Systems

Andreas Harrer[1] and Alke Martens[2]

[1] University of Duisburg-Essen, Dep. of Computer Science and Interactive Systems
[2] University of Rostock, Dep. of Computer Science and Electrical Engineering
harrer@collide.info.de, martens@informatik.uni-rostock.de

Abstract. Intelligent Tutoring Systems (ITSs) are usually based on similar fundamental structures. In contrast to this, software engineering techniques are seldomly used for realizing ITSs. In the last years, some approaches tried to change this: pattern mining took place; methods covering the specifics of ITS project development have been deployed. These approaches usually focus on a specific system type or on a certain application domain. What is missing is a combination of all the different approaches in a pattern language or a pattern catalogue for ITS. The purpose of such a pattern catalogue is to provide pattern for different types of software and to support the software development starting from design and ending with the implementation. The first step towards a pattern language is described in this paper.

1 Introduction

Intelligent Tutoring Systems (ITSs) can look back on a comparably long history and are usually based on similar fundamental structures, reflected in the ITS architecture. This architecture consists: expert module, learner module, user interface module, and steering/tutoring module (see e.g. [5], [27]). This architecture can be seen as a pattern in the broad sense [9]. In contrast to the quite homogeneous usage of terms, the realization of the modules and the interpretation of each module's role and functionality in the ITS is heterogeneous [27]. In some ITSs, e.g. the realization of the expert knowledge module has moved from the integrated expert system to a system module, which resembles a multilayer database. Consequently, the execution layer (former part of the expert system) is then externalized and realized as an own module. The potentially stand-alone expert system and the expert knowledge database are both called expert knowledge modules. Thus, ITS development suffers from the negative effects of homogeneous naming for heterogeneous functionality. This results in hardly comparable ITSs. Re-use of existing systems or system components is in most cases not possible.

In recent years, several approaches have been made towards establishing uniformity in the development of teaching and training systems. Two main directions of research are: the realization of *standards*, and the usage of *software engineering methods*. Standards can be used at different levels of teaching and

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 298–307, 2006.

training system development. Examples are: standardized architectures like the LTSA (Learning Technology system architecture) (see http://www.edutool.com/ltsa), approaches for learning project management like ELM (Essen Learning Model) [34], and XML (EXtensible Markup Language) descriptions for teaching and training content like the Dublin Core Metadata (see http://dublincore.org). The main advantage of standards is the predefinition of terms, which is a step towards exchangeability. Main drawback is that the standard's contribution remains on the level of terms. Standards lack a specific description of how to realize and implement the system parts.

Using methods of software engineering in teaching and training systems is based upon the idea that software development is mainly an engineering science, not an art – although the latter perspective usually describes how software development is done in several different branches of computer science, and also in the area of ITS. To organize work according to the engineering perspective has some consequences: the development of software is based on techniques which are clear, definite, and traceable, and which explicate different aspects of the development process and of the result (i.e. the software).

This paper describes a first step towards a pattern language for intelligent teaching and training systems. In the following section, related work is described. The subsequent section is used to suggest a terminology for describing a generative pattern language. A snapshot of a generative pattern language is given. Afterwards, two examples of ITS are given, which are developed based on software engineering approaches. The paper closes with a discussion.

2 Methods and Terminology

Currently, the situation in e-learning research and development is the following: systems for learning support that do not rely on methods of artificial intelligence, like Web-portals (e.g. Postnuke, http://www.post-nuke.net) and platforms for structured discussion forums (e.g. Future Learning Environment FLE3 [31]) are usually developed according to software engineering techniques (e.g. modularization, extensibility). Functional extensions by other developers can frequently be found (e.g. [35], [7]). By contrast, complex AI-based systems are usually not developed according to software engineering criteria.

ITSs and AIED Systems (Artificial Intelligence for Education) are usually developed specifically for a certain application domain, and based on a cognitive or pedagogical theory. Details of architecture descriptions, explication of interfaces or background information about implementation details cannot be found in most cases. Most of the systems are not extensible and not re-usable. Utterances like "AI kept fighting and losing against Software Engineering" [37] reveal a lot about the feeling of rivalry between these fields.

From the engineering perspective, ITSs are complex software systems, which should be conceptualized and developed based on software engineering. Advantage of this perspective is a reduction of complexity in modules and components of the system. Based on a good design it should be easy to integrate application domain experts. Usually experts of the application domain shouldn't need to know implementation details but should have insight in the specific design of parts, like e.g. the expert knowledge module. An appropriate software design would separate between a application domain oriented specification (e.g. domain knowledge) and technical aspects of implementation.

In software engineering there are several approaches which can support the development of teaching and training systems.

- *Reference architectures* and *architectural patterns* are used to specify basic structure and relationships between the main components of a software system. Reference architectures are e.g. the LTSA, system oriented verbal descriptions [5], or student oriented descriptions [4]. Architectural patterns are on a more abstract level, e.g. [8] and the example of a realization [22].
- *Design patterns* are used to describe typical solutions for recurring design problems. They help to refine and complete system structures. [8] has introduced design patterns in ITS research.
- *Process patterns* and *Learning Design* follow the idea to explicitly model process oriented aspects and to make them re-usable. In ITS, implicit principles (e.g. related to pedagogical principles) are transformed to explicit declarations. This supports and facilitates the work of domain experts; resulting concepts and designs can be re-used. Examples are an exchangable catalogue of tutoring rules [14], and sequencing of learning activities in the Learning Design [6].
- *Ontologies*, *Component based design*, *Frameworks* and *Refactoring* are futher software engineering methods relevant for ITS, which are not discussed here due to space limitations.

To combine the obove mentioned software engineering concepts in a pattern language is a complex task. Thus, in the following, a stepwise approach is described.

A necessary step towards designing and implementing systems and subsystems with a pattern oriented approach is to make the implicit relations between different patterns explicit. These are:

- Patterns can be used to realize patterns
- Patterns complement patterns
- Patterns can be used alternatively
- Patterns have similarities and differences

A collection of patterns which explicitly describes these relations is called a *pattern language* [38]. Pattern-oriented software development means to work stepwise: Based on the requirements of a system to develop, a certain entry point is selected, usually an architectural pattern. This first pattern provides a structure but no details. Thus, a refining pattern is used, which meets the requirements. Step by step, the system design will be refined by selecting subsequent patterns. This procedure continues until all requirements are met or if no refining patterns are available. For ITS currently only few approaches towards pattern language development can be found: e.g. a pattern language for tutoring systems, based on the layer decomposition [8], and the blended learning process pattern [10]. Both are specialized on a certain ITS aspect.

3 Towards a Pattern Language for ITS

When developing a pattern language, the first step is to look for entry points. The basic pattern underlying most ITS is the ITS architecture (e.g. [5], [27], [28]). The elements of the architecture are described as models, modules or as components (see [28]). In the following the term module is used to name the parts of an ITS. A module is not necessarily a well defined and clearly structured model – the term module is used to characterize one entity of a complete system. Accordingly, an ITS consists of: expert module, learner module, user interface module, and process steering module. In figure 1 in part I, the aggregation without label is used – usually only one instance of each module exists. Each module can be concretised by a refining pattern (see figure 1, part II). The expert module can be realized e.g. as pedagogical agent, as case-based expert knowledge module, as expert knowledge model based on production rules. The learner module can e.g. be specialized as CSCL (Computer Supported Collaborative Learning) learner module [15], as cognitive tutor, which uses model tracing for evaluation of learner behavior [1], as the simple learner profile, which has no further information for correction or evaluation, as classical bug library [2], and as overlay module [13].

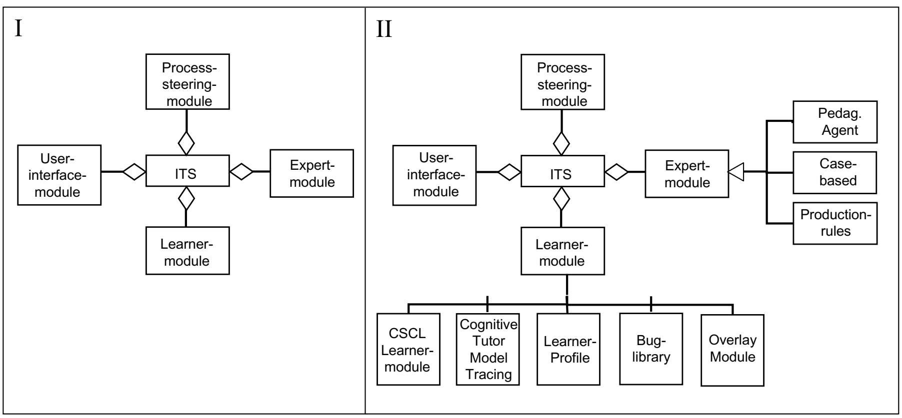


Fig. 1. Part I: Modules of an ITS, Entry point pattern. Part II: Extension of the Modules of an ITS.

The specialization of the process steering module is closer to the implementation level (see figure 2 – everything except the grey parts). The specification of the process itself, which differs dependent on the process description, is separated from the specification of the execution semantics, which should be the same for different processes. The engine part describes the semantics of the process steering module, i.e. how the steering process is realized and executed. Examples are: Learning Design (LD) Engines (for example Coppercore http://www.coppercore.org), an Intelligent Distributed Learning Environment (IDLE) Agent [15], or a Tutoring Process Model (TPM) interpreter [29]. The specification part is given by the (formal) process specification. Examples are: the LD-Documents [6] describing a learning scenario together with activities, services and learning resources, to be used in a LD-Engine; the behavior of the IDLE Agent according to the intended role of the agent in a learning community; a formally described Tutoring Process Model (TPM), which can be refined, e.g. as an adaptive TPM [29] or timed TPM [30].

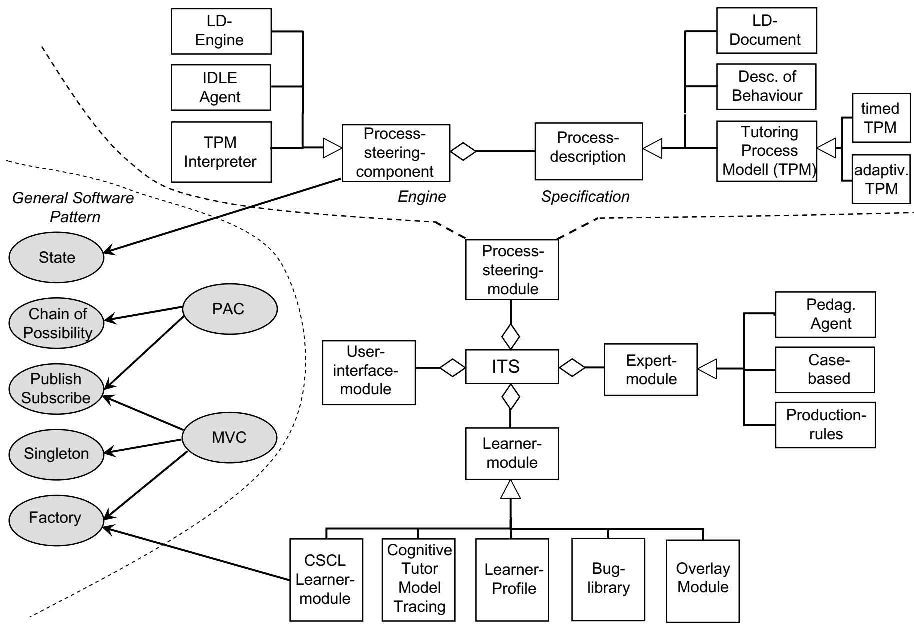


Fig. 2. Separation of Engine and Specification and Extension by General Software Patterns

On the next level of refinement (now including the grey parts in figure 2), general software patterns are integrated in the language. Figure 2 exemplarily shows some patterns, which can be used for the design of the user interface module: the Presentation Abstraction Control (PAC) pattern or the Model View Controller (MVC) pattern (both in [3]). PAC uses for example the pattern Chain of Responsibility [12] and the Publisher-Subscriber pattern [3]. The Publisher-Subscriber pattern can also be used by the alternative MVC oriented implementation. MVC would use the Singleton and the Factory pattern (both in [12]). The Factory pattern also lends itself for the realization of the CSCL learner module. For the realization of the process steering component a State pattern [12] can be used to implement the state dependencies in the tutoring process execution in an elegant way.

4 Applications of the Pattern Language

To show the practical usefulness of our proposed pattern language two extended examples are given. The first one is an analytical usage of the pattern language to explore the structure of the medical ITS Docs 'n Drugs, while the other one is a pattern-oriented extension of a pre-existing system towards an ITS.

Docs 'n Drugs - Case- and Web-Based ITS for Clinical Medicine
"'Docs 'n Drugs - the virtual hospital"' is a web-based ITS for training of clinical medicine in a case-based way [22], [26]. The learner acts as physician who treats patients. His activities range from different anamneses, body examinations, over technical and laboratory examinations, to therapy. He has to decide about how to proceed in a given situation and he has to record and continually actualize a list of differential diagnoses. Since 2000 the system is part of the medical education at the University of Ulm, Germany (see http://www.docs-n-drugs.de). Docs 'n Drugs has been implemented based on the classical ITS architecture [29], see figure 1,I. The implementation of the system was based on the PAC pattern (Presentation-Abstraction-Control) [21]. PAC describes a system based on a set of interacting agents, each of which has presentation, abstraction and control parts. In Docs 'n Drugs, ten agents have been found and described: The *expert module* is realized consisting of two sub-modules: the medical knowledge agent for the expert knowledge module, the case-based knowledge agent for the pedagogical knowledge module. The *learner module* is part of the telecollaboration agent, which is also responsible for the telecollaboration between case authors. The *process steering module* is described by the tutoring process agent. The *user interface module* is part of the execution coordinator agent. Additionally, a repository access agent and an intelligent tutor agent have been identified. The first agent provides an interface between external repositories and system – this part would be an extension of the case-based realizaton of the Expert-module in figure 2. The second agent is an additional feature of the system, comparable to a pedagogical agent for guiding and supporting the learner (see figure 2).

Using the PAC pattern as basic description for ITS development had the advantage of a clear and explicite structure. Docs 'n Drugs had been a project of mid size, consisting of experts, e.g. artificial intelligence, media computer science, computer grafics, and several medical experts. Separating the system in several sub-parts which communicate via interfaces facilitates not only the system design, but also the interaction of the different experts. The approach showed one drawback: several levels of granularity regarding the module design have been mixed in one approach. For example, the tutoring process agent has a completely different functionality and structure than the medical knowledge agent. This has led to the effect that most of the agents had to be refined in a complex way (see [21]). Moreover, the PAC based approach has been quite difficult to communicate to the non-computer scientists of the procject. A better structured approach, like the usage of a pattern language, would shift parts, which are in computer terminology, like e.g. agents, in the responsibility of computer scientists, whereas inter-project communication can take place at a level which is easier to grasp independent of the developer's expertise.

Cool Modes - Towards an ITS for Collaborative Modelling
Cool Modes [36] has been conceived as a collaborative synchronous application for graphical modelling in different domains, e.g. concept mapping, mathematical modelling, computational models like Petrinets and finite automata. The focus was on graphical representational issues and collaborative processes conducted by students in co-located classroom or distance scenarios. Tutoring functionality was not available.

The user interface component of Cool Modes can be analyzed by using the suggested pattern language: The graphical model created by students is separated into visual representation (view), internal data model (model) and the components handling user input (control) – the Model-View-Controller architecture pattern. For the synchronisation of the students' multiple instances of the application an event propagation mechanism (Java RMI) is used – an example for the Publisher-Subscriber design pattern. To guarantee controlled instantiation of graphical objects and uniqueness of a student's identification, the Factory and Singleton design patterns have been used.

After successful use of the application in teacher-supervised classroom scenarios [25], the support of remote and unsupervised scenarios became a topic of interest. Possible modes of desired support are the sequencing of specific collaborative and/or individual work phases and direct feedback with respect to the quality and completeness of the created models. The proposed pattern language has been used to extend the application towards an ITS for graphical modelling – e.g. by adding modules for student information, expert knowledge, and process regulation (see figure 1). These models are realized as follows: For the *student model* a logging mechanism [23] is used that captures all the raw information of the collaborative user actions, e.g. time stamp, user id, type of action, targeted object, and associated parameters. The student model has been realized as a combination of CSCL Learner module and Cognitive Tutor Model Tracing [18], two possible continuation patterns of the student module pattern. The *expert module* is case based, as teachers and/or experts of a specific graphical domain can specify relevant situations and configurations of graphical objects and their relations [19]. Cases are situations directly targeted at specific tasks/problems, but can be also more conceptual entities, e.g. conflicting argumentative structure, in a typed graphical discussion. In complex collaborative learning scenarios, students often need some scaffold, e.g. what to do in a specific phase, which tools to use, and which phases might follow each other. This is similar to pedagogical control in an ITS with interventions, yet more specifically tailored to the needs and specifities of collaborative learning. The IMS/LD specification [6] provides a standardized vocabulary for the definition of (collaborative) learning processes and implementations. Based on the pattern language a *process regulation module* for Cool Modes has been conceptualized. The used refining pattern is LD engine and LD document in figure 2. The pattern approach has then been combined with a component based approach, using the existing Coppercore IMS/LD engine as a loosely coupled process regulation module (see [17]). The Cool Modes example shows that the pattern language is useful for analytical purposes as well as for the pattern oriented development and extension of ITS.

5 Discussion

The investigation of different teaching and training systems, e.g. ITS, AIED, and CSCL systems, has revealed that methods of software engineering are seldomly used. This can be the reason for the following three problems:

1. Re-usability of existing systems in different teaching and training domains and also re-usability of system components is in most cases not possible.
2. System requirements have to be communciated. Communication is difficult between different ITS research groups, and even more between different researchers, e.g. software developers, experts of the application domain, and pedagogical researchers.
3. Comparison of teaching and training systems often takes place by evaluation of the learner's satisfaction and the learner's success. Aspects like re-usability of components, performance, or way of implementation, all of which could be used to compare systems in the same application domain and based on similar didactical approaches, are not taken into account.

Promising regarding all three mentioned problem areas is the idea to combine different software engineering approaches. Example architectures (e.g. reference architectures, architectural patterns, and frameworks), implementation oriented approaches from the field of software engineering (e.g. patterns, component based design, and refactoring), and formal tutoring process descriptions can be combined and related to each other. Together, all these different descriptions can constitute a pattern language. An approach following this direction is described in this paper. Starting with the classical ITS architecture as an entry pattern, examples of refining patterns are given for the expert module, learner module, and for the process steering module. By separating engine and specification part, the possibility to re-use the semantics of process execution arises. The combination of specific ITS patterns with general software patterns is given.

Two examples for the practical application of this pattern based approach with ITS systems have been sketched. In Docs 'n Drugs a pattern based approach has been used for system design. In CoolModes a non-ITS system has been evolved towards an ITS based on the pattern language.

References

1. Anderson, J.R., Boyle, C.F., Corbett, A.T. and Lewis, M.W.: Cognitive Modeling and Intelligent Tutoring. Artificial Intelligence, 42, (1990) 17–49
2. Brown, J.S. and Burton, R.R.: Diagnostic Models for Procedural Bugs in Basic Mathematical Skills. Cognitive Sciences, 2, (1978) 155–192

3. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P. and Stal, M.: A System of Patterns. John Wiley & Sons, Chichester, (1996)
4. Brusilovsky, P.: Intelligent Learning Environments for Programming: the Case for Integration and Adaptation. Proc. of AI-Ed, Washington, US, (1995) 1–7
5. Clancey, W. J.: Methodology for Building an Intelligent Tutoring System. In: Kintsch, W., Miller, J.R., and Polson, P.G. (eds.): Methods and Tactics in Cognitive Sciences, Lawrence Erlbaum Associates, Hillsdale, New Jersey, London, (1984) 51–84
6. IMS Global Learning Consortium. IMS Learning Design Information Model. Report No. 1.0, (2003)
7. Dolonen, J., Chen, W. and Morch, A.: Integrating Software Agents with FLE3. In: Wasson, B., Ludvigsen, S. and Hoppe, U. (eds): Proc. of CSCL 2003, Kluwer Academic Publishers, (2003) 157–161
8. Devedzic, V.: A Pattern Language for Architectures of Intelligent Tutors. In: Moore, J.D., Redfield, C. and Johnson, W.L. (eds): Proc. of AI-Ed, San Antonio, TX, US, (2001) 542–544
9. Devedzic, V. and Harrer, A.: Architecural Patterns in Pedagogical Agents. In: Cerri, S., Gouardéres, G. and Paraguaçu, F. (eds): Proc. of ITS, LNCS 2363, (2002) 81–90
10. Derntl, M. and Motschnig-Pitrik, R.: Patterns for blended, Person-Centered learning: Strategy, Concepts, Experiences, and Evaluation. In: Proc. of the 2004 ACM symposium on Applied Computing, ACM Press, Nicosia, Cyprius, (2004) 916–923
11. Fowler, M., Beck, K., Brant, J., Opdyke, W. and Roberts, D.: Refactoring: Improving the Design of Existing Code, Addison-Wesley, (1999)
12. Gamma, E., Helm, R., Johnson, R. and Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, Reading, MA, (1995)
13. Goldstein, I.P.: Overlays: A Theorie of Modeling for Computer Aided Instruction. Report AI Memo 406, MIT, Cambridge, (1970)
14. Harrer, A.: Both Sides of the Coin - Blending Cognitive and Motivational Aspects into a Tutoring Strategy. In: Ottmann, T., Halim, Z. and Razak, Z. (eds): Proc. of ICCE, Kuching, Malaysia, (1997) 188–195
15. Harrer, A.: Unterstützung von Lerngemeinschaften in verteilten intelligenten Lehrsystemen. PhD thesis, University of Munich, Germany, (2000)
16. Harrer, A.: Software Engineering Methods for re–use of Components and Design in Educational Systems. International Journal of Computers & Applications, Special Issue on Intelligence and Technology in Educational Applications, Anaheim, CA, 25, 1, (2003)
17. Harrer, A., Malzahn, N., Hoeksema, K. and Hoppe, U.: Learning design engines as remote control to learning support environments. Journal of Interactive Media in Education, Special Issue on Advances in Learning Design, (2005) (http://jime.open.ac.uk/2005/05)
18. Harrer, A., McLaren, B., Walker, E., Bollen, L.B. and Sewell, J.: Collaboration and cognitive tutoring: Integration, empirical results, and future directions. In Proc. of AI-Ed, IOS Press, Amsterdam, (2005) 266 – 273
19. Herrmann, K., Hoppe, U. and Pinkwart, N.: A checking mechanism for visual language environments. In Proc. of AI-Ed, IOS Press, Amsterdam, (2003), 97–104
20. Ikeda, M. and Mizoguchi, R.: FITS, A Framework for ITS – A computational model of tutoring. Journal of Artificial Intelligence in Education, 5, 3, (1994) 319–348
21. Illmann, T., Weber, M., Martens, A. and Seitz, A.: A Pattern-Oriented Design of a Web-Based and Case-Oriented Multimedia Training System in Medicine. 4th World Conference on Integrated Design and Process Technology, Dallas, US (2000)

22. Illmann, T., Martens, A., Seitz, A. and Weber, M.: Structure of Training Cases in Web-based Case-oriented Training Systems. Proc. of Advanced Learning Technologies, ICALT, (2001)
23. Jansen, M.: Matchmaker - a framework to support collaborative java applications. In Proc. of AI-Ed, IOS Press, Amsterdam, (2003) 529–530
24. Johnson, W.L., Rickel, J. and Lester, J.: Animated Pedagogical Agents: Face-to-Face Interaction in Interactive Learning Environments. International Journal of Artificial Intelligence in Education, 11, (2000)
25. Lingnau, A., Kuhn, M., Harrer, A., Hofmann, D., Fendrich, M., Hoppe, U.: Enriching traditional classroom scenarios by seamless integration of interactive media. In Proc. of Advanced Learning Technologies, Los Alamitos, CA, (2003) 135–139
26. Martens, A., Bernauer, J., Illmann, T., Seitz, A.: Docs 'n Drugs - The Virtual Polyclinic. In: Proc. of the American Medical Informatics Conference AMIA, Washington, USA, (2001)
27. Martens, A.: Centralize the Tutoring Process in Intelligent Tutoring Systems. In: Proc. of the 5th Internat. Conf. New Educational Environments ICNEE, Lucerne, Switzerland, (2003)
28. Martens, A.: Ein Tutoring Prozess Modell für fallbasierte Intelligente Tutoring Systeme. AKA Verlag infix, DISKI 281, (2004)
29. Martens, A.: Modeling of Adaptive Tutoring Processes. In: Zongmin, M. (ed): Web-Based Intelligent e-Learning Systems: Technologies and Applications, Information Science Publishing, Idea Group Inc., Hershey, London, (2005) 193–215
30. Martens, A.: Time in the Adaptive Tutoring Process Model. Submitted to: Intelligent Tutoring Systems ITS Conference, Taiwan, (2006)
31. Muukkonen, H., Hakkarainen, K. and Lakkala, M.: Collaborative Technology for Facilitating Progressive Inquiry: Future Learning Environment Tools. In: Hoadly, C. and Roschelle, J. (eds): Proc. of CSCL, Stanford University, (1999)
32. Mizoguchi, R., Ikeda, M. and Sinitsa, K.: Roles of Shared Ontology in AI–ED Research. In: du Boulay, B. and Mizoguchi, R. (eds): Proc. of AI-Ed, Kobe, (1997).
33. Mühlenbrock, M., Tewissen, F. and Hoppe, H.U.: A Framework System for Intelligent Support in Open Distributed Learning Environments. Journal of Artificial Intelligence in Education, 9, (1998) 256–274
34. Pawlowski, J.M.: The Essen Learning Model - a Multi-Level Development Model. In: Proc. of the Int. Conf. on Educational Multimedia, Hypermedia & Telecommunications ED-MEDIA, Montreal, Quebec, Canada, (2000)
35. Pinkwart, N., Harrer, A., Lohmann, S. and Vetter, S.: Integrating Portal Based Support Tools to Foster Learning Communities in University Courses. In: Uskov, V. (ed): Proc. of Web-Based Education WBE, ACTA Press, Anaheim, CA, (2005) 201–206
36. Pinkwart, N.: Collaborative Modeling in Graph Based Environments. PhD thesis, Gerhard-Mercator-Universität Duisburg (2005)
37. Rohmer, J.: Yesterday, Today, and Tomorrow of AI Applications. Invited Talk, IFIP World Congress, (2004)
38. Schmidt, D., Stal, M., Rohnert, H. and Buschmann, F.: Pattern–oriented Software Architecture – Patterns for Concurrent and Networked Objects. John Wiley & Sons, Chichester, (2000)

Semantic Web Technologies Applied to Interoperability on an Educational Portal

Elder Rizzon Santos[1], Elisa Boff[1,2], and Rosa Maria Vicari[1]

[1] Instituto de Informática – Federal University of Rio Grande do Sul (UFRGS)
POBox 15.064 – 91.501-970 – Porto Alegre – RS – Brazil
{ersantos,rosa}@inf.ufrgs.br
[2] Departamento de Informática, Caxias do Sul University (UCS), CEP 95.001-970 – Caxias do Sul – RS – Brazil
eboff@ucs.br

Abstract. This paper describes an approach to promote interoperability among heterogeneous agents that are part of an Educational Portal (PortEdu). We focus on a specific agent, the social agent, adding all the necessary functionality for him to interact with agents that aren't fully aware of its context. The social agent belongs to a Multi-agent Learning Environment designed to support training of diagnostic reasoning and modeling of domains with complex and uncertain knowledge, AMPLIA. The knowledge of the social agent is implemented with Bayesian networks, which allows the agent to represent its probabilistic knowledge and make its decisions. However, to communicate with agents outside AMPLIA, it is necessary to express such probabilistic knowledge in a way that all agents may process. Such requirement is addressed using OWL, an ontology language developed by W3C to be used on the Semantic Web.

1 Introduction

The semantic web [7] represents the next step on Internet technology. Nowadays, the content of web pages is understandable by humans only. The purpose of the semantic web is to aggregate meaning to the pages, in a way that computer software may interpret its content. There are several technologies involved in this process, but the main one is the utilization of ontologies.

Ontologies are expected to be used to provide means to explicate concepts and the relationships among them, allowing agents to interpret their meaning flexibly and unambiguously [6]. W3C (World Wide Web Consortium) is developing a stack of recommendations related to the Semantic Web in the effort of making the it a reality.

Currently, the base of the stack is XML, which provides syntax for the web documents; following is XML Schema, a language for restricting the structure of a XML document; next is RDF, which offers a simple graph model consisting of nodes (resources) and relations (statements) between them, providing a small amount of built-in semantics for the data model; RDF Schema augments RDF providing the means to describe properties and classes of RDF resources; last is OWL (Web Ontology Language), a semantic markup language which makes possible the description of classes, properties and their instances, besides that, it allows the definition of relation

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 308 – 317, 2006.

among classes (joint, disjoint), cardinality (none, exactly one), equality, characteristics of properties, among others. [8] [9]

In this work, we apply OWL as the main mechanism to provide semantic interoperability between agents on different contexts. The domain in which these agents exists is an educational portal, Portedu [10], a web portal that provides access to educational contents and systems. It provides a platform for agent based educational systems and several services, among them an information retrieval agent that searches for information on local resources and the web considering the context of the search. Also present as a core service of Portedu is the user profile agent [10], responsible for maintaining user-related information. One of the educational systems present in portedu is AMPLIA.

AMPLIA is an Intelligent Multi-agent Learning Environment designed to support training of diagnostic reasoning and modeling of domains with complex and uncertain knowledge [15]. As part of Portedu, AMPLIA may use the functionalities and services available in the portal and, most important, allow its agents to interact directly with agents from different intelligent educational systems. Besides, the agents may use the portal's service discovery, agent management, communication facilities, etc.

AMPLIA focuses on the medical area. It is a system that deals with uncertainty under the Bayesian network approach, where learner-modelling tasks will consist of creating a Bayesian network for a problem the system will present. The construction of a network involves qualitative and quantitative aspects. The qualitative part concerns the network topology, that is, causal relations among the domain variables. After it is ready, the quantitative part is specified. It is composed of the distribution of conditional probability of the variables represented.

In this paper we describe how we applied OWL as the ontology language used by an AMPLIA's agent, the social agent, in order to communicate with agents outside its domain. The rest of this paper is organized as follows: section 2 contextualizes where this research is applied, the social agent; section 3 presents the related research with this work; in section 4 it is presented our approach to enable a richer communication between social agent and other agents in the portal; section 5 presents our conclusions and future work.

2 Social Agent

The main goal of the social agent is to improve student's learning stimulating his interaction with other students, tutors, professors, etc. At AMPLIA, Each user builds his own Bayesian network for a specific pathology. During this task, the social agent will recommend students to help other students. The social agent creates workgroups to solve tasks cooperatively. Besides, the agent uses the strategy of initiate a network construction to motivate students to interact with the environment.

The social agent reasoning is based on *individual level* and *group level*.

2.1 Individual Level

The individual level has the student features. The information collected that is important to define the right student to recommend are: Social Profile; Acceptance

Degree; Sociability Degree; Mood State; Interest; Commitment Degree; Leadership and Performance.

The *Social Profile* (SP) is built during the students' interaction through a synchronous mechanism (e.g. chat tool). The following information is collected during the students' interaction: number of times that a student had the initiative to talk with another; number of times that a student answered a communication request; individuals with whom the student interacts or has interacted, and number of interactions and individuals with whom the student interacts regularly, and number of interactions.

Based on Maturana [16] we defined the *Acceptance Degree* (AD), which measures the acceptance between students. Such data is collected through a graphical interface that enables each student to indicate his/her acceptance degree for other students. This measurement may also be considered from a point of view of Social Networks. As the AD is indicated by the students themselves based on their affective structures, the measurement can indicate different emotions [22], such as love, envy, hatred, etc. The average of all ADs received by a student influences his/her *Sociability Degree* (SD).

The *Mood State* (MS) represents our belief in the capability of a student to play the role of a tutor if he/she is not in a positive mood state (although the student may have all the technical and social requirements to be a tutor). We consider three values for the MS: "bad mood", "regular mood" and "good mood". These states are indicated by the students in a graphical interface through corresponding clip-arts.

After a helping session, a small questionnaire is submitted to the student who got assistance, with the purpose of collecting information about the performance of the tutor. The questions made are based on concepts from Social Networks and Sociometry, and may be answered by four qualitative values: "excellent", "good", "regular", and "bad". The questions are presented below:

- How do you classify the sociability of your class fellow?
- How do you classify the help given by your class fellow?

The answer to the first question together with the average of the ADs of a student, form his/her *Sociability Degree* (SD). This grade indicates how other individuals see the social capability of this student.

Based on [21], the agent's preferences over world states S are expressed by a utility function U(S), which assigns a single number to express the desirability of a state. Furthermore, for each action *a* available to the agent, and for each possible outcome state S' of that action, P(S'|E, a) represents the agent's belief that action a will result in state S', when the action is performed in a state identified by evidence E. The expected utility (1) of an action *a* is computed as follows

$$U(A) = \sum_{S'} P(S'|E, a)U(S') \tag{1}$$

The socio-affective agent selects the action that maximizes this value when deciding how to act. The influence between nodes is shown in Figure 1. The arrows indicates the influence between the nodes, since it is Bayesian network used for decision making.

The *Interest* feature is given by the initiatives taken for the user (which material it had access without being recommended, with which student he/she initiated an interaction).

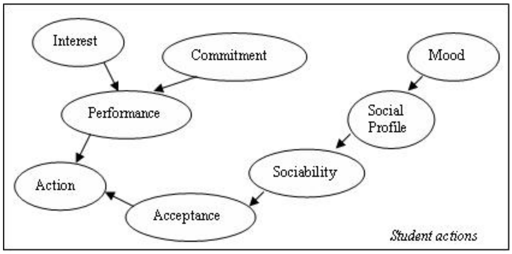


Fig. 1. Decision network of student model

The *Commitment Degree* is based on the consistency theory (theory of persuasion of Social Psychology) which claims that after the people to assume a public commitment, they probably will act in more consistent way with its commitment. The idea is to use this theory as strategy to motivate the students to collaborate with others. Thus, students who collaborate with other more actively (a quantitative measure of interactions can be made here, is not the ideal, but most viable at the moment) they will be inserted in groups with more frequency and will receive aid from better quality.

Leadership occurs when a person becomes capable to modify the beliefs, attitudes and behaviors of other individuals, organizing them and guiding its action for objectives that start to desire to reach [17]. Leadership also is related to the personality of the person, the psychological attributes of the followers and to the situations where it unfolds the leadership process. We can identify a leader in the environment of learning for the relation between leaders, followers, tasks and situations.

When a student behaves actively in the learning environment, he establishes many interactions and contributions as he posts solutions and helps other students. The agent records such information and verifies which student collaborated more actively in the network construction and which student had its work a lesser number of times modified. This fact can indicate the presence of a leader, therefore, they are people who lead the tasks and are accepted by the members of the group as a leader. The bigger the number of accepted contributions, greater the leader pointer.

2.2 Group Level

The group level takes into account the cohesion and the confidence (or trust) in a workgroup. The agent group reasoning is a probabilistic network, where confidence node influences the cohesion node, like group cohesion is based on individual confidence in a group.

The group formation agent can decide who brings more benefits when joining. The confidence is defined as the belief of an agent in the attributes such as the trustworthiness, honesty and the competence of the actual trusted agent. The reputation of an agent defines an expectation on its behavior that is based on the comments of the agent or information on the last behavior of the agent in a specific context in data moment [18] [19]. Mechanisms on confidence and reputation can be used by agents to differentiate collaborative from the non-collaborative users.

Joined groups have a great productivity. The cohesion is the attractiveness that the group exerts for its members, that is if they desire to continue to participate, resisting the idea to abandon it [17]. The cohesion is analyzed using the sociometric test of Jacob Moreno [20]. We can define as group cohesion the solidarity and establishment of loyalty in a group and measure it for the amount of times that the same people choose to interact.

3 Related Research

Since internally the social agent uses a bayesian network to represent its knowledge and make its decisions, it is necessary to provide another representation of his knowledge that is understandable by all others agents. To obtain such interoperability, we use ontology as a bridge between the social agent internal representation and a representation that other agents may interpret.

Before using an ontology as a bridge for the interpretation gap, it is necessary to represent social agent's decision networks (Bayesian networks) in OWL. A probabilistic extension to OWL, called BayesOWL, is proposed in [1]. To support representation of uncertainty and the ability to reason on scenarios where only partial information about a concept is available it is proposed to incorporate Bayesian networks on OWL ontologies.

The approach presented in [1] first augments OWL so that it may allow additional markups that can add probabilities to concepts, individuals, properties and its relationships. The second step is to define a set of translation rules to convert the probabilistic annotated ontology into a Bayesian network.

BayesOWL was developed to be a methodology for automatic ontology mapping. In this context, the ontologies are translated into BNs and the concept mapping between the two ontologies are treated as evidential reasoning between the two translated BNs. Probability tables are automatically created during the translation in order to measure the similarity between concepts.

The focus on ontology mapping limits the BayesOWL markups since it was not necessary to represent variables with states different than true or false. The reason for this is that the probabilistic knowledge associated with each ontology concept was used only for telling if two concepts from different ontologies were the same.

Another approach to represent probabilistic knowledge through OWL is presented in [11], where it is defined PR-OWL, an extension of OWL to express probabilistic knowledge. This approach differs from the first one in the fact that it is based on Multi-Entity Bayesian Networks (MEBN). MEBN [28] combines Bayesian probabilities with First Order Logic. Accordingly to the authors, the use of MEBN logic to base PR-OWL allows the expression of a probability distribution over models of any finitely axiomatizable first-order theory.

The use of MEBN logics as the fundamental semantics for OWL provides both expressiveness and flexibility in order to represent probabilistic knowledge. It allows the representation of the structural (graph) information of the model. As OWL-Full, the actual version of PR-OWL focuses on achieving the most expressiveness, as a consequence, in some cases there are no guarantees that a query will be traceable or even decidable.

Current implementation of PR-OWL provides an upper-class ontology in order to allow the development of probabilistic ontologies using RDF syntax, which is compatible with OWL. Although a Protégé plugin is under development, the lack of tools for MEBN logic and the need for standardization represents a drawback for short-term solutions but also points to a very interesting medium to long term solution, as it fits well (providing the formal foundation of a first-order logic) in the W3C model of standards.

4 Applying Ontologies to Promote Interoperability

The goal of this work was to provide means for the social agent to communicate with agents outside its original scope, the AMPLIA environment. To achieve this goal, we propose the use of semantic web technology. In this section we will present how OWL was applied in order to move towards our goal.

In figure 2 it is shown the social agent's architecture and its interaction with AMPLIA's agents and other agents belonging to the PortEdu. The arrows indicates interaction among components (squares) and agents (circles). The components that make possible the communication with external agents are the Ontology Management Module and the External Communication Module.

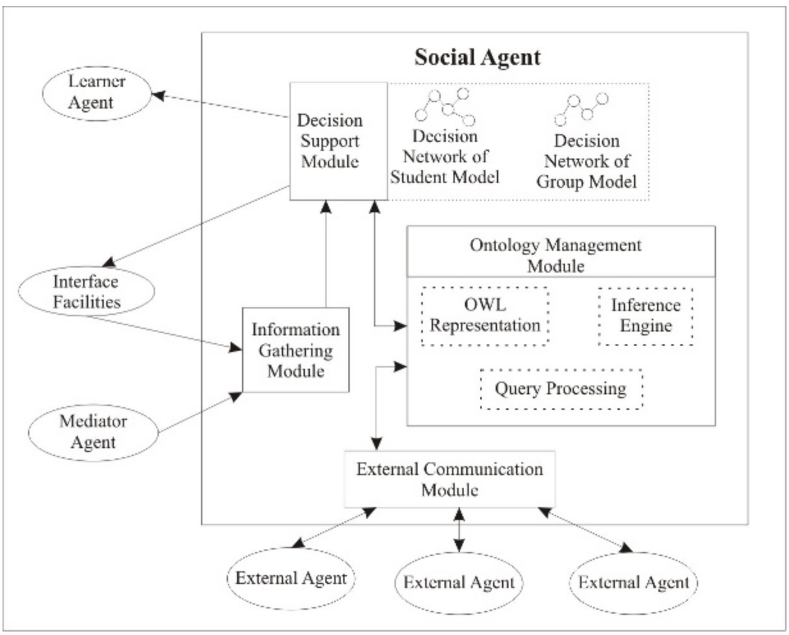


Fig. 2. The social agent architecture

4.1 Ontology Management Module

The OWL representation of the agent's knowledge is obtained extending and adapting the structure presented in [1]. As seen on section 3, BayesOWL was developed to solve the ontology mapping problem. Such focus leaded their approach to a practical but very specific solution. For our case, we used the core concepts of BayesOWL's OWL extension, and added the support to represent multiple state variables and conditional probabilities tables.

We used Protégé [25] to develop the core OWL ontology where all the concepts necessary for representing the social agent's probabilistic knowledge are supplied. The classes defined in this ontology allow the translation from the agent's bayesian networks to OWL. The main classes are:

- Probability Information – a high level abstraction which is the superclass of Prior Probability, Conditional Probability and CPT (Conditional Probability Table)
- Prior Probability – represents the probabilities of states whose nodes doesn't have any parents.
- Conditional Probability – represents a bayesian conditional probability. This class has two properties (a) and (b): State Probability (a), which defines the probabi-lity for every state of the node's variable taking into account its (b) conditions; a Condition (b) is a concept that specifies a variable and a state of this variable in order to be used in conjunction with State Probabality to represent a Conditional Probality.

In the ontology other concepts such as bayesian network node, conditional probability table, variable, state, among others, are defined, but the core functionality lies on the concept that defines the probability information. As stated earlier, our solution extends and adapts BayesOWL to our context, trying to provide a more generic approach for representing probabilistic knowledge in OWL. Besides the probabilistic information of the agent, this ontology may be used by experts to provide more information about AMPLIA's domain to the other systems in PortEdu.

With the concepts defined in our ontology we can now provide a representation of the agent's networks in OWL. Every node of the network is processed and all information about it (parents, children, CPT, etc.) is converted to OWL creating instances of the pre-defined concepts of our ontology.

The conversion of the Bayesian networks to the ontology is done using Jena [2] and the Hugin Java API [26]. The Hugin API is used to gather information about the bayesian network and Jena is used to create the instances based on the data collected. Jena is a semantic web toolkit for Java, it provides APIs for OWL, DAML+OIL and RDFS. Other features of Jena are the inference support for both RDF semantics and OWL semantics, as defined by W3C, and support for a query language SPARQL [27], that can be used on the results of RDFS or OWL reasoning. Using the OWL API we are able not only to easily update the agent's ontology representation but also to provide direct support for other agents to make queries using RDQL.

Although OWL is built on top of RDF, a query language on the RDF level, SPARQL, isn't capable of considering the restrictions expressed in OWL, for example. There is no standard query language for the semantic web yet. OWL-QL [3] is a formal language and protocol for agents, whose knowledge is expressed in OWL, to use on query-answer dialogs. This query language is intended to be a candidate standard query language for OWL and is designed to be suitable for a broad range of query-answering services and applications.

Since the goal of this work is to provide an interoperable way for the social agent to communicate, it is also fundamental that the supported query language to be a standard, as OWL. As soon as a W3C standard or recommendation on query language is released we plan on updating our architecture. Since the agent implementation is modular, it won't be a problem adding support for another query language.

4.2 External Communication Module

The external communication module provides the means to communicate with external agents. In our case, external agents are agents from Portedu and from others learning environments that are attached to the portal. Portedu is developed as multi-agent system that complies with the FIPA standards, in order to promote interoperability dealing with the heterogeneous nature of the different learning environments that it supports. There is no standard on the content language used on the message. The developer is free to use the language that best suits his needs. In our case, to increase the interoperability, we use OWL.

Thus, the communication module is capable to send and receive messages using FIPA ACL. It is responsible for receiving the queries from agents, parsing them and forwarding the content to the ontology management module. Upon receiving a response from the ontology module, already in OWL, a message envelope is assembled, accordingly to the FIPA ACL specification and the message is returned to its sender, following the message protocol.

The implementation of the communication was made easy by using JADE [5], a middleware developed to facilitate the implementation of multi-agent applications that complies with the FIPA specifications. The architecture provides a names service, directory facilitator, message transport, parser services and a library of FIPA interaction protocols. All the mandatory components of a FIPA plataform are available. For agent development, JADE supplies the necessary Java APIs.

5 Conclusions

In this paper we aimed to increase the interoperability of the social agent making possible his communication with agents outside his domain. Such agents are part of an educational portal, Portedu, whose architecture is grounded on a FIPA agent platform. Such platform allows all agents from the educational systems to communicate with each other through FIPA ACL. The use of common language to exchange the messages already provides a significant level of interoperability, but the fact that the content languages doesn't follows any standards can be a drawback.

Our work proposed the use of OWL as the content language used by the social agent. OWL is the latest ontology language proposed by W3C towards the semantic web. The several tools available to work with OWL made it easy to provide several ontology related services, such as query processing and inference engine. Since the social agent's knowledge is expressed using Bayesian networks, it was necessary to provide an alternate OWL representation, which enables our agent to communicate with a standard content language.

This demonstrates one application of the semantic web technology as to improve the interoperability on a multi-agent educational portal. Although being applied to a close heterogeneous domain, our approach to facilitate the interaction among heterogeneous agents may be applied on other contexts.

OWL was not developed considering the possibility of modeling uncertain knowledge. Such limitation reduces the applicability scope of semantic web and is a very important issue to be addressed by standardization committees, such as W3C.

Our approach for representing probabilistic information with OWL is very practical, further work needs to be done to provide solid semantic background to the model. A possible way is using PR-OWL, but more tools are necessary to develop both the multi-entity Bayesian networks and for the utilization of the extended OWL language.

The utilization of OWL by the social agent is a first step towards a semantic educational portal, since it is feasible to consider adopting this ontology language as a standard on the portal, making possible, for example the development of students' personal assistants that automatically gather information from heterogeneous sources (agents) and provide the student with educational resources accordingly to its current needs. Once all the agents present in the portal are able to communicate with one another, apply techniques to merge and exchange concepts [1] [23] [24] from its ontologies, the functionality of one specific educational system will increase due to the additional knowledge available to its agents.

References

1. Rong Pan et al., A Bayesian Network Approach to Ontology Mapping. In: Proceedings of the Fourth International Semantic Web Conference, November 2005.
2. Carroll, Jeremy J. *et al*. Jena: Implementing the Semantic Web Recommendations. In: Proceedings of the 13th international World Wide Web Conference, p. 74-83. ACM Press, 2004.
3. Fikes, Richard; Hayes, Patrick; Horrocks, Ian. OWL-QL - A Language for Deductive Query Answering on the Semantic Web. In: Journal of Web Semantics, vol. 2, issue 1. Science Direct, 2005.
4. Grau, Bernardo Cuenca; Parsia, Bijan; Sirin, Evren. Working with multiple ontologies on the semantic web. In: Proceedings of the Third Internatonal Semantic Web Conference (ISWC2004). Volume 3298 Lecture Notes in Computer Science, 2004.
5. Bellifemine, Fabio; Poggi, Agostino; Rimassa, Giovani. Developing multiagent systems with a FIPA-compliant agent framework. In: Software - Practice & Experience, John Wiley & Sons, Ltd. vol no. 31, 2001, pg. 103-128.
6. Horrocks, Ian; Patel-Schneider, Peter F.; Harmelen, Peter F. From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics, vol. 1, pg7--26, 2003.
7. Berners Lee, Tim; Hendler, James; Lassila, Ora. The Semantic Web. Scientific American 2001; 284(5), pg. 35-43.
8. Ding, Li; Kolari, P.; Ding, Zhongli; Avancha, Sasikanth. Using Ontologies in the Semantic Web: A Survey. In: Ontologies in the Context of Information Systems, Springer-Verlag. October, 2005.

9. Dean, Mike; Schreiber, Guus, eds. OWL Web Ontology Language Reference: http://www.w3.org/ TR/owl-ref/
10. Nakayama, L.; Viccari, R. M.; Coelho, H. An Information Retrieving Service for Distance Learning. Transactions On Internet Research, v. 1, p. 49-56, 2005.
11. Costa, Paulo; Laskey, Kathryn B.; Laskey, Kenneth J. PR-OWL: A Bayesian Ontology Language for the Semantic Web In: Proceedings of the Workshop on Uncertainty Reasoning for the Semantic Web, International Semantic Web Conference, 2005. to appear.
12. FIPA. FIPA ACL Message Structure Specification SC00061. http://www.fipa.org/specs/fipa00061/
13. FIPA-OS Agent Toolkit. http://fipa-os.sourceforge.net/index.htm
14. Miller, L.; Seaborne, A.; Reggiori, A. Three Implementations of SquishQL, a Simple RDF Query Language. In: Proceedings of the First International Semantic Web Conference. Springer-Verlag, 2002, pg 243.
15. Vicari, R. M., Flores, C.D., Seixas, L., Silvestre, A., Ladeira, M., Coelho, H.: A Multi-Agent Intelligent Environment for Medical Knowledge. In: Journal of Artificial Intelligence in Medicine, Vol.27. Elsevier Science, Amsterdam, (2003) 335-366.
16. Maturana, H. and Varela, F. Tree of Knowledge: The Biological Roots of Human Understanding. Shambhala Publications, Boston, MA, 1998.
17. Aronson, E., Wilson, T. D., & Akert, R. M. (2005). Social Psychology (5th ed.). Englewood Cliffs, NJ: Prentice Hall.
18. Vassileva, J. Multi-agent architectures for distributed learning environments. In: AIED Proceedings, vol. 12, pg.1060-1069, 2001.
19. Castelfranchi, C. e Falcone, R. Principles of Trust for MAS: Cognitive Anatomy, Social Importance and Qualification. ICMAS'98.
20. Moreno, Jacob L. (1953). Who Shall Survive? Foundations of Sociometry, Group Psychotherapy and Psychodrama. Beacon, NY: Beacon House.
21. Conati, C. Probabilistic assessment of user's emotions in educational games. Journal of Applied Artificial Intelligence. 16(7-8):555–575. 2002.4
22. Jaques, Patrícia A.; Viccari, R. M. A BDI Approach to Infer Student's Emotions in an Intelligent Learning Environment. In: IBERAMIA 2004, pgs 901-911.
23. Zou, Youyong; Finin, Tim; Ding Li; Chen, Harry; Pan Rong. Using Semantic Web Technology in Multi-Agent Systems: A Case Study in the TAGA Trading Agent Environment. In: Proceeding of the 5th International Conference on Electronic Commerce ICEC 2003, ACM Press, Oct 2003, pgs 95-101.
24. Wiesman, Floris; Roos, Nico. Domain Independent Learning of Ontology Mappings. In**:** Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2, IEEE, 2004, pgs. 346-853.
25. The Protegé Ontology Editor and Knowledge Acquisition System. Available at: http://protege.stanford.edu
26. Madsen, A. L., Jensen, F., Kjærulff, U. B., Lang, M. (2005). HUGIN - The Tool for Bayesian Networks and Influence Diagrams, International Journal of Artificial Intelligence Tools 14 (3), pages 507-543.
27. Prud'hommeaux, E.; Seaborne, A. SPARQL Query Language for RDF. 2006. Available at: http://www.w3.org/TR/rdf-sparql-query/
28. Laskey, Kathryn B.; Costa, Paulo. Of Klingons and Starships: Bayesian Logic for the 23rd Century. In: Uncertainty in Artificial Intelligence: Proceedings of the Twenty-first Conference. 2005, AUAI Press: Edinburgh, Scotland.

Studying the Effects of Personalized Language and Worked Examples in the Context of a Web-Based Intelligent Tutor

Bruce M. McLaren[1], Sung-Joo Lim[1], France Gagnon[2],
David Yaron[3], and Kenneth R. Koedinger[1]

[1] Human-Computer Interaction Institute, 5000 Forbes Avenue, Carnegie Mellon University, Pittsburgh, PA, 15213-3891 United States
bmclaren@cs.cmu.edu, sungjol@andrew.cmu.edu, koedinger@cs.cmu.edu
[2] Faculty of Education, University of British Columbia, Vancouver, BC V6T 1Z4
france.gagnon@shaw.ca
[3] Chemistry Department, 5000 Forbes Avenue, Carnegie Mellon University, Pittsburgh, PA, 15213-3891 United States
yaron@cmu.edu

Abstract. Previous studies have demonstrated the learning benefit of personalized language and worked examples. However, previous investigators have primarily been interested in how these interventions support students as they problem solve with *no other* cognitive support. We hypothesized that personalized language added to a web-based intelligent tutor and worked examples provided as complements to the tutor would improve student (e-)learning. However, in a 2 x 2 factorial study, we found that personalization and worked examples had no significant effects on learning. On the other hand, there was a significant difference between the pretest and posttest across all conditions, suggesting that the online intelligent tutor present in all conditions *did* make a difference in learning. We conjecture why personalization and, especially, the worked examples did not have the hypothesized effect in this preliminary experiment, and discuss a new study we have begun to further investigate these effects.

1 Introduction

In a recent book by Clark and Mayer [1], a number of principles were proposed as guidelines for building e-Learning systems. All are supported by multiple educational psychology and cognitive science studies. We were especially interested in and decided to experiment with two of the Clark and Mayer principles:

Personalization Principle One: Use Conversational Rather than Formal Style

Worked Example Principle One: Replace Some Practice Problems with Worked Examples

In contrast to most previous studies, however, we wished to test these principles in the context of a web-based intelligent tutoring system (ITS), rather than in a standard e-Learning or ITS environment or, as in even earlier studies, in conjunction with problems solved by hand. The key difference is that an intelligent tutoring system

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 318 – 328, 2006.

provides more than just problem solving practice; it also supplies students with context-specific hints and feedback on their progress. In particular, we were interested in whether personalized language within an ITS and worked examples provided as complements to ITS-supported problems might improve learning beyond the gains from the ITS on its own with formal, impersonal feedback. Furthermore, because we are interested in delivering intelligent tutoring systems over the Internet, we were interested in whether we could show results with a web-based ITS deployed in a distance learning environment.

Until relatively recently, providing intelligent tutoring in an e-Learning environment was quite difficult to achieve, given the implementation complexity and computational overhead of the typical ITS. However, advances in ITS authoring tools, an important area of ITS research [2], have started to overcome this obstacle. For instance, authoring software developed in our laboratory, the Cognitive Tutor Authoring Tools (CTAT) [3], make it possible to deliver tutors on the web. CTAT builds on successful research and development of cognitive tutors, intelligent tutoring systems that have been shown to lead to significant improvements in learning of high school math [4] and are currently in use in over 2,000 schools across the U.S. Perhaps the most important contribution of CTAT thus far is its support for developing *example-tracing tutors*, tutors that can be rapidly built by demonstration but are constrained to single-problem use[1]. On the other hand, these example-tracing tutors exhibit behavior that is very similar to full cognitive tutors and are lightweight enough to be deployed on the web. Inspired by our newfound ability to rapidly deploy tutors to the web, our general aim is to explore how we can leverage CTAT and e-Learning principles to improve web-based learning.

The Clark and Mayer personalization principle proposes that informal speech or text (i.e., conversational style) is more supportive of learning than formal speech or text in an e-Learning environment. In other words, instructions, hints, and feedback should employ first or second-person language (e.g., “You might want to try this”) and should be presented informally (e.g., “Hello there, welcome to the Stoichiometry Tutor! …”) rather than in a more formal tone (e.g., “Problems such as these are solved in the following manner”).

Although the personalization principle runs counter to the intuition that information should be “efficiently delivered” and provided in a business-like manner to a learner, it is consistent with cognitive theories of learning. For instance, educational research has demonstrated that people put forth a greater effort to understand information when they feel they are in a dialogue [5]. While consumers of e-Learning content certainly know they are interacting with a computer, and not a human, personalized language helps to create a “dialogue” effect with the computer. E-Learning research in support of the personalization principle is somewhat limited but at least one project has shown positive effects [6]. Students who learned from personalized text in a botany e-Learning system performed better on subsequent transfer tasks than students who learned from more formal text in five out of five studies. Note that this project did not explore the use of personalization in a web-based intelligent tutoring setting, as we are doing in our work.

---

[1] In [3] these specialized tutors were referred as “Pseudo Tutors.” However, we have since renamed them more mnemonically as “example-tracing tutors.”

The Clark and Mayer worked example principle proposes that an e-Learning course should present learners with some step-by-step solutions to problems (i.e., *worked examples*) rather than having them try to solve all problems on their own. Interestingly, this principle also runs counter to many people's intuition and even to research that stresses the importance of "learning by doing" [7].

The theory behind worked examples is that solving problems can overload limited working memory, while studying worked examples does not and, in fact, can help build new knowledge [8]. The empirical evidence in support of worked examples is more established and long standing than that of personalization. For instance, in a study of geometry by Paas [9], students who studied 8 worked examples and solved 4 problems worked for less time and scored higher on a posttest than students who solved all 12 problems. In a study in the domain of probability calculation, Renkl [10] found that students who employed more principle-based self-explanations benefited more from worked examples than those who did not. Research has also shown that mixing worked examples and problem solving is beneficial to learning. In a study on LISP programming [11], it was shown that alternating between worked examples and problem solving was more beneficial to learners than observing a group of worked examples followed by solving a group of problems.

Previous ITS research has investigated how worked examples can be used to help students *as* they problem solve [12][13]. Conati's and VanLehn's SE-Coach demonstrated that an ITS can help students self-explain worked examples [14]. However, none of this prior work explicitly studied how worked examples, presented separately from supported problem solving as complementary learning devices, might provide added value to learning with an ITS and avoid cognitive load [8]. Closest to our approach is that of Mathan and Koedinger [15]. They experimented with two different versions of an Excel ITS, one that employed an expert model and one that used an intelligent novice model, complemented by two different types of worked examples, "active" example walkthroughs (examples in which students complete some of the work) and "passive" examples (examples that are just watched). The "active" example walkthroughs led to better learning but only for the students who used the expert model ITS. However, a follow-up study did not replicate these results [16]. This work, as with the other ITS research mentioned above, was not done in the context of a web-based ITS.

2 The Hypotheses and the Stoichiometry Tutor to Test Them

Given the evidence about personalization and worked examples, our intent in the study described in this paper was to explore the following hypotheses:

H1: The combination of personalized language and worked examples, used in conjunction with a supported problem-solving environment (i.e., an ITS), can improve learning in an e-Learning system.

H2: The use of personalized language in a supported problem-solving environment can improve learning in an e-Learning system.

H3: The use of worked examples in a supported problem-solving environment can improve learning in an e-Learning system.

We tested these hypotheses using a Stoichiometry Tutor developed with CTAT. Stoichiometry is the basic math required to solve elementary chemistry problems and is typically learned in the 10th or 11th grade in U.S. high schools. Solving a stoichiometry problem involves understanding basic chemistry concepts, such as the mole, unit conversions, and Avogadro's number, and applying those concepts in solving simple algebraic equations.

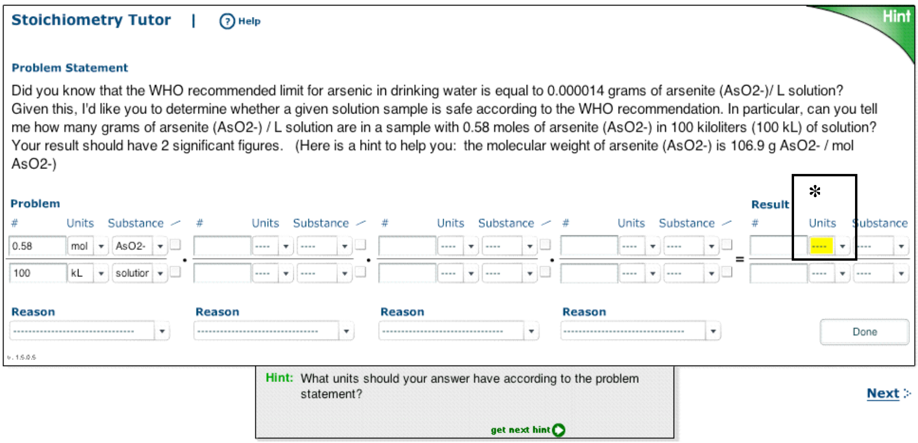


Fig. 1. The Stoichiometry Intelligent Tutor

The Stoichiometry Tutor and an example of a typical stoichiometry problem (a "personal" version used in the study) are shown in Figure 1. To solve this problem the student must first express the goal, given a problem statement and an initial value. The student has requested a hint, which suggests that they should express the units of the result (i.e., grams or "g," see the highlighted cell, further indicated by the box with an asterisk). After providing the goal, the student must fill in the other terms of the equation to convert the given value to the goal value. Each term, expressed as a ratio, is used to cancel the units and substance of previous terms. The full solution to the problem in Figure 1, with cancelled terms highlighted, is:

(0.58 mol AsO2- / 100 kL solution) * (1 kL solution / 1000 L solution) * (106.9 g AsO2- / 1 mol AsO2-) = 0.00062 g AsO2- / 1 L solution

The student is also asked to provide a rationale for each term of the equation (see the "Reason" field below each term in Figure 1). So, for instance, the initial term in the equation of Figure 1 is a "Given Value," since it is provided in the problem statement. Notice that the problem statement and the hint contain first- and second-person pronouns (e.g., "Did you know the WHO recommended limit for arsenic in drinking water is …") instead of more formal, impersonal language (e.g., "Suppose the WHO recommended limit for arsenic in drinking water is…"). The tutor also provides context-specific error messages when the student makes a mistake during problem solving. As with the problem statement and hints, there are personal/impersonal versions of all error messages.

The Stoichiometry Tutor was developed as an example-tracing tutor within CTAT [3]. In particular, after the graphical user interface (GUI) of Figure 1 was created, using the Flash programming language, problem solutions were demonstrated and all hints and error messages were added as “annotations” to the resulting behavior graphs, the knowledge structure that represents individual example-tracing tutor.

3 Description of the Study and Results

To test our hypotheses and the effect of personalized language and worked examples on (e-)learning, we executed a 2 x 2 factorial design, depicted in Figure 2. One independent variable was personalization, with one level being *impersonal* instruction, feedback, and hints and the other *personal* instruction, feedback, and hints. The other independent variable was worked examples, with one level being *supported problem solving only* and the other *supported problem solving and worked examples*. In the former condition, subjects only solve problems; no worked examples are presented. In the latter condition, subjects alternate between observation of a worked example and solving of a problem. As discussed previously, this alternating technique has yielded better learning results in prior research [11].

	Impersonal Instruction, Feedback, and Hints	**Personal Instruction, Feedback, and Hints**
Supported Problem Solving Only	Impersonal / Problem Solving (Condition 1)	Personal / Problem Solving (Condition 2)
Supported Problem Solving and Worked Examples	Impersonal Worked (Condition 3)	Personal / Worked (Condition 4)

Fig. 2. The 2 x 2 Factorial Design

If hypothesis H1 is correct, one would expect the subjects in Condition 4 (Personal / Worked) to exhibit significantly larger learning gains than the other conditions, since this is the only condition with both personalized feedback and worked examples. To confirm hypothesis H2, Conditions 2 and 4 should lead to significantly greater learning gains than Conditions 1 and 3 (i.e., a *main effect* for the personalization independent variable). Finally, to confirm hypothesis H3, one would expect Conditions 3 and 4 to exhibit significantly greater learning gains than Conditions 1 and 2 (i.e., a *main effect* for the worked examples independent variable).

The study was executed at the University of British Columbia (UBC) as an optional, online activity in two courses: Intro to Chemistry for majors and Intro to Chemistry for non-majors. Subjects were offered $20 Canadian for completing the study. A total of 1720 students, primarily freshmen and sophomores, were enrolled in the chemistry-for-majors course and 226 were enrolled in the non-majors course. A total of 240 students started the study, and 69 fully completed it, with “completion” defined as trying at least one step in the final posttest problem. Subjects were randomly assigned to one of the four conditions of Figure 2.

The subjects first watched a video introducing the study and explaining how to use the web-based interface. All subjects were then given an online pretest of nine stoichiometry problems. The pretest (and, later, the posttest) problems were solved using the web-based interface of Figure 1, with feedback and hints disabled. Problems in the pretest (and the posttest) were presented to the subjects in order of difficulty, from relatively easy to fairly difficult. The subjects then worked on 15 "study problems," problems presented according to the different experimental conditions of Figure 2. All of the worked examples in the study were solved using the tutor interface, captured as a video file, and narrated by a chemistry expert. During the solving of the 15 study problems, the subjects were also presented with various instructional videos to instruct them on stoichiometry concepts such as dimensional analysis and molecular weight. After completing the 15 study problems, the subjects were asked to take a posttest of nine problems, analogous in difficulty to the pretest.

All individual steps taken by the students in the stoichiometry interface were logged and scored as correct or incorrect. A score between 0 and 1.0 was calculated for each student's pre and posttest by dividing the number of correct steps by the total number of possibly correct steps. On the pretest there was a possibility of 231 correct steps; on the posttest there was a total of 222.

To test for the effects of worked examples and personalization, a 2 x 2 analysis of variance (ANOVA) was conducted on the difference scores (i.e., post - pre)[2]. For this analysis, there were two between-subjects variables: personalization (personal problems, impersonal problems) and worked examples (worked examples, problem solving). To test for the effect of time (i.e., the overall difference between pre and post across all conditions), a 2 x 2 x 2 repeated measure ANOVA was conducted. For this analysis, the within-subject variable was time (pretest = time 1, posttest = time 2)[3].

Prior to conducting the analyses, six subjects were deleted from the subject pool, four due to sparse pretest and/or posttest responses (indicating they did not seriously attempt to solve the test problems) and two because they were identified as univariate outliers. After data screening, N was adjusted from 69 to 63 (Cond. 1 = 16; Cond. 2 = 16; Cond. 3 = 19; Cond. 4 = 12).

The independence assumption underlying the ANOVA was not violated, as each participant was tested in only one of the two personalization conditions and in only one of the two worked problem conditions. A two-way ANOVA of the pretest scores showed no significant differences between the four conditions, indicating that subjects had been successfully randomized. Descriptive statistics indicated that the posttest scores were slightly skewed. This departure from normality influences the significance of the ANOVA results, especially with a small sample size ($N = 63$).

Table 1 shows the means and standard deviations of the pretest, posttest, and the difference scores across all conditions. Table 2 shows the result of the 2 x 2 ANOVA

---

[2] We also conducted 2x2 ANOVAs using four other common pre/posttest calculations (i.e., (1) (post – pre) / (1.0 – pre), (2) (post-pre) / pre, (3) (post – pre) / ((pre + post) /2), and (4) If post > pre, then (post-pre) / (1.0 – pre) else, (post-pre) / pre [17]). The results were very similar across all analyses, so only the difference score analysis is reported here.

[3] While we could have performed just a 2 x 2 x 2 ANOVA to test for the effects of all of the independent variables, the 2 x 2 ANOVAs provide a much clearer, more intuitive depiction of the effects of worked examples and personalization.

on the difference scores ($N = 63$). As shown in Table 2, the main effect of personalization is non-significant (p= .697). The impersonal conditions performed slightly better than the personal conditions ($M_{impersonal}$ = .1455 vs. $M_{personal}$ = .1292). The main effect of worked examples is also non-significant (p= .828). The worked example conditions performed slightly better than the supported problem solving conditions ($M_{worked\ examples}$ = .1401 vs. $M_{supported\ problem\ solving}$ = .1365). The interaction between problem solving and personalization is also non-significant, (p= .155).

Table 1. Means & Std Dev. for Pretests, Posttests, and Difference Scores, i.e., (post–pre)

	Supported Problem Solving				Worked Examples					
	Impersonal *(Condition 1)*		Personal *(Condition 2)*		Impersonal *(Condition 3)*		Personal *(Condition 4)*		Total	
	M	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*	M	SD
Pretest	.7719	.1043	.8231	.1176	.7810	.1554	.7349	.1358	.7806	.1310
Posttest	.9412	.0411	.9268	.0469	.9066	.1160	.8979	.0783	.9189	.0788
Difference Scores	.1692	.1113	.1037	.1185	.1255	.1853	.1630	.1137	.1383	.1392

Table 2. 2 x 2 ANOVA on Difference Scores

Source	*df*	*SS*	*MS*	*F*	*p*	*η*
Between Subjects						
Personalization	1	.003	.003	.153	.697	.003
Worked Examples	1	.001	.001	.048	.828	.001
Personalization x Worked Examples	1	.041	.041	2.074	.155	.034
Error (Between)	59	1.157	.020			

Table 3 shows the results of the 2 x 2 x 2 ANOVA we conducted to assess the effect of time. As can be seen, the effect of time is significant (p= .000).

Because of the indeterminate results, we decided to further explore the data. In particular, since we suspected that a large portion of our subject pool was relatively experienced in chemistry, we divided our subjects into three groups: "novices" (18 who scored 71% or below on the pretest), "proficient" (22 who scored greater than 71% but less than or equal to 84%), and "expert" (23 who scored equal to or greater than 84%). The novice group achieved the largest mean score gain from pre to posttest by a wide margin (novice increase in mean: 29%, proficient: 12%, expert: 3%). We also ran a two-way ANOVA on the novice group and, while there was still no significant main effect of worked examples, the difference between the worked examples conditions and the supported problem solving conditions was greater than that

Table 3. 2 x 2 x 2 repeated measure ANOVA

Source	*df*	*SS*	*MS*	*F*	*p*	*η*
Between Subjects						
Personalization	1	.001	.001	.046	.830	.001
Worked Examples	1	.039	.039	2.878	.095	.047
Personalization x Worked Examples	1	.016	.016	1.190	.280	.020
Error (Between)	59	.798	.014			
Within Subjects						
Time	1	.604	.604	61.647*	.000	.511
Error (Within)**	59	.578	.010			
Total	125					

* $p < .001$

** Time x Personalization, Time x Worked Examples, Time x Personalization x Worked Examples are not shown because they are not of interest to the analysis of this study

of the population as a whole. Taken together, these findings provide some indication that worked examples may have had more effect on students with less expertise.

Since we suspected that the lack of a worked examples main effect was due, at least in part, to subjects not fully watching and/or processing the worked examples, we also did an analysis of video viewing. We found that percentage of subjects who fully watched the worked examples started at 48% for the first worked example, but dropped, in almost monotonic fashion, to 26% by the last worked example.

4 Discussion of the Study Results

Our hypotheses that personalization and worked examples would improve learning were not supported, as there were no significant main effects or a significant interaction effect. We were particularly surprised about the lack of effect from worked examples, given the substantial supporting body of research. Why might this have occurred? We have three hypotheses.

First, the subjects in our study may have had too much stoichiometry expertise, potentially offsetting the working memory advantage afforded by studying examples. Previous research has shown that the more expert students are, the less they gain from studying worked examples and the more they gain from problem solving [18]. The students in the UBC chemistry class for majors, which likely constituted a larger proportion of the subjects in our study, would almost surely have had stoichiometry in high school. In addition, because the study was optional and a small percentage of the possible subject pool actually finished the study (< 5%), it is also likely that the majority of the subjects who finished participated precisely because they were confident they already knew the material. This conjecture is supported by the relatively high pretest scores across all conditions and an optional post-study survey in which a high percentage of the subjects claimed the material was not difficult.

Second, worked examples have been shown to be most effective when the learner self-explains them [19] or tackles them as "completion" problems (i.e., the subject observes part of the solution, then completes the problem on their own, see e.g., [1], pgs. 177-178). Both approaches help the learner process solutions at a deeper level. The worked example videos in our study were only observed; the subjects were not prompted to self explain or complete them. Also, as discussed above, a high percentage of subjects did not fully watch the videos. On the other hand, active processing of worked examples did not definitively lead to better results in the Mathan studies [15]; only the subjects who used the expert model ITS showed significant learning gains by actively processing examples and this result was not replicated in a second study.

Third, and most intriguing with respect to worked examples employed in conjunction with intelligent tutors, it may be that the tutoring received by the subjects simply had much more effect on learning than the worked examples (or personalization). As mentioned above, there was a significant difference between the pretest and posttest across all conditions. Thus, the subjects did appear to learn, but the personalization and worked example interventions did not appear to make the difference. Since we didn't directly compare supported problem solving (i.e., with tutoring) with regular problem solving (i.e., without tutoring), we can't definitively attribute the positive learning effects to tutoring. On the other hand, the analysis we did on the 18 novice performers indicates that the worked examples may have had the desired effect on at least that class of subjects.

Assuming the explanation of these results is not that tutoring "swamped" the effects of the other interventions, we have several hypotheses as to why personalization did not make a difference. First, many of our subjects may have been non-native English speakers and thus missed the nuances of personalized English. We did not collect demographic information, but the UBC chemistry professor said "perhaps more than 50% of the students were non-native English speakers." Second, as with worked examples, the fact that our subjects were not novices may have made it difficult to get an effect. Finally, perhaps our conceptualization and implementation of personalization was not as socially engaging as we had hoped. In a recent study [20], Mayer and colleagues investigated the role of politeness in the conversational style of an on-screen tutor. In a polite version of their system face-saving constructions were used such as, "You could press the ENTER key", and in the direct version, the tutor used direct constructions such as, "Press the ENTER key." Students learned more with the polite tutor, suggesting that providing an on-screen agent with social intelligence makes a difference. In other words, this study suggests that it is not just conversational first and second person language, such as that employed by the stoichiometry tutor, that makes a difference, but the development of a real social relationship with the learner.

5 Conclusions

In this, our first experiment applying Clark and Mayer's e-Learning principles to web-based intelligent tutoring, our results were somewhat disappointing. None of our three hypotheses, relating to the affordances of personalization and worked examples, was supported. On the other hand, we have demonstrated that web-based tutoring *can* be effective, as shown by the significant learning gains across all conditions. In summary, because of the issues cited above, we view this as a preliminary study whose purpose was to help us develop a workable methodology for testing the effects of personalization and worked examples.

We are currently running a second experiment aimed at U.S. high school students, a pool of subjects that is arguably more appropriate for our stoichiometry materials. We have modified the worked examples, so that subjects must fully watch the videos and correctly answer several self-explanation questions before moving on to the next problem. We conjecture that this will increase the effectiveness of the examples but, as witnessed by the mixed results of the Mathan studies, it is not certain this will make a difference. With respect to personalization, we did not make changes to the study problems, since we believe a more appropriate subject pool of (mostly) native English speakers may lead to different results.

Acknowledgements. This research was conducted within the Pittsburgh Science of Learning Center under NSF Award #0354420.

References

1. Clark, R. C. and Mayer, R. E. (2003). *e-Learning and the Science of Instruction*. Jossey-Bass/Pfeiffer.
2. Murray, T., Ainsworth, S., & Blessing, S. (eds.) (2003). *Authoring Tools for Advanced Technology Learning Environments: Toward Cost-Effective, Adaptive, Interactive, and Intelligent Educational Software*. Kluwer Acad. Publishers.
3. Koedinger, K. R., Aleven, V., Heffernan, N., McLaren, B. M., and Hockenberry, M. (2004) Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration; *Proceedings of ITS-2004*, 162-174.
4. Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *Int'l Journal of Artificial Intelligence in Education, 8*, 30-43.
5. Beck, I., McKeown, M. G., Sandora, C., Kucan, L., and Worthy, J. (1996). Questioning the author: A year long classroom implementation to engage students in text. *Elementary School Journal, 96*, 385-414.
6. Moreno, R. and Mayer, R. E. (2000). Engaging students in active learning: The case for personalized multimedia messages. *Journal of Ed. Psych., 93*, 724-733.
7. Kolb, D. A. (1984). *Experiential Learning - Experience as the Source of Learning and Development*, Prentice-Hall, New Jersey. 1984.
8. Sweller, J. (1994). Cognitive load theory, learning difficulty and instructional design. *Learning and Instruction, 4*, 295-312.
9. Paas, F. G. W. C. (1992). Training strategies for attaining transfer of problem-solving skill in statistics: A cognitive load approach. *Journal of Ed. Psych., 84*, 429-434.
10. Renkl, A. (1997). Learning from Worked-Out Examples: A Study on Individual Differences. *Cognitive Science*, 21, 1-29.
11. Trafton, J. G. and Reiser, B. J. (1993). The contributions of studying examples and solving problems to skill acquisition. In M. Polson (Ed.) *Proceedings of the 15th annual conference of the Cognitive Science Society*, 1017-1022.
12. Gott, S. P., Lesgold, A., & Kane, R. S. (1996). Tutoring for Transfer of Technical Competence. In B. G. Wilson (Ed.), *Constructivist Learning Environments*, 33-48, Englewood Cliffs, NJ: Educational Technology Publications.

13. Aleven, V. & Ashley, K. D. (1997). Teaching Case-Based Argumentation Through a Model and Examples: Empirical Evaluation of an Intelligent Learning Environment, *Proceedings of AIED-97*, 87-94.
14. Conati, C. and VanLehn, K. (2000). Toward Computer-Based Support of Meta-Cognitive Skills: a Computational Framework to Coach Self-Explanation. *Int'l Journal of Artificial Intelligence in Education, 11*, 398-415.
15. Mathan, S. and Koedinger, K. R. (2002). An Empirical Assessment of Comprehension Fostering Features in an Intelligent Tutoring System. *Proceedings of ITS-2002*. Lecture Notes in Computer Science, Vol. 2363, 330-343. Berlin: Springer-Verlag
16. Mathan, S. (2003). *Recasting the Feedback Debate: Benefits of Tutoring Error Detection and Correction Skills*. Ph.D. Dissertation, Carnegie Mellon Univ., Pitts., PA.
17. Bonate, P. L. (2000). *Analysis of Pretest-Posttest Designs*. Chapman & Hall/CRC.
18. Kalyuga, S., Chandler, P., Tuovinen, J., and Sweller, J. (2001). When problem solving is superior to studying worked examples. *Journal of Ed. Psych., 93*, 579-588.
19. Chi, M.T.H., Bassok, M., Lewis, M. W., Reimann, P., and Glaser, R. (1989). Self-Explanations: How Students Study and Use Examples in Learning to Solve Problems. *Cognitive Science*, 13, 145-182.
20. Mayer, R. E., Johnson, L., Shaw, E., & Sahiba, S. (2006). Constructing computer-based tutors that are socially sensitive: Politeness in educational software. International Journal of Human Computer Studies, 64, 36-42.

A Plan Recognition Process, Based on a Task Model, for Detecting Learner's Erroneous Actions

Naïma El-Kechaï and Christophe Després

Laboratoire d'Informatique de l'Université du Maine (LIUM)
Avenue Laënnec, 72085 Le Mans Cedex 9, France
{elkechai, despres}@lium.univ-lemans.fr

Abstract. When a tutoring system aims to provide learners with accurate and appropriate help and assistance, it needs to know what goals the learner is currently trying to achieve, what plans he is implementing and what errors he is making. That is, it must do both plan recognition and error detection. In this paper, we propose a generic framework which supports two main issues (i) the detection of learner's unexpected behavior by using the Hollnagel classification of erroneous actions and (ii) a recognition process based on a task model METISSE that we propose. This model, which is used to describe the tasks the learner has to do according to pedagogical goals, allows learner's unexpected behavior to be detected. The solutions proposed are generic because not dependent on the domain task, and they do not relate to a particular device.

1 Introduction

The general context of this work is the design of Interactive Learning Environments (ILE). More precisely, we are interested in Learning By Doing (LBD) activities [1], that aim at enabling learners to develop skills related to procedural tasks in the vocational training area. For example, the training of train drivers of the SNCF (the French National Railway Company) to handle the switches on high speed train tracks.

In the ILE, the learner who carries out a given task produces an activity. We propose to analyze this activity in order to advise, help, and provide the learner with appropriate and relevant guidance during the task achievement. An important requirement guides this work, which consists in proposing a generic solution.

The main issues of this work are how to represent the tasks the learner has to do, how to implement errors detection during the task achievement, and to define a generic framework that fully supports the first two issues. The goal pursued in this framework is to characterize learners' failure situations in order to make the tutoring system able to (a) determine and provide the learners with the most appropriate assistance and feedback, and (b) to provide a human trainer with relevant information on learners' activity and failure situations.

In this paper, we propose an approach based on a plan recognition process, which is based on a tutoring-oriented task model METISSE that we propose in order to describe the tasks the learner has to do according to pedagogical goals. This process allows unexpected learners' behavior to be detected by using the Hollnagel classification of

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 329 – 338, 2006.

erroneous actions [2] [3]. The solutions proposed are generic because they are not dependent on the domain task and do not relate to a particular device.

This work has been done in the context of the APLG project (Atelier Pédagogique Logiciel Générique, i.e. Computer-Aided Training Virtual Environment Engineering) [4]. APLG concerns the design and development of the next Virtual Environment for Learning (VEL). Its objective is to provide the instructional designers with generic customizable functions in order to assist the learner with pedagogical functions; and the trainer to prepare the pedagogical scenario and manage the learning situations.

We begin by addressing the two core issues of this work, namely the need to have the appropriate formalism to support the description of tasks in order to identify the learners' unexpected behavior. Then, we present a generic framework based on task modeling and plan recognition. Last, we discuss the next developments of this work.

2 Task Modeling and Error Detection

All the learning tasks the learner has to do throughout his course is called the *prescribed task*. It is described by taking into account the set of all possible solutions/paths. A *scenario* is an instance of the *prescribed task*. It corresponds to the trainer's choice about the tasks the learner has to do at a given time of his course.

2.1 Task Modeling

Instructional designers and trainers need a formalism to describe the entire set of possible (or at least acceptable) actions for achieving a task. Formalisms and models to describe human operator activity and tasks have been already developed in various other fields of research. In particular, we have considered the model MAD* [5] developed in ergonomics. Two main arguments are in favour of this interest. The first is that MAD* enables us to represent the hierarchical dimension of planning in the human activity. In the vocational training area, the prescribed tasks are often "codified" in hierarchical procedures aiming at orienting actions and defining what must be done to meet task requirements [6]. The second is that MAD* meets our requirement of generics, it does not relate to a particular device. Although MAD* has several advantages, we believe it suffers from a drawback. MAD* does not allow describing several ways to carry out a given task. Although it proposes alternatives, these depend only on world objects. To describe alternatives depending on teaching strategies, we adapt MAD* to the learning domain by integrating the task/method paradigm (§ 3.1).

2.2 Error Detection

The "human error research" has provided a large set of models to explain and/or predict the occurrence of errors in human behavior. Although these approaches have mostly applied to task performance in the context of critical systems, they could serve as a framework for monitoring errors during task performance in learning situations. In particular, Hollnagel [2] [3] proposed a clear distinction between manifestations, i.e. observable actions, and causes leading to the production of these actions. He proposes the term "erroneous actions" to describe a certain type of action without implying anything about the cause. To highlight this distinction, Hollnagel uses the terms *phenotype* to refer to patterns of erroneous action that are observable, and *genotype* to refer to patterns in the underlying causal mechanisms that lead to *phenotypes*. The *phenotypes* are structured in *simple* (concerning one single action), and *complex* (covering complex sets of actions) *phenotypes* (Fig. 1).

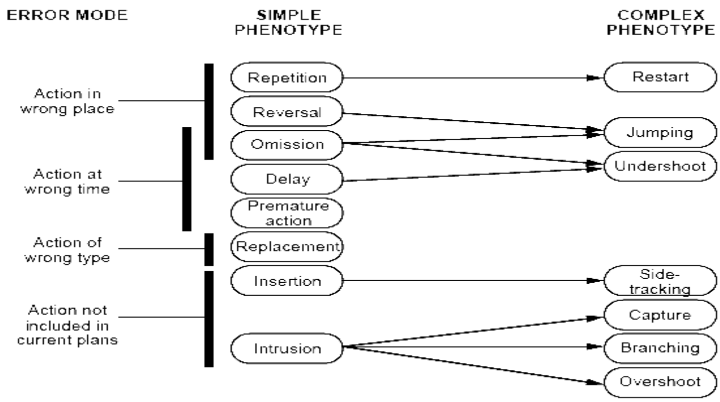


Fig. 1. Hollnagel classification of phenotypes of erroneous actions (from [2])

This classification is relevant to our work for many reasons. First, it fits tasks having a hierarchical decomposition with time and ordering constraints and particularly to the task model METISSE that we propose and which is described below. Second, this classification is based on observable actions. The system did not make any assumptions about causes of erroneous actions. The part of the training process that addresses causality can be left to the human trainer who may nevertheless need relevant information on the type of erroneous actions. Third, erroneous actions are clearly described and the classification is easy to implement satisfactorily.

3 Our Approach

3.1 The Task Model METISSE

We propose METISSE (Modèle de dEscription de Tâches pour l'assIStance et le Suivi de l'apprEnant), a tutoring-oriented task model for describing the tasks the learner has to do according to prescribed pedagogical goals. METISSE is based on the ergonomics task model MAD* [5] adapted to the learning area by integrating the Task/Method paradigm [7] [8]. METISSE describes the prescribed task in terms of *Tasks* and *Methods* which correspond respectively to declarative (what to do) and procedural (how to do it) characteristics of a task. A *task* can be achieved by several *methods*, and achieving a *task* with one *method* or another can influence the accomplishment of the remaining *tasks*. The primitives *Task* and *Method* of METISSE are defined by several items (Fig.2). We detail some of them and give examples of their use for feedback or error detection.

Task	Method
Name ***Objective*** ***Expected Results*** ***Attributes:*** *Option, Interruption, Time [Start, End, Duration], Significance.* ***Conditions:*** *Execution, Triggering, Specific, Stop* ***Associated Methods with preferences*** *(Preference of realization, Pedagogical preference)*	***Name*** ***Results*** ***Input Context*** ***Favorable Context (CF)*** ***Type :*** ***- Simple*** *(describes action).* ***- Decomposition*** : *Constructor + {subtasks list}* *where Constructor* ∈ {SEQ, PAR, ET, OU, SIM}

Fig. 2. Task and Method Items

3.1.1 The Primitive *Task*

• The task *Objective* is used for explanations to provide the learner with, and the task *expected results* are used to compare them to the learner's results;

• Each task has four attributes: *option*, *interruption*, *time*, and *significance*. *Option* indicates whether the task is optional or not. Concretely, this attribute enables us not to take into account an optional task when it is omitted by the learner. *Interruption* indicates if the task can be interrupted or not during execution. For example, an external event such as an emergency signal can trigger the execution of a particular task which can interrupt the current one if it is interruptible. *Time*, which has three slots *start*, *end*, and *duration*, indicates the time window associated with the task. This attribute provides information to compute temporal interdependences between tasks and also to evaluate the tasks' duration. Finally, the *significance* attribute indicates the importance of the task compared to the others. An example of feedback rules using this attribute is {"If a task Ti has a Main significance" and "the learner made many errors when performing it" and "Ti has some subtasks with less significance"} then {propose to the learner a simplified version of Ti by deleting these subtasks}. The aim of this feedback is to lead the learner to turn his attention to what is most important.

• The task *conditions* are constraints on the world objects and on time. They are of four types: *execution* (when the task can be achieved), *triggering* (when the task has to be achieved), *specific* (for optional tasks), and *stop* (for iterative and repetitive tasks) conditions. These conditions are used to detect errors such as when a task is carried out by the learner while its conditions are not fully satisfied.

• The *Associated Methods* with their *preferences* are a list of the methods that can achieve a given task with their preferences. Top-level tasks are associated with one or several methods called *decomposition methods* that split them into subtasks, recursively, until these subtasks are sufficiently simple to be directly tackled by *operational methods*. Each Associated Method has two types of preferences: the first relates to the *realization* of the task, classifying its methods by order of performance; and the second relates to *pedagogical* or *teaching preferences*. These preferences are defined and set by trainers.

3.1.2 The Primitive Method

The primitive *Method* is defined by the *Results* it produces; the *Input Context*, which corresponds to the conditions under which the Method can be achieved; and a *Favorable Context,* which describes when the Method is particularly relevant. For example, we can use this item to induce the learner to perform the task with a given method by ensuring that its favorable context is created. The *Input Context*, *Favorable Context* and *Results* of methods are expressed as features of world objects.

There are two *Types* of methods. *Operational methods* can directly carry out a task, they describe actions. *Decomposition methods* split tasks into subtasks; they consist of a *constructor* and a *list of subtasks*. A *constructor* specifies dependencies and time order of subtasks. There are five constructors: SEQ (sequence), PAR (parallelism), SIM (simultaneity); AND; and OR.

Task and *Method* primitives of METISSE use *world objects* in some of their items. To describe these objects, we use a model composed of three elements (class, instance, attribute) and organized them hierarchically.

3.2 The Plan Recognition Process Uses the Task Model METISSE

To infer the plans the learner is implementing, we use a recognition process entirely based on the prescribed task description done with METISSE. We use a *keyhole recognition* [9] and we assume holes in the sequence of observable actions [10]. We make this assumption because of two linked reasons. First, VEL are considered as open environments where learners can carry out actions which are unrelated to the prescribed task and which are not considered for interpreting their behavior. Second, we do not assume that the task description is exhaustive; there may be unobservable actions with some effects that influence the rest of the task accomplishment.

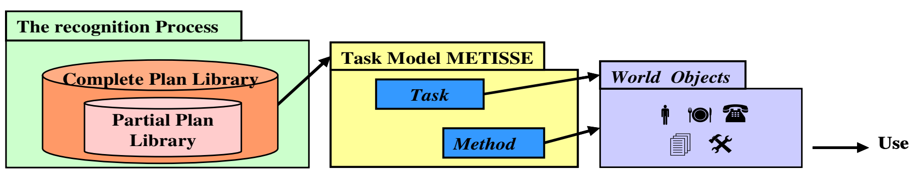


Fig. 3. Plans use the task model METISSE which uses the world objects

To interpret the learner actions, plans (defined below) are derived from tasks and methods (task model), which use the world objects (Fig.3). To consider all plans, we "mark" those belonging to the scenario during the recognition process; that is why there are two libraries: *partial* and *complete libraries*.

3.2.1 Plan: Definition, Type and Conditions

A *Plan* is the association of a task with one of its methods. There are as plans as methods for a given task. Plans are of two types: *simple plans* are associated with *operational methods,* and *composite plans* with *decomposition methods*. Each plan has two types of conditions: *Pre-Conditions* and *Post-Conditions* which are expressed according to task and method conditions, and task and method results.

3.2.2 Plan: States

In order to make plans traceable during the learner activity, we characterize them by states indicating their development during the recognition process. We identified two kinds of states: *conditional* and *dynamic states*.

The *conditional state* relates to the conditions of *activation*, *triggering*, and *realization* of a plan, i.e. provides information on the satisfaction of its *Pre-Conditions* and *Post-Conditions*. For example, we can detect that a plan is started by the learner whereas its *Pre-Conditions* are not satisfied; or that a plan has to be carried out by the learner immediately, for example if the office intruder alarm system is triggered, an emergency procedure should be performed. We can also detect that a plan is not started or is not completely carried out by the learner whereas its results are already produced. This situation may occur if the learner obtained an expected result by following an "unknown" way or after observable effects of unobservable actions.

The *dynamic state* consists of three aspects. The first relates to the plan itself. It indicates when the learner starts, continues, suspends, or finishes a given plan. For example, a *simple plan* becomes *finished* if the learner carried out the corresponding action. There is a difference between a plan which is *finished* (carried out by the learner) and a plan which has its *Post-Conditions* satisfied. The second aspect provides information about two particular relations which can exist between two or several plans: *conflict* and *concurrence*. A *conflict* arises when there is ambiguity regarding how to interpret some given actions. In other words, two or several plans are conflictual when they become *active* or *finished* following the same observed action. A *concurrence* arises when the learner uses two or several methods to carry out the same task, i.e. two or several plans are *concurrent* if they become *active*, *pending* or *finished* and being associated with the same task. Plans can be *conflictual* and *concurrent* at the same time. We consider plans as autonomous entities having a local vision, thus *conflict* and *concurrence* situations are detected by their respective managers, which have a global vision (§ 3.3.2). The last aspect provides information about *errors* that plans can detect. This aspect is detailed in local plans' analysis.

3.3 The Different Stages of the Plan Recognition Process

When an event occurs in the VEL, due to learner's actions or world's changes, the recognition process proceeds as follows (1) the first stage is done locally for each plan and consists in updating plans' states. The *simple plans* are the first to treat provided data because they correspond to observable actions. Analysis continues by bottom-up spreading of local analyses from *simple plans* towards their parent until reaching the root plans. *Composite plans* define their state by taking into account the local analysis of all their child plans; (2) the second stage is the global analysis and consists in identifying and solving *conflict*, *concurrence* and *errors* which have appeared during the last stage. The three problems are treated by their corresponding managers; (3) the last stage consists in sending the interpretation results. The candidate plans (supposed to be followed by the learner) are classified according to the heuristics used by the managers. Consequently, the outcomes of the recognition process can be one or several plans. Another process, currently still in progress and out of the scope of this paper, will use specific knowledge to disambiguate and decide which plans are more plausible, and to decide what assistance can be proposed. This process will influence the progress of the recognition process since the remaining plans (not considered) will be reinitialized and updated.

3.3.1 The Local Analysis of Plans

To make their analyses, *simple* and *composite plans* follow the same general process. They update their state and inform their parent even if their state doest not change. If they detect potential errors, according to the Hollnagel classification, they inform the *error manager* as being erroneous. But to do these analyses, *simple* and *composite plans* do not focus on the same knowledge.

Local Analysis of Simple Plans

A *simple plan* compares directly the observed action to the action defined in its method following the algorithm illustrated by Fig. 4.

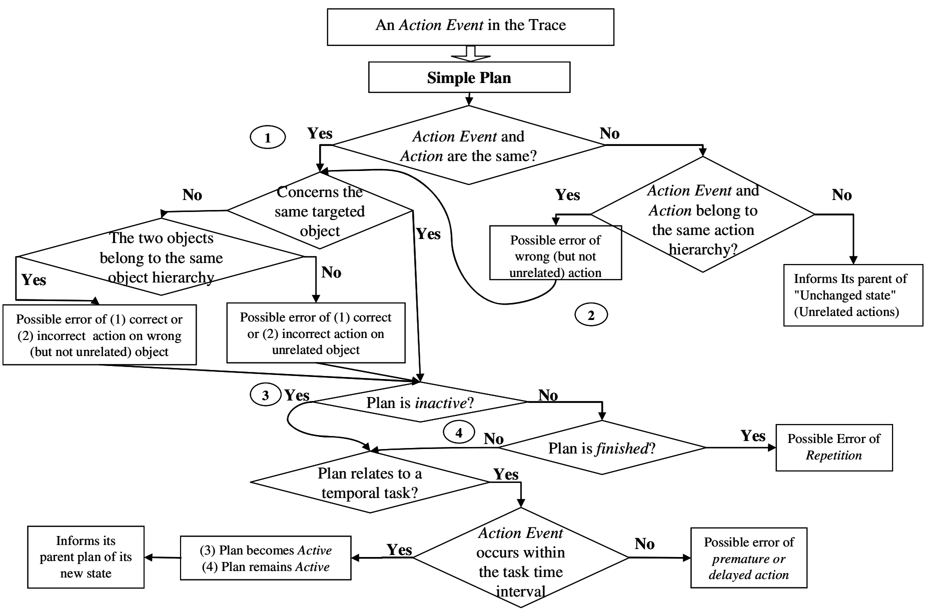


Fig. 4. Simple plan's analysis

Before going further, we precise that we use a *hierarchy of actions* (with the *world objects hierarchy*) in order to detect some execution errors due to similarities of actions, such as correct actions performed on wrong objects or unexpected actions on correct objects. Three elements compose this hierarchy: *ActionClass, Action* and *ActionEvent*. *ActionClass* allows modeling actions disregarding their context, with generalization and specialization links. For example the *ActionClass* "Move" can be applied to any world object having "Movable" *attribute*. An *Action* corresponds to an operational method in the task model, for example "Take the left key". An *ActionEvent* is an occurrence of an *ActionClass* targeted on a world object, for example "Press Emergency button". We remain aware that interpreting the learner's actions in a fine way may imply fastidious work in terms of knowledge representation. But we believe that (i) a small number of disjoined *ActionClass* hierarchies may involve many detections of *conflict*; (ii) a high number may involve several detections of erroneous actions. A good balance between these two solutions depends on what erroneous actions the instructional designer want to detect.

So a simple plan is carried out without errors if before the occurrence of the *Action Event* its *Pre-Conditions* are satisfied, it was *inactive*, its *Action* and the *ActionEvent* are the same and targeted at the same object, and its *Post-Conditions* are satisfied after this action; else it considers itself as erroneous and informs the *error manager.*

Local Analysis of Composite Plans

Composite plans correspond to abstract tasks, i.e. not directly related to the observable actions. When an event occurs in the *Trace*, they wait until all its child plans finish their local analyses.

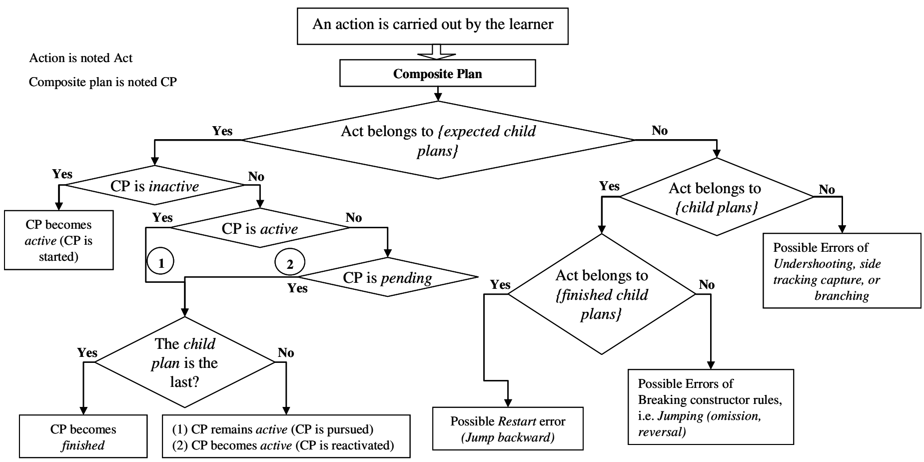


Fig. 5. Composite plan's analysis

To do its analysis, each composite plan needs two lists: *expected child plans* and *finished child plans lists*. The first list represents the various child plans that the learner can follow at time t+1. It is defined by the *constructor* of its *associated method*. The second list represents the child plans already carried out by the learner. To make its analyses, a *composite plan* follows the algorithm of Fig. 5. For example, if the plan concerned by the action does not belong to the *child plans list* of the *composite plan*, then a possible error exists. Two interpretations are possible:

- the learner intended to carry out the plan but, at a given time, followed an unrelated one. It may be an error of "Jump forward", "side tracking", "capture", or "branching".
- the learner temporarily suspended the execution of the plan and followed another one. It is typically the case of parallel tasks (PAR constructor) of a higher level.

In both cases, a learner's unexpected behavior can be identified by the error manager, which has a global vision of all plans.

3.3.2 The Global Analysis of Plans

Plans have a local thus limited vision. To identify problems of *conflict*, *concurrence* and *error* appearing during plans' analysis, it is necessary to have managers with a global vision. These managers focus on different knowledge to classify the candidate plans likely to give the best interpretation of a learner's behavior. They deal with their respective problem in parallel by taking into account the previous analyses. Before underlining the managers' heuristics, we point out that there are two types of conflict.

- The plans are associated with different tasks and methods: the *conflict* is situated at the action level in the sense that the *ActionEvent* of the *Trace* is close to two or several simple methods. For example, the two tasks [Press button1] and [Press button2] correspond to two different simple plans of the *plans library* and have the same *ActionClass*. When the learner presses button1, he may carry out the first task correctly or the second in an erroneous way (he did not press the correct button). We do not presuppose that the action corresponds to the learner's intention.
- The plans are associated with the same task and the same method. In this case the conflict relates to the parent plans. For example, "Press on the button of the elevator's door" belongs to two plans "Go up on the third floor" and "Leave the building".

To classify candidate plans, *conflict* and *concurrence managers* use some heuristics. The *Scenario heuristics* puts in order of priority the plans of the scenario, i.e. those belonging to the partial plans library. The *Learner's activity heuristics* allow plans to be classified by taking into account the effective activity of the learner disregarding what is considered prescribed. They include heuristics based on rate of completion (ratio between finished child plans and total child plans); rate and type of erroneous actions; satisfaction of *Pre-Conditions* of the next expected child plan (anticipation of what the learner may do in the next step). The *Relevance heuristics*, which is the most significant heuristics of the *concurrence manager* is based on the *favorable context* of methods. The plans having the *favorable context* of their method satisfied are better classified than others.

4 Conclusion

The main issues of this work are characterizing learners' unexpected behavior, and specifying the appropriate model to support the description of the prescribed task for this purpose. In order to assess what the learner is doing and to interpret his behavior, we propose an approach of plan recognition process based on the prescribed task model. This model is described with METISSE, a tutoring-oriented formalism we propose. This process also uses the Hollnagel classification which provides a structured way of modeling a space of possible errors and which is adapted to METISSE. The proposed approach is generic because it is not dependent on a domain task and does not relate to a particular device.

Currently, the framework which supports the two main issues is implemented and our work is progressing in two directions. First, modeling and implementing feedback rules corresponding to situations where learners find themselves in failure. These rules are used by the tutoring system to provide learners with accurate and appropriate assistance. They are expressed thanks to a high-level exploitation of the task model (preferences of methods, significance of tasks etc). These rules also use other knowledge; learner's profile (preferences, learning style, frequent errors etc), tutorial tactics and pedagogical strategies which can be implemented, and knowledge about the target vocational training domain. The second direction concerns the way the tutoring system provides the human tutor with appropriate information about the learner's activity. As said before, the part of the training process that addresses causality about the learner's unexpected behavior can be left to the human trainer who may nevertheless need relevant information on the type of erroneous actions.

Acknowledgements. The APLG project is cofunded by the French Ministry of Education and Research under the RIAM (Recherche et Innovation en Audiovisuel et Multimédia) program. The APLG project currently includes the Centre of Robotics of Paris, the LIUM at the University of Maine, the LEI at University of Paris 5, the AFPA (French National Organization for Vocational Training of Adults), CLARTE, SNCF (French National Railway Company), Daesign, and CS.

References

1. Schank, R.C., T.R. Berman and K.A. Macperson, *Learning by doing*, in *Instructional-Design Theories and Models: A New Paradigm of Instructional Theory*, C.M. Reigeluth, Editor. 1999, Mahwah, NJ: Lawrence Erlbaum Associates. p. 161–181.
2. Hollnagel, E., *The phenotype of erroneous actions.* International Journal of Man-Machine Studies, 1993. **39**: p. 1-32.
3. Hollnagel, E., *Looking for errors of omission and commission or the hunting of the Snark revisited.* Reliability Engineering and Systems Safety, 2000. **68**: p. 135-145.
4. Lourdeaux, D., J.-M. Burkhardt, D. Mellet d'Huart, et al. *APLG Project: a library for Learning Virtual Environments*. in *VRIC'05, International VR-Learning Seminar*. 2005. Laval, France.
5. Scapin, D. and J.M.C. Bastien, *Analyse des tâches et aide ergonomique à la conception: l'approche MAD**, in *Analyse et conception de l'IHM*, KOLSKI, Editor. 2001: France. p. 85-116.
6. Leplat, J., *Éléments pour l'étude des documents prescripteurs.* Activités, 2004. **1**(2): p. 195-216.
7. Trichet, F. and P. Tchounikine, *DSTM: a Framework to Operationalize and Refine a Problem-Solving Method modeled in terms of Tasks and Methods.* International Journal of Expert Systems With Applications (ESWA), 1999. **16**: p. 105-120.
8. Choquet, C., F. Danna, P. Tchounikine, et al., *Modeling the Knowledge-Based Components of a Learning Environment within the Task/Method Paradigm*, in *ITS'98*, Springer-Verlag, Editor. 1998: San Antonio (USA). p. 56-65.
9. Kautz, H.A. and J.F. Allen. *Generalized plan recognition*. in *AAAI-86*. 1986. Philadelphia (USA).
10. Pynadath, D.V. and Wellman, P. *Accounting for Context in Plan Recognition with Application to Traffic Monitoring*. in *UAI95*. 1995. Montreal, Canada.

Handling Errors in Mathematical Formulas

Helmut Horacek[1] and Magdalena Wolska[2]

[1] Fachrichtung Informatik, Saarland University, Germany
[2] Fachrichtung Allgemeine Linguistik, Saarland University, Germany
horacek@ags.uni-sb.de, magda@coli.uni-sb.de

Abstract. In tutorial systems, effective progress in teaching the problem-solving target is frequently hindered by expressive sloppiness and low-level errors made by the student, especially in conventionalized expressions such as formulas. In order to improve the effectiveness of tutorial systems in teaching higher-level skills, we present a fault-tolerant formula interpreter that aims at finding plausibly intended, formally correct specifications from student statements containing formal inaccuracies. The interpretation consists of local changes based on categorization of errors, a fault-tolerant structure building, and testing contextually-motivated alternations. The error interpretation component is intended to enhance the analysis component of a tutorial system that teaches mathematical proving skills.

1 Introduction

Teaching problem-solving skills is among the most challenging goals of tutorial systems. Application areas in formal domains include algebra word problems (Ms. Lindquist [5]), elementary geometry [9], electrical engineering [15], and qualitative physics [8]. In practice, effective communication about the problem-solving target is frequently hindered by expressive sloppiness and low-level errors made by the student, especially when specifying formulas. When the given task merely consists in building a single formula as, for example, in the Ms. Lindquist system, interpreting errors can be grounded in expectations about the solution. However, when there is considerable degree of freedom in constructing steps towards a solution, handling errors adequately is more delicate. It requires revealing the probably intended and possibly correct specification, as well as addressing the error according to the tutorial strategy pursued – just reporting it, explicitly asking the student for correction, or producing a contextually-adequate hint.

In order to improve the effectiveness of tutorial systems in teaching higher-level skills, we present a fault-tolerant formula interpreter that aims at finding plausibly intended, formally correct specifications from student statements containing formal inaccuracies. Ingredients of this interpretation are local changes based on categorization of errors, a fault-tolerant structure building, and testing contextually-motivated alternations. We have performed elaborations for some fragment of elementary set theory.

This paper is organized as follows. We first describe the tutorial scenario in which this work is embedded and briefly present the collected data. We motivate formula error categories and methods for their detection. We follow with a presentation of our fault-tolerant formula analyzer. We describe an algorithm that handles the categories of errors and generates hypothesis of intended specifications. Finally, we present an evaluation.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 339–348, 2006.

2 Our Tutorial Environment and Data

The work described in this paper is part of the DIALOG project[1] [2]. The goal of the project is (i) to empirically investigate the use of flexible natural language dialog in tutoring mathematics, and (ii) to develop a prototype tutoring system that incorporates the empirical findings. The experimental system will engage a student in a dialog in written natural language to help him/her understand and construct mathematical proofs.

We envision a modular system architecture, by use of the proof system ΩMEGA [10]. For tutorial purposes, ΩMEGA has been adapted to support proofs in a human-adequate form [12]. Interaction with ΩMEGA is mediated by a *Proof Manager* (PM) [3]. The task of the PM is to communicate with the proof system to check consistency and validity of (possibly ambiguous) interpretations of student utterances within the proof context, and to build and maintain a representation of the constructed proof. Moreover, based on feedback from ΩMEGA, the PM evaluates proof-relevant parts of the utterances with respect to completeness, correctness, and relevance, where *correct* proof steps may not necessarily be *relevant* in that they do not contribute to the progress in finding the proof solution. This categorization is an integral part of our tutorial strategies [11].

Table 1. Examples of flawed formulas from the corpus

	Example Formula	*Error Category*
(1)	$P((A \cup C) \cap (B \cup C)) = PC \cup (A \cap B)$	3
(2)	$(p \cap a) \in P(a \cap b)$	2
(3)	$(x \in b) \notin A \qquad x \subseteq K(A)$	2
(4)	$P((A \cap B) \cup C) = P(A \cap B) \cup P(C)$	1
(5)	$(A \cap B) \subseteq P(A \cap B)$	1
(6)	if $A \subseteq K(B)$ then $A \notin B$	2

To investigate phenomena characterizing written computer-mediated tutorial dialogs, we collected a corpus of tutor-student dialogs in a Wizard-Of-Oz experiment in the domain of naive set theory [4, 14]. An overview of language phenomena observed in the corpus is presented in [1]. In Table 1, we present examples of flawed formulas from our corpus. In (1), a structural error is shown: not only a space between the operator symbol P and the identifier C, but also parentheses are missing. (2) is an example of a typographical error, where an operator symbol p has been used in place of an identifier b. In (3), the types of arguments of the main operator are invalid. In (4), a stronger assertion of set inclusion ($\subseteq$) rather than equality is expected. Finally, (5) and (6) are examples of commonly confused relations of *subset* and *membership*.

In our experiment, it turned out that resolving these errors sometimes required considerable effort through entering longish clarification subdialogs, since an error was spotted, but not specifically investigated. Therefore, we are looking at methods for handling these situations on a somehow higher level, which requires tentatively resolving errors, if possible, so that more emphasis can be given to problem-solving issues.

[1] The DIALOG project is part of the Collaborative Research Center on *Resource-Adaptive Cognitive Processes* (SFB 378) at University of the Saarland http://www.coli.uni-sb.de/sfb378/.

3 Categories of Errors and Associated Correction Attempts

We aim at finding a corrected and possibly intended version of a flawed formula by applying purposeful changes to a that formula. An essential factor in choosing such changes lies in the correctness status of the formula with respect to the statement that embeds it and the task environment. In our scenario, two interpretation components contribute to assessing the role of a formula in a complementary way: 1) the formula analyzer, and 2) the Proof Manager. The formula analyzer delivers one of three results:

1. It reports a *structural (i.e., syntactic)* error (error category 3) if it is impossible to build an analysis tree on the basis of the constructors defined – pre- and infix operators with given arity, variables and constants.
2. It reports a *type (i.e., semantic)* error (category 2) if it succeeded in building an analysis tree which contains a type mismatch according to operator definitions.
3. It reports *no* error if it succeeded in building an analysis tree that is also correct regarding the type requirements.

Only for well-formed formulas consulting the Proof Manager is meaningful. Again, three results are possible: 1) the formula may simply express a *wrong* statement (error category 1), 2) the formula is *correct* in principle, but it does not contribute to the task at hand, and 3) the formula is correct and also *relevant* for solving the given task.

A distinction between the cases assessed as *correct* or *relevant* is of no interest for our purposes (they count as error category 0), and it is up to the embedding tutorial strategies to deal with them. In all other cases, attempts are undertaken to remedy the error by applying local and contextually justified modifications to the formula. These changes are intended to address merely typing errors or conceptual inaccuracies, so that the result can be assessed at the problem-solving level. We structure the task of testing changes according to the error category reported (see above), aiming at changes that achieve an improvement of at least one category level. To constrain the application of meaningful changes, we assume a context consisting of a set of identifiers (i.e., variables and operators), together with type and arity information (the latter only for operators).

Replacement rules and their associated error categories are illustrated in Table 2 with examples taken from our corpus. While we conjecture that some of the rules are of a more general use than only for our application domain, elementary set theory, the concrete elaborations and examples are focused on the observations from our corpus.

Table 2. Replacement rules attempting to remedy errors

Replacement Rules	*Error Categories*	*Examples* (set theory)
dual operators	1	$\cap \Leftrightarrow \cup$, $\subset \Leftrightarrow \supset$
stronger/weaker operators	1	$\supset \Leftrightarrow \supseteq$, $\subseteq \Leftrightarrow =$
confused operators	1,2	$\subset \Leftrightarrow \in$, $K \Leftrightarrow P$
confused identifiers	1,2	$a \Leftrightarrow b$, $P \Leftrightarrow b$
delete character	2	$Pc \Rightarrow P$, $Pc \Rightarrow c$
insert parentheses	3	$Pc \Rightarrow P(c)$
insert a blank	3	$Pc \Rightarrow P\ c$

Typical errors of category 3 are missing separators, and unintended omissions or additions of characters. To keep the number of applicable changes manageable, we only deal with character deletions, when the result is a known identifier, but not with additions of characters. Similarly, we insert a blank as a separator, provided this replaces an unknown by two known identifiers. Typical errors of category 2 are confusion of identifiers, whose repair requires changing variables, of operators, or replacements of variables by operators or vice-versa. In all cases, type compatibility typically restricts the applicable changes considerably. Finally, typical errors of category 1 also include confusions of identifiers, but with the same type for variables, and with the same arity and type restrictions for operators. Moreover, some conceptual relationship, such as duality, or extended/restricted strength must exist between the operator present and the replacement candidate, to motivate the replacement as simulating a mental confusion.

4 Formula Analysis

The formula analysis procedure is part of the input interpretation component [6, 13] whose task is to interpret the content of the student utterances and to produce a representation that can be further analyzed by ΩMEGA in the context of the given task. Formula analysis is part of the input pre-processing procedure. It consists of three stages. Firstly, mathematical expressions are identified within word-tokenized text. Secondly, the identified sequence is verified as to syntactic validity and, in case of a parentheses mismatch, a correction procedure is invoked. Finally, the expression is parsed.

Identification of mathematical expressions. The tagger has access to a list of operation and identifier symbols relevant in the given context (e.g. in the context of naive set theory, characters P and K stand for power set and set complement respectively). Identification of mathematical expressions is based on simple indicators: single character tokens (including parenthesis), multiple-character tokens consisting only of known relevant characters, mathematical symbol unicodes, and new-line characters.

Syntactic analysis. Once a candidate string is identified, "chains" of formulas are separated into individual segments. For example, formula $x \in B \not\subseteq K(b)$ is split into $x \in B \wedge B \not\subseteq K(b)$. Next, parentheses match is verified. In case of a mismatch, missing parentheses are inserted while observing the following preferences: (i) provide parentheses for operators that require bracketed arguments (in our domain, e.g. K or P), (ii) avoid redundant parentheses (i.e. double parentheses around the same substring). For example: the string $P((A \cup B) \cap (B \cup C)) = P(C \cup (A \cap B)$ is corrected in two ways: $P((A \cup B) \cap (B \cup C)) = P(C \cup (A \cap B))$ and $P((A \cup B) \cap (B \cup C)) = P(C) \cup (A \cap B)$, while the sequence $(A \cup C) \cap (B \cup C))$ is corrected uniquely: $((A \cup C) \cap (B \cup C))$.

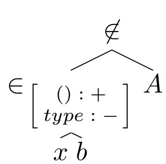


Fig. 1. Tree representation of the sequence $(x \in b) \notin A$

Parsing. Syntactically correct candidate sequences are parsed by a tree-building algorithm. It has access to a list of known operators and operator precedence. For unbracketed operators of the same precedence, all possible bracketings are considered (e.g. $A \cup C \cap B$ will be interpreted as ambiguous between $(A \cup C) \cap B$ and $A \cup (C \cap B)$). Additionally, for every tree node, the procedure stores information on whether the subtree headed by the given node was bracketed in the original string, and whether the types of arguments are consistent with the expected types. The output of the parser is the formula tree with nodes marked as to type compatibility and bracketing where applicable. For example, Figure 1 shows the tree obtained for the sequence $(x \in b) \notin A$. The feature structure associated with the $\in$ node represents the record stored with the formula tree for this node. In this case, the record $() : +$ indicates that the substring associated with the subtree headed by the node was bracketed, while the record $type : -$ indicates that the node is not of compatible type as an argument of the higher node.

5 Formula Modifying Algorithm

In this section, we describe our method for building formulas with modifications that are promising to remedy errors reported in the analysis of the original formula. In a nutshell, modifications to the original formula are made according to appropriate applications of operators considered suitable to remove the reported error, and the resulting new formulas are categorized by consulting the formula analyzer and, if needed, the Proof Manager. Since the latter may be an expensive step, care is taken that more promising alternatives are tested with priority. This process is continued iteratively, but it can be terminated if subsequent calls to the Proof Manager are getting too costly.

The algorithm produces an ordered list of hypotheses which are candidates for representing the intended specification. The ordering among these hypotheses is done by three criteria, the first one dominating the others, whose counts are accumulated:

1. the error-related category of the modified formula,
2. the number of operators applied to obtain the currently considered formula, and
3. the structural similarity of the obtained formula to formulas in the context, which we approximate by simply counting the number of operators and variables appearing in the formulas compared (multiple occurrence counting as frequently as they occur). The context is a set of formulas consisting of the goal expression, the previous proof step, and possible follow-up steps, according to the Proof Manager.

The schema of the algorithm $Generate\text{-}Hypotheses$ is illustrated in Figure 2. The procedure is invoked if the formula analyzer has assigned an error category to the formula to be evaluated. It has two parameters: the original $Formula$, including results obtained by the formula analyzer, and a set of formulas representing the $Context$.

The procedure consists of three parts. First, suitable operators for producing modified formulas are chosen – operators associated with the category of the error observed, for categories 1 and 2. For errors of category 3, the original $Formula$ already contains the results of the first two parts of the procedure since applications of these operators are already carried out within the pre-processing stage of the formula analysis component, so that the second part can be skipped when dealing with this error category.

In the second part, the chosen operators are applied to the formula, possibly at multiple places, but operators addressing category 2 errors only at positions marked as erroneous. All new formulas resulting from one such operator application are collected as $Hypotheses$, excluding results considered $Trivial$ (e.g., an equation with identical left and right sides, and applications of idempotent operators to identical arguments), and their error category is determined by consulting the formula analyzer: $Parse$.

In the third part, available hypotheses are assessed in a two-pass evaluation. In the first one, the similarities to the formulas in $Context$ are computed and for formulas of error category 1 a subsequent call to the Proof Manager is made. Since the latter can be expensive, all formulas created by applying operators are ordered according to contextual similarity, prior to invoking the $Proof\text{-}Manager$. The $Proof\text{-}Manager$ goes over the newly created formulas one after the other, starting with the one(s) ranked highest, so that the procedure can be stopped anytime if resources are exhausted – this criterion is encapsulated in the abstract condition $< Limit >$. The procedure terminates when the problem appears to be solved, that is, the category of some modified formula is superior to the category of the original one, when no more operators can be applied (including all category 3 formulas), or when resources are exceeded. If one of these cases is present, the ordered list of $Hypotheses$ is returned; otherwise, applying the selected operators is repeated interactively to the newly created formulas.

The procedure can accommodate several kinds of limits on the resources involved:

- a maximum number of modified formulas created; since only a rather limited set can be reasonably dealt with in the embedding discourse,
- a limit on the number of calls to the theorem prover,
- a time limit (a sort of an alternative to the preceding criterion),
- a limit on the number of errors, that is, operators to be applied.

6 Evaluation

In this section, we present the results of a quantitative evaluation of the algorithm for errors of categories 1 and 2 (semantic and logical errors; cf. Section 3). We performed the evaluation on a sample of formulas from the corpus study, and on a larger set of formulas into which errors of particular categories were introduced in a principled fashion.

There are a number of examples in which the applicable modifications are quite restricted. This is typically the case for errors of category 2 and 3 when the formula analysis algorithm can determine the nature of the error. For formula (1) in Table 1, we get two interpretations depending on whether PC is separated by inserting parentheses (2 alternatives), or flagged as a type error. In the latter case, replacing PC by any type compatible identifier yields error category 1. The same holds for the parenthesis insertion with narrower scope, $P(C)$, but the other alternative, $P(C \cup (A \cap B))$ yields no error and wins. Formula (2) is even simpler since only replacing the first occurrence of P flagged as a type clash is subject to being changed. Only replacements by A and B yield no error, B winning over A since it gets a better context agreement count. For attempting to correct formula (5), many operators are applicable. Changing one of the variables gives lower agreement scores than changing one of the operators into its dual counterpart – among all these choices, only replacing $=$ by $\supseteq$ yields a correct assertion.

Evaluation data In order to assess the performance of our method, we have conducted a test by applying the procedure to a sample of erroneous formulas. We have built this sample from two sources: (i) a set of erroneous formulas from our corpus (Sample 1), (ii) a set of formulas obtained by systematically introducing errors to valid formulas, according to our error categories and the associated replacement rules (Sample 2).

The choice of these sources is motivated by two complementary factors. The first part of the test sample is intended to provide an insight into the effectiveness for errors that occurred in practice, but this sample is rather small. Hence, the second part is build to support the evaluation in quantitative terms, to illustrate the average behavior of the algorithm over a larger set of related errors.

Sample 2 of erroneous formulas was obtained in the following way: Firstly, we extracted from the corpus 71 valid formulas that occurred in proof contributions evaluated by the tutor as correct. Then, for each of these formulas we generated a set of "mutilated formulas" by systematically changing the operators and identifiers according to the replacement rules (cf. Figure 2). For practical reasons, we introduced at most two errors into one formula to make the correction task manageable. For example, for the valid formula $A \cap B \subseteq P(A \cap B)$, the "mutilated formulas" produced include:

- dual operator errors: $A \cup B \subseteq P(A \cap B)$, $A \cap B \subseteq P(A \cup B)$;
- confused operators errors: $A \cap B \in P(A \cap B)$, $A \cap B \subseteq K(A \cap B)$, $A \cap B \subseteq P(A \cap P)$ (two errors);
- confused identifiers: $A \cap P \subseteq B(A \cap B)$, $A \cup P \subseteq P(A \cap B)$ (two errors), $X \cap B \subseteq P(A \cap B)$ (where X stands for an arbitrary identifier not in context to simulate a typographical error).

From the resulting set of "mutilated formulas", we built Sample 2 for evaluation by randomly selecting 100 in which the number of operators is between 3 and 10, so that formulas in this set are of "interesting" complexity – neither completely trivial nor an extremely involved instance of a formula.

Results. In all tests, we limit the calls to the theorem prover to ten at most since this is the most expensive part of the algorithm; we prefer this qualitative criterion over a time limit since this abstracts from particularities of the system used for searching a proof. The results demonstrate that obtaining or at least guessing the formula intended can be quite delicate. With very few exceptions, all erroneous formulas turned out to be ambiguous with respect to possible corrections – only a few unambiguously corrected formulas were found in Sample 1, all of them being very simple formulas in which only one change of an incorrect operator was applicable.

For such a list of hypotheses, the success of the procedure crucially depends on

1. how likely it is that the intended modification is at least found, but not necessarily identified as the most likely interpretation, and
2. how much effort it takes to generate these hypotheses and to filter out the intended one among the candidates considered.

The first issue can be answered in dependency of the allowed maximum number of calls to the Proof Manager – if the rank of the intended interpretation within the list of candidates generated is not worse than this maximum number, a successful identification is achieved. In Sample 2, this was the case in 64% of all formulas examined. The

Generate-Hypotheses (Formula,Context)

```
1. Collect operators suitable to cure the category of error observed
case <Category of original Formula > of
  3 : Hypotheses ←<List with all alternative analyzes collected in Formula >
      goto Evaluate-and-check-validity
  2 : Operators ← Operators_Category2
  1 : Operators ← Operators_Category1
end case
Hypotheses ←<Analysis result of the original Formula >

2. Iteratively apply operators to the original formula
Iterate:
forall < Hypotheses not yet modified>, Operators do
  New-formulas ←<Apply Operator to formula in Hypothesis >
  forall < New-formulas > do
    if not Trivial(< New-formula >) then
      Parse(< New − formula >)
      Hypotheses ← Hypotheses ∪ <Results of parsing New-formula >
    end if
  end forall
end forall

3. Determine whether a continuation is required/affordable
Evaluate-and-check-validity:
<Evaluate the new formulas in Hypotheses according to Context >
<Sort Hypotheses according to their evaluation score>
forall < Hypotheses not yet modified> do
  while not < Limit > do
    if <Category of Hypothesis > = 1 then
      if < Proof-Manager analysis the formula in Hypothesis as correct> then
        <Category of Hypothesis >← 0
      end if
    end if
  end while
end forall
<Sort Hypotheses according to their evaluation score>
if not <Category of the best Hypothesis is superior to original category>
   and not < Limit > and <New modified formulas built>
   and not <Category of the original Formula > = 3 then
  goto Iterate
end if
return Hypotheses
```

Fig. 2. Pseudo-code of the algorithm for building modified formulas and ranking them

Table 3. Results of hypothesis generation for Sample 2

	Minimum	*Maximum*	*Mode*
Hypotheses generated	5	38	18
Position of target formula in hypothesis list	1	18	14

second issue is illustrated in Table 3, which summarizes the effort required to generate the corrections in terms of the number of generated hypotheses and the position of the intended formula in the list of hypotheses. Mode is the modal number, i.e. the value that occurs most frequently. It is important to note that the first position on the list does not imply that a unique solution is found since multiple candidates may obtain the same final rank.

The results show that automating formula correction is a non-trivial task. While for an objective sample of complex formulas with errors (three to ten operators, up to two errors per formula), the algorithm was able to place the intended formula in the top ten hypotheses in 64% of the cases, there is no guarantee that further evaluation of the top candidates by the Proof Manager yields a unique candidate.

7 Conclusion and Further Research

In this paper, we presented a fault-tolerant formula interpreter that aims at finding plausibly intended, formally correct specifications from student statements containing formal inaccuracies. We have done this by testing the consequences of local changes based on a categorization of errors. This task is carried out by a fault-tolerant structure building, and testing the task contribution of contextually-motivated alternations of original, partly flawed specifications. While we have tried to keep the overall approach as domain-independent as possible, concrete elaborations are tailored to a fragment of elementary set theory for which, informed by analysis of a corpus obtained in Wizard-of-Oz experiments, we have already developed other tutorial components.

The error analysis is intended to enhance the input interpretation component of our tutorial system that teaches mathematical proving skills. This integration is intended to make it possible to handle errors by more specific tutorial methods, prominently concentrating on higher-level goals even in the presence of formal inaccuracies. To obtain a qualitative view of the algorithm's performance, we plan further analysis of the results along two perspectives: (i) firstly, we need to inspect the top ten generated hypotheses for each formula in terms of their correctness status evaluated by the Proof Manager, (ii) secondly, we investigate tutorial dialog strategies with respect to erroneous formulas based on the corpus data. Finally, we plan a comparative quantitative evaluation of the algorithm presented here with our previous Best-First approach presented in [7].

References

1. C. Benzmüller, A. Fiedler, M. Gabsdil, H. Horacek, I. Kruijff-Korbayová, M. Pinkal, J. Siekmann, D. Tsovaltzi, B. Vo, and M. Wolska. Language Phenomena in Tutorial Dialogs on Mathematical Proofs. *Proc. of the 7th Workshop on the semantics and pragmatics of dialogue (DiaBruck)*, pp. 165–166, Saarbrücken, Germany, 2003.

2. C. Benzmüller, A. Fiedler, M. Gabsdil, H. Horacek, I. Kruijff-Korbayová, M. Pinkal, J. Siekmann, D. Tsovaltzi, B. Vo, and M. Wolska. Tutorial Dialogs on Mathematical Proofs. *IJCAI Workshop on Knowledge Representation and Automated Reasoning for E-Learning Systems*, pp. 12–22, 2003.
3. C. Benzmüller and Q.B. Vo. Mathematical Domain Reasoning Tasks in Tutorial Natural Language Dialog on Proofs. *Proc. of the 20th National Conference on Artificial Intelligence (AAAI-05)*, pp. 516-522, Pittsburgh, PA, 2005.
4. C. Benzmüller, A. Fiedler, M. Gabsdil, H. Horacek, I. Kruijff-Korbayová, M. Pinkal, J. Siekmann, D. Tsovaltzi, B. Vo, and M. Wolska. A Wizard-of-Oz Experiment for Tutorial Dialogues in Mathematics. *Suppl. Proc. of the 11th International Conference on Artificial Intelligence in Education*, pp. 471–481, Sydney, Australia, 2003.
5. N. Heffernan, E. Croteau and K. Koedinger. Why Are Algebra Word Problems Difficult? *International Conference on Intelligent Tutorial Systems (ITS-04)*, pp. 240–250, Springer, LNCS 3220, 2004.
6. H. Horacek and M. Wolska. Interpreting Semi-Formal Utterances in Dialogs about Mathematical Proofs. *Natural Language Processing and Information Systems (NLDB-04)*, pp. 26–38, Springer, LNCS 3136, 2004.
7. H. Horacek and M. Wolska. Fault-Tolerant Context-Based Interpretation of Mathematical Formulas. *Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, poster, pp. 1688–1691, Edinburgh, Scotland, 2005.
8. N. Makatchev, P. Jordan and K. VanLehn. Modeling Student's Reasoning about Qualitative Physics: Heuristics for Abductive Proof Search. *International Conference on Intelligent Tutorial Systems (ITS-04)*, Springer, LNCS, 3220, pp. 699–709, 2004.
9. O. Popescu, K.R. Koedinger. Towards understanding geometry explanations. *Building Dialogue Systems for Tutorial Applications, Papers from the 2000 AAAI Fall Symposium.* Menlo Park, California: AAAI Press, pp. 80–86, 2000.
10. J. Siekmann, C. Benzmüller, V. Brezhnev, L. Cheikhrouhou, A. Fiedler, A. Franke, H. Horacek, M. Kohlhase, A. Meier, E. Melis, M. Moschner, I. Normann, M. Pollet, V. Sorge, C. Ullrich, C-P Wirth, J. Zimmer. Proof Development with ΩMEGA. *Proc. of the 18th Conference on Automated Deduction*, pp. 144–149, Copenhagen, Denmark, 2002.
11. D. Tsovaltzi, H. Horacek and A. Fiedler. Building Hint Specifications in an NL Tutorial System for Mathematics. *Proc. of the 17th Florida Artificial Intelligence Research Society Conference (FLAIRS-04)*, pp. 929–934, Menlo Park, CA, AAAI Press, 2004.
12. B. Q. Vo, C. Benzmüller, and S. Autexier. Assertion Application in Theorem Proving and Proof Planning. *Proc. of International Joint Conference on Artificial Intelligence (IJCAI-03)*, pp. 1343-1344, Acapulco, Mexico, 2003.
13. M. Wolska and I. Kruijff-Korbayová. Analysis of Mixed Natural and Symbolic Language Input in Mathematical Dialogs *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pp. 25–32, Barcelona, Spain. 2004.
14. M. Wolska, B. Q. Vo, D. Tsovaltzi, I. Kruijff-Korbayová, E. Karagjosova, H. Horacek, M. Gabsdil, A. Fiedler, and C. Benzmüller. An annotated corpus of tutorial dialogs on mathematical theorem proving. *Proc. of 4th International Conference on Language Resources and Evaluation.*, pp. 1007–1010, Lisbon. 2004.
15. C. Zinn, J.D. Moore, M.G. Core, S. Varges, K. Porayska-Pomsta. The BE&E Tutorial Learning Environment (BEETLE). *Proc. of the 7th Workshop on the Semantics and Pragmatics of Dialogue (DiaBruck)*, pp. 209–210, Saarbrücken, Germany, 2003.

Supporting Tutorial Feedback to Student Help Requests and Errors in Symbolic Differentiation*

Claus Zinn[1]

DFKI — German Research Centre for Artificial Intelligence
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
Tel.: +49 681 3025064
zinn@dfki.de

Abstract. The provision of intelligent, user-adaptive, and effective feedback requires human tutors to exploit their expert knowledge about the domain of instruction, and to diagnose students' actions through a potentially huge space of possible solutions and misconceptions. Designers and developers of intelligent tutoring systems strive to simulate good human tutors, and to replicate their reasoning and diagnosis capabilities as well as their pedagogical expertise. This is a huge undertaking because it requires an adequate acquisition, formalisation, and operationalisation of material that supports reasoning, diagnosis, and natural interaction with the learner. In this paper, we describe SLOPERT, a glass-box reasoner and diagnoser for symbolic differentiation. Its expert task model, which is enriched with buggy rules, has been informed by an analysis of human-human tutorial dialogues. SLOPERT can provide natural step-by-step solutions for any given problem as well as diagnosis support for typical student errors. SLOPERT's capabilities thus support the generation of natural problem-solving hints and scaffolding help.

1 Introduction

The European project "Language-enhanced, user adaptive, interactive e-learning for Mathematics (LeActiveMath)" aims at developing an innovative third generation e-learning system for high-school and university-level self-learners. One of the most prominent features of LeActiveMath is its exercise subsystem that supports students solving problems in the domain of calculus. The LeActiveMath project explores two approaches to implement this subsystem; a more traditional approach where professional math tutors *pre-author* interactive exercises; and a cutting-edge research approach that aims at *dynamically generating* exercises. The traditional approach is quite labour-intensive because exercise authors have to write-down a large number of possible dialogues for a given task. They have to anticipate students' correct and incorrect answers, and provide suitable dialogue continuations for them. The cutting-edge research approach does not rely on explicit representations of dialogue. It aims at providing an *exercise player* that

* Funded by the European Commission's 6th Framework Programme: *IST-507826*.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 349–359, 2006.

can generate future interactions dynamically by exploiting two main knowledge sources: a reasoning and diagnosis engine for the interpretation of student input, and a response system, powered with conversational expertise and tutorial feedback strategies, for the generation of system response.

To better understand effective one-to-one human tutoring in the domain of calculus, in particular, symbolic differentiation, we collected and analysed a corpus of one-to-one dialogues between professional math teachers and students. Our data analysis aims at identifying our tutors' conversational, domain, and pedagogical expertise. In this paper, we focus on modelling observed tutors' problem solving and diagnosis expertise. We describe how we encoded this expertise in SLOPERT to provide glass-box reasoning and diagnosis services for a dialogue manager to generate human-like feedback to student input.

2 Background

Our research focuses on the operationalisation of human tutoring in the domain of symbolic differentiation: what types of input (and errors) do students produce, and what kind of scaffolding and feedback do tutors generate to help learning?

Sinus Rule	$\frac{d}{dx} sin(x) = cos(x)$
Logarithmic Rule	$\frac{d}{dx} log(x) = \frac{1}{x}$
Power Rule	$\frac{d}{dx}\left[x^n\right] = n \cdot x^{n-1}$
Constant Multiple Rule	$\frac{d}{dx}\left[c \cdot f(x)\right] = c \cdot \frac{d}{dx}\left[f(x)\right]$
Sum Rule	$\frac{d}{dx}\left[f(x) \pm g(x)\right] = \frac{d}{dx}\left[f(x)\right] \pm \frac{d}{dx}\left[f(g)\right]$
Chain Rule For Power Functions	$\frac{d}{dx}\left(\left[f(x)\right]^n\right) = n \cdot \left[f(x)\right]^{n-1} \cdot \frac{d}{dx}\left[f(x)\right]$
General Chain Rule	$\frac{d}{dx}\left[f(g(x))\right] = \frac{d}{dg}\left[f(g(x))\right] \cdot \frac{d}{dx}\left[g(x)\right]$

Fig. 1. Differentiation rules; c is any real number, f(x) and g(x) are any functions

We collected a corpus of one-to-one tutoring in this domain.[1] Students were given teaching material on the rules of symbolic differentiation (see Fig. 1). They were made familiar with the overall subject in their normal course of studies. During a computer-mediated session a tutor gave differentiation problems to students and then provided feedback on their problem solving steps.

Figure 2 displays three consecutive fragments of one tutor-student interaction. In the first fragment, we find the student (**S**) solving the task given by the tutor (**T**) in **T-1** without tutorial intervention. In **S-2a**, the student first rewrites the original problem statement into a form that allows him to apply the chain rule for power functions in **S-2b**. In **S2c**, the student then simplifies the resulting term by identifying the common factor 2 in the term $(6x^5 + 2)$, extracting it, and multiplying it with $-\frac{5}{2}$ all in one step. Then, in **S-2d**, the student asks the tutor for feedback, which the tutor provides in **T-3a**.

[1] Collected by Kaska Porayska-Pomsta, Helen Pain and Manolis Mavrikis as part of the LeActiveMath project, using tools developed by Porayska-Pomsta and Mavrikis.

T-1 Try the following one: $\frac{5}{\sqrt{(x^6+2x)}}$
S-2a: $5(x^6 + 2x)^{-\frac{1}{2}}$
S-2b $= -\frac{5}{2}(x^6 + 2x)^{-\frac{3}{2}}(6x^5 + 2)$
S-2c $= -5(x^6 + 2x)^{-\frac{3}{2}}(3x^5 + 1)$
S-2d i think thats right ut im not too sure
T-3a That's very good. You really have got to grips with the chain rule for algebraic expressions.

T-3b Let's move on to other functions. Try to differentiate $log(x^2 + 6x - 1)$.
S-4 im not sure what you get when you differentiate log. It is e^x?
T-5 No. I'll tell you: $\frac{d}{dx}log(x) = \frac{1}{x}$
S-6 im still not sure how to do this one
T-7 First you need to identify the functions in the composition (f(g(x))). By the way, you really need to remember what the derivative of log is...
S-8 i still dont know what to do
T-9a Think about the example that you read in the beginning.
T-9b Try to identify z again and then y as a function of z.
S-10 $z = x^2 + 6x - 1$ im not sure about what y is
T-11 That's good so far. Now think where z appears in the expression
S-12 $y = logz$
T-13 Yes. That's right. Now can you put it all together?
S-14 $\frac{1}{x^2+6x-1}(2x + 6)$
T-15a Yes. That's it. We could write that as $\frac{2(x+3)}{x^2+6x-1}$

T-15b Now let's try one with trig functions. Try $\frac{1}{sin^3x}$ Remember that the derivative of sin is cos
S-16 $(sin^3x)^{-1}$ $-(sin^3x)(3cos^2x)$
T-17 Think of $sin^3(x)$ as $(sin(x))^3$
S-18 $3(sin(x))^2(cosx)$
T-19 That's much better. Now can you solve the original problem (which is a little different)?
S-20 $(sin(x))^{-3} = -3(sin(x))^{-2}(cosx)$
T-21 Almost. Remember that the derivative of x^{-n} is $-nx^{-n-1}$.
T-22 I think you know the answer: $-\frac{-3cos(x)}{sin^4x}$
S-23 yes that is what i was thinking [...]

Fig. 2. Consecutive human-human tutorial dialogue fragments (not spell-corrected)

The tutoring dialogue becomes more interesting with a problem that involves the *log* function, introduced by the tutor in **T-3b**. The student is unsure how to build a derivative for terms that contain logarithmic functions, and makes an incorrect guess in **S-4**. In **T-5**, the tutor acknowledges the incorrect answer with the provision of negative feedback, and the corrected answer. However, the student is still stuck, and in **T-7** a hint is given that presents the problem in a more abstract form. Apparently, the student does not realise that the general chain rule applies and utters **S-8**. After reminding the student in **T-9a** of pre-read material, the tutor elaborates **T-7** by sketching two sub-tasks in **T-9b**. The student gets the first subtask solved in **S-10**, and tutorial help in **T-11** provides more scaffolding to get the student solve the second subtask. In the remaining sub-dialogue, the student then combines the results of the sub-tasks to obtain a solution for the overall task in **S-14**, which the tutor simplifies in **T-15a**.

In the third fragment, the tutor presents a term with a trigonometric function in **T-15b**. The student, however, rewrites the problem statement into a form that is correct, but not adequate for subsequent problem solving. In **S-16**, the student also applies the general rule for power functions to the inappropriately rewritten term (with errors). In **T-17**, the tutor, recognising the student's confusion, hints toward the correct interpretation of the original term by providing the proper reading for its denominator sin^3x. In **S-18**, the student adopts the term $sin(x)^3$ as new problem and solves it. In **T-19a**, the tutor then asks the student to solve the original term. The student's answer in **S-20** contains a common error in applying the power rule for negative exponents. In **T-21**, the tutor then restates the power rule in terms of x^{-n}, and then, corrects the student's answer in **T-22**. Instead of giving instructions to the student, our tutor(s) used hints to get the student to construct the knowledge and solve the problem. Whenever the student is stuck, the tutor hinted at the next possible step towards solving a given problem. In **T-7**, for instance, the tutor hinted at the form of the problem instead of providing the proper term reading; and in **T-9b** and **T11**, the tutor hinted at the substitution variables z and y and their relation, instead of just giving away this substitution step. Remedial advice depends on adequate analysis of student error. The student's inappropriate rewriting in **S-16**, *i.e.*, is recognised and remediated with the proper reading of the problematic sub-term. The student's error in **S-20** is addressed by pointing the student to a version of the power rule that is specialised to polynomials with negative exponents.

3 Formalising Expert Reasoning

The mathematical notation of differentiation rules (c.f. Fig. 1) can be easily translated into the syntax of a symbolic programming language like PROLOG. A possible encoding of the Sinus Rule is `derive(Var, sin(Var), cos(Var)).`

The predicate `derive` has three parameters: the term to differentiate (second argument) with respect to a variable (first argument), and the result (third argument). To compute the derivative of $sin(x)$ with respect to x, we write

```
?-derive(x, sin(x), Result).
```
(note the query prompt "?-")

PROLOG answers this *query* with "Result = cos(x)"; this query *unifies* with the aforementioned PROLOG rule by instantiating the variable `Var` with `x`.

Now consider the encodings of the general power rule and the sum rule:

```
derive(Var, X^N, (N*(X^N1))*XP) :- N1 is N - 1, derive(Var, X, XP).
derive(Var, A+B, A1+B1)         :- derive(Var, A, A1), derive(Var, B, B1).
```

The first rule, or *clause*, has two parts, or *subgoals*. First, `N1` is assigned the value of subtracting `1` from `N`; and second, a *recursive* call computes the derivative `XP` of some term `X` wrt. `Var`. Instantiations that result from the evaluation the subgoals are transported to the third argument of `derive`, which contains the result of the computation, namely, `(N*(X^N1))*XP`. The sum rule has two parts, too. Here, all computation happens in the recursive calls.

Example. The query `?- derive(x, 5*(x^6+2*x)^(-0.5), Answer)` *succeeds* and results into the following instantiation of the variable `Answer`:

```
Answer = 5*(-0.5*(x^6-2*x)^(-1.5)*(6*x^5+2*1)).
```

This answer is unsatisfying for two reasons. First it is not "tidied-up"; the student's answer in **S-2a** and **S-2b** has a simpler and more readable expression; and second, PROLOG only returns the result of the computation, not any intermediate steps. It thus behaves similar to a black-box computer algebra system whereas a glass-box approach would be more useful for tutoring.

Extended Representation. There is a simple but elegant way to extend our representation to add to the result of the computation its computation process:

```
derive(X,sin(X),cos(X), [sinRule(sin(X) = cos(X))]).
derive(X,X^N,(N*(X^N1)),[powerRule(deriv(X^N)=(N*(X^N1)))]) :-
    freeof(X, N), N1 is N - 1.
derive(X,A+B,   A1+B1,  [sumRule(deriv(A+B)=deriv(A)+deriv(B)),Rules]) :-
    derive(X, A, A1, R1), derive(X, B, B1, R2), append(R1, R2, Rules).
```

In this representation, `derive` has four arguments, where the fourth is a description of the rule (or the relevant parts thereof). The effects of this change of representation can be seen by executing the following PROLOG query:

```
?-derive(x, 5*(x^6+2*x)^(-0.5), Answer, Explain).
```

While `Answer` has still the same value, the new parameter `Explain` now contains the *solution graph* that shows each of the rules that were used to compute `Answer`:

$$
\begin{tikzcd}[column sep=small]
\frac{d}{dx}5(x^6+2x)^{-\frac{1}{2}} \arrow[d, "\textit{const. mult. rule}"'] & & & & \\
5\ * & \frac{d}{dx}(x^6+2x)^{-\frac{1}{2}} \arrow[d, "\textit{general power rule}"'] & & & \\
& -\frac{1}{2}(x^6+2x)^{-\frac{3}{2}}\ * & \frac{d}{dx}(x^6+2x) \arrow[d, "\textit{sum rule}"'] & & \\
& & \frac{d}{dx}x^6 \arrow[d, "\textit{power rule}"'] & + & \frac{d}{dx}2x \arrow[d, "\textit{const. mult. rule}"] \\
& & 6x^5 & & 2 * \frac{d}{dx}x \arrow[d, "\textit{linear rule}"'] \\
& & & & 1
\end{tikzcd}
$$

The richer representation enables the problem solver to communicate (parts of) its problem solving strategies. It supports, for example, the generation of feedback given in **T-5**, **T-15b**, and **T21**, where the tutor cites derivation rules. However, our representation is still not sufficient to simulate our tutor's feedback **T-7**, **T-9b**, or **T-11**. This is because our PROLOG-based rule representation is still not explicit enough to inform the generation of such feedback. The tutor's help "identify the form of the statement", for instance, is implicit in PROLOG's unification mechanism. We now extend our rule representation to give a more detailed and explicit account of the tasks involved in computing derivatives.

Extended Representation II. Fig. 3 describes our task model for building derivatives. Its design has been informed by an analysis of our human-human tutorial dialogues. The graph consists of framed nodes (describing tasks or goals)

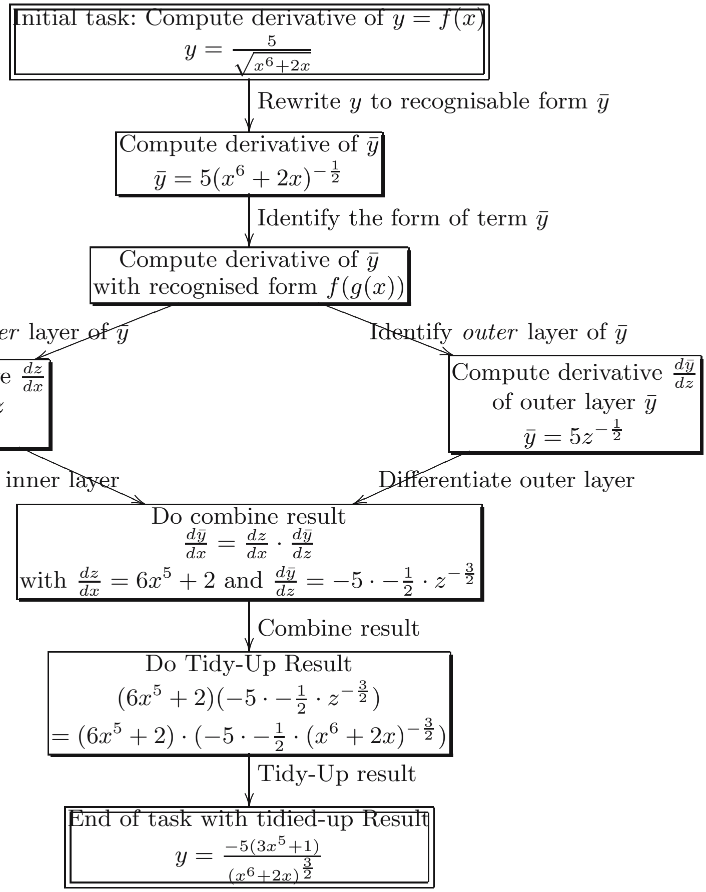


Fig. 3. Task model for building derivatives (with term rewriting)

and labelled arcs (describing actions). We have complemented the generic task model with a specific problem instance, namely, "compute the derivative of $y = \frac{5}{\sqrt{x^6+2x}}$", which is solved within the root's child nodes.

We have extended our representation to encode the task model for the general chain rule: the body of the `derive` clause now contains a more detailed implementation (substitutions, decompositions into separate problems for inner and outer layer, combination of results *etc.*), *and* the last argument of the clause head now contains the relevant explanation for it (where unification and notation is made explicit). As a result, PROLOG now returns the derivative of an input term *and* a solution graph that contains *all* expert problem solving steps necessary to support tutoring. Our enriched representation can now be exploited, for instance, to mimic the interesting scaffolding feedback of our human tutor in the second dialogue fragment. The respective solution graph now contains the nodes that identify the form of the statement in terms of inner and outer layer, *i.e.*, $\mathtt{identify\_form(log(x^2+6x-1), [inner\_layer(x^2+6x-1), outer\_layer(log)])}$ that specify the necessary substitutions (*i.e.*, $\mathtt{z} = \mathtt{x}^2 * \mathtt{6x} - \mathtt{1}$, $\mathtt{y} = \mathtt{log(z)}$), and that compute the sub-derivatives (*e.g.*, `log_rule(deriv(z,log(z))=z^-1))`. In **T-5**, our tutor cites the general rule for differentiating *log* functions, thus abstracting from the specific rule application in the solution graph. In **T-7**, the tutor then exploits the solution graph's first action. A tutorial dialogue manager could use this node to generate hints of increasing specificity. Its generation component could exploit and verbalise this information varying lexicon, grammar, and dialogue move type, *e.g.*, "First, you have to identify the term." (declarative), "Identify the form of the term!" (imperative), or "How about identifying the form of the term?" (rhetorical). Also, the specificity of the hint could be varied, *e.g.*, "Identify the functions in the compositions $f(g(x))$", "Identify the inner and outer layers of the given term!", "We first need to identify the form of the term. The inner layer is $x^2 + 6x - 1$. What's the outer layer?".

The generation of a more specific hint is useful when prior scaffolding fails. We observed also cases where tutors jumped to the next node of a solution graph, as for instance, our tutor in **T-9b**. There are also cases where our tutor interleaves positive feedback with scaffolding help. To support a machine tutor with the generation of such advanced feedback (*e.g.*, in **T-11** and **T-13**), further annotation of the solution graph is needed. Whenever a student answer is diagnosed as correct, the respective node is marked as `visited(diag)`; whenever a hint needs to be generated, the first non-visited node (encountered through graph traversal) is selected, exploited, and then marked as `visited(hint)`.

4 Formalising Buggy Reasoning

Our student in Fig. 2 makes many correct moves, but creates a few errors as well. We observe, for instance, the application of an incorrect basic rule for building the derivative of *log* in **S-4**; a mis-interpretation of the form of the problem statement in **S-16**; and the application of an erroneous power rule in **S-20**. These errors are quite common among our subject population. Other typical misconceptions from our corpus are summarised in Tab. 1. In our corpus, we observed tutors to give corrective or scaffolding help based on their diagnoses. To simulate such behaviour, we complement expert reasoning with *buggy* reasoning.

Fig. 4 depicts our enriched representation. To distinguish expert from buggy rules, we add a new first argument to `derive`. Also, a buggy rule may have

Table 1. Typical student errors in symbolic differentiation

Missing inner layer (chain rule)	$\frac{d}{dx}(x^3-3x)^5 \not\to 5(x^3-3x)^4$
Wrong exponent (chain/power rule)	$\frac{d}{dx}(5x^3-6)^{-3} \not\to -3(5x^3-6)^{-2}(15x^2)$
Missing exponent (chain/power rule)	$\frac{d}{dx}(x^3-3x)^5 \not\to 5(x^3-3x)(3x^2-3)$
Incorrect basic rule (sinus rule)	$\frac{d}{dx}sin(x) \not\to -cos(x)$
Missing bracketing (op. precedence)	$\frac{d}{dx}(x^3-3x)^5 \not\to 5(x^3-3x)^4 3x^2-3$
Erroneous transformation (rewriting)	$\frac{1}{(5x^3-6)^3} \neq (5x^3-6)^{-\frac{1}{3}}$

```
derive(expert,X,X^N,N*(X^N1),[powRule(deriv(X^N)=N*(X^N1))]) :-
    freeof(X, N), N1 is N - 1.
derive(buggy,X,X^N,N*(X^N1), [powRule(deriv(X^N)=N*(X^N1)),wrongPow(N1)]) :-
    N < 0, freeof(X, N), N1 is N + 1.
derive(buggy,X,X^N,N*X, [powRule(deriv(X^N)=N*X), missingExpTerm(N-1)]) :-
    freeof(X, N).
derive(buggy,X,X^N,X^N1,[powRule(deriv(X^N)=X^N1),missingFactor(N)]) :-
    freeof(X, N), N1 is N - 1.
```

Fig. 4. Buggy rules for symbolic differentiation in PROLOG.

additional conditions, and always has a description of its "buggyness". The first buggy rule is only applicable for negative exponents. The fact that students add 1 instead of subtracting it is encoded in the rule's fifth argument `wrongPow(N1)`.

We can exploit our enriched rule representation to diagnose student error. When we ask PROLOG to construct a (potentially buggy) solution graph with the provision of both input and output term, we target PROLOG's search:

```
?- derive(RuleType, x, x^(-4), (-4*(x^(-5))), Explain)
=> RuleType = expert
   Explain = [powerRule(deriv(x^(-4))=(-4*(x^(-5))))]
?- derive(RuleType, x, x^(-4), (-4*(x^(-3))), Explain)
=> RuleType = buggy
   Explain = [powerRule(deriv(x^(-4))=(-4(x^(-3))), wrongPow(-3)]
```

The symbolic differentiation of x^{-4} only requires the use of the power rule. When the student answers $-4 \cdot x^{-5}$, only the expert rule for power functions *fires*, and PROLOG returns its description. When the student answers $-4 \cdot x^{-3}$, the buggy rule that covers wrong powers for negative exponents is applicable, and its argument `Explain` is instantiated appropriately. A machine tutor could then use this information to produce remedial feedback of various specificity, for instance, "There is something wrong.", or "Check the power!", or **T-21**.

5 Evaluation

We encoded all the expert rules of differentiation, including the detailed task model presented in Fig. 3, and many buggy rules, including those of Tab. 1 The resulting PROLOG program, called SLOPERT, can automatically compute a wide range of differentiation problems. It can generate an expert graph that provides natural step-by-step solutions. It can use buggy rules to diagnose student answers, with potentially multiple errors. We have also implemented a graph traversal that can trace through the space of possible solutions. Annotation has been added to mark nodes of the graph as visited, either because they were used for the generation of hints or for the diagnosis of student answers.

Our tutors generated hints at any stage during the student's problem solving. Hints varied from vague ("there is a common factor") to specific ("The expression $5(x^3 - 3x)^4(3x^2 - 3)$ has a common factor of 3"). Sometimes, more specific hints followed vague hints, and sometimes, the first hint given was very specific.

SLOPERT supports the generation of such hints. It can exploit its expert model to generate solution graphs that contains all necessary intermediate steps. How much of the nodes' information should be given to the student is a different matter, and should be answered by the pedagogical module of the ITS, which may take into account other information than problem state and error diagnosis.

Our students seem to be quite familiar with the differentiation rules. In most cases, the chain rule was applied correctly, and at once. When errors were made, they were usually of the type shown in Tab. 1. Students who applied the chain rule step by step (*i.e.*, similar to our task model in Fig. 3), usually made *notational* errors. They were confused with the use of the x's, y's, and z's in the dx notation. Sometimes, dz/dx and $d\bar{y}/dz$ were computed correctly, but the result was incorrectly combined (wrong bracketing, oversights with "back-substitutions"). Most turns occurred when students performed algebraic transformations, either to rewrite terms to forms that allow for the application of derivation rules, or simplifying terms that resulted from the application of such rules.

Given a student's attempt to solve a differentiation problem, SLOPERT can search the solution graph for a node that matches the student's answer. A diagnosis of student input is successful if the input term *matches* the contents of the graph's root node, or the content of any node in the root's subgraph. The notion of *matching* is complex, however. Reconsider the first dialogue fragment in Fig. 2. In **S-2a**, the student succeeds in rewriting the original statement. SLOPERT cannot confirm the correctness of the step by itself, since it only has rules for symbolic differentiation. Therefore, we have interfaced SLOPERT to the PRESS algebra system [3]. Fortunately, PRESS can transform the tutor's initial term to the one given by the student, so that the student's contribution can be judged as correct. The interpretation of **S-2b** is trickier as the student correctly applies the general power rule and does *some* tidying-up, simplifying $5 \cdot -\frac{1}{2} \cdot (x^6 + 2x)^{-\frac{3}{2}}(6x^5 + 2)$ to $-\frac{5}{2}(x^6 + 2x)^{-\frac{3}{2}}(6x^5 + 2)$. SLOPERT's solution graph only contains the first expression, which is semantically equal to the student term, but syntactically different. Asking PRESS to simplify it is of only limited help, since it results into a term that is closer but syntactically different to **S-2c**. It will be thus necessary to extend the PRESS simplification rules in a manner similar to our approach. This will allow PRESS to generate step-by-step solutions for algebraic transformations, and to diagnose errors in this process.

6 Related Work and Conclusion

Instead of giving instructions to the student, (our) tutors use hints to get the student to construct the knowledge. Research shows that hints facilitate active construction of knowledge [4, 7]. *Problem solving hints* can help students retrieve information and then to use it to make an inference and solve the problem [6].

We found off-the-shelves computer algebra systems of only limited use to inform hinting and the provision of other effective remedial feedback in intelligent tutoring systems. While these expert reasoning engines can solve a vast number of math problems, their algorithms are optimised for generality and efficiency, and thus, rarely mirror or mechanise human/student problem solving behaviour.

SLOPERT follows the footsteps of the BUGGY project [2], which started with studying students doing simple algebra, e.g., subtraction. *Procedural networks* were devised to represent each mislearned procedural skill independently. Resulting diagnostic models reflected students' understanding of the skills and sub-skills involved in a task. To reproduce erroneous student behavior, buggy variants of individual sub-skills were added to the procedural network. BUGGY then correctly diagnosed student errors like "the student subtracted the smaller digit in each column from the larger digit regardless of which is on top".

The Cognitive Tutors of Carnegie Learning are based on cognitive models of student problem solving following the ACT-R theory of learning [1]. Production rules capture students multiple strategies and their common error. They describe how a problem solving goal is rewritten to another goal, in a correct or buggy fashion. A model tracer exploits a system of production rules to follow students through their individual approaches to solve a problem. It uses and updates estimates of how well the student knows each (correct and buggy) production rule. This allows for the generation of feedback that is sensitive to the problem solving context. The authoring of production rules usually involves a huge investment from math tutors, as we can also confirm. Moreover, huge rule sets have a detrimental effect on the performance of the model tracer algorithm. Tutoring systems based on model tracing thus tend to provide *immediate* feedback to keep the student on track of recognised problem solving paths.

BUGFIX proves that remedial feedback does not have to be immediate feedback. Henneke's work provides an efficient implementation of a diagnosis algorithm that can consider several billion different student calculations for a given task, without keeping the student waiting [5]. BUGFIX also separates domain from pedagogical expertise. A *separate* tutoring module could exploit its diagnostic capabilities to generate well-informed remedial feedback.

Conclusion. The feedback of our tutors often depended on the problem solving state and the diagnosis of student actions with respect to correctness, and possibly, goal-direction. To mimic this behaviour, an *intelligent* tutoring system needs access to deep reasoning and diagnosis facilities. AI researchers can contribute to the design and implementation of algorithms that search the space of possible solutions efficiently. Pedagogical experts can inform the design of expert and buggy rules, and provide feedback strategies. The separation of domain expertise from pedagogical expertise can help studying, formalising, and operationalising effective human tutorial feedback. This research reported in this paper contributes to this approach of designing intelligent tutoring systems.

References

1. J. R. Anderson, C. F. Boyle, A. T. Corbett, and M. W. Lewis. Cognitive modeling and intelligent tutoring. *Artificial Intelligence*, 42:7–49, 1990.
2. J. S. Brown and R. R. Burton. Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, 2:155–192, 1978.
3. A. Bundy and B. Welham. Using meta-level inference for selective application of multiple rewrite rule sets in algebraic manipulation. *Artificial Intelligence*, 16(2):189–212, 1981.

4. A. C. Graesser, N. K. Person, and J. P. Magliano. Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*, 9:495–522, 1995.
5. M. Henneke. *Online Diagnose in intelligenten mathematischen Lehr-Lern-Systemen*. PhD thesis, Universität Hildesheim, 1999.
6. G. Hume, J. Michael, A. Rovick, and M. Evens. The use of hints as a tutorial tactic. 15th Cognitive Science Conf., pages 563–568. Lawrence Erlbaum Associates, 1993.
7. D. C. Merrill, B. J. Reiser, M. Ranney, and J. G. Trafton. Effective tutoring techniques: Comparison of human tutors and intelligent tutoring systems. *Journal of the Learning Sciences*, 2(3):277–305, 1992.

The Help Tutor: Does Metacognitive Feedback Improve Students' Help-Seeking Actions, Skills and Learning?

Ido Roll, Vincent Aleven, Bruce M. McLaren, Eunjeong Ryu,
Ryan S.J.d. Baker, and Kenneth R. Koedinger

Human Computer Interaction Institute, Carnegie Mellon University
5000 Forbes Ave., Pittsburgh PA 15213
{iroll, eryu}@andrew.cmu.edu, {aleven, mclaren, rsbaker, koedinger}@cs.cmu.edu

Abstract. Students often use available help facilities in an unproductive fashion. To improve students' help-seeking behavior we built the Help Tutor – a domain-independent agent that can be added as an adjunct to Cognitive Tutors. Rather than making help-seeking decisions for the students, the Help Tutor teaches better help-seeking skills by tracing students actions on a (meta)cognitive help-seeking model and giving students appropriate feedback. In a classroom evaluation the Help Tutor captured help-seeking errors that were associated with poorer learning and with poorer declarative and procedural knowledge of help seeking. Also, students performed less help-seeking errors while working with the Help Tutor. However, we did not find evidence that they learned the intended help-seeking skills, or learned the domain knowledge better. A new version of the tutor that includes a self-assessment component and explicit help-seeking instruction, complementary to the metacognitive feedback, is now being evaluated.

1 Introduction

Not only that teaching metacognition holds the promise of improving current learning of the domain of interest, but also, or even mainly, it can promote future learning and successful regulation of independent learning. However, considerable evidence shows that metacognitive skills are in need of better support. For example, while working with Intelligent Tutoring Systems (ITS), students try to game the system [6] or do not self-explain enough [1].

Recently, several researchers have explicitly incorporated metacognitive support into ITS. Conati et al. [8] and Aleven et al. [1] scaffold self-explanation; Baker et al. reduce harmful gaming [6]; Bull et al. [7] and Zapata-Rivera et al. [17] encourage reflection using open learner models; and Gama offers a metacognitive suite in the form of scaffolding self-evaluation, planning and reflection [10]. While many of these components indeed improve learning, they do not focus directly on improving the subset of metacognitive skills that relates to help-seeking. Also, as far as we know, so far there was no evaluation of transfer of metacognitive skills from ITS to other learning environments.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 360 – 369, 2006.

1.1 Help Seeking Behavior

In this paper we focus on supporting metacognitive skills that regulate help-seeking behavior. The need for effective help-seeking strategies is apparent in many aspects of learning, formal or otherwise. The ability to seek help efficiently has been shown to contribute to learning [5; 12], and was correlated with better learning while working with ITS [16].

However, students' help-seeking behavior is often faulty (for an overview, see [4]). Students have a tendency both to overuse and under-use help: they avoid using help when they need it, but when they do seek help, they typically ask for more than is actually required [2].

In the current work we try to improve general help-seeking skills by building the Help Tutor, a domain-independent plug-in agent that can supplement a tutoring system such as a Cognitive Tutor. In this paper we describe a classroom evaluation study we conducted with the Help Tutor, having the dual goals of (a) assessing its effectiveness with respect to improving students' help-seeking behavior, skills and their learning of domain-specific skills and knowledge, and (2) learning about the requirements for and characteristics of a successful metacognitive tutoring system.

1.2 The Cognitive Tutor

The Geometry Cognitive Tutor (see Figure 1) is part of the Cognitive Tutors curriculum, a family of ITS commonly used in high schools around the United States [11]. The main window of the tutor is the Scenario window, which presents the problem and includes the main interaction with the student (on the left). The tutor scaffolds the solution process for the student by outlining the steps that are required to reach the final answer.

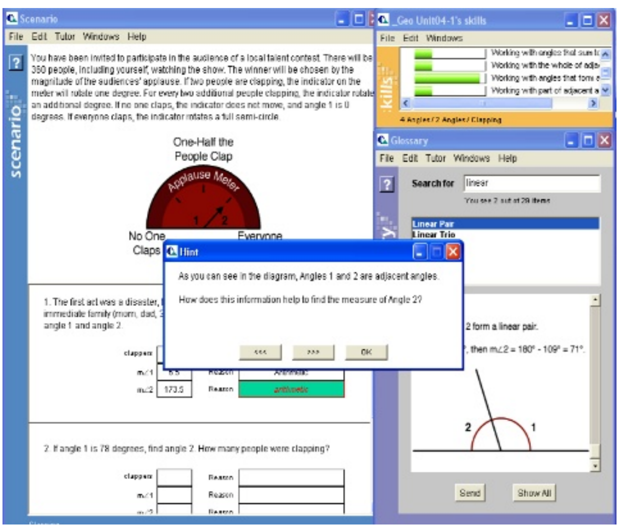


Fig. 1. The Geometry Cognitive Tutor

In the upper-right-hand corner students can see an estimation of their knowledge level. The Cognitive Tutor estimates the student's knowledge-level on the target set of cognitive skills using a Bayesian knowledge-tracing algorithm [9].

The Geometry Cognitive Tutor has two main help-seeking mechanisms: on-demand contextual hints, and a glossary. The on-demand contextual hints provide multiple levels of information that students can browse. The first level is typically very general, intended to remind the student of their current goal. Intermediate hints are increasingly more specific. The last (or "bottom-out") hint is very close to the answer. An example of an intermediate-level hint is shown in figure 1. The second help resource is the Glossary, which resembles a searchable information source (such as the internet, or a dictionary). The Glossary contains all relevant definitions and theorems. However, searching it and applying its content require some available cognitive capacity and ability to transfer the general information to the problem, much like using many real-life information sources.

2 The Help Tutor

When designing the Help Tutor, we chose to leave all help-seeking related decisions to the students. While insisting on help or preventing it can be beneficial for learning specific domain knowledge in the short-term, it will likely not improve students' ability to deal with situations of uncertainty or knowledge gaps in the future. Thus, the goal of the Help Tutor is to teach students to be better help-seekers by offering advice, and not by limiting students to only a certain behavior [3].

Similar to the Cognitive Tutor itself, the Help Tutor supports learning-by-doing, i.e., it teaches help-seeking skills by letting students practice them and then giving appropriate feedback. The Help Tutor is a plug-in agent that is added to an existing Cognitive Tutor, and the student interacts with it during the normal course of interaction with the Cognitive Tutor. In the Help Tutor, students' actions are traced using a metacognitive help-seeking model, in addition to the existing domain-level cognitive model [2]. When a student performs a help-seeking error she receives immediate and tailored feedback, in the form of a help-seeking error message.

The Help Tutor is comprised of two conceptual components - detection and intervention [3].

Detection: The help-seeking model, which is used to trace the student's behavior, determines what the preferred action is at each moment, so the assistance-level the student gets should fit her zone of proximal development [15]. The model often allows for more than one correct action. For example, a student working on a step for which she is estimated to have a high skill level is expected either to attempt the step with no help or to search the glossary, while the same student, on a step for which she has a low estimated skill level, is expected to ask for an elaborated hint.

The model is implemented using eighty production rules. It marks deviations from the set of recommended actions as help-seeking errors, which can be categorized in five families:

- Help abuse – the use of hints or Glossary in an inappropriate manner (for example, by 'drilling down' hints quickly to the bottom-out hint).
- Try-step avoidance – the use of hints when the student seems sufficiently skilled to solve the step on her own.
- Try-step abuse – trying to solve in an inappropriate manner (e.g., by guessing repeatedly)

- Help-avoidance – trying to solve a step when the student should have asked for help (e.g., after making multiple errors on that step).
- General errors – other errors (e.g., the student exhausted all hints and performed high number of errors, and is still trying instead of consulting with the teacher).

An earlier version of the model was shown to be somewhat domain independent, when compared against two different Cognitive Tutors [13]. However, while it correlated with learning, it produced a much-too-high error rate – of all students' actions, 64% in one dataset and 73% in the other were classified as errors. In order to be effective, the current model should reduce the help-seeking error rate drastically, while maintaining its correlation with learning.

Intervention: The other component of the Help Tutor is the help-seeking error messages, which include only domain-independent metacognitive content for several reasons: to encourage students to focus more on the metacognitive feedback (and not be distracted by the cognitive one), to help students generalize the help-seeking skills, and to make the Help Tutor reusable with different Cognitive Tutors.

The help-seeking messages follow few principles:

- Emphasizing the usefulness of effective help seeking behavior (e.g., "it could be that another hint will do the trick for you.")
- Reinforcing correct use of the tools (e.g., "no need to hurry so much. Take your time and read the hint carefully.")
- Being positive and informal (e.g., "could be a good time to give it a try.")

In order to avoid repetitiveness in the messages displayed to students, each error can elicit several different instances of the same message. The Help Tutor messages use the existing hint-window mechanism, and are distinguished from regular hints in their font (color and type) and timing (proactive vs. reactive).

3 Evaluation

3.1 Experimental Design

We evaluated the Help Tutor with 60 students from four classrooms in two high schools in the Pittsburgh area - one urban and one suburban. The students worked with the tutor for six periods. Within each school, the participating classes were taught by the same teacher, and all students were accustomed to the Cognitive Tutor, as they use it regularly in their Geometry classes.

Half of the students in each class worked with a version of the Geometry Cognitive Tutor with the Help Tutor (Help condition), and the other half worked with the Geometry Cognitive Tutor alone (Control condition). Students were counterbalanced between conditions based on their previous achievements in the Cognitive Tutor class. No instruction on help seeking was given in advance.

Students worked with the tutors twice a week. During the other three weekdays students had classroom lectures with their Geometry teachers, which focused on different topics. Due to scheduling considerations at one of the schools, only students in one school completed pre- and post-tests before and after the study (30 students).

For the other school we have only the log-files of the students while interacting with the tutors.

3.2 Assessment Design

In order to evaluate the Help Tutor appropriately, we defined three objectives for the Help Tutor in this study, each of which depends on the previous one.

- **Capture poor help-seeking actions:** The Help Tutor should identify faulty behaviors, while not interrupting the student's workflow too frequently. Also, the model's conclusions should preferably be generalizable across learning environments.
- **Help students improve their help-seeking behavior:** assuming that the Help Tutor successfully captures help-seeking errors, the intervention should eliminate, or reduce, these errors.
- **Improve learning:** The Help Tutor should improve learning. Fewer help-seeking errors should translate to better performance on the posttest. Overall, students should learn the domain knowledge better, as well as become better help-seekers.

In order to assess how well the Help Tutor met these objectives, we included multiple assessments:

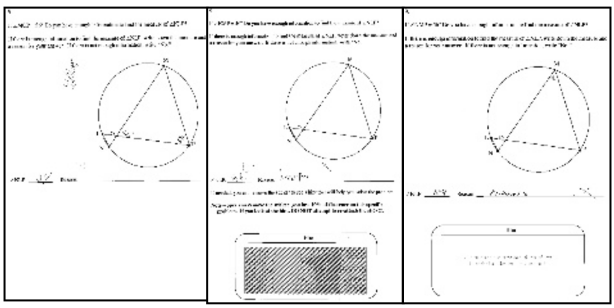


Fig. 2. Embedded hints in pre- and post-tests. From left to right: No Hint, Covered Hint, and Open Hint conditions.

- Students' help-seeking behavior in the tutor was assessed using log files analysis.
- Procedural help-seeking knowledge was assessed also outside the tutor's environment, using a paper test with embedded help-seeking resources. Each problem included three types of hints, counterbalanced between test forms: a problem statement only (No Hint), a problem statement with an open and free hint (Open Hint), and a problem statement with a hint covered by a sticker (Covered Hint). Students were told that removing the sticker costs 10% of their score on that specific item (see figure 2).
- Students' declarative help-seeking knowledge was assessed using questionnaire items. Students were asked five multiple-choice questions, which described situations to which the students were supposed to respond, e.g.:

1. You tried to answer a question that you know, but for some reason the tutor says that your answer is wrong. What should you do?

[ ] First I would review my calculations. Perhaps I can find the mistake myself?
[ ] The Tutor must have made a mistake. I will retype the same answer again.
[ ] I would ask for a hint, to understand my mistake.

3.3 Results

Overall students performed 59,034 actions with the tutor during the study, an average of approximately 1,000 actions, or 350 steps, per student. A typical tutor problem consists of 6-10 steps.

Although there was a significant improvement in scores from pre- to post- test, it was rather small: on average, students improved from 36% on the pretest to 41% on the posttest ($F(2,28)=6.4$, $p=0.015$). Also the log-files from the interaction with the tutor reveal rather little learning. On average, students mastered only 6 of all 49 skills that were practiced during that period.

As seen in figure 3, students scored significantly better on test items with embedded hints, compared to the No-Hint condition ($t(29)=2.1$, $p=0.04$). Students revealed hints on 24% of the Covered Hints problems.

Table 1. Help-seeking error rate and correlations with posttest scores, controlling for pretest

	Help-seeking errors overall	General Errors	Help Avoid-ance	Help Abuse	Try-Step Avoid-ance	Try-Step Abuse
Error rate	**17%**	1%	5%	6%	6%	<0.5%
Correlation with learning	**-0.42****	-0.34*	-0.41**	-0.17	-0.27	-0.10

* - marginally significant ($p<0.1$); ** - statistically significant ($p<0.05$)

Objective 1: Capturing erroneous help-seeking actions. The Help Tutor identified 17% of the actions as help-seeking errors (table 1). Higher frequency of help-seeking errors was negatively correlated with posttest scores (controlling for pretest scores). This is a significant improvement from the old model, which as noted before, captured about 70% of the actions and yielded similar correlation with learning.

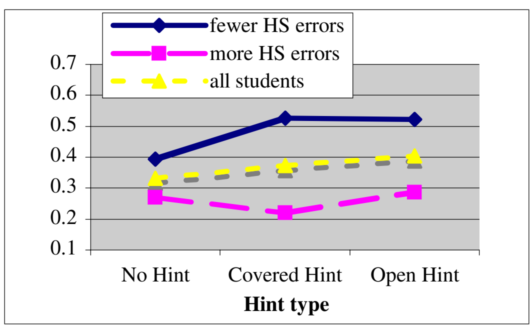


Fig. 3. Better use of hints during the posttest correlates with better help-seeking behavior in the tutor

Students' help-seeking performance as assessed by the Help Tutor also correlated with both declarative and procedural help-seeking knowledge outside the tutor environment. Scoring high on test items with embedded hints (controlling for score on items with no hints) was correlated to performing better help-seeking actions while working with the Help Tutor ($r=0.5$, $F(2,27)=10$, $p<0.01$). In other words, the same students who had better help-seeking skills while working with the tutor (according to the Help Tutor) also had better help-seeking procedural knowledge, as measured by comparing their scores on paper-test items with hints to those with no hints (figure 3).

Also, students who made fewer help-seeking errors in the tutor scored significantly higher on the help-seeking multiple choice declarative knowledge questionnaire (78% vs. 59%, t(28)=2.2, p=0.04), showing that they were aware of the correct use of help.

Objective 2: Helping students improve their help-seeking behavior. In order to be effective, the Help Tutor feedback should reduce the rate of help-seeking errors students make. This objective was only partially met: on the face of it, the difference in help-seeking error rate between the Help and Control groups is not significant: 16% vs. 19% (F(2,57)=2.5, p=0.12). However, when examining students' help-seeking errors more closely, we find that the Help Tutor had a different effect on different actions: When asking for hints, students working with the Help Tutor made significantly fewer help-seeking errors (see Table 2). However, errors can also be made when trying, for example, when students try too fast, or avoid needed help. There was no improvement in the rate of such errors while working with the help-tutor. Perhaps the low rate of errors related to try-step does not leave much room for improvement on these errors (Controlling for school, the interaction between condition and action is significant (F(3,54) =21.0, p<0.0001)).

When looking at the context of the actions, the Help Tutor was effective only after a hint (F(3,53)=7.0, p=0.02).

These analyses show that the Help Tutor influenced students' behavior mainly during or following hint requests, and not as much on other actions. This can be best viewed when looking at the depth of hints students are viewing: The overall number of steps on which students asked to see hints was indifferent to the Help Tutor (14% for the Help group vs. 17% for the control group, not significant). However, the ratio of bottom-out hints (where students drill-down to the bottom-out hint) to all hints dropped drastically following the use of the Help Tutor: from 72% in the Control group to 46% in the Help group (F(3,53)=35, p<0.0001).

Table 2. Help seeking error rate per action type and context, and rate of drilling down to bottom-out hint

		Control group	Help group
Action type	Try-step	9%	9%
	Hint	45%	33%**
	(first hint)	(27%)	(17%)*
	(following hints)	(52%)	(37%)**
Context	On first action	9%	8%
	After an error	18%	18%
	After a hint	40%	31%**
% drilling down to bottom-out hint		72%	46%**

* p < 0.01; ** p < 0.001

Objective 3: improve learning. Besides improving the help-seeking behavior, the Help Tutor should promote learning in both dimensions: learning of the domain knowledge, and learning of the help-seeking skills. While we observed overall learning from pre- to post-test, we were not able to identify any effect of the Help Tutor on learning (t(28)=0.1, p=0.95). Both groups improved from 36% on pretest to 41% on posttest.

While the error-rate on actions involving hints was lower for the Help group, we did not see any evidence for metacognitive learning with time – that is the help-seeking error rate was lower in the Help group throughout the study and did not have a significant learning effect. This finding suggests that rather than learning the help-seeking skills, students only followed suggestions. To evaluate this we looked at the frequency with which the Help Tutor's recommendations were followed. In this analysis we looked at the subset of actions that were performed after the Help Tutor displayed a message advising the student to act differently. We compared the actions of the Help group students to those of the Control group students in similar situations (i.e., situations in which the Help Tutor, if was used by this group of students, would have recommended them to act differently). The Help group students followed the Help Tutor recommendation when it advised them to ask for a hint ($t(44)=2.5$, $p=0.02$), but did not follow Try-Step recommendations any more than they would have done anyhow (as evaluated by the Control group).

There was also no improvement of help-seeking declarative knowledge. The differences between groups were not significant, and changes from pre- to post-test were not significant either: the Help group changed from 60% to 64% ($t(16)=0.7$, $p=0.5$); the Control group changed from 64% to 73%. ($t(12)=1$, $p=0.3$).

4 Discussion

The Help Tutor was successful in improving behavior - it captures hint usage errors, which are correlated with poorer learning, and reduces their rate significantly. Even more encouraging is the environment-independent nature of the tutor - the erroneous behavior the Help Tutor captures in the Cognitive Tutor environment is negatively correlated to successful hint-usage in the paper-and-pencil test and to declarative help-seeking knowledge. However, the Help Tutor did not yet achieve its broader goal, i.e., improving all help-seeking related actions, including faulty try-steps attempts, helping the students learn transferable help-seeking skills, and improving learning.

It appears that the Help Tutor achieved positive effects mainly because students followed its advice, and not because they assimilated the help-seeking principles. One possible explanation is the timing of the Help-Tutor messages. We did not expect students to be attentive after a successful completion of a step at the domain level, so the Help Tutor does not interfere in these instances. As a result, the Help Tutor interfered when students may have been consumed with problem solving, and thus were less likely to give the messages sufficient attention. The student might have used the Help-Tutor messages in the local scope in which they were given, to assist them in the domain level and did not internalize the rule or principle governing the specific situation. Hence, the student did not learn to evaluate her own needs and to regulate her learning. More reflective feedback at the end of the problem-solving process or before starting to solve might have been helpful.

5 Conclusions and Future Work

In this line of research we built a model of desired help-seeking behavior and used it to create the Help Tutor, which provides students with feedback on their help-seeking behavior in addition to any other feedback that the tutor provides. The results of the study show that the Help Tutor successfully captured help-seeking errors that were negatively correlated with learning, and furthermore, were correlated to differences in help usage in a paper and pencil test. Students who worked with the Help Tutor reduced their errors in using hints, as compared to students who used the regular tutor, but did not reduce their errors in faulty solution attempts, and did not learn better help-seeking techniques over time. We hypothesize that this might be due to inadequate preparation prior to working on the problems and a lack of a reflective process after the domain-problems were solved.

We have re-designed the Help Tutor based on the findings from this study. First, conceptual instruction on help-seeking is provided to students by the teacher using a short video in advance. The instruction focuses on successful help-seeking principles, and adopting positive dispositions towards help seeking. We have also incorporated self-assessment into the Help Tutor, which encourages students to reflect upon their needs for assistance [14]. In addition, we have attempted to improve the model by allowing it to catch more Try-Step errors. One last change is the scope of the study. To emphasize the domain-independent nature of help-seeking behavior, the current evaluation stretches across two different units of the Geometry Cognitive Tutor class.

Acknowledgements. We would like to thank Ido Jamar, Jo Bodnar, Dale Walters, Mike Konieczki, Kathy Dickensheets and Grant McKinney for their help carrying out this study. The research reported in this paper is supported by NSF Award No. IIS-0308200. The contents of the paper are solely the responsibility of the authors and do not necessarily represent the official views of the NSF.

References

1. Aleven, V., & Koedinger, K.R. (2002). An effective metacognitive strategy: learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science* 26(2), 147-79
2. Aleven, V., McLaren, B.M., Roll, I., & Koedinger, K.R. (to appear). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education*
3. Aleven, V., Roll, I., McLaren, B.M., Ryu, E.J., & Koedinger, K.R. (2005) An architecture to combine meta-cognitive and cognitive tutoring: Pilot testing the Help Tutor. *in proceedings of 12th International Conference on Artificial Intelligence in Education (AIED 2005),* Amsterdam, The Netherlands: IOS press.
4. Aleven, V., Stahl, E., Schworm, S., Fischer, F., & Wallace, R. (2003). Help seeking and help design in interactive learning environments. *Review of Educational Research* 73(2), 277-320
5. Arbreton, A. (1998). Student goal orientation and help-seeking strategy use, in S.A. Karabenick (Eds.), *Implications for learning and teaching* (pp. 95-116). Mahwah: Erlbaum.
6. Baker, R.S., Corbett, A.T., Koedinger, K.R., & Roll, I. (2005) Detecting When Students Game The System, Across Tutor Subjects and Classroom Cohorts. *in proceedings of User Modeling 2005,* Berlin: Springer-Verlag.

7. Bull, S., Dimitrova, V., & Brna, P. (2002). Enhancing Reflective Modelling through Communicative Interaction in Learning Environments, in P. Brna, M. Baker, K. Stenning & A. Tiberghien (Eds.), *The Role of Communication in Learning to Model* (pp. 183-211). Mahwah NJ: Lawrence Erlbaum Associates.
8. Conati, C., & VanLehn, K. (1999) Teaching meta-cognitive skills: implementation and evaluation of a tutoring system to guide self-explanation while learning from examples. *in proceedings of International Conference on Artificial Intelligence in Education,*
9. Corbett, A., & Anderson, J.R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4(4), 253-78
10. Gama, C. (2004) Metacognition in Interactive Learning Environments: The Reflection Assistant Model. *in proceedings of 7th Conference on Intelligent Tutoring Systems,* 668-77. Berlin: Springer-Verlag.
11. Koedinger, K.R., & Anderson, J.R. (1997). Intelligent Tutoring Goes to School in the Big City. *International Journal of Artificial Intelligence in Education* 8, 30-43
12. Renkl, A. (2002). Learning from worked-out examples: Instructional explanations supplement self-explanations. *Learning and Instruction* 12, 529-56
13. Roll, I., Baker, R.S., Aleven, V., McLaren, B.M., & Koedinger, K.R. (2005) Modeling Students' Metacognitive Errors in Two Intelligent Tutoring Systems. *in proceedings of User Modeling 2005,* 379-88. Berlin: Springer-Verlag.
14. Roll, I., Ryu, E., Sewall, J., Leber, B., McLaren, B.M., Aleven, V., & Koedinger, K.R. (submitted). Towards Teaching Metacognition: Supporting Spontaneous Self-Assessment. *in proceedings of the International Conference on Intelligent Tutoring Systems 2006*
15. Vygotsky (1978). *Mind in society* Cambridge, MA: Harvard University Press.
16. Wood, H.A., & Wood, D.J. (1999). Help seeking, learning, and contingent tutoring. *Computers and Education* 33, 153-69
17. Zapata-Rivera, D., & Greer, J.E. (2002). Exploring Various Guidance Mechanisms to Support Interaction with Inspectable Learner Models. *6th International Conference on Intelligent Tutoring Systems*, 442-52

The Role of Feedback in Preparation for Future Learning: A Case Study in Learning by Teaching Environments

Jason Tan and Gautam Biswas

Dept. of EECS & ISIS, Box 1824 Sta B,
Vanderbilt University
Nashville, TN 37235. USA
Tel.: (615)-343-6204
{jason.tan, gautam.biswas}@vanderbilt.edu
http://www.teachableagents.org

Abstract. Past research on the timing and content of feedback on student learning in computer-based learning environments has shown that directed or corrective feedback helps with immediate learning, whereas guided and metacognitive feedback help in gaining deep understanding of the domain and developing the ability to transfer this knowledge. Feedback becomes important in discovery learning environments, where novice students are often overwhelmed by the cognitive load associated with learning and organizing new knowledge while at the same time monitoring their own learning progress. We focus on feedback mechanisms in Betty's Brain, a teachable agent system in the domain of river ecosystems. Our goal is to help improve students' abilities to monitor their agent, Betty's knowledge, and, in the process their own learning and understanding. Our studies demonstrate the effectiveness of guided metacognitive feedback in preparing students for future learning.

1 Introduction

For the last five years we have been designing, implementing, and evaluating computer-based learning environments called Teachable Agents (TAs) that are based on the *learning by teaching* paradigm. In the TA paradigm, students learn science and math by teaching a computer agent through well-structured visual representations that help them organize their knowledge and thinking. The TA, using artificial intelligence techniques, reasons with the facts and relations it is taught to answer questions and solve problems. The TA can also illustrate its reasoning using graphics and animation, and this provides valuable feedback to the students. One of our TAs, called Betty's Brain, has been successfully used to teach about river ecosystems in 5th grade science classrooms [5]. This paper discusses the effectiveness of feedback mechanisms that we have built into the Betty's Brain system to help students with their assessment and learning tasks.

The cognitive science and education research literature supports the idea that teaching others is a powerful way to learn [4]. The literature on tutoring has shown that tutors benefit as much from tutoring as their tutees [7]. Beyond preparing to

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 370 – 381, 2006.

teach, the act of teaching taps into three aspects of learning interactions – *structuring, taking responsibility, and reflecting* [3]. Whereas preparation to teach involves significant amounts of learning and organizing [3], our studies have found that for novices, a great deal of learning can occur through assessment and reflection during teaching [5, 13]. The feedback students receive by observing their TA's performance helps them discover what to prepare, and how to structure what they are learning to ensure the agent can understand and apply what it has been taught. However, 5th grade students, who are both domain novices and novices in teaching practices, often do not possess the skills to monitor their own knowledge. As a result, they often fail to analyze relevant pointers to errors and omissions in their knowledge. We focus on explicit feedback mechanisms that help improve students' abilities to monitor their agent's, and as a result, their own learning in a way that improves their understanding and problem solving capabilities. Feedback in the Betty's Brain environment comes from a Mentor agent who answers student's queries, and Betty who demonstrates the use of metacognitive learning strategies while being taught by the student.

This paper focuses on the impact of directed versus guided feedback on student learning in TA environments. The results of an experimental study run in a 5th grade science classroom are discussed in terms of the students' immediate learning abilities and their preparation for future learning [12].

2 Betty's Brain

Our TA, Betty, shown in Fig. 1, is taught using a concept map representation [10]. Students teach her about entities, such as fish and algae, and their relations, (e.g., fish consume dissolved oxygen, algae replenish it) as they pertain to river ecosystems. Once taught, Betty uses qualitative reasoning methods to reason through chains of links [5], which helps her answer questions, such as "*if macroinvertebrates increase what happens to bacteria?*" Learning by teaching is implemented as three primary components: (i) *teach* Betty using a concept map, (ii) *query* Betty with your own questions to see how much she has understood, and (iii) *quiz* Betty with provided tests to see how well she does on questions you may not have considered. These activities are usually embedded within a larger narrative (e.g., teach Betty so she can pass a test to join a science club) [5].

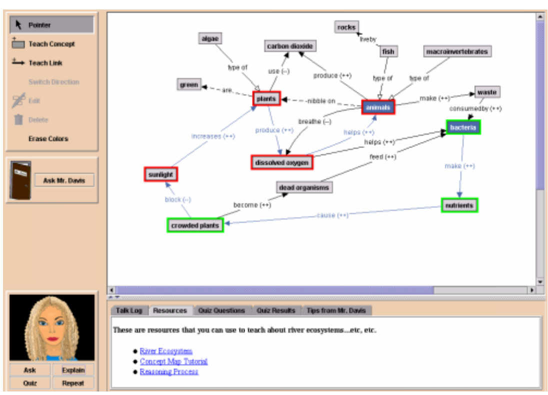


Fig. 1. Betty's Brain – Interface

When asked, Betty explains her answers using text and animation. Students reflect on Betty's answers and revise their own knowledge as they make changes to the concept maps. Details of the Betty's Brain system are summarized in [5]. Our work has demonstrated that one of the primary benefits of learning by teaching a TA is the need to structure knowledge in a compact and communicable format so that the student-teacher may develop important explanatory structures for the domain. The fact that TAs have independent performance and can show their reasoning based on how they have been taught also helps students (and teachers) assess their teaching. This should provide metacognitive and self-assessment opportunities for students that can lead to superior learning and transfer.

To help novice students with their learning and teaching tasks, we built additional resources into the environment: (i) domain resources organized as searchable hypertext so that students can look up information as they teach Betty, (ii) a concept map tutorial that provides students information on causal structures and how to reason with these structures, and (iii) a Mentor agent, Mr. Davis, who provides on-demand feedback about learning, teaching, and domain knowledge ("Ask Mr. Davis"). The Mentor also provides feedback immediately after Betty takes a quiz.

An experimental study conducted in a 5th grade science classroom has demonstrated that learning-by-teaching with metacognitive support for self-regulated learning helps students develop better learning strategies, and better prepares them for future learning on related topics, even when this learning happens outside of the support provided by the TA environment [5]. Students were divided into groups to work on three versions of the system: (i) Intelligent Tutoring System (ITS), where the Mentor asked students to create a concept map that would correctly answer a set of test questions, (ii) Learning by Teaching (LBT), where students taught Betty to help her pass a test so she could become a member of the school Science club, and (iii) Self-Regulated Learning (SRL), where students taught a Betty persona that incorporated metacognitive learning strategies [11, 15]. All three groups had access to identical resources on river ecosystems, the same quiz questions, and similar access to the query feature and the Mentor agent. The differences in performance for the three groups in the main study were not significant (we studied the quality of the concept maps students generated and the quiz scores). However, in a *preparation for future learning* task, where all students had to construct a concept map to answer questions about the land-based nitrogen cycle (a topic they had not studied before), the SRL group created maps with more concepts and links than the ITS and LBT groups. The effects of teaching self-regulation strategies had an impact on the students' abilities to learn a new domain [5]. These results were very encouraging, and prompted us to study metacognitive feedback in a more systematic way.

3 Previous Studies on Feedback

A number of researchers have studied the effects of timing and content of feedback provided by computer-based tutors and pedagogical agents on student learning and problem solving performance. With their Lisp Tutor, Corbett and Anderson [8] found that students who received immediate feedback (the tutor intervened as soon as students made errors and forced them to correct the error before they could move on) went through the tutoring lessons faster than students who had to ask the tutor for feedback. However, the latter group made lesser errors in post-test debugging and code generation tasks. This study demonstrated that immediate error feedback helped with immediate learning, but there were indications that providing students with more control (on-demand feedback) led to better retention and deeper understanding.

Aleven and Koedinger [1] performed studies on students' help-seeking behavior with a Cognitive Tutor for Geometry that provided on-demand help at multiple levels of detail, starting with general strategies relevant to a problem solving step to specific hints, which explicitly outlined the correct solution to the step. In initial studies, the researchers found that most students did not know when to ask for help and when prompted, they quickly clicked down to the most detailed hint. Students with higher pretest scores made fewer errors, asked for less hints, and did better on post tests. However, it was not clear that feedback helped students improve their overall learning. In later work [2], they incorporated self-explanation, where the students were required to explain their problem solving steps. In addition to error feedback, the system provided self-explanation hints centered on general strategies related to the current problem. Students showed deeper learning when the tutor required them to explain their steps. Students in the explanation condition spent more time on the system than students who were not required to provide self-explanations, but they needed fewer problems to achieve predetermined mastery levels for skills.

Moreno [9] used feedback as a mechanism for decreasing the cognitive load of novice students using software agents in discovery-based multimedia environments. Her study included: (i) a guided learning environment, where the agent provided explanatory feedback and (ii) a directed learning environment, where the agent provided corrective feedback. Her guided discovery hypothesis centered on the belief that learning occurs when learners actively construct a coherent knowledge representation by meaningful interactions with resource materials, converting the information extracted into representations, and integrating new information into existing representations. Typically, discovery learning results in high cognitive load for students with low prior knowledge, making it hard for them to learn. Her studies showed that the guided feedback group found the instructional material easier to follow, made significantly fewer errors on post-test questions, and was much better than the directed feedback group at transfer tasks that involved novel situations.

These studies support our early findings with the Betty's Brain system. On-demand, guided feedback is very likely to lead to better learning and transfer in discovery learning environments, where students are involved in the construction of knowledge. However, the complexity of the constructive discovery and problem solving tasks may overwhelm students who have low prior knowledge of the domain. The resulting frustration may distract the students and cause them to abandon their learning tasks. In the remainder of this paper, we discuss three versions of the Betty's Brain system designed to compare the differences between directed and guided feedback and study the effects of the content of guided feedback on students' understanding of domain knowledge and their preparation for future learning.

4 Different Forms of Feedback in Betty's Brain

For the three versions of Betty's Brain, we started with the previous LBT and SRL versions of the system. The LBT system is designed to provide directed or corrective feedback. Betty provides answers to queries and explanations of how she derived her answers using text, speech, and animation when asked by the student [5]. The Mentor provides feedback after Betty takes a quiz by overlaying the part of the expert map used to answer the quiz questions on the concept map the student created to teach Betty. For every incorrect answer, the Mentor looks to see if the concepts in the quiz question appear in the student's map. If they do not, the Mentor suggests that the student study these concepts in the resources. Otherwise, the Mentor looks for the first (i) missing expert concept, (ii) missing expert link, and (iii) incorrect expert links, in that order, to generate the appropriate feedback content. Like the Cognitive Tutors, the Mentor is designed to provide hints that range from general (e.g., "read about algae and dissolved oxygen") to specific ("you are missing a link between algae and dissolved oxygen in your concept map").

Our SRL system feedback is designed to teach students a set of comprehensive skills that includes setting goals for learning new materials and applying them to problem solving tasks, deliberating about strategies to enable this learning, monitoring one's learning progress, and then revising one's knowledge, beliefs, and strategies as new materials and strategies are learnt. Betty's persona in the SRL version incorporates self-regulation [5] and metacognitive strategies [13]. For example, when the student is building the concept map, she occasionally responds by demonstrating reasoning through chains of events. She may query the user, and sometimes remark (right or wrong) that the answer she is deriving does not seem to make sense. The idea of these spontaneous prompts is to get the student to reflect on what they are teaching, and perhaps, like a good teacher, check on their tutee's learning progress. At times, Betty may directly suggest to the students that they need to query her to ensure that she can reason correctly with the current concept map. At other times, Betty refuses to take a quiz, because she feels that she has not been taught enough, or that the student has not given her sufficient practice by asking queries.

In addition to comparing directed versus guided feedback, we were also interested in determining how the content of the feedback provided by the TA would affect the students' learning behaviors. Feedback content can be categorized as either cognitive or affective. Cognitive feedback is based on beliefs, thoughts, and rational arguments

Table 1. Example patterns of behavior andBetty's SRL-C and SRL-A responses

Pattern	Cognitive Response	Affective Response
If after four questions, Betty has not been queried on an unlinked concept	Excuse me. You taught me a concept, but didn't teach me any relationships between it and other concepts. Please teach me more, and ask me questions to make sure I understand	Hey, I'm confused and I don't understand what you taught me. Please teach me more, and ask me some questions.
If quiz and causal query but no update	Hey, you haven't taught me anything new since my last quiz. My score will surely be the same. Teach me something, and ask me some questions to make sure I understand, before you send me to take another quiz	Hey! You're making me do really hard things and I don't like it.
If no resource access and no improvement on previous quiz score	Excuse me. I like what you are teaching me, but it may not help me pass the quiz. I would like to be better prepared when I take it again. Could you check the resources and teach me about what you find there? Thanks.	Excuse me, but that quiz is very difficult. I really don't want to take it now. Can we do something else?

that are related to a problem-solving task while affective feedback is based on feelings or emotions that arise when performing a task [14]. Cognitive feedback is directed to helping the student develop better skills whereas affective feedback more likely invokes empathy and feelings for the agent. Both kinds of feedback may promote better learning among students, even if they are for different reasons.

To study the differences between affective and cognitive feedback, we created two versions of the SRL system that take their cues from the same set of patterns, but provide either cognitive or affective feedback. Several examples of the two kinds of feedback are illustrated in Table 1. The SRL-Cognitive (SRL-C) feedback is more content-directed, and students are provided with hints that help them apply metacognitive strategies to improve their learning, monitoring, and debugging tasks. The SRL-Affective (SRL-A) feedback is triggered by the same patterns as the SRL-C system, but Betty's responses are emotional and based on her feelings. The next section presents the experimental study we conducted to compare the effectiveness of the different kinds of feedback on student learning.

5 Experimental Study and Results

This new study, conducted with two 5th grade classrooms, was designed to compare the effects of the different types of feedback. 37 students from the two classrooms were divided into the three groups (SRL-C, SRL-A, and LBT) using a stratified sampling method based on standard achievement scores in mathematics and language. The students worked with Betty's Brain for seven 45-minute sessions. Their goal was to successfully teach Betty about river ecosystems and get her to pass three quizzes (answer all questions correctly). Approximately 10 weeks later, the students were given the transfer test, where they again taught Betty about a new domain: the land-based nitrogen cycle. The students worked for three sessions and this permitted us to determine which group was better prepared to learn in situations where scaffolds and feedback from their previous environments were removed. We measured student learning along two dimensions: (1) Students' performance in creating concept maps and (2) students' learning behaviors (in the form of quiz attempts, queries, and resource accesses). We believed that students previously in the SRL-C condition would demonstrate the best performance for future learning, and students in LBT condition with directed feedback would perform better than the students in the SRL-A condition, who received no useful feedback.

Experimental Results

Students' activities on the TA systems in the main and transfer study were recorded in log files, along with Betty's and Mr. Davis' feedback. The students' concept maps were also saved at the end of each session. In evaluating the students' concept maps we considered both "expert" and "relevant" concepts and links. Concepts and links that appeared in the expert map were labeled as "expert." However, other concepts and links that corresponded to a correct understanding about the domain (though they were not required to answer quiz questions) were coded as "relevant." The set of expert and relevant concepts and links are called "valid."

Table 2 summarizes the average number of expert and valid concepts and links for each condition at the end of the final session. Overall, the LBT group had more expert links in their maps than the two SRL groups. However, the SRL-C group had more valid links than the LBT and SRL-A groups. We believe that the directed feedback focused on the quiz questions directed the LBT students only to the missing expert concepts and links in their maps. The metacognitive feedback for the SRL-C group was more general, and focused students on acquiring knowledge from the resources and organizing it into their concept maps. We believe that learning such behavior would promote better abilities to learn on one's own.

Table 2. Means and standard deviations (sd) of the number of concepts and links in the final concept map for the main study. (Differences were not statistically significant).

Main study	SRL Cognitive Mean (sd)	SRL Affective Mean (sd)	LBT Mean (sd)
Expert concepts	9.23 (1.7)	9.15 (1.8)	8.27 (2.2)
Valid concepts	21.69 (11.0)	18.00 (9.9)	14.27 (6.2)
Expert links	5.38 (3.0)	5.62 (3.9)	7.18 (4.7)
Valid links	18.15 (10.5)	14.92 (12.6)	13.36 (8.0)

To study this, we performed a similar analysis on the transfer test concept maps. All students used identical barebones systems with no feedback. Analysis of the expert concepts and links produced no statistically significant differences. However, the SRL-C group had more valid concepts and links than the other two groups, and the differences in the number of valid concepts was statistically significant (see Table 3). This demonstrates the effectiveness of metacognitive, guided feedback in preparation for future learning.

Table 3. Means and standard deviations (sd) in students' final concept map by group for the transfer test concepts and links. Significant differences are indicated.

Transfer test	SRL Cognitive Mean (sd)	SRL Affective Mean (sd)	LBT Mean (sd)
Expert concepts	6.54 (1.9)	5.31 (2.7)	6.09 (4.1)
Valid concepts	14.69[ab] (5.5)	10.23 (4.9)	10.27 (5.6)
Expert links	1.923 (1.4)	2.25 (2.7)	3.80 (5.0)
Valid links	10.85 (7.6)	8.5 (6.5)	9.3 (6.9)

[a] Significantly greater than SRL-A, $p < .05$; [b] Significantly greater than LBT, $p < .05$

In addition to evaluating students' performance, we also monitored their behaviors in the main study and the transfer test. Analyzing the student log files revealed differences between the three groups that can be attributed to the differences in feedback they received. We focus on quiz attempts, queries asked, and resource accesses as they demonstrate the students' abilities to monitor their learning and seek new information. We computed the average counts of each behavior across all sessions in both the main study and the transfer test.

Correlations were computed to see if the behavior patterns could be linked to improvements in the students' concept maps on a session by session basis. The number of new valid links added in each session for each student was compared against the number of times the specific actions were performed for that session. These pairs were aggregated for each group, and a Pearson's r test was performed to determine the significance of the association between the two sets of variables.

Figure 2a shows that the SRL-A group made a much larger number of quiz attempts when compared to the SRL-C group in all of the main study sessions. Asking Betty to take a quiz allowed the students to monitor Betty's progress and their own teaching and learning. However, making Betty repeatedly take quizzes showed that the student resorted to a trial-and-error pattern. To avoid this behavior, we created an SRL behavior pattern (see Table 1) where Betty refuses consecutive quiz attempts if the concept map is not updated and the student has not queried Betty between the quiz attempts. The number of quiz attempts refused shows an even larger difference between the two SRL groups (Figure 2d and Table 4). Although Betty refuses to take a quiz in exactly the same situations for both SRL groups, the SRL-C group seemed to avoid the trial-and-error strategy and focus on more systematic and effective learning methods. Interestingly, when the SRL patterns and feedback were removed in the transfer test, the SRL-C group continued to use the quiz feature more effectively than the SRL-A group (see Figure 3a). The difference in quiz attempts, combined with the fact that the SRL-C students were more successful in adding valid links to their concept maps in the transfer test, shows that they effectively used the quiz feature even when the scaffolding and feedback were removed.

Table 4. Means, standard deviations, and statistical significance for the main study counts

Main study	SRL Cognitive Mean (sd)	SRL Affective Mean (sd)
Quiz attempts refused (session 2)	1.77 (1.4)[a]	3.54 (2.2)
Quiz attempts refused (session 3)	1.92 (1.8)[a]	4.15 (3.3)
Quiz attempts refused (session 4)	4.31 (4.8)	10.54 (10.0)
Quiz attempts refused (session 7)	4.08 (3.9)[a]	9.33 (6.8)

[a] Significantly less than SRL-A, $p < .05$

Main study	SRL Cognitive Mean (sd)	SRL Affective Mean (sd)	LBT Mean (sd)
Queries (session 6)	7.00 (6.0)	12.92 (9.1)[ab]	3.73 (3.8)
Queries (session 7)	6.62 (6.7)	16.17 (13.7)[ab]	4.00 (4.6)

[a] Significantly greater than SRL-C, $p < .05$; [b] Significantly greater than LBT, $p < .05$

Main study	SRL Cognitive Mean (sd)	SRL Affective Mean (sd)	LBT Mean (sd)
Resource accesses (session 5)	4.92 (4.6)	1.92 (2.2)[ab]	5.00 (3.2)
Resource accesses (session 6)	5.08 (3.3)	2.00 (2.6)[a]	3.55 (2.7)

[a] Significantly less than SRL-C, $p < .05$; [b] Significantly less than LBT, $p < .05$

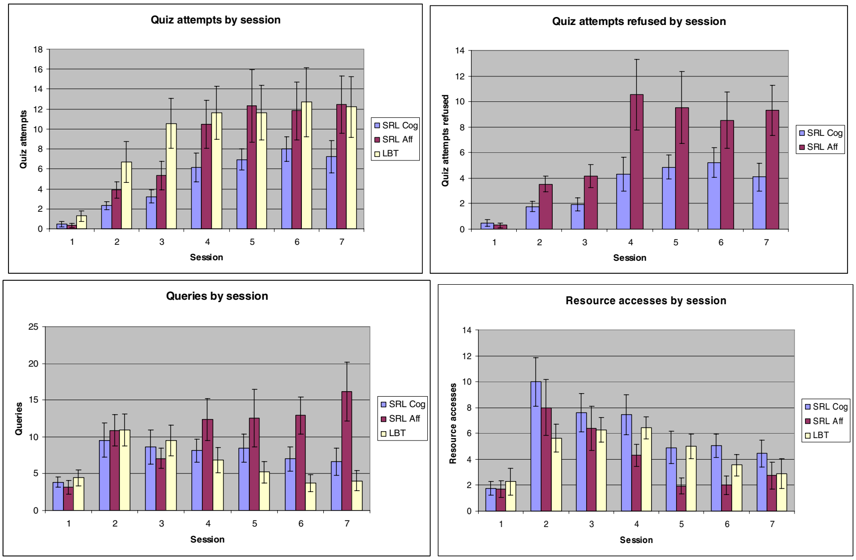


Fig. 2. Number of quiz attempts (a – top left), queries (b – bottom left), and resource accesses (c – bottom right) by session for the main study. Quiz attempts refused per session (d – top right) for the two SRL groups (the LBT system does not have SRL patterns).

In support of this claim, the correlation analysis showed that only the SRL-C group demonstrates a significant correlation between quiz attempts per session and number of added valid links for that same session. This correlation is negative, indicating that the students who were successful in adding valid links did not have to repeatedly use the quiz feature (Pearson r=-0.247, n=91, p=0.009). This behavior persisted in the transfer test.

Figure 2b shows that the SRL-A group used the query feature excessively in the later sessions. In particular, sessions 6 and 7 show a statistically significant difference between the use of the query feature between groups (see Table 4). Since the query feature is an important mechanism for monitoring Betty's knowledge, and, therefore, the student's teaching performance, the SRL Betty prompts the student to query her from time to time. This occurs most often after Betty refuses to take a quiz because the student has not checked to see if she has learnt what she was taught. Whereas the SRL-C group seemed to realize the role of the query feature in monitoring Betty's and their own knowledge, it is clear that the SRL-A group were generating queries just to get Betty to take the quiz. This group had many more quiz attempts refused, despite their efforts to use the query feature to get Betty to take the quiz. In the transfer test, there is no significant difference in the number of queries asked by the groups (see Figure 3b). This further supports the fact that the SRL-A group did not understand the role of queries. A correlation analysis for the main study showed that both the SRL-C and LBT groups exhibited a positive correlation between queries asked per session and number of added valid links for that same session (Pearson R for SRL-C: r=0.196, n=91, p=0.031; for LBT: r=0.226, n=77, p=0.024). This did not hold for the SRL-A group.

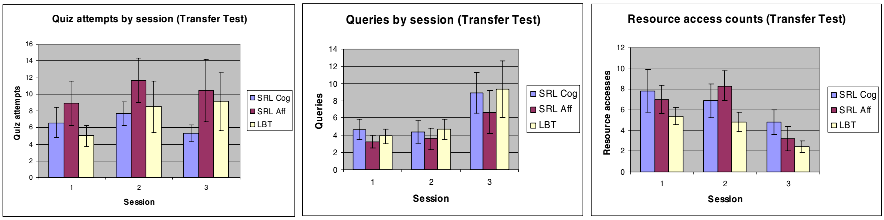


Fig. 3. Number of quiz attempts (a – left), queries (b – middle), and resource accesses (c – right) for all three groups across all sessions for the transfer test

Finally, examining students' resource accesses across the three groups reveals an interesting pattern in the students' behavior. The key to successfully teaching Betty hinges upon extracting information from the resources and transferring it into the concept map structure. The correlation analysis for the resource accesses per session and number of added valid links for that same session showed a positive correlation for all groups, although the data is statistically significant only for the SRL-A and SRL-C groups (Pearson R for SRL-C: $r=0.319$, $n=91$, $p=0.001$; for SRL-A: $r=0.260$, $n=88$, $p=0.007$; for LBT: $r=0.099$, $n=73$, $p=0.202$). This supports the obvious claim that the amount of resource use is directly related to the quality of the student's concept map. However, Figure 2c shows that the SRL-A group's resource accesses showed a significant drop from session 4 on (see Table 4). This suggests that Betty's affective explanations may have been more of a hindrance than a help for the SRL-A students, and their enthusiasm for teaching Betty may have dropped because of their lack of success. It is interesting to see the LBT group with no metacognitive prompting used the resources more often than the SRL-A group, despite the fact that they received corrective feedback from the Mentor.

6 Conclusions

The results from this study demonstrate the performance and behavioral differences in learning that can be associated with the different types of feedback provided by our TA system. Although directed or corrective feedback may allow the student to quickly achieve immediate goals set by the learning environment, like earlier work [9], we have demonstrated that guided metacognitive feedback better prepares the student for learning even when the student is removed from the learning environment. This was illustrated in the transfer study, when the students from the three different conditions were asked to learn a new domain in an environment with no scaffolding and very little feedback.

We have also demonstrated that the *presence* of guided feedback based on metacognitive cues is not enough. Students have to be explicitly taught the metacognitive strategies, and be given enough opportunities to practice them like the main study. The differences between the SRL-C and SRL-A groups indicate that the *type* of feedback received has a significant effect on learning outcomes. Students receiving cognitive content feedback were better able to learn from the teachable agent's metacognitive behaviors, but it is clear that the SRL-A students did not learn even though feedback was given in the same situations for both groups.

There are other issues in the learning by teaching framework with computer-based Teachable Agents that may contribute to student learning. One issue is the social aspect of the teaching process that very likely contributes positively to the student's motivation, and therefore, enhances his or her ability to learn. A second issue has to do with the notion of shared responsibility. In the TA environment, the agent knows only what the student has taught her. On the other hand, the student typically knows little of how to reason with causal structures, but learns by observing the agent answer questions and explain her answers. This results in a significant decrease in the student's cognitive load during initial learning. Last, there is the issue of the student monitoring their agent's knowledge, as opposed to their own (though, in reality it is their own knowledge). This again may result in a reduction of cognitive load, since the student is not problem solving and debugging their problem steps at the same time. (They would have to do this if they were generating a self-explanation for a problem they had solved). In future work, we would like to bring together all of these issues in redesigning our learning environments with guided, metacognitive feedback to provide better mechanisms that enable learning with deep understanding.

Acknowledgements

This work has been supported by a NSF ROLE Award #0231771. The assistance provided by the Teachable Agents group, especially Daniel Schwartz, Kadira Belynne, Bilikiss Adebiyi, Krittaya Leelawong, Marc Santoro, John Bransford and Nancy Vye are gratefully acknowledged.

References

1. Aleven V. & Koedinger, K. R., Investigations into Help Seeking and Learning with a Cognitive Tutor. *Working Notes AIED Workshop on Help Provision and Help Seeking In Interactive Learning Environments*. San Antonio, Texas, May 2001.
2. Aleven, V., McLaren, B., Roll, I., & Koedinger, K., Toward tutoring help seeking: Applying cognitive modeling to meta-cognitive skills. *Proceedings of Seventh International Conference on Intelligent Tutoring Systems*, pp. 227-239, 2004.
3. Artzt, A. F., & Armour-Thomas, E. Cognitive Model for Examining Teachers' Instructional Practice in Mathematics: A Guide for Facilitating Teacher Reflection. *Educational Studies in Mathematics*, 1999: 40(3), pp. 211-335.
4. Bargh, J.A. and Y. Schul, On the cognitive benefits of teaching. *Journal of Educational Psychology*, 1980. 72(5): pp. 593-604.
5. Biswas, G., D. Schwartz, K. Leelawong, N. Vye, and TAG-V, "Learning by Teaching: A New Agent Paradigm for Educational Software," *Applied Artificial Intelligence*, special issue on Educational Agents, vol. 19, no. 3, pp. 363-392, March 2005.
6. Bransford, J. D., & Schwartz, D. L. Rethinking transfer: A simple proposal with multiple implications. In A. Iran-Nejad & P. D. Pearson (Eds.), *Review of Research in Education*, vol. 24: 1999, pp. 61-101. Washington DC: American Educational Research Association.

7. Chi, M.T.H., Siler, S. A., Jeong, H., Yamauchi, T., & Hausmann, R. G., Learning from Human Tutoring. *Cognitive Science*, 2001. 25(4): pp. 471-533.
8. Corbett, A.T. and Anderson, J.R., Locus of Feedback Control in Computer-Based Tutoring: Impact on Learning Rate, Achievement and Attitudes, *Proc. CHI Letters*, vol. 3, no. 1, pp. 245-352, 2001.
9. Moreno, R., Decreasing Cognitive Load for Novice Students: Effects of Explanatory versus Corrective Feedback in Discovery-Based Multimedia. *Instructional Science*, vol. 32, pp. 99-113, 2004.
10. Novak, J.D., Concept Mapping as a tool for improving science teaching and learning, in *Improving Teaching and Learning in Science and Mathematics*, D.F. Treagust, R. Duit, and B.J. Fraser, Editors. 1996, Teachers College Press: London. pp. 32-43.
11. Pintrich, P. R. and E. V. DeGroot, Motivational and self-regulated learning components of classroom academic performance, *Journal of Educational Psychology*, 1990, vol. 82: pp. 33-40.
12. Schwartz, D. L. and Martin, T. Inventing to prepare for learning: The hidden efficiency of original student production in statistics instruction. *Cognition & Instruction*, 2004: 22, pp. 129-184.
13. Schwartz, D.L., K.B. Pilner, G. Biswas, K. Leelawong, and J.P. Davis, Animation of Thought: Interactivity in the Teachable Agent Paradigm. *Learning with Animation: Research and Implications for Design*, Lowe, Cambridge Univ. Press, to appear.
14. Verplanken, B., Hofstee, G. & Janssen, H. Accessibility of affective versus cognitive components of attitudes. *Eur. Journal of Social Psychology*, vol. 28, pp. 23-35, 1998.
15. Zimmerman, B. J. A Social Cognitive View of Self-Regulated Academic Learning. *Journal of Educational Psychology*, vol. 81(3), pp. 329-339, 1989.

Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems

Jason A. Walonoski and Neil T. Heffernan

Worcester Polytechnic Institute, Computer Science Department, 100 Institute Rd, Worcester, MA 01601 USA
{jwalon, nth}@wpi.edu

Abstract. A major issue in Intelligent Tutoring Systems is off-task student behavior, especially performance-based gaming, where students systematically exploit tutor behavior in order to advance through a curriculum quickly and easily, with as little active thought directed at the educational content as possible. The goal of this research was to explore the phenomena of off-task gaming behavior within the *Assistments* system. Machine-learned gaming-detection models were developed to investigate underlying factors related to gaming, and an analysis of gaming within the Assistments system was conducted to compare some of the findings of prior studies.

1 Introduction

Intelligent Tutoring Systems (ITS) have been shown to have a positive effect on student learning [1], however these effects may be negated by a lack of student motivation or student misuse. One area of research examining these issues involves studying student "gaming" of the system, especially recognition of gaming behavior [2]. A student is gaming if they are attempting to systematically use the tutor's feedback and help methods as a means to obtain a correct answer with little or no work, in order to advance through the curriculum as fast or easily as possible. Student gaming has been correlated with substantially less learning [3]; therefore it is of particular importance to understand in order to maximize tutor effectiveness.

One objective of this research was to apply existing methodologies of gaming behavior detection to the *Assistments* mathematics ITS [4]. These methods involved the construction of machine-learned models to identify gaming behavior. Although gaming behavior has only two hallmark appearances (help abuse and systematic guessing and checking), there may be various hidden factors at work: some students are harmed by their gaming while others are not. Machine learning has been shown to be able to differentiate between these two types of gamers [2].

Additionally, previous work by Baker et al [2][3] has resulted in documentation of the phenomenon of gaming within ITS and theories about why students game. Our second objective was to corroborate or contradict those findings by profiling the typical gaming student in the *Assistments* system and exploring the relation of gaming behavior with student learning, and then comparing the results of the studies.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 382 – 391, 2006.

2 Detection of Gaming

If tutoring software is outfitted with a model that can reliably identify gaming behavior, then intervention strategies can be developed and deployed with reasonable assurance that they are invoked under appropriate circumstances. Rather than manually constructing a model by authoring rules based on the surface features of gaming (systematic guessing and checking or requesting help until answers are directly supplied), machine-learning methods were employed to identify the underlying hidden variables that lead students to game and illustrate how they are affected by their behavior. A prior study by Baker et al has shown that gaming behavior can be reliably detected with machine-learned models [2], and in the course of constructing a model for the *Assistments* system we adapted those methods for general verification of their findings, to determine if gaming behavior has the same causes, appearances, and resulting effects in different tutoring systems.

The methodology used was essentially a four-step process: (1) classroom observation of students using the tutoring software, (2) dataset creation based upon those observations to be used by machine-learning algorithms, (3) the construction of classifiers (prediction models) using the datasets, and (4) analysis of the results.

2.1 Classroom Observation

Students were observed using the *Assistments* ITS in real classroom sessions. Each observation was a triplet of observation time, student identity (alias), and recorded behavior. Observation was conducted as unobtrusively as possible, with students unaware that surveillance was taking place (students treated observers as assistant teachers and displayed no knowledge of being systematically observed). Observations were taken from a modest distance to (1) minimize student self-consciousness and awareness of being watched, (2) allow the observer flexibility in positioning themselves within the environment to maximize line of sight, and (3) allow the simultaneous observation of groups of students. For consistency purposes, each observer was given an instruction sheet on how to conduct observations. Groups of students were observed for approximately 20 to 30 seconds per student; so a group of 3 students would have been observed for 60 to 90 seconds. The possible variation of observation times was left to the observer depending on the consistency or deviation in the students' behavior in order to get a representative measurement. A numerical code from Table 1 (adapted from measurements in [3]) was recorded for each student per observation period. During a given observation period, a student might exhibit multiple behaviors. In that case, rather than record all the behaviors, observers were instructed to give priority where gaming behavior was given the highest priority, followed by the three on-task behaviors (sorted by least engaged to most engaged), followed by the remaining two off-task behaviors (sorted by most active to least active).

To ensure that this methodology was not subjective to observer bias, an inter-rater reliability study was performed. Two observers (one of which was the first author) were provided with the observation instructions and then observed two classes, with students observed in the exact same order and at the exact same time. The two observers made 71 observations each. The consistency across all six numerical behaviors was 77% (57 out of 71 observations matched). The consistency across the three

Table 1. Measurement Definitions

Category	Observation	Definition	Priority
A - On Task	1	On Task with the Tutor	4
A - On Task	2	On Task with Paper or Teacher (including talking about the problem)	3
A - On Task	3	On Task, but talking while working (subject matter of conversation is irrelevant)	2
B - Off Task	4	Off Task and Talking	5
B - Off Task	5	Off Task and Inactive (including web-surfing, staring into space, sleeping, et cetera)	6
C - Gaming	6	Gaming (guessing-and-checking or bottom-out-hinting)	1

alpha-encoded categorical behaviors was 97% (69 out of 71 observations matched), while there was 100% agreement in the identification of gaming behavior. Since our classifier was aimed purely at the identification of gaming behavior, as opposed to distinction between all behaviors, these results have a suitable level of consistency.

Overall, 850 observations were recorded, spanning 8 classes that lasted approximately 50 minutes each. Those 8 classes consisted of experienced users of the *Assistments* system, who had been using it biweekly for the entire 2004-2005 academic year.

2.2 Dataset Creation

The *Assistments* system automatically logs all user actions and events except mouse movements. From these logs we can distill information such as a student's number of attempts (including whether the attempt was correct or incorrect, or if it was the first attempt on a given problem), numbers of hint requests (including bottom-out hint requests that directly supply the correct answer), and action response time in milliseconds. Actions were joined to the recorded classroom observations by user identification and time to create training instances for the machine-learning algorithms.

Given the length of time spent observing particular students, it is not clear which actions should be matched with a particular observation. To resolve this issue, actions were joined to observations using an "unsupervised action filter" based on a variable "time window." Informally, a time window is defined as a dilation of time around a recorded observation time. Not being sure what size time windows are reasonable, five sizes were utilized: 30 seconds, and 1, 2, 4, and 6 minutes. For example, given a 2-minute time window, all actions made between 1 minute before and after each observation were included in the generation of the training instances. The filter is considered "unsupervised" because no attempt is made to filter in or out actions based on their applicability to the recorded observed behavior.

Using the observations and action logs, a number of datasets were generated via unsupervised action filtering using time windows. The datasets had 1430 attributes and 1 classification value (gaming, *true* or *false*). The six observation types in Table 1 were rolled up into either gaming is *true* (observation #6) or gaming is *false* (all other observations). Machine-learning algorithms are dependent on relevant attributes, so the selection of attributes is an important exercise. We adapted the attributes of Baker et al [2] to the particulars of the *Assistments* system and variable time windows. For an observation within a particular time window, the attributes were as defined:

- Actions: the total number of all actions.
- Attempts: six attributes for the total number of all attempts, correct attempts, incorrect attempts, correct first attempts, and incorrect first attempts.
- Attempt Time: five attributes for the sum, minimum, maximum, average, and standard deviation of all attempt times in milliseconds. Also four Boolean attributes were included indicating whether attempt times were slow, extra-slow, quick, or extra-quick, which were calculated by comparing the student response time with the average response time of all students on the given problems (and plus or minus the standard deviation of all student response times for the extra-slow and extra-quick attributes).
- Hints: two attributes for the total number of hint requests and bottom-out hint requests.
- Hint Time: five attributes for the sum, minimum, maximum, average, and standard deviation of all hint request times in milliseconds. Also four Boolean attributes were included indicating whether hint request times were slow, extra-slow, quick, or extra-quick, which were calculated by comparing the student response time with the average response time of all students on the given problems (and plus or minus the standard deviation of all student response times for the extra-slow and extra-quick attributes).
- Problems: two attributes for the total number of top-level problem questions, and the total number of follow-through helping questions.
- User-Interfaces: two attributes for the total number of questions that featured a multiple-choice user-interface and another for the total number of questions that featured a textbox user-interface.
- Replays: the total number of times a problem was "replayed" by the *Assistments* tutor runtime [5] (this generally indicates that the student tried to exit the system and the runtime had to "replay" the students actions on a given problem to reconstruct the tutors agenda *exactly* for the given problem).
- *pmpKnow*: "poor man's prior knowledge," the probability that the student possesses the prior knowledge required to answer the given question correctly. Prior knowledge in ITS is often determined by knowledge tracing, however the *Assistments* system currently lacks dynamic knowledge model tracing, so as a substitute we use the poor man's version: the student's percent correct across all previous problems. Also four Boolean attributes were included indicating whether the prior knowledge was high, extra-high, low, or extra-low in comparison to the average prior knowledge of all students and in combination with the standard deviation of that average (for the extra-high and extra-low variables).

- Problem-Difficulty: four attributes for the minimum, maximum, average, and standard deviation of problem difficulties for all problems encompassed within the observation time window. Problem difficulty is a number between 0 (easy) and 1 (hard) that is the percent correct on first attempt by all previous students. The combination of these values would hopefully represent the range of difficulty during the observation.
- Ratios: six attributes representing the following ratios: the number of attempts per problem, the number of correct attempts per problem, the number of incorrect attempts per problem, the number of hints per problem, the number of bottom-hints per problem, and the number of replays per top-level problem.
- Pair-wise Interaction Effects: 1378 attributes representing the quadratic effects between any two of the attributes listed above. For example, the total number of hints times the average problem difficulty. The list of pair-wise interaction effect attributes is comprehensive (all the original attributes have a pair-wise interaction effect attribute with every other original attribute, including itself).

2.3 Machine-Learning Algorithms

We used 12 different algorithms from the WEKA machine-learning system [6] on our datasets to generate models including decision tree methods, lazy methods (k-nearest neighbors), locally weighted learning, Bayesian methods, a neural network, a propositional-logic rule learning algorithm (PRISM) [7], as well as logistic regression. A large number of algorithms were used out of curiosity because each has advantages and disadvantages (which are outside of the scope of this document) that could potentially reveal different kinds of relationships within the data. Some of the algorithms generate human-readable rules while others produce mathematical models that are often difficult to interpret by humans. The results were then examined for (1) classification accuracy, (2) accuracy of the confusion matrices, and (3) reasonable rules, especially those that might corroborate or contradict expected findings based on previous studies, or other interesting results.

2.4 Results and Discussion

None of the algorithms used significantly outperformed any of the others (according to statistical tests automatically performed by WEKA). Therefore, choosing a final model rested on a selecting a classifier that generated reasonable rules that corroborated both the surface-level hallmark characteristics of gaming and the findings of previous studies.

The classifier that was ultimately selected as our preferred model was generated using the J48 decision tree algorithm (based on Quinlan's C45 algorithm [8]). Although other algorithms had faster classification times or higher accuracies, this model was chosen because across all training and testing folds it produced reasonably clean confusion matrices, generated human-readable rules that upon interpretation seemed to corroborate findings from past studies.

Several of our resulting rules offered further support to the hypotheses of Baker et al [2][3] that suggest that one cause of gaming is low prior knowledge combined with problem difficulty. Other rules could be interpreted in such a way as to identify the class of "gamed-not-hurt" students, which supports the Baker et al distinction between students whose learning is affected by gaming and those who are not. Finally, results at the four and six minute intervals suggests that using longer time windows does not adversely effect the detection of gaming, and in fact improves as those students who game tend to make a habit of it and identifying them becomes easier and easier as they continue their off-task behavior.

After being selected as the preferred model, the J48 algorithm was rerun using leave-one-out cross-validation (LOOCV) as the testing method. This involved generating our model 850 times, each time using 849 of the 850 instances for training purposes and leaving out one different instance for testing, and then using the model to predict whether the 850th instance was gaming or not. This process was repeated for each of the datasets (one for each time window).

While most of the models resulting from LOOCV had 100% classification accuracy, averaging out the results of all models results in about 96% accuracy. Given the low rate of observed gaming (19 out of 850 observations, ~2.2%), the effectiveness of the models becomes questionable. Analysis of the confusion matrices helps our understanding of how the models perform. On average, the models tend to correctly identify non-gaming instances about 98% of the time, while correctly identifying gaming instances only about 19% of the time. Although this is not ideal, if we consider that gaming is much more harmful to learning than other behaviors [3] and it is such an infrequent behavior, then 19% of gaming instances may seem better than what might be expected from chance alone. So, while the model accuracy leaves something to be desired, we are at least satisfied in the general reasonability of the resulting "rules" given what is known about gaming behavior.

Ultimately, results of our final model were satisfactory since construction of the datasets and models verified some of the underlying hidden variables that lead students to game (e.g. low prior knowledge), and the generated rules were human-readable and reasonably captured the hallmark surface-level characteristics and other known causes of gaming behavior. Although we would like to improve the accuracy and strength of our final model, it could be outfitted as-is into the *Assistments* system to dynamically detect gaming behavior and trigger various intervention strategies, as a post-tutoring reporting device, or as an objective evaluator of various intervention strategies within controlled experiments.

3 Gaming Within the Assistments System

The last portion of this research was a general examination of gaming behavior within the *Assistments* system. This examination made use of a *prima facie* algorithm (as opposed to the machine-learned model) that calculates how frequently individual students gamed based on surface-level features of hint abuse and guessing-and-checking only. If a student asks for a hint on, or answers incorrectly (possible guess), any step within a given problem three consecutive times, then they are assumed to be gaming that problem. When a problem is gamed, a *possibly-gaming* index is increased by one. If an entire problem is completed without any step being gamed, then the *possibly-gaming* index is reduced by one. If at any time the *possibly-gaming* index is above three, any further identified gaming increases a student's *total-gaming-score* by one.

3.1 Assistments System Survey Responses

A survey was administered to students who had been using the *Assistments* system throughout the 2004-2005 academic year on a biweekly basis. The survey consisted of 32 Likert-scale questions and some open response items. Gaming scores were calculated for those students who completed the entire survey using the *prima facie* algorithm. Depending on where a students average score fell in relation to the overall average and the overall standard deviation, they were classified as very-high, above-average, below-average, or very-low gamers. Out of 365 students, 53 were very-high gamers, 91 were above-average, 179 were below-average, and 42 were very-low gamers. By analyzing the distribution of responses by gaming-classification we constructed the following profile:

- Mathematics: Students who gamed were more likely to believe that they were not good at math and less likely to believe they could do well at math if they worked hard. Students who gamed often said they were less likely to do homework in math class, and the more students gamed, the less they said they liked math class. The less a student gamed the more they were likely to strongly agree that they would use math in a job when they grew up. Students who gamed often were much more likely than other students to strongly agree that their parents thought it important for them to do well in math, which may explain the performance-based motivation behind some gaming.
- Computers: Even though students who gamed often were less likely to have a computer at home, they were also less likely to report having trouble concentrating on the computer. Students who gamed often agreed more often and more strenuously that they liked learning from a computer than those who gamed very little.
- Educational Medium: Students who tended not to game were more likely to say that they preferred using the *Assistments* system to doing homework. In a similar question, there were no differences between the groups when asked if they would prefer to use the tutor rather than take a test – they mostly all strongly agreed that they would. The less a student gamed, the more strongly they would prefer using the *Assistments* system to normal classroom activity.
- Help Seeking: Students who gamed very little were more likely to strongly agree that they would seek help when they didn't understand something. The more a student gamed, the more they thought that being told the answer was more helpful than reading the hints. The more a student gamed, the more they agreed that the hints aided in their understanding of similar problems.
- Problem Difficulty: Students who gamed often tended to strongly agree that the items were frustrating because they were too hard, while students who gamed very little were more likely to disagree. This is probably partially

related to student prior knowledge. Students who gamed often were more likely to agree or strongly agree that they tried to get through difficult problems as quickly as possible.

- Goals: Students who gamed often tended more than other students to say that their goal was to get through as many items as possible. Interestingly, the more students gamed, the more they also tended to strongly agree that their goal was to learn new things.
- Students who gamed often had a slight tendency to say that they prefer facts and data to concepts and ideas more than other students.

Some of these results are interesting merely because they either corroborate or disagree with past findings. For example, Baker et al have reported that students who game do not like computers [9], while our survey suggests that those students who appeared to be heavily gaming *prima facie*, agreed more often and more strenuously that they liked learning from a computer than those who gamed very little. However, our survey results show that gamers were less likely to own a computer at home, and were more likely to dislike math class (also inconsistent with previous findings).

3.2 Gaming and Learning

Off-task gaming behavior has been correlated with substantially less learning in several prior studies [3]. In an attempt to validate those findings, learning rates that had been previously calculated for [10] using traditional methods as well as longitudinal data analysis, were grouped by the very-high, above-average, below-average, and very-low gaming categories. The results are summarized in Table 2.

These results seem to corroborate previous findings that indicate that off-task gaming behavior is correlated with substantially less learning. However, a few statistical tests were run to examine the significance of these results. Before those tests were run, students were classified as being gamers or not. If a student's score put them at the level of very-high gaming, then they were a considered a gamer, and otherwise they were not. This was done to simplify the results and make them easier to interpret.

Table 2. Learning Rates by Gaming Category

Category	Traditional Slope	Traditional Intercept	SW Slope	SW Intercept	Actual MCAS	Scaled Ans.
Very Low Gaming	1.68	26.92	0.34	24.04	35.17	0.76
Below Avg Gaming	1.62	19.53	0.33	19.31	30.56	0.80
Above Avg Gaming	1.29	15.07	0.33	14.23	24.24	0.70
Very High Gaming	0.95	11.99	0.26	11.71	19.66	0.77
Overall Average	1.44	17.88	0.32	17.25	27.69	0.76

The first test was an analysis of variance (ANOVA) of learning (SW slope) by gaming status. The results suggest that the learning rates of students who game and those that do not are reasonably different than mere chance alone, and they show that gaming behavior is correlated with less learning ($p < 0.19$).

The second test was an ANOVA of knowledge (SW intercept) by gaming status. The results very strongly indicate that students who engage in gaming behavior are more likely to come to the *Assistments* system with lower prior knowledge than other students ($p < 0.0001$).

One last test examined the correlation of gaming with a students actual MCAS score via Bartlett's Test of Sphericity, which very strongly showed that gamers do not perform well on the actual MCAS state administered mathematics exam ($p < 0.0001$).

These three tests show that *prima facie* gamers start with less knowledge, learn less, and perform worse on the actual MCAS examination.

4 Conclusions

Off-task gaming behavior is a major issue within the field of ITS, since it has been correlated with poor learning. The goal of this research was to explore this important phenomenon within the *Assistments* system. A machine-learned decision-tree model for gaming detection was developed, and while the practicality of this model was questionable, the resulting rules corroborated the connection of low prior knowledge and problem difficulty with gaming. Further analysis of gaming and its effects within the *Assistments* system was undertaken, via student survey responses and student learning-rates. The survey results provide some agreement and disagreement with previous studies about the nature of gaming, and the learning rates corroborated findings that indicate that off-task gaming behavior is correlated with lower prior knowledge and less learning.

Acknowledgements

This research was made possible by the US Dept of Education, Institute of Education Science, "Effective Mathematics Education Research" program grant #R305K03140, the Office of Naval Research grant # N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. All the opinions in this article are those of the authors only. This work would not have been possible without the assistance of the 2004-2005 WPI/CMU *Assistments* team including Mingyu Feng, Andrea Knight, Ken Koedigner at CMU, Abraao Lourenco, Michael Macasek, Goss Nuzzo-Jones, Kai Rasmussen, Leena Razzaq, Steven Ritter at Carnegie Learning, Carolyn Rose at CMU, Terrence Turner, and Ruta Upalekar.

References

1. Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). *Intelligent tutoring goes to school in the big city*. International Journal of Artificial Intelligence in Education, 8, 30-43.
2. Baker, R.S., Corbett, A.T., Koedinger, K.R. (2004) *Detecting Student Misuse of Intelligent Tutoring Systems*. Proceedings of the 7th International Conference on Intelligent Tutoring Systems.

3. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. (2004) *Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System"*. Proceedings of ACM CHI 2004: Computer-Human Interaction.
4. Razzaq, L, Feng, M., Nuzzo-Jones, G., Heffernan, N.T. et. al (2005). *The Assistment Project: Blending Assessment and Assisting*. Proceedings of the 12th Annual Conference on Artificial Intelligence in Education, Amsterdam.
5. Nuzzo-Jones, G., Walonoski, J.A., Heffernan, N.T., Livak, T. (2005). *The eXtensible Tutor Architecture: A New Foundation for ITS*. Proceedings of the 12th Annual International Conference on Artificial Intelligence in Education, Amsterdam.
6. Ian H. Witten and Eibe Frank (2005). "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005.
7. J. Cendrowska (1987). *PRISM: An algorithm for inducing modular rules*. International Journal of Man-Machine Studies. Vol.27, No.4, 349-370.
8. JR Quinlan. "C4. 5: Programs for Machine Learning." The Morgan Kaufmann Series in Machine Learning, San Mateo, 1993.
9. Baker, R.S., Roll, I., Corbett, A.T., Koedinger, K.R. (2005) *Do Performance Goals Lead Students to Game the System?* Proceedings of the 12th International Conference on Artificial Intelligence and Education, Amsterdam.
10. Feng, M., Heffernan, N.T, Koedinger, K.R. (2006) *Addressing the Testing Challenge with a Web-Based E-Assessment System that Tutors as it Assesses*, Proceedings of WWW2006, Edinburgh, Scotland.

Adapting to When Students Game an Intelligent Tutoring System

Ryan S.J.d. Baker[1], Albert T. Corbett[2], Kenneth R. Koedinger[2], Shelley Evenson[3], Ido Roll[2], Angela Z. Wagner[2], Meghan Naim[4], Jay Raspat[4], Daniel J. Baker[5], and Joseph E. Beck[6]

[1] Learning Sciences Research Institute, University of Nottingham, Nottingham, UK
Ryan.Baker@nottingham.ac.uk
[2] Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA
{corbett, koedinger, iroll, awagner}@cmu.edu
[3] School of Design, Carnegie Mellon University, Pittsburgh, PA, USA
evenson@andrew.cmu.edu
[4] North Hills Junior High, Pittsburgh, PA, USA
{raspatj, naimm}@nhsd.k12.pa.us
[5] Department of Pediatrics, University of Medicine and Dentistry of New Jersey, New Brunswick, NJ, USA
bakerd1@umdnj.edu
[6] Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA, USA
joseph.beck@gmail.com

Abstract. It has been found in recent years that many students who use intelligent tutoring systems game the system, attempting to succeed in the educational environment by exploiting properties of the system rather than by learning the material and trying to use that knowledge to answer correctly. In this paper, we introduce a system which gives a gaming student supplementary exercises focused on exactly the material the student bypassed by gaming, and which also expresses negative emotion to gaming students through an animated agent. Students using this system engage in less gaming, and students who receive many supplemental exercises have considerably better learning than is associated with gaming in the control condition or prior studies.

1 Introduction

In recent years, increasing attention has been paid to the subject of how students choose to use intelligent tutoring systems. Recent models have suggested that students adopt a variety of strategies for using intelligent tutoring systems and other interactive learning environments, with different strategies potentially leading to different learning outcomes [2,3,7,14]. One strategy in particular, gaming the system, has been found to be associated with poorer learning gains in intelligent tutoring systems [5,7]. We define gaming the system as attempting to succeed in an educational environment by exploiting properties of the system rather than by learning the material and trying to use that knowledge to answer correctly. Gaming has been observed in a variety of types of learning environments, from educational games [10] to online newsgroups [9], and has been repeatedly documented in intelligent tutoring systems [1,7,8,12,13].

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 392 – 401, 2006.

Within the specific intelligent tutoring system that we will discuss in this paper, gaming behavior consists of systematic guessing and rapid-fire hint requests [4].

Baker and his colleagues [4] have determined that gaming can be divided in some systems into two distinct behaviors – “harmful” gaming, which typically occurs on the problem steps the student knows least well, and is associated with poor learning outcomes, and “non-harmful” gaming, which typically occurs on problem steps the student already knows, and is not associated with poor learning outcomes.

In this paper, we present a tutor component that responds to harmful gaming, in order to improve gaming students’ learning. This tutor incorporates an animated agent, Scooter the Tutor, who observes students as they interact with the tutor, looks increasingly unhappy when students game and gives a student supplementary exercises on the exact steps of the problem-solving process that the student gamed.

2 Design

Two previous attempts to address gaming in intelligent tutoring systems took a “preventative” approach to addressing gaming, attempting to directly prevent known gaming behaviors [1,8]. Researchers at Carnegie Mellon and Carnegie Learning introduced a two-second delay between each level of a multi-level hint, to prevent a student from clicking through hints at high speed, and gave mandatory hints (“proactive help”) when a student commits more than three errors on a single step, preventing systematic guessing [1]. Researchers at the University of Massachusetts re-designed their system to not give help until a student had spent a minimum amount of time on the current problem [8].

In [7], we hypothesized that students using a system re-designed to directly prevent gaming would attempt to discover new ways to game. Shortly after, [13] found that students using a tutor with two-second help delays developed new strategies for gaming – for example, rapidly repeating the same error several times in a row in order to elicit delay-free proactive help. An additional concern with direct prevention is that students game features which are used in more positive ways by the majority of students who do not game.

Our design approach, by contrast, attempted to meet two conditions: First, the design must improve the learning of students who currently game. Second, the design must change the tutor minimally for students who do not game.

In accordance with these design goals, we developed a new component for the students’ intelligent tutoring software – an animated agent named “Scooter the Tutor”, developed using graphics from the Microsoft Office Assistant [11] but modifying those graphics to enable a wider range of emotions. Scooter was designed to both reduce the incentive to game, and to help students learn the material that they were avoiding by gaming, while affecting non-gaming students as minimally as possible.

When the student is not gaming, Scooter looks happy and occasionally gives the student positive messages (see the top-left of Figure 1). Scooter’s behavior changes when the student is detected to be gaming harmfully (using an updated version of the gaming detector presented in [4,6]). If the detector assesses that the student has been gaming harmfully, but the student has not yet obtained the answer, Scooter displays increasing levels of displeasure (culminating in the expression shown on the bottom-left of Figure 1), to signal to the student that he or she should now stop gaming, and try to get the answer in a more appropriate fashion.

If the student obtains a correct answer through gaming, Scooter gives the student a set of supplementary exercises designed to give the student another chance to cover the material that the student bypassed by gaming this step. The supplementary exercises have three levels, each multiple-choice – the student is only given one chance to answer each level. In each of the first two levels of an exercise, the student is asked to answer a question that either requires understanding one of the concepts required to answer the step the student gamed through, or a question which is about what role the step they gamed through plays in the overall problem-solving process. If the student gets both the first and second levels wrong, he or she is given a third level, which is still relevant to the step the student gamed through, but which is very easy, in order to prevent indefinite floundering.

If the student gets any level right on the first try, Scooter lets the student return to the regular tutor exercise; if the student gets all three levels (including the very easy third level) wrong, Scooter assumes that the student was trying to game him, asks the student to attempt to get his exercises correct on the first try, and marks the problem step involved to receive supplementary exercises in future problems. If the student tries to game a supplementary exercise, Scooter displays anger.

Our goal, in designing Scooter, was to benefit students in three fashions. First, by representing how much each student had been gaming, Scooter both serves as acontinual reminder that the student should not game, and lets teachers know which

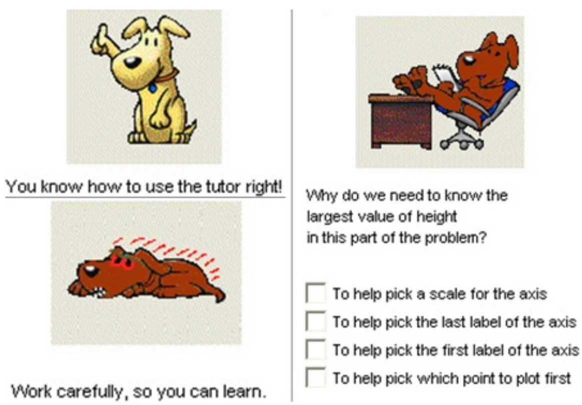


Fig. 1. Scooter the Tutor – looking happy when the student has not been gaming harmfully (top-left), giving a supplementary exercise to a gaming student (right), and looking angry when the student is believed to have been gaming heavily, or attempted to game Scooter during a supplementary exercise (bottom-left)

students were gaming recently. Second, Scooter was intended to invoke social norms in students by expressing negative emotion when students game. Scooter's display of anger is a natural social behavior in this context; if a student systematically guessed every number from 1 to 38 when working with a human tutor, it seems reasonable to expect that the human tutor would become impatient or upset. Therefore, we hypothesized that when Scooter becomes angry, he will invoke social norms, leading the student to game the system less. Third, by giving students supplemental exercises targeted to the material the student was gaming through, Scooter gives students a second chance and another way to learn material he or she may otherwise miss entirely. Additionally, supplemental exercises may change the incentive to game – whereas gaming might previously have been seen as a way to avoid work, it now leads to extra work. Thus, we predicted that Scooter would both reduce gaming and improve gaming students' learning, either by reducing their gaming or giving them a second chance to learn the material they miss by gaming.

3 Study Methods

We studied Scooter's effectiveness in the context of a year-long Cognitive Tutor curriculum for middle school mathematics, within 5 classes at 2 schools in the Pittsburgh suburbs. The study was conducted in the spring semester, after students had already used the Cognitive Tutor for several months.

Initially, the study was designed such that every student used both a version of the tutor with Scooter (experimental condition), and a version of the tutor without Scooter (control condition). Each student was randomly assigned to use one of two lessons (a lesson on percents, and a lesson on scatterplots) with Scooter, and the other lesson without Scooter. All students completed the control condition of the study first, and the experimental condition second. However, due to a scheduling error, the experimental condition of the study took place in the same week as subject material on percents was being taught in class. To avoid bias in favor of the experimental condition, we will therefore limit our discussion to data from the scatterplot lesson. 51 students participated in the experimental condition for the scatterplot lesson (12 were absent for either the pre-test or post-test, and thus their data will not be included in analyses relevant to learning gains); 51 students participated in the control condition for the scatterplot lesson (17 were absent for either the pre-test or post-test).

Before using the tutor, all students first viewed conceptual instruction, delivered via a PowerPoint presentation with voiceover and simple animations [cf. 4]. In the experimental condition, a brief description of Scooter was incorporated into the instruction. Then students completed a pre-test, used the tutor lesson for 80 minutes across multiple class periods, and completed a post-test. Test items were counterbalanced across the pre-test and post-test, and were identical to items used in past studies using this tutor lesson [4]. Log files were used to distill measures of Scooter's interactions with each student, including the frequency with which Scooter got angry, and the frequency with which Scooter gave a student supplementary exercises. In addition, observational data was collected to determine each student's frequency of gaming, using the quantitative observational method as in [7], in order to analyze Scooter's effects on gaming frequency. Another potential measure, the gaming detector [4], was not used because of risk of bias in using the same metric both to drive interventions and as a measure of the intervention's effectiveness.

4 Results

Scooter was associated with a sizeable, though only marginally significant, reduction in the frequency of observed gaming. 33% of students were seen gaming in the control condition, while 18% of students were seen gaming in the experimental condition, a marginally significant difference, $\chi^2(1,N=102)= 3.30$, $p=0.07$. However, although fewer students gamed, those students who did game did not appear to game less. The average gamer in the control condition gamed 17% of the time, while the average gamer in the experimental condition gamed 14% of the time, which was not a significant difference, $t(23)=0.74$, $p=0.47$.

Despite the apparent reduction in gaming, however, there was not an overall improvement in learning. Overall, students in the control condition averaged a 22 point pre-post gain (44%->66%), while students in the experimental condition averaged a 25 point pre-post gain (37%->62%), which was not a significant difference, $t(70)=0.34$, $p=0.73$. However, analyzing overall learning may not be the most appropriate way to test the intervention's effect on learning. Gamers are a fairly small subset of the overall population, both in this study and past studies [cf. 6,7].

Therefore, differences in gamers' learning may be swamped by normal variation in the rest of the population. Additionally, since students engaged in different degrees of gaming, and the detector was accurate but not perfect [cf.4], not all students who in engaged in harmful gaming received the same number of interventions from Scooter. Thus, in the following sections, we will look at the students who got a considerable amount of each type of intervention from Scooter, to see if and how the students' behavior and learning was affected by Scooter. We will analyze the two types of interventions separately, since the two types of interventions were given in different situations and may have had different effects.

4.1 Supplementary Exercises

Overall, Scooter gave a fairly small number of exercises: no student received a set of exercises from Scooter on more than 3.2% of problem steps (12 sets), and the median student received a set of exercises on only 1.1% of problem steps (3 sets). However,

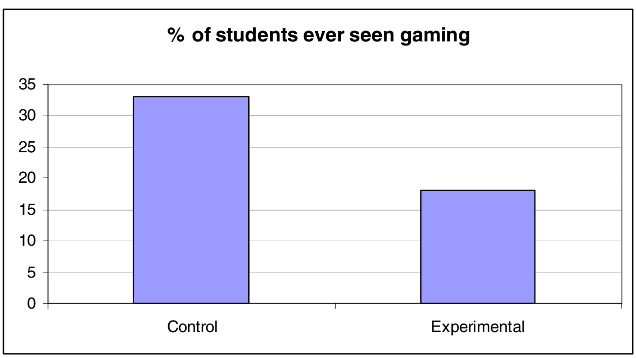


Fig. 2. The frequency of gaming (observed) in each condition

Scooter's exercises were assigned to exactly the problem steps students gamed on (according to the detector), and were significantly correlated to the frequency of observed gaming, r=0.43, F(1,38)=8.24, p<0.01, so the exercises might have had more effect on learning than their low frequency might otherwise indicate.

One possible model for how learning could relate to the number of supplementary exercises received is a linear relationship – the more supplementary exercises a student receives, the more they learn. However, students who never receive supplementary exercises don't receive supplementary exercises precisely because they don't engage in harmful gaming, and not engaging in harmful gaming is generally associated with better learning [cf. 4]. Therefore, if supplementary exercises positively affect learning, it may be more reasonable to expect students who receive either many or very few supplementary exercises to show good learning, with the students in the middle showing poorer learning.

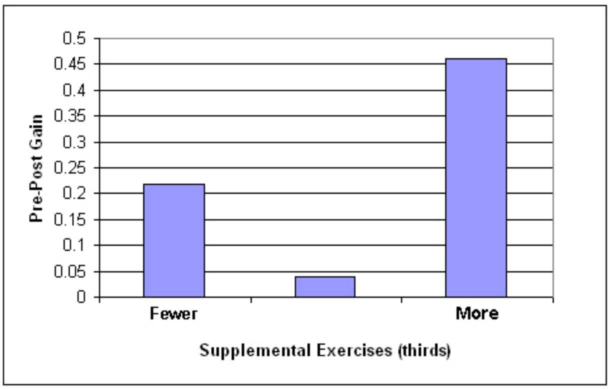


Fig. 3. The Learning Gains Associated With Receiving Different Levels of Supplemental Exercises From Scooter

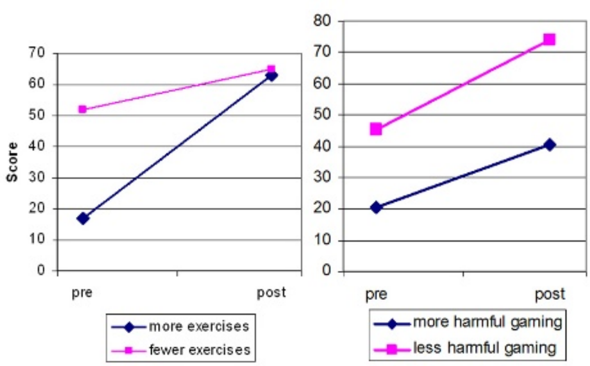


Fig. 4. Left: The Learning Gains Associated With Receiving Different Levels of Supplemental Exercises From Scooter (Top Third versus Other Two Thirds)**.** Right: The Learning Gains Associated With Different Levels of Harmful Gaming, in the Control Condition (Top Half of Harmful Gaming Versus Other Students).

In fact, this is exactly the relationship we find, as shown in Figure 3. The third of students that received the most supplementary exercises had significantly better learning than the other two thirds, t(37)=2.25, p=0.03; the overall difference between all three groups was also significant, F(2,36)=3.10, p=0.06.

This occurred because the students who received the most supplementary exercises started out behind the rest of the class (common among students who frequently game [cf. 7]), but caught up by the post-test (see Figure 4 Left). There was a statistically significant interaction between pre-test and post-test scores, and how many supplementary exercises the student received (top third versus other two thirds), F(1,37) = 5.07, p=0.03, for a repeated measures ANOVA. Note that there was not a ceiling in the mid-60s, nor a post-test floor effect: students in each group had perfect post-test scores, or low post-test scores.

In considering the evidence that students who received many supplemental exercises caught up to the rest of the class, it is worth remembering that students receive supplemental exercises because they are detected to be engaging in a large amount of harmful gaming. In both the control condition (see Figure 4 Right), and in prior studies involving the same tutor lesson [4,5], frequent harmful gaming is associated with starting out lower than the rest of the class, and falling further behind by the post-test, rather than catching up. As shown in Table 1, students in the control condition and past studies who did not use Scooter and engaged in more than the median amount of harmful gaming (among harmful gamers) averaged a 22 point learning gain, less than half of the average learning gain (46 points) of students who received many supplementary exercises, a statistically significant difference, t(47)=2.09, p=0.04.

Table 1. Learning gains for students who received large numbers of supplementary exercises from Scooter, and for students who did not use Scooter and engaged in more than the median amount of harmful gaming, among harmful gamers. All students used the same lesson on Scatterplots.

Group	Learning Gain
Experimental condition: more supplementary exercises	46 points
Control condition: more harmful gaming	20 points
2004: more harmful gaming [e.g. 5]	18 points
2003: more harmful gaming [e.g. 7]	25 points

Interestingly, although Scooter's exercises appear to be associated with improved learning, Scooter's exercises were not directly associated with the decrease in gaming reported in the previous section. If receiving an exercise from Scooter led a student to reduce his/her gaming, we would expect the students who received more exercises to reduce their gaming over time. There is no evidence of such a decrease. Figure 5 (left) shows the frequency in gaming over the 3 days of the study among the students who received many exercises (top third) in the experimental condition, compared to the other students. Among the students who received more exercises, neither the apparent increase in gaming from day 1 to day 2, nor the apparent decrease in gaming from day 2 to day 3, was statistically significant, $\chi^2(1,N=155)= 0.31$, $p=0.58$, $\chi^2(1,N=105)= 0.17$, $p=0.68$. Overall, the students who received more exercises gamed significantly more often than the students who received fewer exercises, $\chi^2(1,N=388)= 24.33$, $p<0.001$.

4.2 Expressions of Anger

Overall, Scooter became angry considerably more often than he gave supplementary exercises. The median student saw an angry Scooter 13% of the time, and the student who saw an angry Scooter the most often saw an angry Scooter 38% of the time.

There did not appear to be an association between viewing an angry Scooter more often, and better learning. Students who received more expressions of anger did not have a significantly larger average learning gain than other students, whether we compared the top quartile to the other students, $t(37)=0.48$, $p=0.63$, effect size = 0.20σ, the top third, $t(37)=0.16$, $p=0.87$, or the top half, $t(37)=0.15$, $p=0.88$.

Additionally, there was no evidence of a relationship between Scooter's frequency of expressions of anger, and a reduction in gaming over time (as shown in Figure 5, right). Among the students who saw an angry Scooter the most often (top quartile), there was not a significant change either from day 1 to day 2, or day 2 to day 3, $\chi^2(1,N=79)= 0.04$, $p=0.84$, $\chi^2(1,N=50)= 0.83$, $p=0.36$.

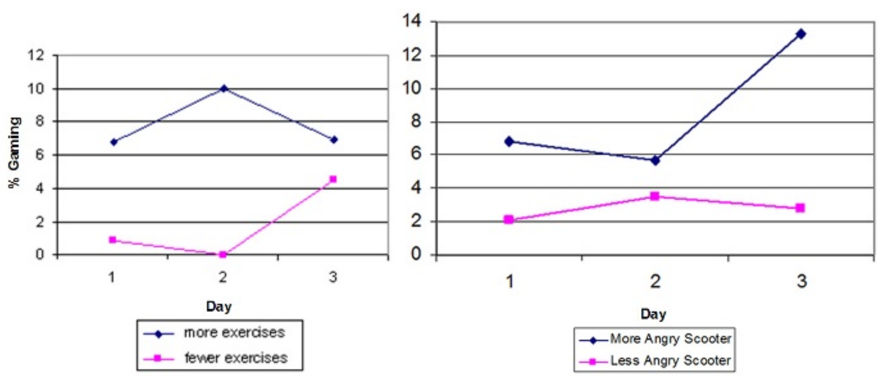


Fig. 5. Observed Gaming Over Time, in the Experimental Condition

5 Conclusions

In this paper, we present a re-designed tutor that responds to when students game the system, incorporating an animated agent, Scooter the Tutor. Students who received a large number of supplementary exercises from Scooter had high learning gains, and caught up to the rest of the class. This result is quite different from the pattern observed in the control condition and past studies [4,5], where students who game harmfully start out with lower pre-test scores, and fall further behind the rest of the class by the post-test.

Since students tend to game harmfully on the steps they know least well [4], the supplementary exercises may have been effective in large part because they offered additional learning support (and, perhaps, different learning support) for each student on the exact steps which that student found most difficult. Hence, we may be able to use a student's choice to game as an opportunity to learn more about where the student is having difficulty.

Incorporating Scooter into the tutor also led to about half as many students choosing to game. It is not entirely clear what aspect of the modified tutor led to the reduction in gaming. Neither students who saw an angry Scooter more often, nor students who received more supplementary exercises, reduced their gaming over time. One possibility is that simply knowing Scooter was present, and that he would make it impossible to hide gaming, led some students to game less. Thus, although Scooter's actions may not have directly affected the students who saw an angry Scooter, Scooter's presence may have motivated some students to avoid gaming during the entire lesson.

Overall, these results suggest that there is value to detecting and responding to differences in how students choose to use intelligent tutoring systems. By responding to gaming, we can develop tutors that help lower-performing students catch up to the rest of the class, and come closer to the goal of developing educational systems that help all students achieve.

Acknowledgements

We would like to thank Steve Ritter, Tom McGinnis, Sue Cameron, Russ Hall, Dina Crimone, Pat Battaglia, Shaaron Ainsworth, Cecily Heiner, Peter Scupelli, and Cristen Torrey for helpful suggestions and assistance. This work was funded by NSF grant REC-043779 to "IERI: Learning-Oriented Dialogs in Cognitive Tutors: Toward a Scalable Solution to Performance Orientation".

References

1. Aleven, V. (2001) Helping Students to Become Better Help Seekers: Towards Supporting Metacognition in a Cognitive Tutor. Paper presented at *German-USA Early Career Research Exchange Program: Research on Learning Technologies and Technology-Supported Education,* Tubingen, Germany.
2. Aleven, V., McLaren, B.M., Roll, I., Koedinger, K.R. (2004) Toward tutoring help seeking: Applying cognitive modeling to meta-cognitive skills. *Proceedings of the 7th International Conference on Intelligent Tutoring Systems (ITS 2004)*, 227-239.
3. Arroyo, I., Woolf, B. (2005) Inferring learning and attitudes from a Bayesian Network of log file data. *Proceedings of the 12th International Conference on Artificial Intelligence in Education,* 33-40.
4. Baker, R.S., Corbett, A.T., Koedinger, K.R. (2004) Detecting Student Misuse of Intelligent Tutoring Systems. *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, 531-540.
5. Baker, R.S., Roll, I., Corbett, A.T., Koedinger, K.R. (2005) Do Performance Goals Lead Students to Game the System? *Proceedings of the International Conference on Artificial Intelligence and Education (AIED2005)*, 57-64.

6. Baker, R.S., Corbett, A., Koedinger, K., Roll, I. (2005) *Detecting When Students Game The System, Across Tutor Subjects and Classroom Cohorts*. Proceedings of User Modeling 2005, 220-224.
7. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. (2004) Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System". *Proceedings of ACM CHI 2004: Computer-Human Interaction*, 383-390.
8. Beck, J. (2005). Engagement tracing: using response times to model student disengagement. *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, 88-95.
9. Cheng, R., Vassileva, J. (2005) Adaptive Reward Mechanism for Sustainable Online Learning Community. *Proc. of the International Conference on Artificial Intelligence in Education*, 152-159.
10. Klawe, M.M. (1998) Designing Game-based Interactive Multimedia Mathematics Learning Activities. *Proceedings of UCSMP International Conference on Mathematics Education.*
11. Microsoft Corporation. (1997) *Microsoft Office 97*. Seattle, WA: Microsoft Corporation.
12. Mostow, J., Aist, G., Beck, J., Chalasani, R., Cuneo, A., Jia, P., Kadaru, K. (2002) A La Recherche du Temps Perdu, or As Time Goes By: Where does the time go in a Reading Tutor that listens? Paper presented at *Sixth International Conference on Intelligent Tutoring Systems (ITS'2002).*
13. Murray, R.C., vanLehn, K. (2005) Effects of Dissuading Unnecessary Help Requests While Providing Proactive Help. *Proc. of the International Conference on Artificial Intelligence in Education*, 887-889.
14. Stevens, R., Soller, A., Cooper, M., & Sprang, M. (2004). Modeling the Development of Problem-Solving Skills in Chemistry with a Web-Based Tutor. *Proceedings of the 7th International Conference on Intelligent Tutoring Systems (ITS 2004)*, 580-591.

Generalizing Detection of Gaming the System Across a Tutoring Curriculum

Ryan S.J.d. Baker[1], Albert T. Corbett[2], Kenneth R. Koedinger[2], and Ido Roll[2]

[1] Learning Sciences Research Institute, University of Nottingham, Nottingham, UK
Ryan.Baker@nottingham.ac.uk
[2] Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA
{corbett, koedinger, iroll}@cmu.edu

Abstract. In recent years, a number of systems have been developed to detect differences in how students choose to use intelligent tutoring systems, and the attitudes and goals which underlie these decisions. These systems, when trained using data from human observations and questionnaires, can detect specific behaviors and attitudes with high accuracy. However, such data is time-consuming to collect, especially across an entire tutor curriculum. Therefore, to deploy a detector of behaviors or attitudes across an entire tutor curriculum, the detector must be able to transfer to a new tutor lesson without being re-trained using data from that lesson. In this paper, we present evidence that detectors of gaming the system can transfer to new lessons without re-training, and that training detectors with data from multiple lessons improves generalization, beyond just the gains from training with additional data.

1 Introduction

Developing models that can reliably detect differences in how students choose to use intelligent tutoring systems, and the attitudes and goals which underlie these decisions, has received considerable attention in recent years [1,3,4,7,8]. A number of models have been developed which can reliably detect specific student behaviors – from avoiding help [cf. 1], to gaming the system [4], to competing with other students [7]. These models have supported the development of systems that influence students to learn to use intelligent tutoring systems more effectively [2].

However, to be widely useful, detectors of student behaviors and motivation need to be generalizable. Thus far, most such detectors have been developed using data from individual lessons from a tutoring curriculum, or from fairly small-scale intelligent tutors. However, intelligent tutors are increasingly being used as major components in year-long curricula. A model of help-seeking behavior developed using only log file data has been shown to generalize effectively across lessons [11], but many of the models developed to detect student behaviors and attitudes have been trained using additional data such as human observations [4,8], improving accuracy [11]. Unfortunately, human observations are time-consuming to collect for an entire year-long curriculum. Therefore, to be maximally useful – and used – detectors of behaviors and motivation need to be able to take advantage of observational data, while generalizing to new tutor lessons without the collection of additional data.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 402 – 411, 2006.

In this paper, we will discuss our work to generalize a behavior detector which detects whether a student is "gaming the system", attempting to succeed in an educational environment by exploiting properties of the system rather than by learning the material and trying to use that knowledge to answer correctly [6]. Within the set of intelligent tutor lessons that we will discuss in this paper, gaming behavior consists of systematic guessing and rapid-fire hint requests. Prior analyses have also found that gaming can be divided into two distinct categories of behavior: harmful gaming, which is associated with poor learning outcomes and appears to occur on the problem steps the student knows least well, and non-harmful gaming, which is not associated with poor learning outcomes and appears to occur on problem steps the student already knows [4].

Additionally, we will consider the question of what data is most useful for developing generalizable detectors. A considerable amount of machine learning research treats generalizability largely as a function of the sheer amount of data trained on, and the degree to which the training technique over-fits to that data. In this paper, we examine whether additional advantage can be gained by collecting a broader, more heterogeneous data set – in specific, presenting analyses suggesting that training on data from multiple tutor lessons improves generalizability more than would occur simply from increasing the sample size.

2 Methods

2.1 Data Sources

The first gaming detector [4] was developed using data from a tutor lesson on scatterplots, drawn from a middle-school Cognitive Tutor mathematics curriculum. In order to study issues of generalizability, we collected data from three additional lessons (on geometry, percents, and probability) from the same tutoring curriculum. All data came from classes in two school districts in suburban Pittsburgh. For the scatterplot lesson, we had data from classes in 2003, 2004, and 2005. For each of the other lessons, we had data from a single year (2004 for geometry and probability, 2005 for percents). In total, we had data from 300 students (with 113 students represented in multiple lessons), with 128,887 actions across the 473 student/lesson pairs. Each student completed between 50 and 500 actions in the tutor.

Table 1. Quantity of data obtained for each tutor lesson

Lesson	Number of students	Number of actions
SCATTERPLOT	268	71,236
PERCENTS	53	16,196
GEOMETRY	111	30,696
PROBABILITY	41	10,759

For each lesson, we collected quantitative field observations (using the method in [6]), to estimate what percentage of time each student gamed the system. Pre-tests and post-tests were given for each lesson – in all cases, test items were counterbalanced across the pre-test and post-test. Data on learning gains enabled us to distinguish between harmful gaming and non-harmful gaming [cf. 4], both during training and when evaluating goodness-of-fit. In our analyses, we will refer to students who engaged in harmful gaming as “GAMED-HURT”, and students who engaged in non-harmful gaming as “GAMED-NOT-HURT”.

Finally, we obtained logs of each student’s actions within the tutor. For each student action recorded in the logs, we distilled a set of 26 features (listed in [4 and 5]) describing that action, including information about the action itself (time taken, type of interface widget) and the action’s historical context (for instance, how many errors the student had made on the same skill in past problems).

2.2 Modeling Framework

Using this combination of data, we trained a set of detectors to predict how frequently an arbitrary student gamed the system. Each detector of gaming, within our framework, is a hierarchical Latent Response Model [10] with one observable level and two hidden (“latent”) levels. In a gaming detector’s outermost/observable layer, the detector predicts how frequently each student is gaming the system, labeling these predictions $G'_0 \ldots G'_n$. These predictions can then be compared to the observed

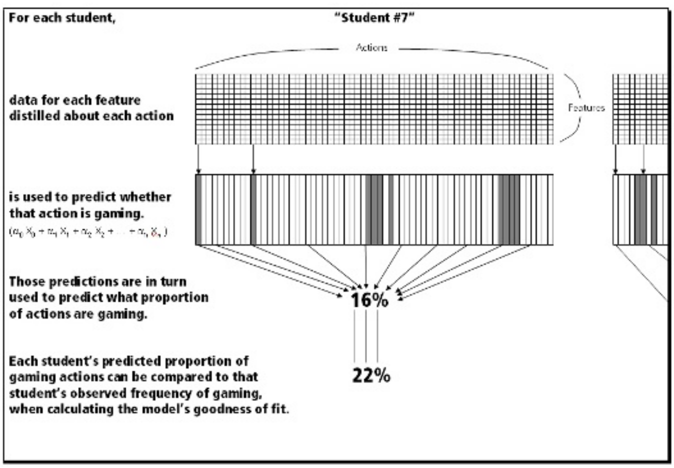


Fig. 1. The Gaming Detector

proportions of time each student spent gaming the system, $G_0 \ldots G_n$ (the metrics used will be discussed momentarily). The middle layer consists of a set of binary predictions as to whether each individual student action (denoted P'_m) is an instance of gaming. The observable predictions $G'_0 \ldots G'_n$ are derived by taking the percentage of actions which are predicted to be instances of gaming, for each student. The innermost layer is a function on features drawn from each action’s characteristics, which are used to make the binary predictions in the middle layer. Each parameter in a model of gaming is either a linear effect on a feature (a parameter value α_i multiplied by the corresponding feature value X_i – $\alpha_i X_i$), a quadratic effect (parameter value α_i multiplied by feature value X_i, squared – $\alpha_i X_i^2$), or an interaction effect on two features (parameter value α_i multiplied by feature value X_i, multiplied by feature value X_j – $\alpha_i X_i X_j$).

A prediction P_m (in the innermost layer) as to whether action m is an instance of gaming is computed as $P_m = \alpha_0 X_0 + \alpha_1 X_1 + \alpha_2 X_2 + \ldots + \alpha_n X_n$, where α_i is a parameter value and X_i is the data value for the corresponding feature, for this action, in the log files. Each prediction P_m is then thresholded using a step function to form the binary predictions that form the middle layer, such that if $P_m \leq 0.5$, $P'_m = 0$, otherwise $P'_m = 1$. This gives us a set of classifications P'_m for each action within the tutor, which are then used to create the predictions of each student's proportion of gaming, $G'_0 \ldots G'_n$ which are compared to their observed frequency of gaming.

2.3 Detector Selection

Detectors are trained as follows: First, a set of single-parameter detectors are selected (using Fast Correlation-Based Filtering [13]) such that each single-parameter gaming detector is at least 60% as good as the best single-parameter detector found (in terms of linear correlation to the observed data). If two parameters have a closer correlation than 0.7 to each other, only the better-fitting single-parameter detector is used. Then, for each single-parameter detector, we repeatedly add the parameter that most improves the linear correlation between the detector's predictions and the original data, using Iterative Gradient Descent to find the best value for each candidate parameter. Generally, when selecting detectors, we continue adding parameters until the most recent parameter worsens the model's fit under Leave-One-Out-Cross-Validation (LOOCV); however, for the analyses in this paper, we stopped when a detector had six parameters, for tractability in training a large number of detectors. Generally, the detectors had very little absolute improvement in fit after the first three or four parameters, regardless of the results of LOOCV. This process resulted in a set of detectors with comparable correlation, from which the model with the best A' [1] is selected (averaging A' across the model's ability to distinguish GAMED-HURT students from non-gamers, and the model's ability to distinguish GAMED-HURT students from GAMED-NOT-HURT students).

3 Detector Comparisons

3.1 Statistical Techniques for Detector Comparison

In the remainder of this paper, we will investigate how well gaming detectors transfer across different tutor lessons, examining detectors trained on single lessons, detectors trained on multiple lessons (but not all lessons), and a detector trained on all available lessons. Conducting these comparisons in a statistically appropriate fashion requires meta-analytic techniques, which we discuss in this section.

When comparing detectors to one another across multiple test lessons, the data from different test lessons cannot simply be collapsed into a single data set, since this

[1] A' is both the area under the ROC curve, and the probability that the detector can successfully distinguish between an arbitrary student from each of the two groups being classified.

will bias towards detectors that do best on the lesson with the most data; additionally, since gaming may occur with different frequency in different lessons, the A' value of a combined data set will be substantially lower than the A' values of the individual data sets, underestimating all detectors' effectiveness. Hence, we will in all cases determine our measures of interest for each test lesson individually, compare the detectors to each other within each test lesson, and then use meta-analytic techniques to combine these comparisons into a single statistical comparison.

In the analyses to follow, we will compare detectors to each other in terms of their A' and correlation. In order to use common meta-analytic techniques, we will convert these metrics to Z-scores. Two A' values can be compared to each other, giving a Z-score as the result, by using the standard Z-score formula in combination with Hanley and McNeil's technique for estimating the variance of an A' value [9]. Correlations can be compared to each other, giving a Z-score, by converting the correlations to Z-scores via the Fisher Zr transformation [12], and then comparing those Z-scores to one another.

Once all values are Z-scores, comparisons between results from different test lessons (for example, to estimate whether a detector performs significantly better than chance, across multiple test lessons) will be made using Stouffer's method [12] and denoted Z_s. Comparisons between results within the same test lesson (for example, to compare two detectors to each other) will be made using the mean Z-score method [12] and denoted Z_m. Comparisons of multiple detectors (such as the set of detectors trained using data from three lessons) across multiple test sets will be denoted Z_{ms}. In these cases, all within-lesson comparisons will be made before any between-lesson comparisons, in order to avoid comparing Z-scores estimated with methods which have different assumptions to each other. Z-scores derived without meta-analytic aggregations or comparisons will be denoted Z.

3.2 Transferring Models Trained on a Single Lesson

We begin our analysis by investigating how well a detector trained on a single tutor lesson will transfer to other tutor lessons. We trained four detectors – one on each of the four lessons. We then tested how well each detector detected gaming within its training lesson, and within each of the 3 other lessons.

The four detectors trained on a single lesson had an average A' of 0.86, in the training lessons, at distinguishing GAMED-HURT students from non-gamers, significantly better than chance, Z_s=10.74, $p<0.001$. The detectors were significantly worse at making this same distinction in the transfer lessons (A' =0.71), Z_{ms} =3.63, $p<0.001$, though their performance in the transfer lessons was still better than chance, Z_m = 2.12, p=0.03. The detectors had an average A' of 0.79, in the training lessons, at distinguishing GAMED-HURT students from GAMED-NOT-HURT students, significantly better than chance, Z_s =5.07, $p<0.001$. The detectors were not significantly worse at making this distinction in the transfer lessons (A' =0.74), Z_{ms} =0.56, p=0.58, and were significantly better than chance, Z_m =2.86, $p<0.01$. The detectors had an average correlation of 0.57 between the observed and predicted frequencies of harmful gaming, in the training lessons, significantly better than chance, Z_s = 12.08, $p<0.001$. The detectors were significantly worse at making this same distinction in the transfer lessons (r=0.22), Z_{ms} =5.15, $p<0.001$, though their performance in the transfer lessons was still better than chance, Z_m =2.40, p=0.02.

Hence, a detector trained on one lesson performs significantly better than chance when transferred to other lessons. However, there is a significant and substantial drop in performance from training lessons to transfer lessons, on 2 of the 3 metrics of interest. The overall pattern of results from the comparisons is shown in Table 2.

3.3 Training a Detector on All Four Lessons

One potential explanation for the relatively poor transfer of detectors trained on single lessons is that it is simply not possible to develop a single gaming detector which is highly effective at detecting harmful gaming in multiple lessons, using our techniques. To investigate this possibility, we trained a detector on all four lessons together.

The detector trained on all four lessons had an average A' of 0.85, across the four lessons, at distinguishing GAMED-HURT students from non-gaming students. This was not significantly lower than the average A' (0.86) of the models trained on single lessons, when tested on the training lessons, $Z_{ms} = 0.38$, p=0.70. The detector trained on all four lessons had an average A' of 0.80, across the four lessons, at distinguishing GAMED-HURT students from GAMED-NOT-HURT students. This was also not significantly lower than the average A' (0.79) of the models trained on single lessons, when tested on the training lessons, $Z_{ms} = 0.12$, p=0.90. Finally, the model trained on all four lessons had an average correlation of 0.60, across the four lessons, between the observed and predicted frequencies of harmful gaming, in the training lessons. This was again not significantly different than the average correlation (0.57) of the models trained on single lessons, when tested on the training lessons, $Z_{ms} = 0.53$, p=0.60.

Table 2. Detectors trained on just one of the four lessons. Italics denotes when detectors were, in aggregate, statistically significantly better than chance. Boldface denotes when detectors were significantly better for training lessons than transfer lessons.

Metric	Training lesson average	Transfer lesson average
A' (GAMED-HURT versus NON-GAMING)	***0.86***	*0.71*
A' (GAMED-HURT versus GAMED-NOT-HURT)	*0.79*	*0.74*
Correlation	***0.57***	*0.22*

Table 3. Comparing a detector trained on all four lessons to detectors trained on just one of the four lessons, within the training lessons. All detectors were statistically significantly better than chance, on each metric. There were no statistically significant differences between detectors, on any metric.

Metric	Training on one lesson	Training on all lessons
A' (GAMED-HURT versus NON-GAMING)	0.86	0.85
A' (GAMED-HURT versus GAMED-NOT-HURT)	0.79	0.80
Correlation	0.57	0.60

Hence, a model trained on all four lessons is equally as effective as four models trained on individual lessons, within the training lessons. This indicates that it is possible to develop a gaming detector which is effective in multiple lessons. The overall pattern of results from these comparisons is shown in Table 3.

3.4 Training a Detector on Three of Four Lessons

The next question to consider is whether we can develop a gaming detector which is not just effective across multiple lessons, but which can also transfer effectively to lessons it was not trained on. To this end, we trained a set of detectors on three of four of the lessons together, and then tested each of these detectors on the fourth, left-out, lesson.

We will compare these detectors' effectiveness at transferring to two other conditions. The first comparison condition is how well detectors perform when trained on a single lesson and then tested on the same lesson. We view this level of performance as a reasonable "gold standard" for how well a detector can do on any lesson. The second comparison condition is how well detectors perform when trained on a single lesson and then tested on the other lessons. Our goal is to obtain significant and substantial improvements on this level of performance.

The detectors trained on three lessons had an average A' of 0.84 at distinguishing GAMED-HURT students from non-gamers, in the training lessons, and an average A' of 0.80 at making the same distinction in the test lessons. The test set performance of detectors trained on three lessons (A'=0.80) was not significantly lower than the training set performance of detectors trained on one lesson (A'=0.86), Z_{ms} = 1.36, p=0.17. However, the test set performance of detectors trained on three lessons (A'=0.80) *was* significantly higher than the test set performance of detectors trained on one lesson (A'=0.71), Z_{ms} = 1.98, p=0.05.

The detectors trained on three lessons had an average A' of 0.78 at distinguishing GAMED-HURT students from GAMED-NOT-HURT students, in the training lessons, and an average A' of 0.80 at making the same distinction in the test lessons. The test set performance of the detectors trained on three lessons (A'=0.80) was not significantly lower than the training set performance of the detectors trained on one lesson (A'=0.79), Z_{ms} = 0.67, p=0.50.

The detectors trained on three lessons had an average correlation of 0.55 between the observed and predicted frequencies of harmful gaming, in the training lessons, and an average correlation of 0.41 in the test lessons. In this case, the test set performance of the detectors trained on three lessons (r=0.41) was marginally significantly lower than the training set performance of the detectors trained on one lesson (r=0.57), Z_{ms} = 1.74, p=0.08. However, the test set performance of detectors trained on three lessons (r=0.41) was still significantly higher than the test set performance of detectors trained on one lesson (r=0.22), Z_{ms} = 2.46, p=0.01.

Overall, detectors trained on three lessons suffered considerably less degradation in performance when transferred to new lessons than detectors trained on a single lesson. Detectors trained on a single lesson had large and significant drops on 2 of 3 metrics when transferred to new lessons; the detectors trained on three lessons had much smaller and less significant drops in performance when transferred to new lessons. The overall pattern of results is shown in Table 4.

Table 4. Comparing detectors trained on three of the four lessons to detectors trained on just one of the four lessons. All detectors were statistically significantly better than chance, on each metric. Grey boxes denote indicate when a detector was worse than the best detector for that metric (light grey=marginal significance, dark grey = significance).

Metric	Training on one lesson (training-set performance)	Training on 3 of 4 lessons (test-set performance)	Training on one lesson (test-set performance)
A' (GAMED-HURT versus NON-GAMING)	0.86	0.80	0.71
A' (GAMED-HURT versus GAMED-NOT-HURT)	0.79	0.80	0.74
Correlation	0.57	0.41	0.22

3.5 For a More Generalizable Detector, Should We Collect More Data or More Representative Data?

In the previous section, we showed that detectors trained on multiple lessons transfer better than detectors trained on a single lesson. While it is tempting to conclude that training on multiple lessons led to the better performance, it is also possible that the better performance came simply from training using more data. We developed linear regression models to distinguish between these hypotheses, predicting each detector's A' (GAMED-HURT vs non-gaming) and correlation to observed harmful gaming, within each lesson it was not trained on. These models can distinguish the relative contribution of sample size and number of lessons, because each of the four lessons had a different sample size (see Table 1). In these analyses, we define sample size as the number of observed gaming frequencies in the training set (for which there is one per student, per lesson), since this was the value correlated to during training.

A model which predicts A' using only the sample size (A' = α_0*SampleSize) achieves an r^2 of 0.02; a model which predicts A' using both the sample size and the number of lessons used in training (A' = α_0*SampleSize + α_1*Lessons) achieves an r^2 of 0.13. The model which includes the number of lessons is a significantly better predictor of A', $F(1,13)=8.61$, $p=0.01$, for an extra-sum-of-squares F-test. A model which predicts correlation to observed harmful gaming using only the sample size ($r = \alpha_0$*SampleSize) achieves an r^2 of 0.22; a model which predicts correlation to observed harmful gaming using both sample size and the number of lessons used in training ($r = \alpha_0$*SampleSize + α_1*Lessons) achieves an r^2 of 0.26. The model which includes the number of lessons is a marginally significantly better predictor of a detector's correlation to observed harmful gaming, $F(1,13)=4.19$, $p=0.06$, for an extra-sum-of-squares F-test.

These results indicate that training with more lessons improves a detector's generalizability, even when we control for the size of the training set. This pattern is consistent, whether A' or correlation is the measure of interest.

4 Discussion and Conclusions

Our results show that detectors of harmful gaming trained on single tutor lessons perform well in the lesson they were trained on, but considerably more poorly on other lessons. However, if a detector is trained using data from multiple lessons, the detector is effective both within the lessons it was trained for, and on a new lesson that it was not trained for. We have also presented analyses which suggest that the improvement in transferrability arises not just from training on more data, but from training on a broader cross-section of data.

The general implication is that, for developing detectors of complex student behaviors, it is not optimal to use data from only one segment of a larger curriculum – even if it is possible to obtain a very large amount of student data from that curricular segment. Training on just one curricular section or tutor lesson risks over-fitting to the specific features of that tutor lesson. By training on a larger cross-section of data from a curriculum, a developer can develop a behavioral detector which will generalize better to the rest of the entire curriculum.

Often, it is assumed that the best way to improve a machine-learned detector is to collect more data. We do not question that more data can lead to better detectors; however, the results of our investigation suggest that if there is a choice between collecting more data from a single tutor lesson (or curricular section) or collecting data from a variety of lessons, it is preferable to collect the broader data set.

Acknowledgements

We would like to thank Tom Mitchell, Darren Gergle, Irina Shklovski, and David Andre for helpful suggestions and assistance. This work was funded by NSF grant REC-043779 to "IERI: Learning-Oriented Dialogs in Cognitive Tutors: Toward a Scalable Solution to Performance Orientation".

References

1. Aleven, V., McLaren, B.M., Roll, I., Koedinger, K.R. (2004) Toward tutoring help seeking: Applying cognitive modeling to meta-cognitive skills. *Proceedings of the 7th International Conference on Intelligent Tutoring Systems (ITS 2004)*, 227-239.
2. Aleven, V., Roll, I.,, McLaren, B.M., Ryu, E.J., Koedinger, K.. (2005) An Architecture to Combine Meta-Cognitive and Cognitive Tutoring: Pilot Testing the Help Tutor. *Proceedings of the 12th International Conference on Artificial Intelligence in Education,* 17-24.
3. Arroyo, I., Woolf, B. (2005) Inferring learning and attitudes from a Bayesian Network of log file data. *Proceedings of the 12th International Conference on Artificial Intelligence in Education,* 33-40.
4. Baker, R.S., Corbett, A.T., Koedinger, K.R. (2004) Detecting Student Misuse of Intelligent Tutoring Systems. *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, 531-540.
5. Baker, R.S., Corbett, A., Koedinger, K., Roll, I. (2005) *Detecting When Students Game The System, Across Tutor Subjects and Classroom Cohorts*. Proceedings of User Modeling 2005, 220-224.
6. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. (2004) Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System". *Proceedings of ACM CHI 2004: Computer-Human Interaction*, 383-390.

7. Conati, C., McLaren, H. (2005) Data-Driven Refinement of a Probabilistic Model of User Affect. *Proceedings of the Tenth International Conference on User Modeling (UM2005)*, 40-49.
8. de Vicente, A., Pain, H. (2002) Informing the detection of the students' motivational state: an empirical study. *Proceedings of the Sixth International Conference on Intelligent Tutoring Systems*, 933-943.
9. Hanley, J.A., McNeil, B.J. (1982) The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology,* 143, 29-36.
10. 10.Maris, E. (1995) Psychometric Latent Response Models. *Psychometrika*, 60 (4), 523-547.
11. 11.Roll, I., Baker, R.S., Aleven, V., McLaren, B.M., Koedinger, K.R. (2005) Modeling Students' Metacognitive Errors in Two Intelligent Tutoring Systems. *Proceedings of User Modeling*, 379-388.
12. Rosenthal, R., Rosnow, R. (1991) *Essentials of Behavioral Research: Methods and Data Analysis.* Boston, MA: McGraw-Hill.
13. 13.Yu, L., Liu, H. (2003) Feature selection for high-dimensional data: a fast correlation-based filter solution. *Proceedings of the International Conference on Machine Learning*, 856-863.

Using Multiple Intelligence Informed Resources in an Adaptive System

Declan Kelly[1] and Brendan Tangney[2]

[1] National College of Ireland, Dublin, Ireland
dkelly@ncirl.ie
[2] University of Dublin, Trinity College, Ireland
tangney@tcd.ie

Abstract. Adaptive educational systems capture and represent, for each student, various characteristics such as knowledge and traits in an individual learner model. However, there are some unresolved issues in building adaptive educational systems that adapt to individual traits. For example it is not obvious what is the appropriate educational theory with which to develop instructional resources and model individual traits. This paper describes an experiment using the Multiple Intelligence (MI) based adaptive intelligent educational system, EDUCE, that explores how different categories of resources are used when the learner has complete control and when adaptive presentation strategies are employed. In particular it explores how Musical/Rhythmic traits and resources impact on performance. Results suggest that students prefer using Musical/ Rhythmic resources to other types of resources, however it is not clear how this preference can be best employed to enhance learning performance.

1 Introduction

Educational research informs us "one size does not fit all". It states that learners, reflecting individual traits, possess different learning characteristics, process and represent knowledge in different ways, prefer to use different type of resources and exhibit consistent observable patterns of behaviour [12]. Research also suggests that it is possible to diagnose a student's learning traits and that some students learn more effectively when instruction is adapted to the way they learn [13].

Within the field of technology enhanced learning, adaptive educational systems offer an advanced form of learning environment that attempts to meet the needs of different students [2]. Such systems capture and represent, for each student, various characteristics such as knowledge and traits in an individual learner model. Subsequently, using the resulting model it dynamically adapts the learning environment for each student in a manner that attempts to best support learning. Typical strategies that could be used to adapt the environment include adapting the presentation of content in order to hide information not relevant to the user's knowledge and providing navigation support using annotated links that suggest the most relevant path to follow [5].

Several adaptive educational systems that adapt to different traits have been developed [14] [7] [15] [16]. However, building adaptive educational systems that

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 412 – 421, 2006.

adapt to individual traits is not easy. Major research questions still outstanding include: what is the relevant educational theory with which to model individual traits, how are the relevant learning characteristics identified and in what way should the learning environment change for users with different learning characteristics [1]? For example it is not obvious what is the appropriate educational theory with which to develop instructional resources and model individual traits.

This paper describes an experiment that explores one of these challenges, namely what is the appropriate educational theory with which to develop instructional resources. Specifically, it describes an experiment using the Multiple Intelligence (MI) based adaptive intelligent educational system, EDUCE [8] [9] [5], that explores how different categories of resources, and in particular musical resources, are used when the learner has complete control and when adaptive presentation strategies are employed.

EDUCE uses Gardner's theory of Multiple Intelligences (MI) as the basis for dynamically modelling learning characteristics and for designing instructional material [7]. The theory of Multiple Intelligences reflects an effort to rethink the theory of measurable intelligence embodied in intelligence testing. It is also a rich concept that offers a framework and a language for developing adaptive educational systems that supports creative, multimodal teaching [11]. In the past 20 years since its inception, its use in the classroom has been significant [3] but, surprisingly, its application to online learning and adaptive educational systems is still in the early stages of research [8].

This paper describes the results of an empirical study that explores how different categories of resources are used and the impact on learning performance when the learner has complete control over the learning environment and when different adaptive matching and mismatching presentation strategies are used. In particular it explores how Musical/Rhythmic traits and resources impact on performance. Results suggest that students prefer using Musical/Rhythmic resources to other types of resources, however it is not clear how this preference can be best employed to enhance learning performance.

2 EDUCE

In EDUCE, a student model of learning characteristics is created using the MI theory. The theory identifies eight intelligences that are involved in solving problems, in producing material such as compositions, music or poetry and other educational activities. In contrast to learning styles, intelligences refer to abilities in what one can do such as execute skills or strategies, whereas styles refer to preferences in the use of abilities. Moreover, an intelligence is usually limited to a particular domain of content, such as verbal ability, whereas style cuts across domains of ability. Currently EDUCE uses the four intelligences in modelling the student:

- Logical/Mathematical intelligence (LM) - This consists of the ability to detect patterns, reason deductively and think logically.
- Verbal/Linguistic intelligence (VL) - This involves having a mastery of the language and includes the ability to manipulate language to express oneself.

- Visual/Spatial intelligence (VS) - This is the ability to manipulate and create mental images in order to solve problems.
- Musical/Rhythmic intelligence (MR) - This encompasses the capability to recognise and compose musical pitches, tones and rhythms.

The three intelligences, LM, VL and VS were chosen as they reflect the abilities that are historically designated as intelligences. The musical/rhythmic intelligence was chosen because it is not considered as an intelligence that can be used to deliver and inform the design of content yet the emotive power of music is widely acknowledged [4].

EDUCE builds a dynamic model of the student's MI profile by observing, analysing and recording the student's choice of MI differentiated material. Other information also stored in the student model includes the navigation history, the time spent on each learning unit, answers to interactive questions and feedback given by the student on navigation choices.

EDUCE holds a number of tutorials designed with help of subject matter experts. Each tutorial contains a set of content explaining a particular subject area. For the experiment described in this paper, Science is the subject matter and the content is developed for the 13-15 age group. A tutorial consists of learning units that explain a particular concept. In each unit there are four different sets of learning resources, each based predominantly on one of the intelligences. The different resources explain a topic from a different angle or display the same information in a different way.

Different instructional design strategies and techniques were used to create the content [9]. For example, verbal/linguistic content is developed using explanations, descriptions, highlighted keywords, term definitions and audio recordings. Logical/mathematical content is developed using number, pattern recognition, relationships, questioning and exploration. Visual/spatial content is developed using photographs, pictures, visual organisers and colour. Musical/rhythmic content is developed using musical metaphors, raps and rhythms.

In more detail, musical/rhythmic content can be developed using music tuning, content illustration, musical metaphor and sounds. Music tuning involves the use of background music, mood setting music, sound breaks and jingles to relax, invigorate and focus attention. Content illustration employs the use of songs, raps, chants and lyrics to convey information and content (Fig. 1). Musical metaphors convey concepts through the use of tones, notes, rhythms and clapping. Instrumental, environmental and nature sounds can be used to musically augment concept and ideas.

All resources developed were validated and identified as compatible with the principles of MI theory by expert practitioners.

Each learning unit consists of several distinct stages. The first stage aims to attract the learner's attention, the second stage provides a set of different MI resources, the third stage re-enforces the key message in the lesson and the final stage presents interactive questions on the topic. After accessing the second stage, students may repeatedly go back and use the same or different MI resource. The presentation strategy controls the movement from the first to the second stage. Different strategies guide students to resources they like to use and do not like to use. In this process, different versions of EDUCE can be used. One version of EDUCE uses the static MI profile to identify the learning preference, another version uses the dynamically generated student model. The dynamic student model is generated from a set of navigational and temporal features that act as behavioural indicators of the student's learning characteristics. EDUCE's predictive engine [8], with these features as input and the Naïve Bayes algorithm as its inference engine, dynamically detects patterns in the learning behaviour and determines the learner's preferences.

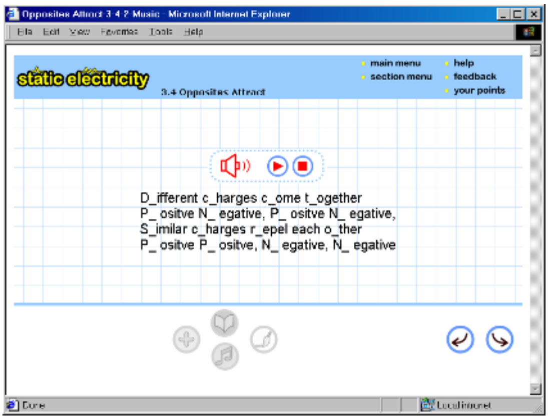


Fig. 1. Rap illustrating the use of Musical/Rhythmic Intelligence

3 Experimental Design

Using EDUCE, an experiment was designed to explore how different categories of resources are used when the learner has complete control and when adaptive presentation strategies are employed. In relation to adaptive control, the experiment also investigated the effect of matching/mismatching student preferences to learning resources. In particular the experiment explores how the use of Musical/Rhythmic resources compares to the use of other resources and how it impacts on learning performance.

In order to investigate the issues of adaptivity versus learner control and adaptive matching versus adaptive mismatching, two independent variables are defined: level of choice and presentation strategy. When looking at the definitions of these variables it is useful to remember that within each learning unit there are four multiple MI based learning resources for the student to use.

The independent variable level of *choice* provides for two different levels of choice and adaptivity. These are:

- *Free* – student has the choice to view any resource in any order. No adaptive presentation decisions are made as the learner has complete control.
- *Adaptive Dynamic* – the student is first given one resource but has the option to go back and view alternative resources. The resource first given to the student is determined by using the dynamic MI profile that is continuously updated based on the student's behaviour. The predictive engine within EDUCE identifies the most preferred and least preferred resource from the online student computer interaction.

Two different versions of EDUCE correspond to the two different levels of choice. The dynamic version can be considered as an adaptive system as the system takes the initiative in deciding which resource to present.

The independent variable *presentation strategy* encompasses two main strategies for adaptively delivering material. These strategies are:

- *Most preferred*: - showing resources the student prefers to use or matching resources with preferences
- *Least preferred*: - showing resources the student least prefers to use or mismatching resources with preferences.

The different presentation strategies are only used in the adaptive dynamic version of EDUCE. Here the dynamically generated MI profile determines which resource is shown first to the student.

The dependent variable *learning performance* is defined by the post-test score. Each student sits the post-test after the tutorial. The post-test consist of the 10 multi-choice questions, where are mostly factual. These same questions also appear throughout the tutorial.

Table 1. Variables used and their values

Variable	*Value*
Presentation Strategy	Least Preferred, Most Preferred
Choice Level	Free, Adaptive Dynamic

Students have been randomly assigned to one of the two groups defined by the levels of choice. Students assigned to the *free* group experience the same learning environment during both tutorials. Students assigned to the *adaptive dynamic* version experience both presentation strategies of least preferred and most preferred. To ensure order effects are balanced out, students are also assigned to systematically varying sequence of conditions. The design of the experiment can be described as a mixed between/within subject design with counterbalance.

For each student the experiment will consist of 4 sessions of approximately 25 minutes. the sessions are conducted over three or four days. In Session-1, students are introduced to the MI concept. In Session-2, students explore one tutorial on electricity. Before the session, the students are given a 2 minute induction on how to navigate through EDUCE. The session is followed by a post-test. Session-3 repeats the same format as Session-2, except that the student explores a different tutorial. Session-2 and Session-3 are conducted on different days. During Session-2 and Session-3, the groups using the adaptive versions receive the most preferred and least preferred presentation strategies on different days. In Session-4 students are asked to reflect on their experiences and their MI profile. This session was recorded by video camera or audio tape.

4 Results

70 boys and girls participated in the study. The ages ranged from 12 to 17, with an average age of 14. The students were randomly assigned to one of the two versions. 39 students (18 boys and 21 girls) were assigned to the free version and 31 students (15 boys and 16 girls) were assigned to the dynamic version. The students were participating in a "Discovering University" programme being run in the author's place of work. The objective of the programme was to give students the experience of third level education and to encourage them to continue education in university. The students attending this programme would primarily be from areas designated as disadvantaged in terms of the number of students who participate in third level education. The study itself was conducted in the computer laboratories in the college and took place within the 'Computer' sessions on the Discovering University programme. No reward incentives were provided to the students who participated. The following two sections examine how, for the free and adaptive group, the use of different types of resources influence learning performance. In addition, qualitative feedback from students is analysed to determine preferences for different categories of resources.

Free Group

To answer the question on how particular types of resources have greater influence on learning performance, analysis was conducted on the resources used by students in the free group. Only students in the free group were used because with the adaptive dynamic group, the adaptive presentation strategy was a factor in the choice of resources.

Table 2 displays the statistics for how much each resource category was used, aggregated over all students. As illustrated MR resources are very popular, with on average each student using 58 % of MR resources available, 24 % of VS resources, 19 % of LM resources and only 14 % of VL resources. MR resources, it appears are very attractive to students and indicates the power of music to stimulate students. However further examination is needed to determine how this preference influences learning performance.

Table 2. Resources used by students in the Free group

Resource Used	N	% Used	Std. Dev.
VL	39	14.1	16.9
LM	39	18.7	25.0
VS	39	23.9	22.4
MR	39	57.8	32.7

To analyse the influence of the most used resource (MR) and least used resource (VL), for each student the amount of each resource type used (VLUse, MRUse) was calculated by getting the average over the two tutorial sittings.

First, using the MRUse variable, students were divided up into three groups determined by how much they used the MR resource type: high, medium and low. A one-way ANOVA was conducted to explore the impact of MRUse on the average post-test score. The results were statistically significant: F (2, 36) =.4974, p=.012. Post-hoc comparisons using the Tukey HSD test indicated that the mean score for the low use MR group (M=64.6, SD=6.91) was significantly different from the medium (M=45.77, SD=19.0) and high use MR group (M=47.08, SD=20.47). The results suggest that students who did not just use the MR resource to the exclusion of all others had the greater learning performance.

A similar analysis was performed on the VLUse variable. Students were again divided into three groups determined by how much the VL resource type was used: high, medium and low. A one-way ANOVA was conducted to explore the impact of VL use on the average post-test score. The results were statistically significant: F (2, 36) =3.56, p=.039. Post-hoc comparisons using the Tukey HSD test indicated that the mean score for the high use VL group (M=63.8, SD=13.67) was significantly different from the low use VL group (M=46.92, SD=18.66). The results suggest that students who used the VL resource a lot had the greater learning performance.

Summarising the results above, it seems that for this group of students, high use of the VL resource type and low use of the MR resource type result in greater learning performance. It is significant to note the popularity of the MR resources and a promising research challenge is to identify how the motivating power of MR can be used to enhance learning performance.

Adaptive Group

The results for the free group suggest that adaptive strategies should guide students away from MR to VL and other resources. To evaluate this hypothesis, the resources used by the adaptive group are analysed. The resources used with the most and least preferred strategy are analysed separately.

First, analysis was conducted on the use of resources when the most preferred presentation strategy was used. Examining the relationships between the use of different resource categories, it was discovered that the only significant correlation was between the use of LM and MR resources [r=-.393, n=31, p=.029]. This result suggests that high use of MR resources is correlated with low use of LM resources, which also agrees with the results for the free group.

The relationship between the use of the different resources and the post-test score was next analysed. No significant correlations were found between the use of VL or MR resources and post-test scores. Indeed the only correlation that approached significance was the relationship between the use of VL resources and post-test score [r=-.343, n=31, p=.059] and in this case it was a negative correlation. This result surprisingly suggests that high use of VL resources result in a low post-test score, a direct contradiction to what was reported in the free group. One reason for this could be that VL resources were not initially presented as it was not the preferred resource for the majority of students and subsequently students did not bother to use them. No significant correlations were found between the use of resources and relative gain.

Second, analysis was conducted on the use of resources when the least preferred presentation strategy was used. Significant correlations were found between the use of VL and LM resources [r=.487, n=31, p=.005] and VL and VS resources [r=.404, n=31, p=.024]. This suggests that high use of VL resources is correlated with high use of LM and VS resources, which supports the results for the free group.

When examining the relationship between the use of resources and post-test scores, no significant correlations were found. However, the positive correlations between the use of VL, LM or VS resources and post-test scores approached significance: for VL [r=.318, n=31, p=.082], for LM [r=.32, n=31, p=.08] and for VS [r=.348, n=31, p=.055]. The correlation between use of MR resources and post-test score was very weak [r=.002, n=31, p=.992]. The results suggest that high use of VL, LM or VS resources are related to high post-test scores. No significant correlations were found between use of resources and relative gain.

Table 3. Correlations for least and most preferred strategies

	Significant Correlations	Amount
Most Preferred Strategy	Use of LM and MR	r=-.393, n=31, p=.029
	Use of VL and Post-Test	r=-.343, n=31, p=.059
Least Preferred Strategy	Use of VL and LM	r=.487, n=31, p=.005
	Use of VL and VS	r=.404, n=31, p=.024
	Use of VL and Post-Test	r=.318, n=31, p=.082
	Use of LM and Post-Test	r=.32, n=31, p=.08
	Use of VS and Post-test	r=.348, n=31, p=.055

The results are summarised together in Table 3. With the least preferred strategy, high use of VL, LM and VS resources is related to high post-test scores. With the most preferred strategy, low use of VL resources is related to high post-test scores. With the least preferred strategy, the use of VL, LM and VS resources are related to each other and with the most preferred strategy, high use of MR resources is correlated with low use of LM resources.

Returning to the original hypothesis that the best adaptive presentation strategy is to guide students away from MR to VL and other resources, the results from the least preferred sitting are in agreement. These results suggest that the use of VL, LM and VS resources can result in higher learning performance. In contrast, it was found that with the most preferred strategy low use of VL resources is correlated with high post-test scores. It appears that students, when given options, did not choose the VL resource type and were able to learn from other resources. Concerning the use of MR resources, nothing definitive can be said as no significant correlations with post-test score were discovered.

Examining the results for the free and adaptive group together, there are indications that students who prefer to work with VL resources achieve higher post-test scores (except for the adaptive group with the most preferred strategy) but this could be related to the verbal mode of assessment based on written multi-choice questions. No indications could be found about how the use of different resources is related to relative gain. It is clear that MR resources are extremely popular. MR resources seem to captivate students, maybe because of the novelty effect or because music conveys an emotional power that normal text does not. Further research is required to understand how the power of music can be tapped into for education purposes and how music can be best employed to enhance learning performance.

5 Conclusions

This paper has presented an experiment conducted primarily to determine if the use of particular categories of MI resources, and in particular Musical/Rhythmic, influenced learning performance. Hence, an experiment was designed to compare performance and the use of resources between students who have complete learner control over the learning environment and students who use an adaptive system that matches and mismatches resources with preferences.

The results for the free learner control group reveal that MR resources are extremely popular with VL resources being the least popular. However on conducting analysis, it was found that high use of the VL resource type and low use of the MR resource type resulted in greater learning performance. In addition no relationship was discovered between the use of resources and the relative gain.

These results suggest that adaptive strategies should guide students away from MR to VL and other resources. Analysis of the adaptive dynamic group did indicate that the use of VL, LM and VS resources could result in higher post-test scores. However, concerning the use of MR resources, no significant correlations with post-test score were discovered and no conclusions could be drawn.

Taken together, the results suggests that students do have different strengths and preferences and the challenge is to find out best to adapt to this diversity. It suggests that a wide approach to learning is necessary so that all students can find something attractive and beneficial. In particular, a promising research challenge is to identify how the motivating power of music can be used to enhance learning performance.

References

1. Brusilovsky, P. (2001). Adaptive Hypermedia. User Modeling and User-Adapted Instruction, 11(1-2), 87-110.
2. Brusilovsky, P., & Peylo, C. (2003). Adaptive and Intelligent Web-based Educational Systems. International Journal of Arificial Intelligence in Education, 13(2-4), 159-172.
3. Cambpell, L. & Campbell, B. (2000): Multiple Intelligences and student achievement: Success stories from six schools, Association for Supervision and Curriculum Development.
4. Carroll, K., &. (1999). Sing a Song of Science: Zephyr Press.
5. De Bra, P. (2002). Adaptive Educational Hypermedia on the Web. Communications of the ACM, 45(5), 60-61.
6. Gardner H. (1983) Frames of Mind: The theory of multiple intelligences. New York. Basic Books.
7. Gilbert, J. E., & Han, C. Y. (1999). Adapting Instruction in search of ‘a significant difference’. Journal of Network and Computer Applications, 22(3), 149-160.

8. Kelly, D. (2003). A Framework for using Multiple Intelligences in an ITS. Proceedings of EDMedia'03, World Conference on Educational Multimedia, Hypermedia & Telecommunications, Honolulu, HI.
9. Kelly, D., & Tangney, B. (2004). Predicting Learning Characteristics in a Multiple Intelligence based Tutoring System. Paper presented at the Seventh International Conference on Intelligent Tutoring Systems, ITS'04, Maceio, Brazil, 679-688.
10. Kelly, D., & Tangney, B. (2005). Matching and Mismatching Learning Characteristics with Multiple Intelligence Based Content. Paper presented at the Twelveth International Conference on Artificial Intelligence in Education, AIED'05, Amsterdam, Netherlands, 354-361.Reigeluth, C.M. (1996): A new paradigm of ISD ? Educational Technology, 36(3)
11. Lazaer, D. (1999): Eight Ways of Teaching: The Artistry of Teaching with Multiple Intelligences, SkyLight.
12. Riding, R. & Rayner. S, (1997): Cognitive Styles and learning strategies. David Fulton. .
13. Rasmussen, K. L. (1998): Hypermedia and learning styles: Can performance be influenced? Journal of Multimedia and Hypermedia, 7(4).
14. Specht, M. and Oppermann, R. (1998): ACE: Adaptive CourseWare Environment, New Review of HyperMedia & MultiMedia 4,
15. Stern, M & Woolf. B. (2000): Adaptive Content in an Online lecture system. In: Proceedings of the First Adpative Hypermedia Conference, AH2000.
16. Papanikolaou, K. A., Grigoriadou, M., Kornilakis, H., & Magoula, G. D. (2003). Personalising the inter-action in a Web-based educational hypermedia system: the case of INSPIRE. User-Modeling and User-Adapted Interaction, 13(3), 213-267.

20000 Inspections of a Domain-Independent Open Learner Model with Individual and Comparison Views

Susan Bull and Andrew Mabbott

Electronic, Electrical and Computer Engineering, University of Birmingham,
Edgbaston, Birmingham, B15 2TT, U.K.
{s.bull, axm891}@bham.ac.uk

Abstract. This paper introduces a domain-independent open learner model with multiple simple views on individual learner model data. Learners can also compare their knowledge level to their peer group, and to instructor expectations for different stages of their course. The aim is to help learners identify their knowledge, difficulties and misconceptions; prompt reflection on their knowledge and learning; and facilitate planning. We present a study of OLMlets in 4 university courses, with 114 users making over 20000 learner model inspections.

1 Introduction

There is an increasing trend towards opening the learner model of intelligent tutoring systems (ITS) to the learner that the model represents. The views of such 'open learner models' (OLM) may be simple overviews of knowledge level, or more detailed representations of knowledge, concepts, interrelationships between concepts, misconceptions, etc. In this paper we are concerned with students' use of simple learner model presentations that can be easily deployed into a range of courses.

Most simple open learner models are in the form of a skill meter, e.g. [1,2,3,4,5]. However, as yet there has been no investigation into whether learners find this the most useful presentation of simple-format learner model information. Students have clear individual preferences when multiple detailed views are available, but with none of the views standing out as most useful for most students, or generally less useful [6]. Given that skill meters are now becoming more common, it is important to investigate the use of multiple views of simple learner model presentations, to find out whether differences in learner preferences also exist with a simple open learner model, and hence whether the more widespread use of skill meters over other formats, is justified.

This paper introduces OLMlets (small OLMs - as in piglets or rootlets). OLMlets can be used in a range of courses for which multiple choice questions are appropriate. It is necessarily simple in order to encourage instructors to input the multiple choice questions required, and deploy the system in their courses. As a simple OLM, it is not currently part of an ITS, although the OLMlets approach could be harnessed for use in full systems. The aim of using OLMlets independently of an ITS is to prompt students to reflect on their knowledge (including lack of knowledge and misconceptions), facilitate planning of future learning episodes, and encourage learners to take greater responsibility for their learning. It has also been suggested that learners might like to

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 422 – 432, 2006.

compare their knowledge to that of their peers, or to the expectations of the instructor for the current stage of their course [7]. OLMlets also has such comparison views.

This paper presents a study of the OLMlets logs with 114 users in 4 university courses. 13113 questions were answered; 17000 (exactly) inspections of the individual views of knowledge level were made; 520 additional inspections of misconception descriptions; 1637 inspections of the peer comparison view; and 1296 inspections of the instructor expectations view - a total of over 20000 model inspections.

2 A Domain-Independent Simple Open Learner Model

OLMlets has an interface through which instructors can enter multiple choice questions and responses; indicate which answer is correct; define misconceptions and assign these to incorrect answers; and indicate which responses are incorrect (but where no associated misconceptions are identified). Images, superscript/subscript and limited special characters (e.g. Ω,Π) may be used. Thus OLMlets is suitable for many courses for which multiple choice questions are suitable. Instructors may also define expected knowledge levels for each topic, at different stages of the course.

Based on a student's answers to the questions, OLMlets constructs a model of the learner's knowledge level of each topic - represented by a number between 0 and 1, with 1 representing probable mastery and 0 indicating no knowledge. Misconceptions are identified by comparing user input to a misconceptions library for a course (created when an instructor defines misconceptions). The probability of a misconception being held is also represented by a number between 0 and 1, with 1 indicating high probability that the learner holds the misconception. The last five attempts at questions for each topic contribute to the learner model of an individual, with successively heavier weighting on the most recent of those five attempts. An 'unsure' option is automatically included with the response choices created by an instructor, in order that a learner is not forced to guess if they do not know the answer, thus avoiding knowledge or misconceptions being represented in the model based on guessing.

Because OLMlets is domain-independent, relying on instructors to input questions and building the learner model as defined according to instructor input, the learner modelling is not complex. There is no domain model for the differing content of the various courses. The views of the learner model for an individual are therefore simple. There are five views, as shown in Figure 1: skill meter (the most common form of simple OLM); graph (the bars from the skill meter located over, or to one side of a 'neutral' axis to help visualisation of positive and negative data); boxes (coloured to indicate knowledge level); table (knowledge level ranked according to proficiency); text (a summary of knowledge in the order topics were sequenced by the instructor).

The large image shows the skill meter view in the full window. The smaller images show the other forms for comparison. The learner may change view by clicking on the links at the top of the page. Clicking on the misconceptions link next to a topic results in a textual description of the misconception being displayed at that location. Clicking on the compare link provides a comparison of the user's own knowledge for a topic, to that of the other users in their course, as shown in Figure 2. The star indicates the learner's own knowledge level for the topic, for ease of comparison against the rest of the group. Clicking on the numbers below the view links in Figure 1 (for week, day or lecture number - method selected by the instructor), displays the learner's knowledge level against the expectations for that stage of the course, as shown in Figure 3 for the skill meters and text views. Thus the learner can compare their current knowledge to the current expectations, and to expectations for previous or future stages of the course. This is useful, for example, if they are behind: the student may wish to try to 'catch up', and viewing expectations for different points of a course may help them to set realistic goals not only according to the current expectations, but also taking into account their position with respect to the expectations at different stages of learning.

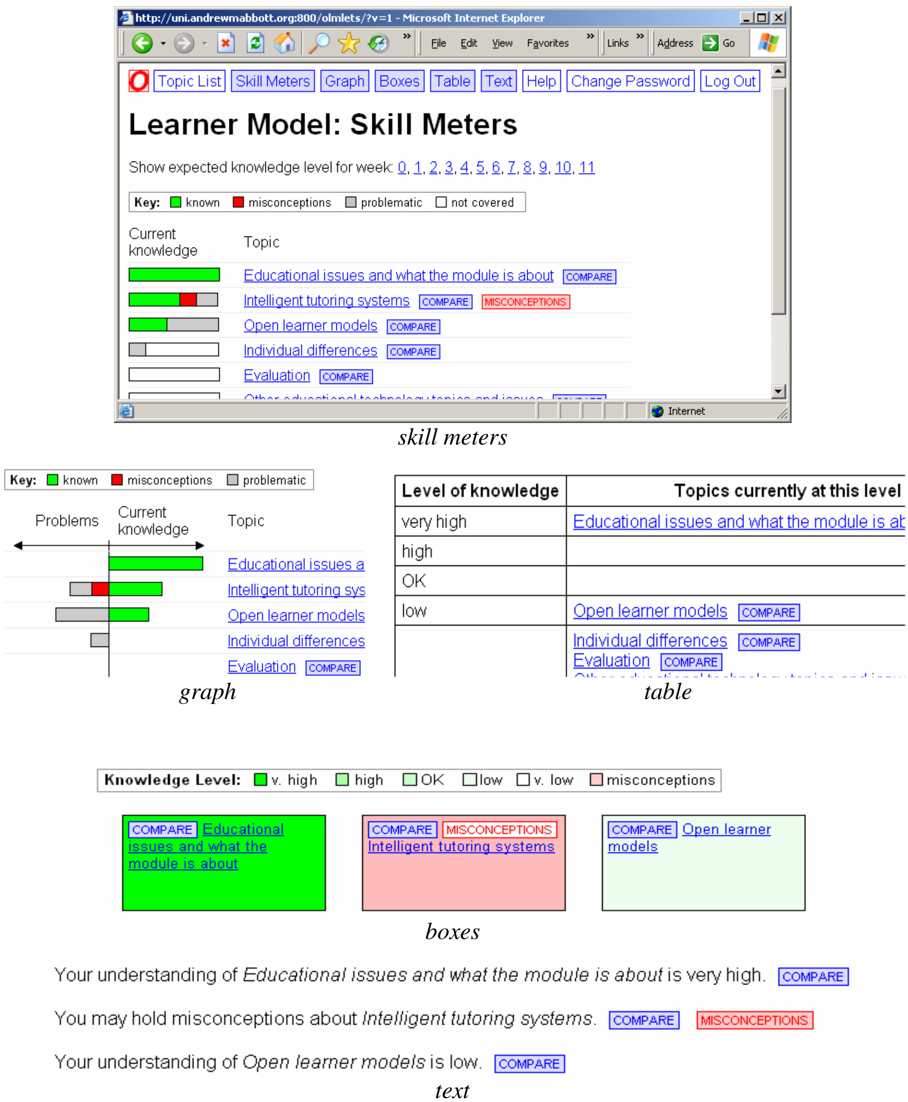


Fig. 1. The individual learner model views

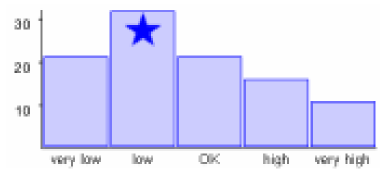


Fig. 2. Comparison to peers

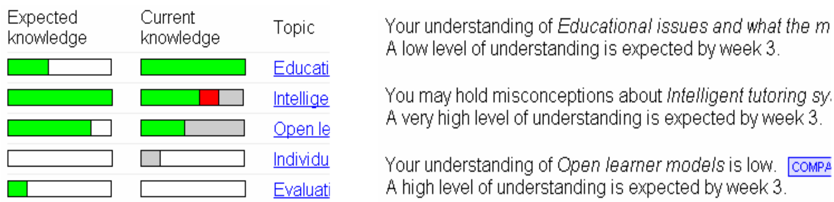


Fig. 3. Comparison to instructor expectations (skill meters and text views)

3 Use of the Individual and Comparison Leaner Model Views

While OLMlets could be incorporated into an ITS, it is currently deployed independently of a full system, alongside a range of courses. We present an overview of use in the first 4 courses in which OLMlets was deployed - the courses in which the teaching (but not necessarily the assessment) was completed at the time of the study.

3.1 Participants, Materials and Methods

Participants were 114 students in Electronic, Electrical and Computer Engineering at the University of Birmingham, using OLMlets in one of 4 courses as in Table 1.

Table 1. Courses in which OLMlets was used

Year	*Course*	*Taking Course*	*Used OLMlets*
1	Circuit Analysis	70	28 (40%)
1	Communication, IT & Lab Skills	108	47 (44%)
3	Interactive Learning Environments	29	29 (100%)
4/MSc/MRes	User Modelling	10	10 (100%)

Electronic, Electrical and Computer Engineering has 3 year BEng, 4 year MEng, and MSc/MRes degrees in subjects such as Electronic and Electrical Engineering, Communications Systems Engineering, Computer Systems Engineering, Computer Interactive Systems; and combinations with Computer Science, Languages, and Business Management. The 1st year Communication, IT & Lab Skills had students from the range of degrees as it is compulsory for all. The 1st year Circuit Analysis had students from the more traditional engineering degrees. The 3rd year Interactive Learning Environments had mainly students from the Computer Interactive Systems degree (which has a stronger focus on human factors and psychology in computing), but included some taking the Computer Systems Engineering degree, who chose it optionally. Students taking 4th year/MSc/MRes User Modelling were from a range of degrees.

The two 1st year courses introduced the OLMlets URL in a lecture, but had no further support. Those participating were therefore self-selecting. However, their data is still valid as we do not wish to argue that OLMlets should necessarily be used by everyone, if they have alternative successful learning strategies. Furthermore, the level of uptake (40% & 44%) is not especially low when considering that use was optional, and students had to familiarise themselves with OLMlets without support. There was no upcoming assessment for Circuit Analysis at the time of the study, thus students had no immediate assessment goal. The Communication, IT & Lab Skills course introduced OLMlets shortly before a multiple choice test was administered, the question types in OLMlets and the test being similar. The test contributed 25% to the final course mark. There may have been students who did not know about OLMlets in that course, as on the day that the URL was given out, there were only about 60 in attendance. Thus the percentage may be underestimating uptake from amongst those who knew about OLMlets. The 3rd year Interactive Learning Environments course introduced OLMlets in a lab at the start of the course, several weeks before the learner model of each student was assessed, the learner model counting for 10% of the course mark. The 4th year/MSc/MRes User Modelling also introduced OLMlets in a lab, a month before assessed reports were due, which comprised 100% of the assessment. The form of assessment (written report) was quite different from the OLMlets questions, though the questions addressed issues relating to the report requirements. Because these two courses introduced OLMlets in a lab, there was 100% uptake.

All groups had access to the five individual learner model views, and the group comparison - automatically generated based on all models. Expected knowledge was set in all courses. Misconceptions libraries were identified for 1st year Circuit Analysis and 3rd year Interactive Learning Environments. Students were not advised how to use OLMlets. Only in the 3rd year course did use contribute directly to assessment. Questionnaires were completed by the 3rd years at the end of the course. The analysis below is taken mostly from the system logs for all courses, but with some reference to the questionnaires. Questions required answers on a 5 point scale (strongly agree, agree, neutral, disagree, strongly disagree); and solicited open-ended comments.

3.2 Results

13113 questions were answered. 20523 learner model inspections were made. 1241 questions were answered in 1st year Circuit Analysis (average 44); 2051 in 1st year Communication, IT & Lab Skills (average 44); 9340 in 3rd year Interactive Learning Environments (average 322); 481 in 4th year/MSc/MRes User Modelling (average 48). Table 2 gives the breakdown for viewing the individual model views, which comprised 17000 of the model inspections. The final columns show the mean, median and range of inspections of individual views. The skill meter was the most frequently used by all groups. Apart from 3rd year Interactive Learning Environments, there was also usage, albeit at a lower level, of the graph view. There was occasional use of the other views (though these need to be accessed at least once before users can decide whether they are helpful). In the courses where the learner models were not assessed, the mean viewings per student was similar, at 31, 33 & 37, though the ranges were more varied. The median number of viewings was 16-28. The generally higher number of model inspections was in the more senior course, which had students with an interest in user modelling. In the course in which the learner models were assessed, the mean number of model inspections was much higher, at 491. Here the number of viewings ranged from 16 to 1112. However, only 3 learners made fewer than 100 inspections of their model, and only 5 made fewer than 200. The median was high, at 517. In this course more viewings were made of the learner model, than questions answered.

Table 2. Use of the individual model views

Course	*Skill M*	*Graph*	*Boxes*	*Table*	*Text*	*Mean: all views*	*Median: all views*	*Range: all views*
1 Circuit	534 *59%*	142 *16%*	92 *10%*	81 *9%*	62 *7%*	33	16	0 - 273
1 IT/Lab	997 *68%*	197 *13%*	90 *6%*	98 *7%*	95 *6%*	31	24	0 - 143
3 ILE	12110 *85%*	694 *5%*	590 *4%*	485 *3%*	357 *3%*	491	517	16 - 1112
4 UM	208 *55%*	63 *17%*	40 *11%*	37 *10%*	28 *7%*	37	28	14 - 82

45 of the 114 students used 1 view mainly, 44 of whom using the skill meter. 1 used mainly the boxes (usage defined by a view being selected at least 10% of times the individual model was accessed). 16 used 2 views, the most common combination being skill meter and graph, with 11; then other combinations, 2 of which did not include the skill meter, and 3 of which did not include the graph. All combinations included at least one of these views. 13 students used 3 views, each combination including the skill meter, with the second most frequent being the graph (9). 20 students used 4 views, with 2 not using the skill meter and 3 not using the graph; and 13 used all 5 views. Only 15 users did not use the skill meter as their most frequent view, and 72 were using the skill meter for over 50% of viewings of their individual model. Only the graph and boxes also had users accessing them at least 50% of the time, with 4 and 3 users respectively. 7 students did not access their individual learner model.

The most common strategy was to observe the model update after each question had been answered, with 5056 such occurrences, representing 30% of model viewings. 14 (12%) of the 114 users usually checked their model after each question, 11 of these in the 3rd year course in which the learner model was assessed. These users had attempted varying numbers of questions, the highest being a learner who attempted 626, where the model was checked after a single question had been answered on 449 occasions. 11 (10%) were checking their model on average after 10 or more questions, with the highest average being 16 (3 of these were from the 3rd year course). The remainder checked their model after answering 2-9 questions. The average for 1st year Circuit Analysis was 6.6; 1st year Communication, IT & Lab Skills, 4.5; 3rd year Interactive Learning Environments, 3.2; 4th year/MSc/MRes User Modelling, 7.8.

Misconception descriptions can be accessed in all views. Once opened, they remain visible until the user hides them. Thus there may be more occasions on which learners paid attention to misconceptions, than shown in the logs. Table 3 shows the number of times students opened a misconception in the courses in which misconceptions were modelled, and the mean, median and range. This includes only students who had misconceptions. Viewing misconceptions was higher in the 3rd year course. Only 1 student had no misconceptions. There was no clustering of viewing, with a spread over the range of 3-54 viewings - but with fewer at the higher end. In the 1st year course, 12 users had no misconceptions. Of the remainder, half viewed misconception descriptions once or twice, with most of the rest viewing fewer than 8 times.

Table 3. Inspection of misconception descriptions

Course	*Misconceptions*	*Mean*	*Median*	*Range*
1 Circuit Analysis	79	5	8.5	1 - 27
3 ILE	441	16	12.5	3 - 54

Table 4. Use of the peer comparison view

Course	*Peer View*	*Mean*	*Median*	*Range*
1 Circuit Analysis	217	8	5	0 - 68
1 IT/Lab Skills	326	7	5	0 - 34
3 ILE	1011	35	25	2 - 144
4 User Modelling	83	8	4	2 - 40

Table 5. Use of the instructor expectations comparison view

Course	*Expect. View*	*Mean*	*Median*	*Range*
1 Circuit Analysis	169	6	2	0 - 22
1 IT/Lab Skills	178	4	0	0 - 24
3 ILE	841	29	24	1 - 63
4 User Modelling	108	11	13	1 - 23

Table 4 shows use of the peer comparison view. There were 1637 inspections of the peer comparison. Some students were not interested in comparing their knowledge to that of others, with 19 of the 114 students not accessing this view. However, most accessed it several times - with means of 7 and 8; and medians of 4 and 5 for the courses in which the learner model was not assessed. In the course in which the learner model was assessed, students compared their knowledge to others' more often.

Table 5 gives the usage of the instructor expectations view. As with the misconceptions, the expectations comparison remains visible until the learner closes it. Therefore the logs may underestimate the attention users paid to this information. 1296 inspections were made. Most of these were in the 3rd year course where the learner model was assessed, although it was only the final model that contributed to the mark (i.e. assessment was *not* made at various points in the term to coincide with what students were expected to know). In the 1st year courses, inspections of the comparison to expectations were made occasionally, though 38 students were not interested in this information. In the Communication, IT & Lab Skills course, OLMlets was not introduced until shortly before the end of the teaching period (and so students had little opportunity to make use of this view). In the 4th year/MSc/MRes course, interest was higher, with 6 students making 12 or more inspections, but 4 making 5 or fewer.

The 3rd year group filled in a questionnaire at the end of the course. This was the only group that had completed all teaching and assessment at the time of the study. 23 questionnaires were returned, a 79% return rate. This paper has focussed on the logs rather than questionnaires. However, to demonstrate students' perceptions of the utility of OLMlets, and differences in preferences between individuals, we provide the results of 3 of the questions and excerpts from students' open-ended comments:

1. 22 students agreed or strongly agreed that using one or more of the individual views was useful in helping them identify their knowledge/difficulties; 1 user gave a neutral response.
2. 18 learners agreed or strongly agreed that the comparison to instructor expectations was useful; 5 gave a neutral response.
3. 20 students agreed or strongly agreed that the peer comparison view was useful; 2 gave a neutral response; and 1 disagreed.

- The graph view was less useful. The reason was because by having the grey and red areas separate it was difficult to see how much of the topic I had good knowledge of. In the skill meter view I could tell more easily as the 3 areas were all part of the same rectangle.
- The graph view is very much similar to the skill meter. They only differ as the graph has an axis. I found this view easier to interpret as there are results on both sides of the y axis.
- The only problem with the skill meters was that it was difficult to know what knowledge level I was in. This meant having to use other views such as the table for this information.
- I only used the table view towards the end when trying to get all my topics into very high.
- The boxes view was useful to me, as it gave me a graphical view of my progress as well as a textual description. This gave me much more information than the graph view on its own and a clearer description of how I was progressing than the skill meter. However, I did not find the boxes method useful for comparing my progress to what the lecturer wanted.
- [The boxes view] provides quite an easy way to compare current knowledge level to the expected ones, because it only contains a single colour tone rather than having to compare the proportion of good knowledge.
- The ability of checking my own progress against the rest of the students gave me a great deal of motivation, as I could actively view my rise in learning. Being quite competitive in nature this aid gave me a greater incentive to achieve.
- I found it reassuring to compare my results and see all students were struggling on a topic, also on the flip side it made me work harder on subjects in which I was below the average.

- Although I did prefer the skill meter view, the other views being available when I wanted them was of great use, as when I wanted to show my peers my model they could choose their favourite view to aid their understanding of my model.
- I did not think comparing my knowledge against what the lecturer expected me to know and to the knowledge of the rest of the group was helpful. When my knowledge of a topic was low, I felt low in confidence.

3.3 Discussion

Over 13000 questions were answered and over 20000 learner model inspections were made. It is perhaps not surprising that students viewed their learner model frequently in OLMlets, as it was the only feedback available. However, it was not expected that learners would make more model inspections than they answered questions, as occurred in the 3rd year course. In that course students were sensitive to the contents of their learner model, as it was assessed. Nevertheless, even in the other courses students were keen to see their learner model updating after only a few questions, as illustrated by the logs showing that only 11 of the 114 users answered 10 or more questions on average, before viewing their model. Most were viewing it more frequently, with the highest frequency (apart from the 3rd year course) being the course in which there was an upcoming test. Not surprisingly students felt motivated to perform self-evaluation that could lead to a higher result in the test. As stated above, learner model inspection was likely to be because students would otherwise have had no indication of their progress. The key question is, then, was inspection of the learner model made only because students would otherwise not receive feedback - i.e. is it simply that *some* information is better than *no* information? The questionnaire responses from the 3rd years suggest that students were indeed finding the learner model useful - both for viewing one or more of the individual views, and the comparisons. While in that course students were required to use OLMlets, use in the other courses - particularly the two 1st year courses where the URL was simply given out in a lecture, and the students had no particular interest in user/learner modelling - suggest that OLMlets was considered a helpful support alongside a lecture course. It would be interesting to see how use would compare if OLMlets was incorporated into an ITS.

Of the individual model views, the skill meter was the most commonly used, with most having it as their preferred view. Although the skill meter is the first in the series of links, and so more likely to be selected first, there was sufficient use of OLMlets that students tried all views. None is presented as a default, and so users have to make a choice about how to access their model. Skill meter usage provides hitherto lacking justification for the incorporation of skill meters ITSs or hypermedia systems that use this simple OLM format. Nevertheless, the graph was also used in combination with other views by 52 students, and 58 of those who used the skill meter were also using other views. Not all users had the skill meter as their first preference. Furthermore, questionnaire comments revealed that some students used different views for different purposes (e.g. to focus on their own knowledge and for comparison to the expectations; for a quick overview and to gauge knowledge level more precisely later). It is therefore suggested that, while it seems skill meters are a good choice for a simple OLM, additional views could be considered when designing the views of the OLM.

For the 2 courses in which misconceptions were defined, many students did view the misconception descriptions. In the 3rd year course where the learner model was assessed, the misconception inspections were higher as students had extrinsic motivation to ensure that their learner model was as good as they could get it. In 1st year Circuit Analysis, the mean and median viewings of misconceptions were 5 and 8.5 respectively (range 1-27). It should be noted that misconceptions remain visible once opened, until the learner hides them. Thus the actual attention paid to misconceptions may be higher than the logs suggest. Although not necessarily useful to all learners, it seems likely that there are sufficient who will use this information if it is available.

While some learners did not make use of the peer comparison view, most made several comparisons. Higher use was in the course that assessed the learner model. It is not clear whether this is higher simply because students used OLMlets more frequently, and so made the comparison more frequently, or whether they were really concerned about obtaining an equal or higher mark than the rest of the group. It would be interesting to pursue this question further. Several of the open-ended questionnaire comments certainly suggested that some students were motivated by being able to see their position in the group, and the fixed-response questions also suggested that students generally found this comparison useful. However, there was one student who found it demotivating to see that their knowledge was lower than that of other people.

The lecturer expectations view was used by many, most heavily in the course assessing the learner model. Nevertheless, there were students in the other courses using it. This may be because students find it useful to be able to make continued reference to their developing knowledge with regard to a target such as an upcoming test, as this provides external judgement rather than their own possibly inaccurate self-evaluation.

Where the learner model was assessed, there was higher use of the system and viewing of the model. More interesting is that students in other courses used OLMlets despite it not being directly assessed. Of course, the similarity of questions to the upcoming assessment in one of the 1st year courses no doubt contributed to students' motivation to use it. Nevertheless, it is unlikely that they would have continued if they had perceived no benefit. It appears, then, that a simple OLM can be a useful resource alongside a lecture course, even when not part of an ITS. However, our students are computer-literate, so the generality of this finding needs to be investigated in other course types, before recommendation for wider use of OLMlets can be made.

4 Summary

This paper has introduced OLMlets, a simple domain-independent open learner model with multiple individual and comparison views. Use of OLMlets in 4 university courses was described. Unlike with more detailed learner model information, where it has been found that students have preferences for a range of views [6], with this simple OLM the skill meter view was the most commonly used, with the graph in second place. It appears less critical, then, to provide a broad range of views if a simple format is used, and the growing use of skill meters in ITSs and educational hypermedia seems justified. Nevertheless, some users did use other views in preference to, or in combination with skill meters. Thus provision of a choice of view is still worthwhile. It was also found that students viewed a comparison of their own knowledge level to their peers; and instructor expectations of their knowledge level at different stages of the course. These features could therefore be usefully incorporated into systems.

References

1. Bull, S. & McEvoy, A. (2003). An Intelligent Learning Environment with an Open Learner Model for the Desktop PC and Pocket PC, *Artificial Intelligence in Education (Proceedings)*, IOS Press, Amsterdam, 389-391.
2. Corbett, A.T. & Anderson, J. (1995). Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge, *User Modeling and User-Adapted Interaction* 4, 253-278.
3. Mitrovic, A. & Martin, B. (2002). Evaluating the Effects of Open Student Models on Learning, *Adaptive Hypermedia and Adaptive Web-Based Systems, Proc. Second International Conference*, Springer-Verlag, Berlin Heidelberg, 296-305.
4. Papanikolaou, K.A., Grigoriadou, M., Kornilakis, H. & Magoulas, G.D. (2003). Personalizing the Interaction in a Web-Based Educational Hypermedia System: The Case of INSPIRE, *User Modeling and User-Adapted Interaction* 13(3), 213-267.
5. Weber, G. & Brusilovsky, P. (2001). ELM-ART: An Adaptive Versatile System for Web-Based Instruction, *Int. Journal of Artificial Intelligence in Education* 12(4), 351-384.
6. Mabbott, A. & Bull, S. (2004). Alternative Views on Knowledge: Presentation of Open Learner Models, *Proc. Intelligent Tutoring Systems*, Springer, Berlin Heidelberg, 689-698.
7. Kay, J. (1997). Learner Know Thyself: Student Models to Give Learner Control and Responsibility, *Proc. International Conference on Computers in Education*, AACE, 17-24.

Automatic Calculation of Students' Conceptions in Elementary Algebra from Aplusix Log Files

Jean-François Nicaud[1], Hamid Chaachoua[2], and Marilena Bittar[3]

[1] Université Joseph Fourier, Leibniz-MeTAH, 46 Ave Viallet, 38031 Grenoble, France
Jean-Francois.Nicaud@imag.fr
[2] IUFM-Leibniz-MeTAH, 46 Ave Viallet, 38031 Grenoble, France
Hamid.Chaachoua@imag.fr
[3] Universidade Federal de Mato Grosso do Sul, Campo Grande
marilena@nin.ufms.br

Abstract. We present a student's modeling process in algebra. This work is situated in the framework of the deployment of the Aplusix system, a learning environment for algebra. The process has two phases. The first phase is a local diagnosis where a student's transformation of an expression A into an expression B is diagnosed with a sequence of rewriting rules. A library of correct and incorrect rules has been built for that purpose. The second phase uses conceptions for modeling students more globally. Conceptions are attributed to students according to a mechanism using the local diagnoses as input. This modeling process has been applied to data (log files) gathered in France and Brazil with 13-16 years old students who used the Aplusix learning environment. The results are described and discussed.

1 Introduction

The Aplusix software [7], http://aplusix.imag.fr is devoted to the learning of algebra. It has been used in experiments in several contexts and countries, and has proved to be efficient: students work with pleasure, gain autonomy and improve their knowledge [8]. Furthermore, Aplusix facilitates the teacher's work because of students' autonomy and of ready-made lists of exercises. These good results led us to enter in a deployment phase in 2005. Aplusix is now distributed in France (since early 2005). It will be distributed in more than 10 countries in 2006, and in as many languages and countries as possible later.

In that context, we have launched a student modeling project[1] that aims at defining a map of conceptions in elementary algebra and at automatically calculating conceptions of students working with Aplusix in the usual framework of the class. The main goals are: (1) to have a map of possible students' conceptions that can be published in a mathematics education journal, with statistics concerning conceptions of real students from several countries and grades; (2) to inform the teachers who use Aplusix of the conceptions of their students after a few training and test sessions with the

[1] This project is funded by the programme 'ACI, Ecole et Sciences Cognitives' of the French Ministry of Research.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 433 – 442, 2006.

system in order to help them to take didactical decisions (re-teaching of a part of algebra or choice of new activities) at an individual or at a class level; (3) and to allow Aplusix choosing adequate exercises for favoring a self-correction of misconceptions.

Conceptions. Conceptions have been studied in mathematics education [1, 2]. According to Artigue [1], a conception is related to a concept and is characterized by three components: (1) a set of situations which give meaning to the concept; (2) a set of significations (mental images, representations, symbolic expressions); (3) tools (rules, theorems-in-act, algorithms). Our current focus concerns theorems-in-act. An example in algebra is the following: *When a sub-expression is moved from one side to the other side in an (in)equation, its sign is always changed.* Note that this is correct for additive movements (e.g., $x–3 = 5 \rightarrow x = 5+3$) and incorrect for multiplicative movements (e.g., $–3x = 5 \rightarrow x = 5/3$ or $x/2 = 5 \rightarrow x = –2*5$). Such theorem-in-act is not a rewriting rule: it applies to all the movement rules in (in)equations. In the rest of the paper, we will use the term *conception* instead of the *theorem-in-act* (which is a component of a conception) for fluidity reasons. With regard to reference knowledge, a conception has a domain of validity (the domain where it performs correct actions). A *misconception* (a term often used in the AI-ED community [3]) is considered here as a conception which is not 100% correct.

Student modeling. Student modeling with rewriting rules in elementary algebra has a long history. Sets of rules [10, 11] have been produced by researchers; student models are used in ITSs [5] having the capacity to help students learn. However, we cannot consider that the research problem is solved today: there is not a published set of rules established as a reference for student modeling even for a specific grade and part of algebra. We think that the main raison comes from the fact that the choice of an adequate set of rules is a difficult problem (see section 3).

Conceptions appear to be a better framework for student modeling. This framework has been investigated in mathematics education and some researches in automatic diagnosis of conceptions have been carried out [6].

Paper organization. This paper presents a complex process for designing and automatically calculating conceptions in elementary algebra. This work has been realized by researchers in mathematics education and in computer science. It uses rules as intermediate constructions for calculating conceptions. Section 2 describes the students' activities and the data gathered in France and Brazil. The construction of rules and a diagnosis process with rules is presented in section 3. Section 4 is devoted to the construction of conceptions and to diagnosis with conceptions. Note that a detailed description of this study is available in French [9].

2 Students' Activities and Data

When they use Aplusix, students solve exercises in a way similar to the usual paper context: they make the calculations and they produce several steps. An advanced editor of algebraic expressions in their usual two-dimension form facilitates this process. In the training mode, students receive two fundamental feedbacks: the indication of the correctness of their calculation steps and the indication of the correct end of the

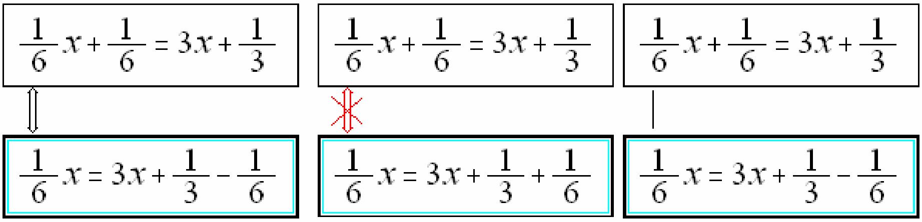


Fig. 1. A correct step in the left, with an equivalence sign; an incorrect step in the middle, with a red crossed equivalence sign; a step without feedback, in the right

exercise. In the test mode, they receive no feedback, but can later enter in a self-correction mode to check their errors. See figure 1.

Aplusix records all the students' action in log files and contains a replay system allowing seeing later the students' behavior with many details.

During years 2003 and 2004, a lot of experiments have been performed in France (540 students) and Brazil (2500 students) at grades 8, 9 and 10 to analyze the use of Aplusix, and to gather data for modeling students (about 13 000 hours of students' work). Some experiments mixed the training and test modes to help students learn algebra; other were limited to the test mode to study the students' behavior in a more stable context where learning is not supposed to occur, because of the absence of feedback. Some experiments were conducted by teachers in the usual functioning of the class; others were conducted by assistants after the class.

3 A Library of Rules and a Rule Diagnosis Algorithm

From the log files containing all the students' actions, we have extracted the steps. In a step, an expression A is transformed into an expression B. A rule diagnosis of a step consists of providing an adequate sequence of correct or incorrect rules that transforms A into B. For that purpose, we have built a library of correct and incorrect rules and designed a diagnosis algorithm. Our methodology for building the library of rules combines an epistemic approach, which consists of analyzing the algebraic elements, and a cognitive approach, which consists of studying the students' behaviors.

The granularity problem. An important question in rule construction is the following: *What is the adequate granularity of the rules to be produced?*

Let us see an example. For the *collect like terms* action, we can write the following rules (where x is the algebraic variable, and a and b are rule variables matching numbers): $x+x \rightarrow 2x$; $x-x \rightarrow 0$; $x+bx \rightarrow (1+b)x$; $ax+x \rightarrow (a+1)x$; $ax+bx \rightarrow (a+b)x$. This is adequate for degree 1 and integer coefficients. However, we can refine these rules by using only positive integers and placing the minus signs (e.g., $-ax+x \rightarrow (-a+1)x$). For higher degrees, we need to duplicate these rules, like: $ax^n+bx^n \rightarrow (a+b)x^n$. For fractions, some rules are usable (e.g.,: ax matches $(2/3)x$) but we have new situations like $(ax)/b$; x/b ; $(-x)/b$; $x/(-b)$; etc. This produces more than 300 correct rules. We can build incorrect rules by perturbing correct rules in several ways that can be checked by observing the students' behaviors, e.g., forgot or introduce minus signs, make a calculation error in a+b, a+1, etc. By this manner, we obtain more than 2000 rules. The context also has an influence, for example some students correctly transform x+x into 2x but transform x–2+x in 2 (the minus sign which is out of the terms in x is associated to them). We did not estimate the amount of rules we can produce if we introduce many contexts in rules.

This model with more than 300 correct rules for the *collect like terms* action, which is a very syntactical model, does not seem to be adequate: it would mean that students have to learn all these rules and that we have to present 300 exercises to verify his/her knowledge (and more when we introduce contexts). There is a lack of conceptualization in that model. It can be overcome by introducing a *coefficient* concept such that ax will match, using this concept, either with x, –x, 3x, –3x, (3/2)x, etc., providing a coefficient, respectively 1, –1, 3, –3, 3/2, and a variable, here x. In this situation, the 320 rules are transformed in a unique rule: $ax+bx \rightarrow (a+b)x$. Such model, with a unique rule, may be adequate for non novice students and too abstract for novice students. However, it is not easy to introduce incorrect rules by perturbing this unique rule.

Our choices. Concerning the *collect like terms* action, we have chosen the above unique rule and we have built incorrect rules by studying the behavior of students (sometime with the replay system of Aplusix). We have built correct and incorrect rules for reductions, expansions and (in)equations.

We have realized a specific work concerning the movement concept in (in)equations leading to a unique abstract rule that moves a sub-expression (the argument of the movement) from one side to the other. This rule is specified by several features described in table 1. For example, the incorrect transformation $2x–4 < 5 \rightarrow 2x > 5–4$ is represented by a movement with the argument –4 and the following vector of features: (<, LeftToRight, NumToNumerator, InitAdditive, FinAdditive, SignUnchanged, OrientationChanged). This unique movement rule represents 648 correct and incorrect detailed rules. Note that the cardinality does not really decrease, because the description of an application of the movement rule needs to indicate the features that concern some goal, but the description level is more conceptual.

Our library contains 260 correct and incorrect rules expressed in the SIM language (the rule language used by Aplusix). There are 22 SIM rules for the abstract unique movement rule).

The rule diagnosis algorithm. This algorithm aims at providing a sequence of rules transforming an expression A into an expression B. It is a heuristic search algorithm which develops a tree from A. At each step, the node of the search space which is the closest to B, according to a distance between expressions, is chosen and developed. The development consists of applying the applicable rules of the library. When the development produces the expression B, the goal is reached and the path from A to B in the tree is a sequence of rules that explains the transformation of A into B. The search phase stops when a chosen number of nodes have been developed or when no more rules are applicable. So this phase can fail. It can fail because of: a missing incorrect rule, an early stop of the process, a student's behavior that has not been understood. For example, the transformation of 2x–6 = 7x–8 into –5x = –14 is diagnosed with 4 rules: (1) Incorrect additive move of 6 leading to 2x = 7x–8–6; (2) Correct additive move of 7x leading to 2x–7x = –8–6; (3) Correct like terms collection leading to –5x = –8–6; (4) Correct calculation leading to –5x = –14.

Table 1. The seven dimensions of the unique abstract movement rule

Dimension	Possible values
Symbol of relation	$= \neq < \leq > \geq$
Horizontal orientation	LeftToRight, RightToLeft
Vertical orientation	NumToNumerator, DenoToNumerator, NumToDenominator, DenoToDenominator
Initial position of argument	InitAdditive, InitMultiplicative
Final position of argument	FinAdditive, FinMultiplicative
Change sign of argument	SignChanged, SignUnchanged
Change inequality orientation	OrientationChanged, OrientationUnchanged

The current performance of the local diagnosis, in terms of success/failure, for classes of grades 8 and 9, in France (540 students) and Brazil (2500 students), is the following: between 90% and 100% of success for correct transformations, depending of the class, and between 74% and 93% of success for incorrect transformations. Note that a failure is sometimes the best diagnosis (researchers don't always explain a student's transformation). The appropriateness of the diagnoses has been studied by two researchers for two French classes, grades 8 and 9, for incorrect expansions, incorrect reductions and incorrect transformations on in(equation): between 82% and 97% diagnoses have been considered to be correct, depending of the class and the category of rule (expansions, movement, etc.).

4 The Design and Diagnosis of Conceptions

Design of conceptions. Rule diagnoses described in section 3 are not enough to model students in a usable way. There are too many rules or too many rule features. As a consequence, statistics built with these rules are poor because the elements of the students' conceptions are distributed over too many rules or features.

In order to model students with conceptions, we have studied in detail the rule diagnoses of many students. At the present time, we have worked on the movement concept in (in)equations. We have built a model for this concept containing three aspects: the *sign aspect* (whether the syntactic sign[2] of the argument is changed or not), the *inequality orientation* for inequations (whether the orientation of the inequality is changed or not) and the *operator evolution* (what happens to the operator linking the argument to the (in)equation in the movement). For each aspect, we have defined conceptions corresponding to many students' behaviors. They are presented in table 2.

A conception is attributed to a student for a *domain of application* which can be global, or limited to equations, or limited to equations and additive arguments, etc. We have built a lattice of conceptions and domain for representing the main possibilities, see figure 2.

A conception is attributed to a student if it is verified by several behaviors and contradicted by none or just a few ones.

[2] The syntactic sign is the visible sign of the number or the monomial (it is "–" for –2 and –x).

Table 2. The defined conceptions of the 3 aspects of movement. Only 3 conceptions are correct. The other conceptions produce correct or incorrect calculations depending on the context. The argument is the sub-expression which is moved; "sign" is its syntactical sign.

Aspect	**Conception name**	**Description**
Sign aspect	CorrectSign	Correct treatment of the sign
Sign aspect	AbsoluteValue	Change the sign if it is "–"
Sign aspect	SemiAbsoluteValue	Change the sign if it is "–" and the argument is multiplicative; or if the argument is additive
Sign aspect	SaveSign	Never change the sign
Sign aspect	ChangeSign	Always change the sign
Orientation	CorrectOrientation	Correct treatment the inequality orientation
Orientation	UnifiedOrientation	Change the inequality orientation if the argument sign is "–"
Orientation	SaveOrientation	Never change the inequality orientation
Orientation	ChangeOrientation	Always change the inequality orientation
Operator evolution	CorrectOperator	Correct treatment of the operator
Operator evolution	AdditiveOperator	Final position of the argument is always additive
Operator evolution	NumeratorOperator	Final position of the argument is always numerator
Operator evolution	DenominatorOperator	Final position of the argument is always denominator
Operator evolution	SaveOperator	Final operator of the argument is always identical to initial

Diagnosis of conceptions. In order to automatically diagnose conceptions, we have built the lattice of conceptions introduced above such as the lowest nodes correspond to precise behaviors called Local Behavior Vector (LBV), see figure 2. An example of LBV for the sign aspect is: the movement occurs in an equation; the argument has an additive position; the argument has a "+" sign; the sign of the argument is changed. Such LBVs have three important properties: they are 100% correct or 100% incorrect; each LBV has an opposite LBV (the opposite of the example has the same three first elements and a last element which is "its sign is not changed"); they can be indicated by rules.

Having defined this lattice, we have built an algorithm for determining conceptions from the rule diagnosis. During a first phase, the movement rules of the rule diagnosis are used to mark LBVs. A second phase studies the couples of opposite LBV and activates LBVs according to the following rule (where LBV1 and LBV2 are opposite LBVs; LBV1 has n1 rule marks and LBV2 has n2 rule marks):

IF n1+n2 = 0 THEN there is no LBV activation
ELSE IF n1 / (n1+n2) ≥ 2/3 THEN LBV1 is activated with coefficient n1 / (n1+n2)
ELSE IF n2 / (n1+n2) ≥ 2/3 THEN LBV2 is activated with coefficient n2 / (n1+n2)
ELSE there is no LBV activation

The third phase activates other nodes of the lattice (remember that LBVs are the lowest nodes of this lattice). The activation consist of calculating the coefficient of a node as the geometrical average of the coefficients of its two direct descendants (in this lattice, all the nodes which are not bottom nodes have two descendants – if the coefficients of the two direct descendants are *a* and *b*, the coefficients of the node is sqrt(*ab*)). A node is activated when its coefficient is not null. In the last phase, the activated conceptions which are included in the more general activated conceptions are eliminated.

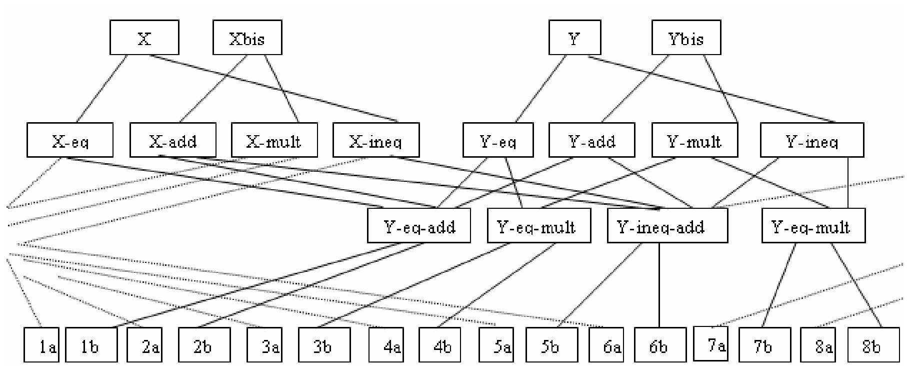


Fig. 2. Part of the lattice for the sign aspect: X is CorrectSign and Y is ChangeSign. X is decomposed in X-eq (equation) and X-ineq (inequation). With an Xbis name, it is also decomposed in X-add (additive) and X-mult (multiplicative). The bottom nodes are LBVs. Opposite LBVs are named 1a and 1b, 2a and 2b, etc.

At the end of this process, we obtain a list of conceptions for a student. Each conception has a level in the lattice (from 1 for the global ones to 3 or 4 to the most contextual ones). This level is important: level 1 means a large application field and a real conception; at the opposite, level 3 or 4 means a small application field and not a real conception. Of course, the ideal student has the 3 level 1 correct conceptions.

5 The Results

We have applied the diagnosis of conceptions to a part of our data: French and Brazilian classes of grades 8, 9 and 10. It produced a description of each student with a list of conceptions, and a summary table containing the number of occurrences of each conception.

Brazilian results. A group of 342 Brazilian students of grade 9 used Aplusix with 20 minutes familiarization and 1 hour in the test mode (where no feedback is given). The modeling process has been applied to the data of the test. The analysis of the distribution of the conceptions shows what follows:

○ Type of exercise: 56% conceptions concern equations (97% correct and 3% incorrect); 44% conceptions concern inequations (71% correct and 29% incorrect).
○ Initial position of the argument: 62% conceptions concern an additive position (95% correct and 5% incorrect); 38% conceptions concern a multiplicative position (68% correct and 32% incorrect).

Most of the conceptions concern equations with an additive position of the argument. The high rate of correct conceptions cannot be viewed as certitude of a good result because the level of the conceptions has to be taken into account. Actually, we had only 2% level 1 correct conceptions (e.g., CorrectSign). At the other levels, we have 32% correct conceptions for level 2 (e.g., CorrectSign in equations), and 66% correct conceptions concerning very specific contexts of levels 3 and 4. These results are consistent with the hand analysis made for a part of the students. The distribution of the number of conceptions per student is an approximately Gaussian distribution, see table 3.

Table 3. Distribution of the numbers of conceptions. The ideal student has 3 conceptions but most of the 80 students having 3 conceptions do not have these 3 conceptions.

Number of conceptions	0	1	2	3	4	5	6	7	8	Total
Number of students	11	37	58	80	38	47	48	22	1	342
Percentage	3.2	10.8	16.9	23.4	11.1	13.7	14	6.4	0.3	100

The distribution of the conceptions with respect to the aspects of the movement concept is shown in table 4. There are many correct conceptions for *Sign aspect* and *Operator evolution*, but just a few of them are at level 1. There is an important amount of incorrect conceptions at level 1 for *Inequality orientation*. This is consistent with the fact that these students have had many equations, and not many inequations, to solve in the preceding school year.

Table 4. Distribution of the conceptions with respect to the aspects of the movement concept

Level of the conception	Sign aspect		Inequality orientation		Operator evolution	
	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
1	20	0	0	37	0	0
2	158	0	85	4	158	0
3	129	11	0	0	186	0
4	0	0	0	0	335	9

French results. We have applied the modeling process to 221 French grade 10 students, obtaining similar results.

Last, we have modeled 30 French grade 10 students who used Aplusix during the whole school year 2003-2004. The analysis of the collected data shows:

- ○ A total of 171 conceptions: 18% for level 1, 15.2% for level 2, 30% for level 3;
- ○ 70% students having the CorrectSign, level 1, correct conception;
- ○ 40% having the SaveOrientation, level 1, incorrect conception (never change the sign of an inequality).

Note that these students, who had a longer training, have more general conceptions, and that the sign aspect is rather well acquired but the orientation aspect of the inequality is not.

6 Discussion and Future Work

This work is a significant step towards the achievement of the goals we have presented in the introduction. The obtained results are consistent with opinions of teachers and with analyses by hand of a part of the data. However, we need to analyze in depth the data that are not captured by the process. For example when we find that a student has 3 conceptions, we have an interesting result, but we would like to have an opinion about the behaviors of this student that do not contribute to these 3 conceptions. Some of them may be random behaviors, others rational behaviors not captured by the model.

We aim at covering the domain of algebra for grades 8 to 10. We are currently working on conceptions for expansion and reduction. When this coverage will be achieved, we will be able to produce statistics for various situations (we currently have experimentations in Brazil, France, India, Italy, Vietnam). At this moment, we will be also able to add a new module to Aplusix for the teacher's use. These two objectives demand to produce conceptions understandable by common teachers, i.e., teachers who are not involved in research projects, and who are just willing to teach their students. Of course, pioneer teachers and mathematic education researchers are also concerned by the results. Our third goal, consisting of automatically choosing adequate exercises for favoring students' self-correction of misconceptions, will be pursue by an automatic indexation of exercises with conceptions and adequate feedbacks. At the present time, we do not envisage giving students access to their models (like in the open model theme [4]). We will think of that when our map of conceptions will be achieved. In algebra, many concepts are *in action*, so it is not sure that the description of conceptions, correct and incorrect, will help students. It may be more efficient to provide good feedback coming from the calculated conceptions in specific situations.

References

1. Artigue M. (1991). Epistémologie et didactique. Recherche en didactique des mathématique, 10 (2/3).
2. Balacheff N., Gaudin N. (2002). Students conceptions: an introduction to a formal characterization. Cahier 65 du laboratoire Leibniz, http://www-leibniz.imag.fr.

3. Dillenbourg P. & Self J.A. (1992) A framework for learner modelling. Interactive Learning Environments. vol. 2, nº 2, pp. 111-137
4. Dimitrova V., Self J., Brna P. (2001). Applying Interactive Open Learner Models to Learning Technical Terminology. UM 2001, Springer, p 148-157.
5. Koedinger, K. R., Anderson, J. R., Hadley, W. H., Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education, 8, p. 30-43.
6. Jean S., Delozanne E., Jacobini P., Grugeon (1999). A diagnostic based on a qualitative model of competence in elementary algebra. AI-ED 1999, IOS Press, Amsterdam, p. 491-498.
7. Nicaud, J.F., Bouhineau, D., and Chaachoua H. (2004). Mixing microworld and CAS features in building computer systems that help students to learn algebra, in Int. Journal of Computers for Mathematical Learning, Volume 9 (2), Springer-Verlag.
8. Nicaud J.F., Bittar M., Chaachoua H, Inamdar P., Maffei L. (2005). Experiments of Aplusix in Four Countries. 7th International Conference on Technology in Mathematics Teaching, Bristol.
9. Nicaud et al. (2005). Modélisation cognitive d'élèves en algèbre et construction de stratégies d'enseignement dans un contexte technologique. Project report of the "Ecole et sciences cognitives" research programme. Cahier 123 du laboratoire Leibniz, http://www-leibniz.imag.fr
10. Payne S. and Squibb H. (1994) Algebra Mal-Rules and Cognitive Accounts of Error. Cognitive Science, 14.
11. Sleeman D.H. (1983). Inferring student model for intelligent aided instruction. In Michalski, Carbonell and Mitchell (Eds) Machine Learning: An Artificial Intelligence Approach. Morgan Kaufman.

The Potential for Chatbots in Negotiated Learner Modelling: A Wizard-of-Oz Study

Alice Kerly and Susan Bull

Electronic, Electrical and Computer Engineering, University of Birmingham,
Edgbaston, Birmingham, B15 2TT, UK
{alk584, s.bull}@bham.ac.uk

Abstract. This paper explores the feasibility of using conversational agents, or chatbots, in negotiated learner modelling. This approach aims to combine the motivational, intuitive and domain-independent benefits of natural language dialogue using a chatbot, with the opportunities for learner reflection and increased model accuracy that can be achieved through negotiation of the learner model contents. A Wizard-of-Oz paradigm allowed investigation into the interactions between learners and their learner model in order to highlight key issues for the design of a chatbot for this purpose. Users appreciated interacting with a chatbot, and found it useable and an aid to negotiation. The study suggested many avenues for future investigation of the role of conversational agents in facilitating user-system dialogue about learner understanding.

1 Introduction

It has been argued that making visible ("opening") the learner model of an intelligent tutoring system (ITS) to its users can provide opportunities for learner reflection that strengthen and enhance the learning experience (e.g. [1], [2], [3], [4]). Mr. Collins [5] and STyLE-OLM [2] proposed and implemented the learner's participation in the construction and maintenance of the learner model through a process of negotiation of the model contents. They allowed learners to discuss their beliefs with the system, and to argue against the system's assessment of their knowledge if they disagreed with it, with the aim of enhancing learner reflection and improving learner model accuracy.

Mr. Collins was developed with the open learner model (OLM) central to the system. In order to discuss beliefs, the system must have knowledge of the student's real beliefs (in case these differed from the system inferences), as well as its own beliefs about the student's knowledge. Thus, two belief measures were maintained, each of which the system used to determine appropriate adaptive interactions [5]. This allowed both student and system to retain autonomy over their individual beliefs. With this ethos, the student is able to alter their self-assessment as they wish (though the system will challenge them if it disagrees), and must engage in negotiation if they wish to alter the system's belief. Baker's [6] notion of interaction symmetry was applied by making all negotiation moves available to both student and system. The Mr. Collins system utilised menu-selection for negotiation. This provided an efficient and usable interaction (as demonstrated in a laboratory study [5]), but may have restricted some users by being an unnatural method for communicating beliefs.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 443 – 452, 2006.

STyLE-OLM [2] also provided negotiation of the learner model, through graphical representations of conceptual graphs. Both learner and system formed propositions about learner understanding by constructing conceptual graphs and selecting an appropriate dialogue statement. In a laboratory trial STyLE-OLM was successful in engaging learner reflection and in improving the resultant learner model.
However, the use of a graphical approach may be difficult for some learners, and may sometimes detract from learning of the domain in question.

Given the positive reactions to negotiated learner modelling in Mr. Collins and STyLE-OLM, it would be useful to consider allowing negotiation in a more flexible, intuitive and naturalistic way than previously implemented. A method that does not require learning additional communication skills, and that does not detract from the target domain, is required. Natural language negotiation may be a way to enable this.

Conversational agents, or chatbots, provide a natural language interface to their users. Their design has become increasingly complex, and modern commercial chatbots, such as Lingubot technology [7] offer sophisticated development environments, allowing the building of intelligent conversational agents with complex, goal driven behaviour. In 'Lingubots' both the words and grammatical structure of user input are analysed. This allows the development of a user model, which is used in conjunction with conversational context and dialogue content to determine the chatbot's response. This seems particularly appropriate for student-system collaborative construction of the learner model.

2 A Wizard-of-Oz Study

This paper describes a Wizard-of-Oz experiment to explore the feasibility of using a chat-based interface to an OLM system. Participants interacted with the system to view their learner model and discuss or negotiate over the contents with the system 'chatbot'. The role of the 'chatbot' was taken by a human 'Wizard', a fact not revealed to the participants. All interactions were logged. Participants completed questionnaires indicating their opinions on the interaction with the 'chatbot'.

2.1 The Wizard-of-Oz Paradigm

The Wizard-of-Oz method allowed the collection of data on the human-computer interactions that arose in negotiating the learner model. Computer mediated human-human interaction data can be an unreliable source of information for some important aspects of dialogue design as humans and computers can be expected to perform differently in conversation [8]. The dialogues required to negotiate over the learner model are expected to be complex and varied in terms of language, tasks, and domain content. Wizard-of-Oz studies have been shown to be an appropriate approach in collecting data about dialogues in complex domains such as this [9].

In this study, the 'Wizard' (experimenter taking the role of the chatbot) followed a protocol designed to ensure that responses to students remained consistent between users, and that the 'chatbot' was believable to users. The Wizard-of-Oz paradigm necessarily involves deception of the participants about their true activities in the experiment. Work was conducted within British Psychological Society Ethical Principles for Research with Human Participants [10].

2.2 The Learner Modelling System

This study employed a modified version of Flexi-OLM [11], which operates in the domain of C programming. The system's beliefs are inferred from students' answers to multiple-choice and short-answer questions. For this study Flexi-OLM was extended to also allow learners to record their own beliefs about their knowledge. These two belief sets make up the learner model contents, viewable in seven formats.

2.3 Participants, Materials and Methods

The participants were 11 final year undergraduates from the University of Birmingham Electronic, Electrical and Computer Engineering Department. All had taken courses in educational technology and C programming. All were competent English language speakers, though in some cases English was not their first language.

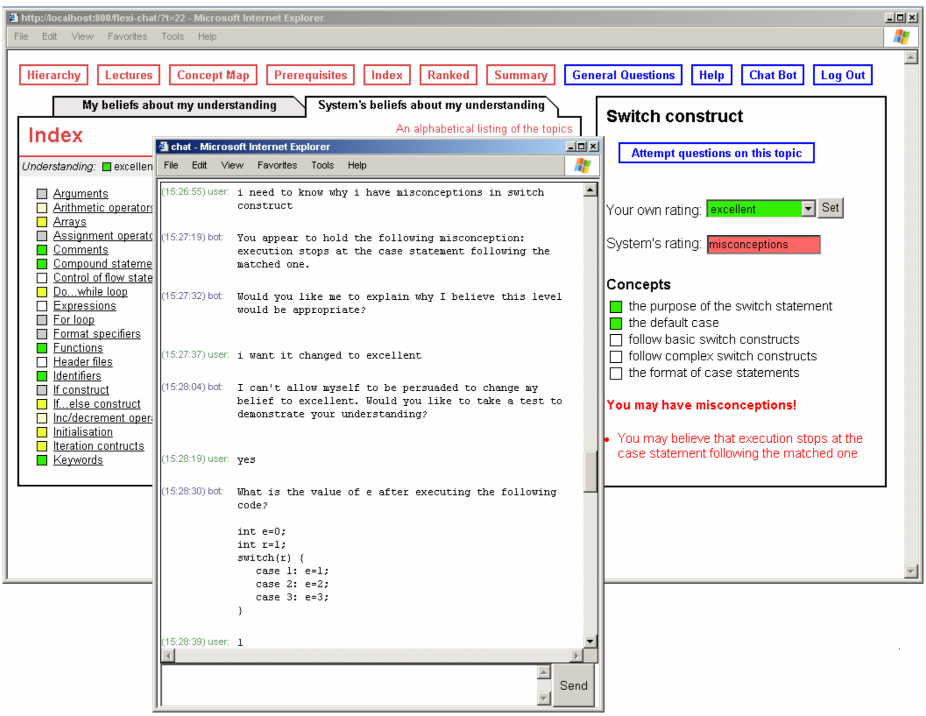


Fig. 1. User's screen in Index view of System beliefs on student knowledge & chat interface

Figure 1 shows a screenshot of user activity in Flexi-OLM. Buttons at the top of the screen allow the user to view their learner model contents as either: lecture structure, topic hierarchy, concept map, pre-requisites, index, ranked or textual summary views. Although these views differ in their presentation, the same model information is available in each. Below these buttons are tabs allowing the user to switch between their own or the system's beliefs about their knowledge.

Topics group together several smaller concepts. For each topic or concept a learner's knowledge is represented at one of four levels by a node, coloured between white (very limited), through yellow and green to bright green (excellent). If the learner holds a misconception about a concept, the parent topic is changed to red.

The current topic (here, "Switch construct") is shown on the right. A button allows answering questions on this topic, and the user's and system's beliefs are shown for ease of comparison. To allow investigation of negotiation through a chatbot interface, a chat window was provided. Students viewed the 'chatbot's' conversation in the top part of the window, and typed their responses to the 'chatbot' in the bottom section.

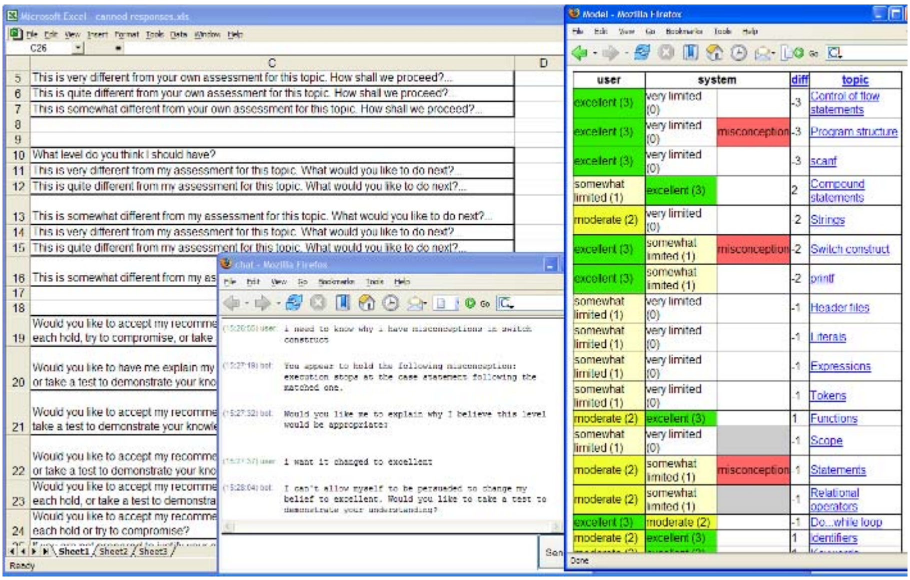


Fig. 2. Wizard's view showing (left) part of the canned responses; (centre) Chat interface - excerpt reproduced below; (right) Learner Model (part)

- Chatbot: Would you like me to explain why I believe this level would be appropriate?
- User: I want it changed to excellent.
- Chatbot: I can't allow myself to be persuaded to change my belief to excellent. Would you like to take a test to demonstrate your understanding?

Mr. Collins [1] allowed negotiation by a range of strategies: ask user if they wish to accept system view; offer compromise; ask user to justify their belief (by taking a test to demonstrate knowledge); system justify its belief; or offer student opportunity to view learner model. These strategies formed the conversational basis of the 'chatbot'.

Table 1 shows the key structure of the decision tree constructed to assist the Wizard in providing appropriate responses. To enact the protocol, the Wizard was provided with some 350 pre-authored 'chatbot' negotiation initiations and responses to user inputs. These allowed the Wizard to respond to user inputs quickly and consistently, and to avoid betraying the fact that the 'chatbot' was controlled by a human. These can be seen in the left section of Figure 2. The learner model (right of Figure 2) was presented to enable the Wizard to compare user and system beliefs for each topic at a glance. User and system beliefs were colour coded using the previously described scheme. Unanswered questions were available to allow the student to demonstrate their knowledge in a further test. The window provided for 'chatbot' responses was identical to that of the users.

Table 1. Key actions from the Wizard's decision protocol

If:	User has challenged system belief	
Then:	Respond with an appropriate strategy (view model, compromise, user accept, system justify, user justify)	
	System can be persuaded without evidence to move one level up or down to match user belief, except if user wishes to change system belief to excellent, insist on test to demonstrate knowledge	
If:	Discrepancies between system and user beliefs AND user has not already initiated negotiation AND user is not currently answering questions OR user has changed own belief, causing discrepancy	
Then:	Initiate interaction on topics with largest discrepancy/misconception first	
	Offer the five strategies to proceed for user to choose between:	
	User accepts system view	Update user belief
	System justify itself	Show user's previous answers
	User justify self	Give test questions
	Compromise	Set user's and system's beliefs to mid-point
	View learner model	Remind user how to view
If:	All options exhausted for topic, but no resolution to negotiation	
Then:	Agree to differ	

2.3.1 Experimental Setup

Participants were instructed that they would be able to use a chatbot feature of Flexi-OLM to discuss their learning of C programming and which could lead to modification of the learner model. They were told that they could answer questions; view the system's model of their knowledge, or their own belief model in seven formats; edit their own beliefs for a topic (though the system may challenge this); and negotiate over the model by challenging the system's beliefs, seeking justification for the system's beliefs, offering their own justifications of their knowledge, taking additional tests on a topic, attempting to compromise with the system, or by discussing the differences between the belief models. Participants were told that they could initiate discussion at any stage, or that the chatbot may do so. Participants were requested to be natural in their language, but precise and clear.

Each participant was shown how to use the chatbot and asked to interact with it for at least 20 minutes. Prior to using the system, each participant completed a self-assessment, rating their knowledge of each concept, providing the initial content of the user's beliefs in the learner model. Nine users had previously used Flexi-OLM and data from this usage allowed construction of the system's beliefs in the learner model. Two users interacted with the system before beginning the experiment to enable the system to model their understanding.

A post-experiment questionnaire comprised statements requiring participants to indicate their level of agreement (strongly agree, agree, neutral, disagree, strongly disagree) and open-ended questions to elicit further comments or suggestions.

2.4 Results

This section presents results relating to negotiation and interaction with the 'chatbot'.

2.4.1 Negotiation Issues Findings

The logs of interactions show there were 20 occasions when the system initiated discussion of the learner model, a mean of 1.8 per user session. The user initiated negotiation on 29 occasions, a mean of 2.6 per user session. The greatest number of initiations by a single user was eight. Two users did not attempt to initiate negotiation.

Table 2. User opinions of the negotiation facilities

		<strongly agree....strongly disagree>					Mean score
		(1)	(2)	(3)	(4)	(5)	
I was frequently convinced by the Chatbot's arguments		1	6	3	1	0	2.4
The negotiation changed my view of my understanding		2	5	4	0	0	2.2
In negotiation, I wanted to discuss	my knowledge	7	3	1	0	0	1.5
	my knowledge level	7	3	1	0	0	1.5
	my misconceptions	7	3	0	1	0	1.5
	the gaps in my knowledge	4	5	0	2	0	2.0
I wanted to be able to negotiate at a topic overview level		3	6	0	2	0	2.1
I wanted to be able to negotiate at a more detailed (concept) level		3	3	1	4	0	2.5
I used the Chatbot to help me understand my learner model		2	4	3	1	1	2.5
I always challenged the Chatbot when I disagreed with it (or I would)		7	3	0	1	0	1.5

As shown in Table 2, in the questionnaire seven users agreed or strongly agreed that they were frequently convinced by the chatbot's arguments; one user disagreed. User comments supported this, including indicating that the chatbot was "right most of the time, and never unreasonable". Another user said, "I could see *why* it was disagreeing".

Users were also asked to comment on what they thought of the chatbot when it agreed or disagreed with them. When there was disagreement, users made comments such as "I saw it as an opportunity to put it right if it was wrong and correct myself if it proved I was wrong". Users also wanted to understand the reasons for any disagreement, variously stating, "I could see why it was disagreeing so it wasn't too bad" and "when it disagreed it was justified".

When users found the chatbot in agreement with their views, opinions were mainly positive. Users' comments included "it only agreed when the difference was marginal so it was good" and "I liked that it wasn't just agreeing with me all the time". As with situations when the chatbot disagreed with the user, users wanted to understand the reasons for the chatbot's belief, for example, "[I] could see why it was agreeing, so I was pleased", and requested "feedback of how many answers got right or wrong".

Regarding when and what the users wanted to discuss, most users indicated that they would always challenge the chatbot when its belief differed from their own, that they wanted to discuss each of their knowledge, their knowledge level or their misconceptions, and that they wanted to discuss the gaps in their knowledge. Nine students wanted to be able to negotiate the learner model at a topic overview level, with six wanting to negotiate at a more detailed (concept) level.

Users suggested that the chatbot was "an interesting approach to learning – more enjoyable than just using the original Flexi-OLM" and that it was a "good way of negotiation and justification".

One of the principal aims of opening the learner model is to promote user reflection. Seven users stated that the negotiation had changed their view of their understanding while four remained neutral on this. Six users indicated that they used the negotiation to understand their learner model, while three remained neutral, and two disagreed.

2.4.1 Chatbot Interaction Findings

As Table 3 shows, all users stated they liked the chatbot. Most indicated that they enjoyed interacting with it.

More users liked the chatbot when it agreed with them than when it disagreed. Only three users stated they liked the chatbot when it disagreed with them, while seven liked the chatbot when it was in agreement with their beliefs.

Table 3. User opinions of the chatbot

	<strongly agree...strongly disagree>					Mean score
	(1)	(2)	(3)	(4)	(5)	
I liked the Chatbot	2	9	0	0	0	1.8
I liked the Chatbot when it disagreed with me	1	2	6	1	1	2.9
I liked the Chatbot when it agreed with me	2	5	4	0	0	2.2
I enjoyed interacting with the Chatbot	5	5	1	0	0	1.6
I was able to achieve the negotiation tasks I wished with the Chatbot	4	5	2	0	0	1.8
The Chatbot made negotiation easy	4	6	0	1	0	1.8
The Chatbot was a convenient way to provide my opinions to the system	5	4	2	0	0	1.7

An indicator of user enjoyment of the chatbot is the users' language that shows their level of relaxation in the interaction. Examples of inputs include "lol" (meaning laugh out loud, indicating amusement), and "lol, never mind then :-)". Users also commented that the interaction "felt more relaxed" and even that they would like to be able to "discuss other stuff, e.g. weather, just for a little break from working!". However, a few students reported that the interaction seemed rigid.

A number of users felt that they were not alone in the use of the system, and it appears that the presence of the chatbot was felt to be companionable. For example, users commented, "I liked the fact that there was an interaction going on. I wasn't in the learning area by myself", "it makes learning system active, such as chatting with real people", and "it was very good to use chatbot because you can interact with [it]".

Given the intention of the chatbot is to facilitate communication, it was important to discover whether this was the user experience. Nine users said that the chatbot was a convenient way to provide their opinions to the system. Their comments included, "The chatbot is a fast and effective yet enjoyable way of creating an accurate learner model", and "A more effective way of giving information into the learner model".

Most users agreed that the chatbot made negotiation easy and that they were able to satisfactorily complete their desired tasks.

3 Discussion

The central concern of this study was whether a chatbot might effectively provide or enhance negotiation in an open learner model, to explore user reactions to the 'chatbot', and to investigate the technical challenges for implementation.

3.1 Issues Raised by Study

The results of this study suggest that users found the negotiation easy and useful, and appreciated the facilitation of the chatbot. One user commented that she saw disagreements with the system as "an opportunity to put it right if it was wrong and correct myself if it proved I was wrong". This, and the fact that users were often convinced by the chatbot's arguments, demonstrates a willingness to revise personal beliefs that is essential for the success of negotiated learner modelling.

When the system and user disagreed, users were clear in their wish to see why this was. This indicates users' need to understand the reasoning behind system beliefs. A capacity for explanation could prove a valuable addition to OLM systems.

The finding that most users would always challenge the system if they disagreed with it is an important indicator that users were sufficiently concerned about the chatbot's opinion to make the effort to challenge it. This accords with findings for the menu-based negotiation of Mr. Collins [5].

3.2 Implications for the Development of Chatbots in Negotiated Learner Modelling

The user comments indicate support for the inclusion of a chatbot in a negotiated OLM system. However, several points indicated areas for particular consideration for an implementation to be successful in user acceptance and fitness for purpose.

A number of users commented that the chatbot sometimes appeared rigid or formal. This is not surprising given the limited scope of the Wizard's responses, but should be rectified in future work. Smalltalk (conversation in areas unrelated to the subject in hand) may be useful in helping students build a rapport with the chatbot. When combined with effective methods to draw the user back on topic, incorporation of smalltalk gives a far more rounded and naturalistic conversation, more akin to a discussion and less of a clever question answering system [12].

The user interactions in this study were relatively short, and were single sessions. Although the limited canned responses were necessary for the structure of this experiment, if users are to remain engaged with the chatbot, and to find its comments both useful and interesting, a richer range of chatbot responses should be provided.

It is also worth noting that many types of interaction are potentially valuable to the learning experience, and it is therefore reasonable to expect the chatbot to be able to deal with these. Although negotiation of the learner model is the primary task, it would be reasonable to expect the chatbot to cope with smalltalk, information requests, being told when its response is wrong, and clarification when things are not understood. Future development of the chatbot may enable it to extract and provide teaching material or point users to it, perhaps by opening relevant web pages.

4 Summary

This paper has described a Wizard-of-Oz experiment exploring the potential of using natural language conversation to negotiate an OLM system. Eleven participants used a simple ‘chatbot’ interface to engage with an existing OLM system.

Users were happy to engage with the system, found the ‘chatbot’ easy to interact with, and saw it as an effective way of providing information to the system. Most found the ‘chatbot’ friendly and helpful, and were happy to accept the ‘chatbot’s’ arguments, revising their beliefs where necessary. Users’ comments suggested a strong need to understand the reasoning behind system beliefs. A chatbot provides excellent facility for this type of explanation.

The study demonstrated the potential for using a ‘chatbot’ to take a central role in negotiating the learner model. Users indicated building rapport with the chatbot, which could be improved through adding smalltalk capabilities to core negotiation.

This work has provided support for implementation of a real chatbot, although limitations of the participant sample size are acknowledged. Future work will address shortcomings identified by users, and consider how negotiation can be enhanced through the inclusion of a wider range of natural language dialogue strategies. The results showed the potential extension a natural language interface could provide an OLM system, a useful first step in the investigation of the potential of chatbots in negotiated learner modelling. Work has commenced to consolidate the findings and, in collaboration with Elzware, a leading UK developer of Lingubot chatbots, development of a prototype OLM system with embedded chatbot negotiation.

Acknowledgement

We thank Andrew Mabbott for extending Flexi-OLM [11] for this study. The first author is funded by the UK Engineering and Physical Sciences Research Council.

References

1. Bull, S., Pain, H., & Brna, P.: Mr. Collins: A collaboratively constructed, inspectable student model for intelligent computer assisted language learning. Instructional Science 23. Kluwer Academic Publishers. Netherlands (1995) 65-87
2. Dimitrova, V.: STyLE-OLM: Interactive Open Learner Modelling. International Journal of Artificial Intelligence in Education, Vol 13 (2003) 35-78
3. Morales, R.: Exploring participative learner modelling and its effects on learner behaviour, unpublished PhD Thesis, University of Edinburgh (2000)
4. Zapata-Rivera, J-D. & Greer, J.: Externalising Learner Modelling Representations, Proc. of Workshop on External Representations in AIED: Multiple Forms and Multiple Roles. Intl Conference on Artificial Intelligence in Education (2001) 71-76
5. Bull, S. & Pain, H.: 'Did I say what I think I said, and do you agree with me?': Inspecting and Questioning the Student Model. Proceedings of World Conference on Artificial Intelligence in Education, Charlottesville, VA (1995) 501-508
6. Baker, M. J.: Negotiated Tutoring, and Approach to Interaction in Intelligent Tutoring Systems, unpublished PhD thesis, Open University, Milton Keynes, UK (1990)
7. Lingubot technologies UK distributor. http://www.creativevirtual.com/ accessed 14/12/05
8. Dahlbäck, N., Jönsson, A. & Ahrenberg, L.: Wizard of Oz Studies – Why and How. Proceedings of Intelligent User Interfaces (1993) 193-200
9. Bernsen, N. O., Dybkjaer, H., & Dybkjaer, L. Designing Interactive Speech Systems – From First Ideas to User Testing. Springer. (1998), cited in: Fiedler, A., & Gabsdil, M.: Supporting Progressive Refinement of Wizard-of-Oz Experiments. Proceedings of the ITS 2002 - Workshop on Empirical Methods for Tutorial Dialogue Systems, San Sebastian, Spain (2002) 62-69
10. British Psychological Society Ethical Principles for Conducting Research with Human Participants. Available: http://www.bps.org.uk/the-society/ethics-rules-charter-code-of-conduct/code-of-conduct/ethical-principles-for-conducting-research-with-human-participants.cfm accessed 18/10/05
11. Mabbott, A & Bull, S.: Student Preferences for Editing, Persuading, and Negotiating the Open Learner Model. 8th International Conference of Intelligent Tutoring Systems, Springer-Verlag, Berlin Heidelberg (in press)
12. Hall, P.: Director, Elzware (http://www.elzware.com) Personal communication 20/12/05

Improving Intelligent Tutoring Systems: Using Expectation Maximization to Learn Student Skill Levels

Kimberly Ferguson, Ivon Arroyo, Sridhar Mahadevan, Beverly Woolf, and Andy Barto

University of Massachusetts Amherst, Computer Science Department
140 Governor's Drive, Amherst, MA 01002
{kferguso, ivon, mahadeva, bev, barto}@cs.umass.edu

Abstract. This paper describes research to analyze students' initial skill level and to predict their hidden characteristics while working with an intelligent tutor. Based only on pre-test problems, a learned network was able to evaluate a students mastery of twelve geometry skills. This model will be used online by an Intelligent Tutoring System to dynamically determine a policy for individualizing selection of problems/hints, based on a students learning needs. Using Expectation Maximization, we learned the hidden parameters of several Bayesian networks that linked observed student actions with inferences about unobserved features. Bayesian Information Criterion was used to evaluate different skill models. The contribution of this work includes learning the parameters of the best network, whereas in previous work, the structure of a student model was fixed.

1 Introduction

Intelligent tutoring systems provide individualized instruction based on students' knowledge level. This is a hard problem mainly because knowledge is measured in terms of skill mastery, which are unobservable abstractions. Previous approaches include belief networks [12] and Bayesian networks [3],[1]. Traditional Bayesian approaches to determining students' understanding of a skill consist of the evaluation of observable student behavior on problems that are thought to require specific skills to be solved correctly. For example, the cognitive mastery approach performs Bayesian estimations of mastery given some observed evidence about students' correct or incorrect responses to problems [4]. Such models rely on parameters that link problems to skills, such as the probability of answering a problem correctly even though the skill is not mastered (guessing) and the probability of answering incorrectly given that the skill is mastered (slipping). These parameters are easier to estimate when a problem involves just one skill, but get more complicated as the number of skills involved in a problem increases.

We propose that a full model of student mastery of skills can be learned with machine learning techniques that deal with missing data. Our past work showed

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 453–462, 2006.

that student models can be learned from simulated data of students' responses on problems [8]. In this paper, we present the results of learning a student model from real student data. We take students' actual responses from a pencil-and-paper pre-test and learn the parameters that link problems to skills, producing a model that allows us to make inferences about students' mastery levels. Last, we use the resulting model for inference and show how the mastery of skills improves after using our tutoring system.

One concrete future use of this model is a proper initialization of the student model before the tutoring session starts. Students will take an online pre-test, skill mastery will be inferred, and then the tutoring session will start, well informed about each student's strengths and weaknesses. More precisely, the ITS will have information about which skills a student has already mastered and what level of improvement is still needed for others.

2 Problem Definition

The goal of this research is to gain knowledge of the student's skill level and thereby improve the problem and hint selector of the Intelligent Tutoring System. Decisions made by the problem selector improve the tutor's ability to customize problems for an individual student.

Wayang Outpost is an Intelligent Tutoring System that emphasizes SAT-math preparation [2]. The system can individualize the tutoring based upon a student's specific needs. In particular, the intent is to develop a tutor that will select an action (i.e. a particular problem or hint) based on knowledge about a student's learning needs (i.e. problems he or she tend to get correct/incorrect/skip, inferred knowledge and skills, motivation level, mathematics fact retrieval time and learning style). The information we know about students increases as they use the system, but we also need to gather some student characteristics before the tutoring session begins, to initialize the student model and start proper tutoring. This information is collected by giving a pre-test to the students that includes short-answer problems similar to the problems within the tutoring session. This helps determine which skills a student initially has mastered so this information can be used to discern the best policy for optimizing the student's learning.

We cannot observe a student's mastery of a skill directly, so the tutor must infer this knowledge from student answers to problems involving these skills. Twelve basic geometry skills were identified by psychology and education researchers based on skills commonly used on the math portion of the SAT. These skills include: Skill 1. area of a square; Skill 2. area of a right triangle; Skill 3. properties of an isosceles triangle; Skill 4. identify rectangle; Skill 5. area of a rectangle; Skill 6. perimeter of a rectangle; Skill 7. identify right triangle; Skill 8. area of a triangle; Skill 9. Pythagorean theorem; Skill 10: corresponding angles; Skill 11: supplementary angles; Skill 12: sum of interior angles of a triangle.

Geometry problems were created utilizing these 12 skills, in the following fashion: 12 one-skill problems each of which used only 1 of each of the 12 skills;

12 two-skill problems which combine 2 of the 12 skills; 4 three-skill problems. Each set of three skills {1,2,3}, {4,5,6}, {7,8,9}, and {10,11,12} (see Figure 2) was combined to create the one-skill, two-skill and three-skill problems as described above for each set. This totals 28 problems for the student pre-test, 7 from each set of 3 skills. Some skills are obviously simpler than others (i.e. identify rectangle versus Pythagorean theorem); thus, we expect to find results that these skills are initially mastered more than others. Similarly, the one-skill problems were generally easier for students than the two-skill and three-skill problems.

We know exactly which problems involve which skills, and we know how each student answers each problem: incorrect, correct, or skipped. However, the actual mastery of a skill, which is what we really want to know, is a hidden variable. We treated these hidden skill variables as missing data and used Expectaction Maximization (EM) to estimate the parameters of the Bayesian network to learn skill mastery. This methodology is described in Section 4.

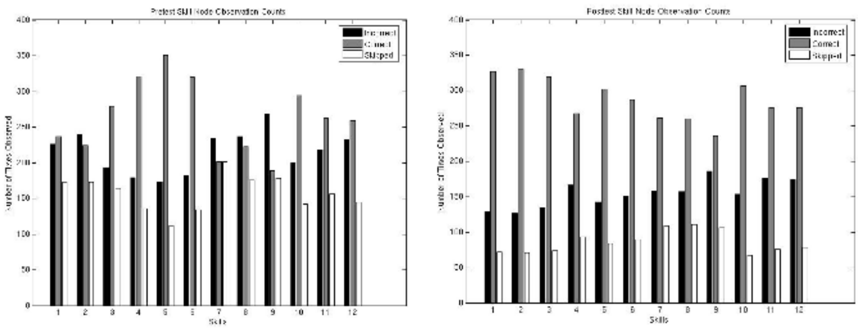
3 Data

The data used to create the models comes from a Spring 2005 study in an urban area High School in Massachusetts. Students took the pre-test, then spent two days using the tutor (50 minutes the first day and 30 minutes the next day) and then took the post-test. A total of 159 students took the pre-tests that were used to learn the Bayesian network and 132 students took the post-tests that were used to learn a second Bayesian network. The post-test involves different problems than the pre-test, but each problem is associated with the same set of skills as in the pre-test. Each problem resulted in one of four observable student actions: incorrect, correct, skipped or left blank[1]. Problems that are *skipped* supply information about the associated skill(s) being possibly unmastered, as opposed to problems that are *left blank* that we discount as uninformative.

Some information can be gathered based on the raw counts of how many student observed answers for problems involving each skill were incorrect, correct, or skipped (See Figure 1). Note when examining the raw counts that due to various reasons 27 more students took the pretest than the posttest, therefore small decreases/increases may not be significant.

The difference between the pre-test and post-test graphs suggests that the Intelligent Tutoring System is teaching students to improve their performance on these types of problems. More problems were answered correctly and less skipped from pre-test to post-test. More importantly, the non-uniform distribution of skill levels suggests that students improve more on certain skills than others, highlighting the value in modeling and learning the distribution of skill mastery levels across all students as well as within individual students.

[1] *Left blank* is different than *skipped* since a problem is *left blank* when the student runs out of time before reaching the problem (and thus is left blank at the end of the test); instead, a problem is considered *skipped* when the student may not know how to solve the problem and then skips to the following problem (and leaves the problem blank in between other answered problems).



(a) Pretest Observed Counts Across Skills

(b) Posttest Observed Counts Across Skills

Fig. 1. Improvement in student skill mastery from pre-test to post-test

4 Methodology

As stated earlier, the goal is to infer a student's skills based on evaluating only the pre-test. We now describe how to learn a Bayesian model of skill mastery that links problem outcomes to skill mastery levels. Such a network has hidden nodes representing the mastery of each skill (modeled by a binomial distribution) and observed nodes representing student answers (modeled by a multinomial distribution). Problems were created by hand to be associated with specific skills, so we have a clear idea of how problems are linked to skills. We constructed a Bayesian network (BN) linking each of the 12 skills to each of the problems that are associated with it. Each skill is used in four problems: 1 one-skill problem, 2 two-skill problems, and 1 three-skill problems (see Figure 2).

Even though the dependencies between skills and problems are known, it is not clear what the structure of the skills should be–whether the domain skills should be linked within the network (i.e., does the mastery of one skill affect the mastery of another skill?). Thus, we analyzed different models using the following methodology: 1) Expectation Maximization method (EM) was used to learn the

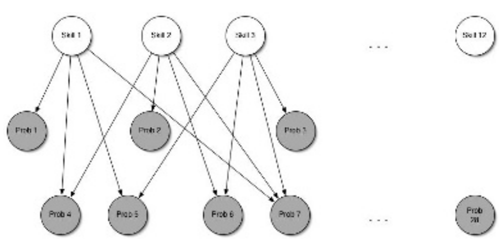


Fig. 2. Bayesian Network for Flat Skill Model: This identical linking pattern is repeated for each group of 3 skills and 7 problems. In total, there are 12 Skills (hidden nodes), and 28 Problems (observed nodes). Skills have 2 possible values: Not Mastered, Mastered. Problems have 3 possible observed actions: Incorrect, Correct, Skipped.

parameters of each Bayesian network on the pre-test data; 2) different BN models were evaluated using the Bayesian Information Criterion (BIC) to decide which model fit the data best; 3) the best fitting model was used for inference of student mastery levels given some outcome of responses to problems.

- *Parameter Learning using Expectation Maximization:* EM is a framework for maximum-likelihood parameter estimation with missing data [5]. EM finds the model parameters that maximize the *expected* value of the log-likelihood, where the data for the missing parameters are "filled in" by using their expected value given the observed data. In general, EM is trying to learn the pattern that best associates the observed data and the hidden parameters within the context of the specified graphical model. The log-likelihood value maximized though EM is used to calculate the BIC score of that model.
- *Model Evaluation using Bayesian Information Criterion:* The BIC [13] is used to evaluate different models. Simply put, given a set of data and probability distribution generating the likelihood, the larger the likelihood, the better the model fits. The BIC score is calculated with the following formula: $-2 * ll + npar * log(nobs)$, where ll is the log-liklihood, $npar$ represents the number of parameters and $nobs$ the number of observations in the model. Thus, the model with the highest BIC score is assumed to have the best structure for this task.
- *Inferring Mastery Levels:* Inference can be thought of as querying the model. The junction tree inference algorithm was used, which is an exact inference engine that uses dynamic programming to avoid redundant computation. Given a set of observations, the algorithm provides the probability of other events. For example, when a student answers Problem 1 (using only Skill 1) and Problem 2 (using only Skill 2) correctly, how likely is it that he or she will answer Problem 4 (using Skills 1 and 2) correctly as well. Inference is used to predict a student's overall skill mastery. Given a pre-test for a new student, inference is run on the learned model given the new student answers to estimate this student's skill levels.

Flat Skill Model

Based on the parameters learned when using uniform priors (where the probabilities of *mastered* and *not mastered* are both initially 50%), we discovered that it is not the *incorrect* answers from which we infer a lack of mastery, but the *skipped* answers instead. See technical report for more results [6].

Hierarchical Skill Difficulty Model

Figures 3 and 4 capture the idea that the skills are not mutually exclusive and may have different difficulty levels. For example, if the skill of identifying a triangle is not mastered, it seems probable that the skill of finding the area of a triangle is not mastered either. In particular, this Hierarchical Skill Difficulty Model is arranged so that each parent node is a skill which should be mastered before its child node can be mastered.

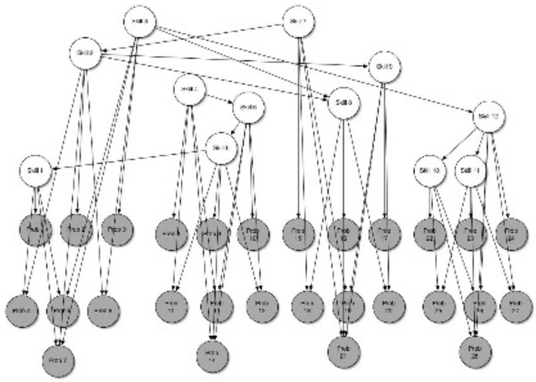


Fig. 3. Bayesian Network for Hierarchical Skill Difficulty Model: The hierarchy is based on the order in which skills should be learned (i.e. a student should know how to identify a right triangle (Skill 7) before learning how to take its area (Skill 8) which should all be mastered before learning the Pythagorean Theorem (Skill 9)). See Figure 4 for a higher level view.

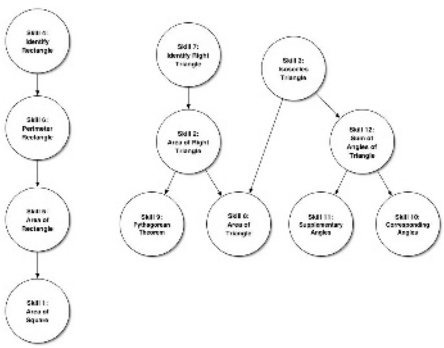


Fig. 4. Higher Level Look at Skill Difficulty Hierarchy Model (hidden skill nodes shown only)

5 Results and Discussion

Flat Skill Model With Simulated Priors. This section describes and compares the results achieved for the different models. As a first sanity check, we created an experiment using hand-coded priors to get baseline conditional probability Tables (CPTs). The structure of the model is identical to that of the flat skill model (Figure 2) only with user-specified parameters for all 40 nodes. We then sampled from this network to create simulated training data and built an identical model with the skill nodes hidden and initialized randomly. Finally, we found the maximum likelihood estimates of the parameters using the generated data on the model with random parameters. We compared the learned parameters to the

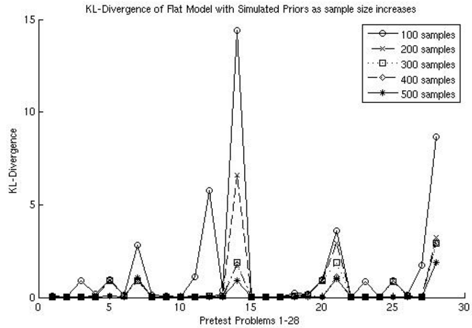


Fig. 5. Kullback-Leibler Divergence across 28 problems for increasing sample size

parameters set in the initial model using the Kullback-Leibler Divergence (see Figure 5). The KL-Divergence is a distance metric used to measure the difference between two probability distributions. We see that the learned parameters are fairly close to the "true" ones. The divergence decreases as the number of samples is increased, but begins to converge at an achievable sample size (400 students). The improvement to estimating the distribution with just 100 more samples is significant. We are in the process of collecting additional data which will increase our model's accuracy.

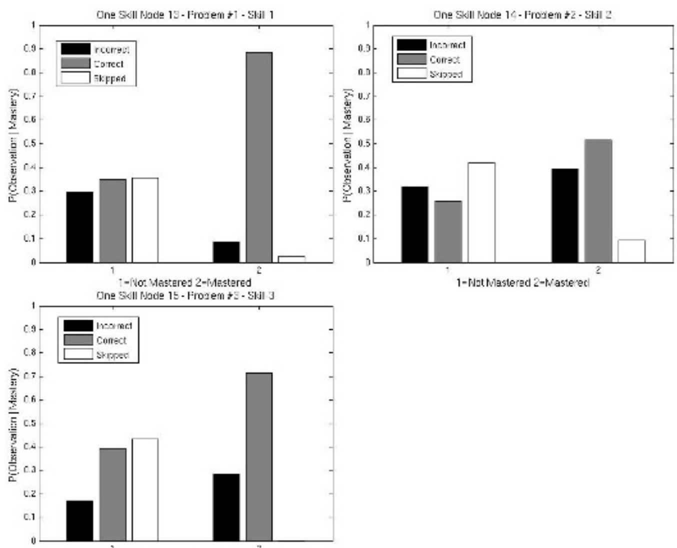


Fig. 6. Reflects the learned conditional probabilities of seeing each of the observations given it is know whether the skill is mastered or not: Flat Skill Model with uniform priors, One-skill problems involving Skills 1, 2 & 3

Flat Skill Model With Uniform Priors
Learned Model
In Figure 6, we consider one-skill problems. We have learned the conditional probabilities of a student getting a specific problem incorrect/correct/skipped assuming we know if the skill is mastered or not. Notice that it is not the *incorrect* answers from which we infer a lack of mastery, but *skipped* instead. This makes sense since the pre-test problems are not multiple choice, so if a student has no mastery of the skill needed, he or she will usually skip it. When the observation is *correct* we can usually infer the skill is mastered and when the observation is *skipped* we can usually infer the skill is unmastered, but *incorrect* can mean either. Perhaps, the student does not have the skill fully mastered and thus answers incorrectly, or he or she may make a simple math error or misunderstand the wording of the problem. For example, the graph on the top left of Figure 6 shows that a correct answer on problem 1 indicates a 90% certainty that Skill 1 (area of a square) is mastered.

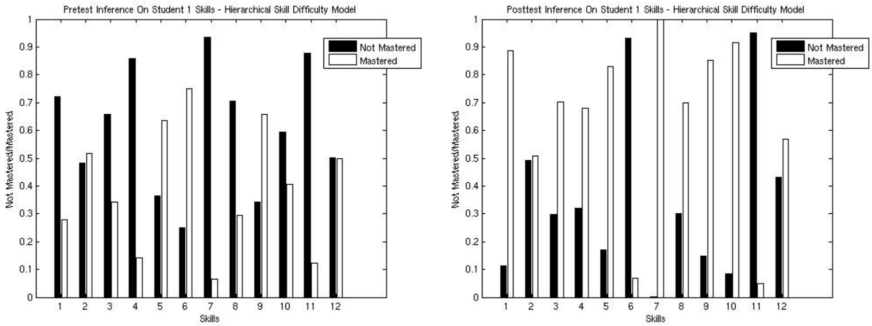
Table 1. Bayesian Information Criterion For Flat and Hierarchical Skill Model (maximum BICs and average BICs are based on 50 random runs)

	Max BIC	Average BIC	Standard Deviation	Variance
Flat Skill Model	−5202.4	−5294.7	47.022	2216.7
Hierarchical Skill Difficulty Model	−4927.7	−5041.1	47.4178	2248.4

Hierarchical Skill Difficulty Model
In Table 1 we see that the Hierarchical Skill Difficulty Model yields the maximum BIC, as well as the highest average BIC. Thus, we conclude that this has the best structure to fit our data. This may be because some sets of skills/problems which were independent previously are now relevant to each other. For example, Skill 2 (area of a right triangle) is actually a subset of Skill 8 (area of a triangle), but in the flat BN these skills are not linked, and are no way dependent on each other. However, in this model, these skills are conditionally dependent on each other. Recall that the skills and their associated problems were originally split up into the following 4 independent sets of skills: {1,2,3}, {4,5,6}, {7,8,9}, {10,11,12}. In this model, these sets of skills are no longer independent so this model is actually more informative than the flat skill model.

Inference. We will look at a student's results to show the inference results of the best model: Hierarchical Skill Difficulty Model. We will look at the observations of the last set of 7 problems involving Skills 10, 11 and 12.

For Student 1 (Figure 7) we see an overall improvement in skill mastery from pre-test to post-test. This aligns with the student's performance on the tests. The test scores are calculated to evaluate individual improvement as follows: *corr/attmp*, the number of problems the students answered correctly divided by the number of problem the student attempted to answer (did not skip). Student 1 got a score of 27 on the pre-test and 83 on the post-test. For most skills,



(a) Student 1 Pretest Skill Mastery Inferred Probabilities

(b) Student 1 Posttest Skill Mastery Inferred Probabilities

Fig. 7. Student 1 Improvement in the probabilities of overall student skill mastery from pre-test to post-test

Student 1 shows a higher certainty of mastery in the post-test than in the pre-test. However, the certainty that Skill 6 (perimeter of a rectangle) is mastered has lowered from pre-test to post-test. This may be because the student did poorly on the post-test problem involving this skill. It is unclear as to whether this assumption should lead to the conclusion of mastery or non-mastery of Skill 6. Notice that Skill 7 (identify right triangle) initially had the most certainty of being unmastered in the pre-test, but shows a high certainty of being mastered in the post-test. This is an easy skill, and this result allows us to assume our tutor does a good job of teaching it. However, Skill 11 (supplementary angles) actually increases in it's probability of being not mastered from pre-test to post-test. This may show that the tutor is not doing a good job of teaching Skill 11, but may also be caused simply by the answers this student supplied on the tests and not the tutor's capability.

6 Conclusions

In summary, a data-centric approach was demonstrated to build a model of how student outcomes in a pre-test are linked to skill mastery. Graphical models were built to infer student mastery of skills, and the Bayesian Information Criterion used to evaluate several models and pick the most accurate one. A hierarchical model that links skills that are dependent on each other gave a better fit than a flat skill model that did not use this dependency information about the domain. In this best fitting model, related skills were linked so that parent nodes represented more basic skills that should be mastered before their more difficult child node skills are mastered (prerequisites).

We can now make predictions not only of how a student will do on a particular problem, but also of how much a student's performance on problems reflects their ability of individual specific skills. With the sparsity of data our current predictions are not optimally accurate, however, the Kullback-Leibler divergence results prove that with a little more data, accuracy will greatly improve. Something interesting learned from this data-centric approach to estimation of skill mastery was the fact that the resulting models automatically found that if a skill is not mastered the student is more likely to skip the problem than answer incorrectly. This makes sense if we think that when students do not know the underlying skills, then they do not even attempt it; meanwhile, if the student has some idea, they probably attempt an answer and give an incorrect response. In general, data-centric models can help reveal valuable information such as this, which was not obvious at first sight. Finally, student improvement from pre- to post-test was demonstrated through raw counts, learned parameters, and inference, again highlighting the positive effect of the Intelligent Tutoring System.

References

1. Arroyo,I., Murray, T., Woolf, B., Beal, C.: Inferring Unobservable Learning Variables from Students Help Seeking Behavior. Intelligent Tutoring Systems (2004)
2. Beal, C., Arroyo, I., Royer, J., Woolf, B.: Wayang Outpost: An intelligent multimedia tutor for high stakes math achievement tests. American Educational Research Association annual meeting, Chicago IL (2003)
3. Conati, C., Gertner, A., VanLehn, K.: Using Bayesian Networks to Manage Uncertainty in Student Modeling. Journal of User Modeling and User-Adapted Interaction (2002), 12, 371-417
4. Corbett, A., Anderson, J., O'Brien, A.: Student modeling in the ACT Programming Tutor. Cognitively Diagnostic Assessment (1995) Hillsdale, NJ: Erlbaum, 19-41
5. Dempster, A., Laird, N., Rubin, D.: Maximization-likelihood from Incomplete Data via the EM Algorithm. Journal of Royal Statistical Society, Series B (1977)
6. Ferguson, K., Arroyo, I., Mahadevan, S., Woolf, B., Barto, A.,: Improving Intelligent Tutoring Systems: Using Expectation Maximization To Learn Student Skill Levels. University of Massachusetts Amherst, Technical Report (2006) TR-2006-09
7. Friedman, N.: Learning Belief networks in the Presence of Missing Values and Hidden Variables. Fourteenth International Conference on Machine Learning (1997)
8. Jonsson, A., John, J., Mehranian, H., Arroyo, I., Woolf, B., Barto, A., Fisher, D., Mahadevan, S.: Evaluating the Feasibility of Learning Student Models from Data. AAAI Workshop on Educational Data Mining (2005) Pittsburgh, PA
9. Jordan, M.: An Introduction to Graphical Models. unpublished book (2001)
10. Mayo, M., Mitrovic, A.: Optimising its behaviour with Bayesain networks and decision theory. International Journal of Artificial Intelligence in Education (2001)
11. Murphy, K.: The Bayes Net Toolbox for Matlab. Computing Science and Statistics (2001) vol. 33
12. Reye, J.: Student Modeling based on Belief Networks. International Journal of Artificial Intelligence in Education (2004) 14, 1-33
13. Schwarz, G.: Estimating the Dimension of a Model. Annals of Statistics (1978)

Automatic Recognition of Learner Groups in Exploratory Learning Environments

Saleema Amershi and Cristina Conati

Dept. of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, BC, V6T 1Z4, Canada
{samershi, conati}@cs.ubc.ca

Abstract. In this paper, we present the application of unsupervised learning techniques to automatically recognize behaviors that may be detrimental to learning during interaction with an Exploratory Learning Environment (ELE). First, we describe how we use the *k*-means clustering algorithm for off-line identification of learner groups with distinguishing interaction patterns who also show similar learning improvements with an ELE. We then discuss how a *k*-means on-line classifier, trained with the learner groups detected off-line, can be used for adaptive support in ELEs. We aim to show the value of a data-based approach for recognizing learners as an alternative to knowledge-based approaches that tend to be complex and time-consuming even for domain experts, especially in highly unstructured ELEs.

1 Introduction

Exploratory learning environments (ELEs) provide facilities for student-led exploration of a target domain with the premise that active discovery of knowledge promotes deeper understandings than more controlled instruction. Through the use of graphs and animations, algorithm visualization (AV) systems aim to better demonstrate algorithm dynamics than traditionally static media, and there has been interest in using them within ELEs to promote interactive learning of algorithms [1,8]. Despite theories and intuitions behind AVs and ELEs, reports on their pedagogical effectiveness has been mixed [5,8]. Research suggests that their pedagogical effectiveness depends upon the way in which these systems are used, which in turn is influenced by distinguishing student characteristics such as meta-cognitive abilities [5] and learning styles [11,8]. For example, some students often find such unstructured environments difficult to navigate effectively and so they may not learn well with them.

Such findings highlight the need for ELEs in general, and specifically for ELEs that use interactive AVs, to provide adaptive support for students with diverse abilities or learning styles. This is a challenging goal because of the difficulty in observing distinct student behaviors in such highly unstructured environments. The few efforts made towards this goal mostly rely on hand-constructing detailed student models that can monitor student behaviors, assess individual needs, and inform adaptive help facilities. This is a complex and time-consuming task that typically requires the collaborative efforts of domain, application and model experts. For example, Bunt et al. [5] hand-constructed a Bayesian network to model student exploration and

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 463 – 472, 2006.

understanding in the Adaptive Coach for Exploration, an ELE for mathematical functions. Model construction required enumerating all possible exploration cases and domain concepts in the form of network nodes, specifying all node interdependencies, and manually estimating multi-valued conditional probability tables for each node. Though the model can be used to provide students with personalized help in exploration, this entire process would have to be repeated for each new application. In the absence of experts or in large applications, the difficulties of constructing such models can be exacerbated.

To circumvent the difficulties in hand-constructing models, some researchers have turned to the field of machine learning [4,2]. Typically, supervised learning algorithms are used to approximate functions that map observable input data to observable output data such as the correctness of student answers [4]. These functions can then predict student behavior and inform adaptive facilities. However, when output values are unobservable, domain experts must resort back to manual labeling to supply them [2]. This is again time-consuming and can be error prone.

We explore an alternative method for informing adaptive support for ELEs by employing *k*-means [6], an unsupervised machine learning technique to recognize patterns of student behaviors that may affect learning. Because ELEs provide unconstrained environments for exploration, the space of user behaviors can be very large, and characteristic behaviors of different learner types may not be obvious or easily observable even by application experts. For this reason we use *k*-means clustering for the automatic, off-line recognition of user groups. Once groups are detected, we analyze similarities within and dissimilarities between them in terms of both learning and interface usage, to show that clustering identifies meaningful patterns along these two dimensions. Then, we show how these distinct learner groups can be used for on-line classification of individual users. The long-term goal is automatic interface adaptation to encourage effective behaviors and prevent detrimental ones.

We begin by discussing related work. Next, we illustrate the ELE and the experimental data we use. Then, we describe and present results on the use of *k*-means for both the off-line identification of learner groups as well as for on-line recognition. We conclude with a summary and suggestions for future research.

2 Related Work

Gorniak and Poole [7] identify several issues concerning the development of sophisticated user or application models for intelligent user interfaces, including limited transferability across applications and effort required to hand construct the models. They address these concerns by learning a stochastic state space model of application use from user interactions with an earlier version of the same pedagogical tool that we use here, the CIspace Constraint Satisfaction Problem applet (see Section 3). The model they propose can predict future user actions, but unlike our work, it does not allow for assessing quality or relevance of these actions for the interaction goals.

Closely related to our work is research in the emerging field of educational data mining (e.g. [3]). For example, in [9] the authors cluster student responses to multiple choice tests and then analyze the clusters to assess student understanding and misconceptions. Our work differs in that we aim to cluster on higher dimensional interaction behaviors as opposed to answers to questions, as ELEs avoid this type of structured learning tasks by nature. In [12], the authors use clustering on behaviors, specifically action frequencies of students using a collaborative learning tool. These clusters provide behavioral summaries to instructors who can then interpret the results to manage student collaborations. In our research, we take into account temporal data as well as activity frequency when clustering interactions. We also take this process one step further by demonstrating how detected clusters can be used to provide automatic, online adaptive support.

3 Experimental Data

The ELE we use as a testbed for our approach is the Constraint Satisfaction Problem (CSP) Applet, one of a collection of interactive AV tools for learning common Artificial Intelligence algorithms called CIspace [1]. Algorithm dynamics are interactively demonstrated on graphs by the use of color and highlighting, and graphical state changes are reinforced through textual messages (see Figure 1 for an example).

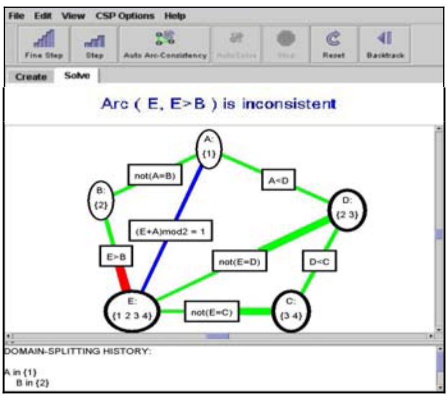


Fig. 1. CSP applet with example CSP

A CSP consists of a set of variables, variable domains and a set of constraints on legal variable-value assignments. The goal is to find an assignment that satisfies all constraints. The CSP applet illustrates the Arc Consistency 3 (AC-3) algorithm for solving CSPs [10] represented as networks of variable nodes and constraint arcs. AC-3 iteratively makes individual arcs consistent by removing variable domain values inconsistent with a given constraint until all arcs have been considered and the network is consistent. Then, if there remains a variable with more than one domain value, a procedure called domain splitting can be applied to that variable to split the CSP into disjoint cases so that AC-3 can recursively solve each case or sub-network.

The CSP applet provides several mechanisms for interactive execution of the AC-3 algorithm, accessible through the toolbar shown at the top of Figure 1 or through direct manipulation of graph elements. These mechanisms include:

- *Fine Stepping.* Cycles through three detailed algorithm steps: selecting an arc, testing it for consistency, and removing variable domain values when necessary.
- *Direct Arc Clicking.* Allows the user to decide which arc to test, and then performs three *Fine Steps* on that arc to make it consistent.
- *Auto Arc Consistency (Auto AC).* Automatically *Fine Steps* through the network.
- *Stop*. Stops *Auto AC*.
- *Domain Splitting (DS).* Allows the user to select a variable domain to split, and specify a sub-network for further application of AC-3.
- *Backtracking*. Recovers the alternative sub-network set aside by *DS*.
- *Resetting.* Resets the CSP network to its initial state.

The data we use for this research was obtained from a previous experiment investigating the effects of studying sample problems with the CSP applet [1]. The experiment typified a study scenario in which students first learn from text based materials, and then study relevant sample problems with the applet. We use the following data collected from 24 students who participated in the study: time-stamped logs of user interactions with the applet, and learning gains computed from pre and post test scores. From the logged data we obtained 1931 actions of users over 205.3 minutes.

4 Behavior Recognition Through *K*-Means Cluster Analysis

Clustering is a class of machine learning techniques used for automatically recognizing patterns in unlabelled data. Clustering operates on data points (feature vectors) in a feature space. Features can be any measurable property of the data. Similarities correspond to distances between data points in the feature space. We use a popular clustering algorithm, *k*-means [6], on our experimental data.

K-means clustering takes as input feature vectors and a user-specified *k* value, and returns *k* clusters of feature vectors. From our logged data, we have 24 feature vectors corresponding to the 24 study participants. Typically the *k* value is determined by intuition about the data or through cross-validation. We experimented with *k* set to 2 and 3 because our data set was relatively small and so we expected to find only a few clear groups with distinct learning outcomes.

Initially, *k*-means randomly selects *k* data points to be the current cluster means. The remaining data points are then assigned to the cluster whose mean minimizes some specified distance metric. Here we minimize Euclidean distances in a 21 dimensional, normalized feature space where the dimensions are the average frequencies of use, and the mean and standard deviations of the pause durations after use of the seven mechanisms described in Section 3. The two latter dimensions have been chosen to capture both the speed of use (which could indicate student attention) and its selectiveness, since varied speed may indicate planned rather than impulsive or inattentive behavior and may not be as detrimental for learning. After all data points are assigned to a cluster, new cluster means are computed from these groupings. The process then repeats for a given number of iterations or until there are little or no changes in the clusters.

K-means can often converge at local optima depending on the selection of the initial cluster means and so several trials are typically executed and the highest quality clusters are used. We executed 20 trials of *k*-means on the data and measured quality by selecting the clusters that resulted in the lowest Euclidean Sum of Squares and highest Euclidean distance between the clusters as the solution groups [6].

Results (*k*=2). A statistically significant[1] (p<.006) difference was found in learning gains (pre to post test improvements) between students in the two clusters found by *k*-means with *k*=2. In terms of practical significance [13], the magnitude of this difference in learning gains is large[2] (Cohen's *d*=1.48).

In order to characterize the different learner groups found, we examined the differences between the groups on each of the 21 dimensions. Figures 2 and 3 show the *frequency* and *pause duration* dimensions that were found to have significant (p<.05) or marginally significant (p<.07) statistical differences, and significant practical differences (Cohen's *d*≥0.8) between the group with high average learning outcomes (HL) and the group with low average learning outcomes (LL).

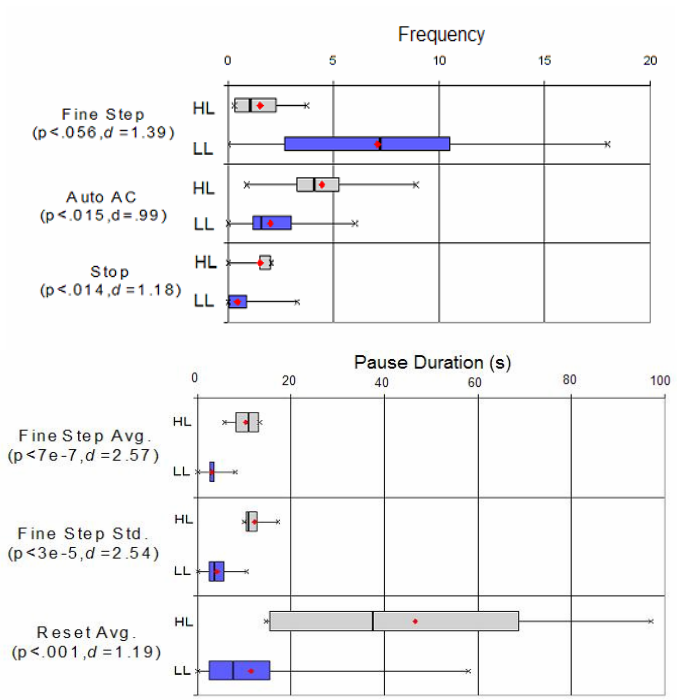


Fig. 2 and 3. Dimensions with significant differences between HL and LL (*k*=2). Fig. 2 (above) shows box plots of frequencies[3], and Fig. 3 (below) shows plots of pause durations.

The results on the use of the *Fine Step* feature are quite intuitive. It is reasonable that students who *Fine Stepped* frequently and consistently too quickly (given by the

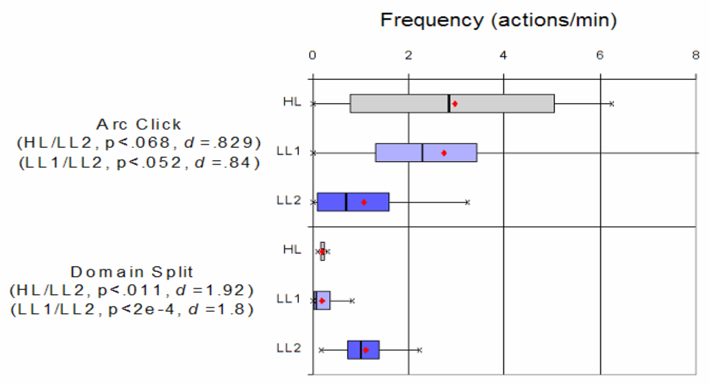
[1] Unless otherwise stated, all tests for significance are one-tailed Student's t-tests.

[2] Cohen's standard suggests *d*=.2, *d*=.5 and *d*=.8 are small, medium, and large effects resp.

[3] *Fine Step* is plotted in actions per minute, whereas *Auto AC* and *Stop* are plotted in actions per 10 minute intervals because *Auto AC* runs AC-3 in its entirety and so fewer *Auto ACs* are typically performed, and *Stop* is used when running *Auto AC*.

combination of low *Fine Step* pause average and standard deviation) may be over using this feature mechanically, without pausing long enough to consider the effects of each *Fine Step*. Such behavior may negatively affect learning, as is evident with the LL group. The higher frequency of *Auto AC* by the HL group in isolation appears unintuitive, but in combination with the higher frequency of *Stopping*, this behavior suggests that students could be using these features to forward through the AC-3 algorithm in larger steps and to analyze it at a coarser scale. The HL group also paused longer after *Resetting* than the LL group. With the hindsight that these students were successful learners, we can interpret this behavior as an indication that they were reflecting on each problem more than the LL group. However, without prescience of learning outcomes, it is likely that an application expert or educator observing the students would overlook this less obvious behavior.

Results (k=3). For *k* set to 3, significant differences in learning gains were found between one group with high learning gains and the two other groups with lower learning gains (p<.014 and Cohen's *d*>1.4 in both cases). We will call these groups 'HL', 'LL1' and 'LL2'. No significant differences in learning gains were found between the two groups with low learning outcomes, suggesting that students may use/misuse pedagogical software in a variety of distinctive ways.



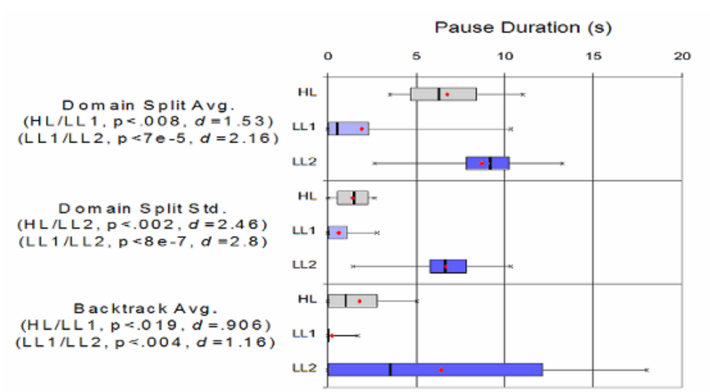


Fig. 4 and 5. Dimensions with significant differences between HL, LL1 and/or LL2 (*k*=3). Fig. 4 (above) shows box plots of frequencies, and Fig.5 (below) shows pause durations.

The same distinguishing behaviors identified by k=2 were replicated[4] between the HL group and both LL groups. For instance, both the LL1 and LL2 group had a significantly higher frequency of *Fine Stepping*, and a significantly and consistently shorter average pause duration after *Fine Stepping* than the HL group. This clustering also revealed several additional patterns, not only between the HL and LL groups, but also between the two LL groups, indicating that k=3 was better at discriminating between different behaviors. Figures 4 and 5 show the additional *frequency* and *pause duration* dimensions for which we found significant/marginally significant statistical and practical differences between the groups.

Interestingly, no differences were found on the frequency of use of the *Arc Click* feature between group HL and LL1, but differences were found along this dimension between HL and LL2 as well as LL1 and LL2. The HL and LL2 difference is reasonable because this feature involves more active engagement on the part of the learner and so using it more often could entail increased learning. However, the LL1 group used this feature comparably as frequently as the HL group but had significantly lower learning outcomes, suggesting that the LL1 students may be using it, but only passively. This is consistent with the passive operation of the *Fine Step* feature exhibited by the LL1 group, not shown in Figure 4 but analogous to the results presented in the previous section. Similarly, the HL group used the *Domain Splitting* feature as frequently as the LL1 group, but paused longer after each split on average. This feature is intended to require thought about efficiency in solving a CSP given different possible splits, and so longer pauses may be needed to thoroughly consider the choices. The LL2 group, however, paused longer and more selectively after *Domain Splitting* than the LL1 group, yet still had low learning gains. The LL2 group is also characterized by significantly longer pauses after *Backtracking* than LL1, and so in this case, the very long pauses may indicate that these students were confused about these applet features or the concepts of domain splitting and backtracking. Once again, these behaviors may be difficult to identify through mere observation.

Discussion. Though our sample size is small, and as a result the power to achieve statistical significance is reduced, k-means is still able to detect groups of users that achieved statistically and, arguably more importantly [13], practically different learning outcomes. And several of the behavioral differences found reasonably explained both effective and poor learning outcomes. However, as expected, some findings were less intuitive, requiring consideration of combinations of dimensions (as k-means does to determine its clusters). This makes interpreting meaningful characteristics a complex task, even for application experts, and highlights the benefits of using clustering to automatically identify learner groups.

5 On-Line Classification Through K-Means Classifier

Understanding the effectiveness of a student's behavior for learning is mostly useful if an ELE can provide adaptive support to improve behavior *while* the student is interacting with the system. Here we discuss the use of a version of k-means for on-line learner classification that can help provide this real time adaptive support.

[4] For space considerations these are not shown in Figs. 4 and 5. Refer to Figs. 2 and 3.

Once *k*-means clustering has found learner groups off-line, an on-line *k*-means classifier can incrementally update a student's classification within these groups as the student interacts with the applet. As interface actions are observed, the student's 21D feature vector is updated to reflect the new observation and classification is computed by simply recalculating the distances between the updated vector and the cluster means. The vector is then assigned to the cluster with the nearest mean.

We use leave one out cross validation (LOOCV) to see how the classifier generalizes to unseen data. We performed a 24 fold LOOCV for both the *k*-means classifiers with *k*=2 and *k*=3. For each trial the training data consists of the *k*-means clusters found off-line (Section 4) with one student removed, and the test data is the logged interface actions of the removed student.

Results (*k*=2 and *k*=3). Figure 6 shows the percentage of correct classifications as a function of the percentage of actions seen by the *k*-means classifiers over time. Note that we should not expect to achieve 100% accuracy after seeing all the actions because we are not re-clustering the data on-line, instead we are classifying incoming data given the clusters found off-line by *k*-means over the data points given by LOOCV. Thus, some error is expected reflecting the possibility of different clusters being found with one data point removed. The trend for *k*=2 suggests that this classifier's accuracy improves as more evidence is accumulated. More notably, the accuracy of the classifier is already around 80% after seeing only 10% of the actions.

It should be noted that there is an unbalance in classification accuracy between the individual clusters. With *k*=2, the accuracy is higher (93.5%) for the group with low learning gains (see Table 1, first row). This means that while this method currently would be very effective in detecting behaviors that eventually result in suboptimal learning, it would more often interfere with learners that show these behaviors sporadically but may eventually be successful. However, the lower accuracy (62.4%) for classification of the high learning gains groups' actions is likely the consequence of our small sample size. Only four students were clustered in this group, and so removing even one student for LOOCV purposes may produce different clusters than those found using all the data. Thus with small data sets, larger clusters may be more stable [6], suggesting that the accuracy of the classifier would increase with more training data. Further investigation is necessary to evaluate this hypothesis, but the fact remains that even with the current limited amount of data, *k*-means with *k*=2 reaches very high accuracy in detecting behaviors that eventually result in suboptimal learning.

Table 1. Classification accuracy for individual clusters. Clusters names appear with the number of members within that group.

	Learner Groups/number of available data points				
Classifier	HL/4	LL/20	LL1/8	LL2/12	Overall/24
k=2	62.4%	93.5%			88.3%
k=3	63.3%		62.1%	85.7%	74.1%

As Figure 6 shows, the trend for *k*=3 initially improves but then dips slightly as student actions are observed. Overall, this *k*-means classifier was able to detect the correct group labels 74.1% of the time (see Table 1, second row). For the high average learning gains group, classification accuracy was 63.3%. For the two low average learning gains groups, the classification accuracies were 62.1% and 85.7% respectively. Group sizes decrease as the number of clusters increase and so the classifier with k=3 faces the same problem as the high learning gains group does for k=2 discussed above. The group with the highest classification accuracy was the group with 12 members, further supporting the hypothesis that lower accuracy for the other groups is an artifact of fewer data points.

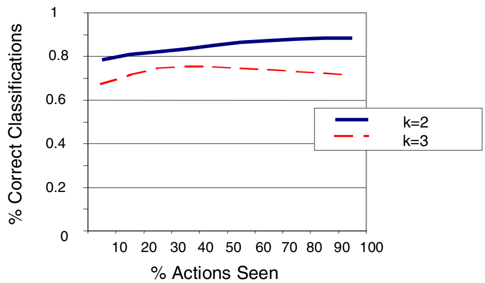


Fig. 6. *K*-means Classifier Performance Over Time

Discussion. From the results of the on-line k-means classifiers we can assert that this technique for classifying student actions appears to be a promising avenue worth further investigations. Despite the limited amount of available data, the classifiers were able to achieve reasonable accuracy, especially in detecting behaviors detrimental for learning, even after seeing only 10% of a student's overall actions. An adaptive ELE could use these classifications for interface adaptations to promote more effective behavior. For example, the ELE could employ a multi-layered interface design, where each layer's features are tailored to facilitate learning for a given learner group. Based on a learner's classification, the ELE could select the most appropriate interface layer. For instance, the ELE may select a layer with *Fine Step* disabled or with a subsequent delay to encourage careful thought for the students in either of our "low learning" groups, or could choose a layer with additional textual explanations of *Domain Splitting* and *Backtracking* for students classified in our LL2 group.

6 Conclusion and Future Work

In this work we have explored a data-based approach to automatic behavioral recognition in an ELE that uses interactive AVs to help students learn constraint satisfaction algorithms. We have described the use of k-means clustering to detect groups of learners with distinct interaction patterns and with significantly different learning outcomes. The clusters identified were then used for on-line behavioral classification. We found that this approach achieved good accuracy even after seeing only a small fraction of student actions, despite the low amount of data available for training.

The next step of this research is to collect more data and verify that this will substantially improve k-means performance. In addition, we plan on experimenting with other unsupervised techniques including a probabilistic version of *k*-means called Expectation Maximization. We also intend to examine how well our approach transfers to other educational applications with different input dimensions including eye tracking and physiological data. Finally, we wish to design an adaptive support facility that takes as input on-line classification information, and empirically evaluate the classifier's effectiveness in a real world setting.

References

1. Amershi, S., Arksey, N., Carenini, G., Conati, C., Mackworth, A., Maclaren, H., Poole, D.: Designing CIspace: Pedagogy and Usability in a Learning Environment for AI. Innovation and Technology in CS Education (2005) 178-182
2. Baker, R.S., Corbett, A.T., Koedinger, K.R.: Detecting Student Misuse of Intelligent Tutoring Systems. Intelligent Tutoring Systems (2004) 531-540
3. Beck, J.E. (ed.): Educational Data Mining: AAAI Workshop. WS-05-02 (2005)
4. Beck, J.E.: Engagement Tracing: Using Response Times to Model Student Disengagement. AI in Education (2005) 88-95
5. Bunt, A., Conati, C., Huggett, M., Muldner, K.: On Improving the Effectiveness of OLEs Through Tailored Support for Exploration. AI in Education (2001)
6. Duda, R., Hart, P., Stork, D.: Pattern Classification. 2nd edn. Wiley-Interscience, NY (2001)
7. Gorniak, P.J., Poole, D.: Building a Stochastic Dynamic Model of Application Use. Uncertainty in Artificial Intelligence (2000) 230-237
8. Hundhausen, C.D., Douglas, S.A., Stasko J.T.: A Meta-Study of Algorithm Visualization Effectiveness. J. Visual Languages and Computing 13, 3 (2002) 259-290
9. Hunt, E., Madhyastha, T.: Data Mining Patterns of Thought. In Beck, J.E. (ed.): AAAI Workshop on Educational Data Mining (2005)
10. Poole, D., Mackworth, A, Goebel, R.: Computational Intelligence: A Logical Approach. Oxford University Press, New York (1998)
11. Stern, L., Markham, S., Hanewald, R.: You Can Lead a Horse to Water: How Students Really Use Pedagogical Software. Innovation and Technology in CS Ed. (2005) 246-250
12. Talavera, L., Gaudioso, E.: Mining Student Data to Characterize Similar Behavior Groups In Unstructured Collaboration Spaces. Workshop on AI in CSCL, European Conference on Artificial Intelligence (2004) 17-23
13. Vicente, K.J., Torenvliet, G.L.: The Earth is Spherical ($p<0.05$): Alternative Methods of Statistical Inference. Theoretical Issues in Ergonomics Science 1, 3 (2000) 248-271

Estimating Student Proficiency Using an Item Response Theory Model

Jeff Johns, Sridhar Mahadevan, and Beverly Woolf

Computer Science Department
University of Massachusetts Amherst
Amherst, MA 01003 U.S.A.
{johns, mahadeva, bev}@cs.umass.edu

Abstract. Item Response Theory (IRT) models were investigated as a tool for student modeling in an intelligent tutoring system (ITS). The models were tested using real data of high school students using the Wayang Outpost, a computer-based tutor for the mathematics portion of the Scholastic Aptitude Test (SAT). A cross-validation framework was developed and three metrics to measure prediction accuracy were compared. The trained models predicted with 72% accuracy whether a student would answer a multiple choice problem correctly.

1 Introduction

Student modeling is defined as the system's belief about a learner's state of knowledge. This is one of the most important aspects of an intelligent tutoring system. Any pedagogical strategy must rely on an accurate model to understand the effect of different tutorial actions on student performance. Student models can be categorized into two broad categories: expert-centric or data-centric [14]. The expert-centric approach, which includes cognitive modeling and knowledge tracing [1, 10], relies on an expert to identify the skills required to solve each problem. The expert provides the structure of the model and possibly the parameters. The data-centric approach relies on using the data to uncover the structure relating student ability to performance. Examples of data-centric student models are structure-learned dynamic Bayesian networks [14], models learned using the Q-Matrix method [6], and Item Response Theory [16, 17] models. Data-centric models typically have far fewer parameters compared to expert-centric models.

In this paper, we evaluate the predictive power of IRT models. A data-centric model was selected to contrast with our previous work [13] using an expert-centric model. From [13], we concluded that robust parameter estimation was difficult given the ratio of the amount of data available from student logs to the model complexity (i.e. number of parameters). IRT models are an attractive alternative because they have a relatively small number of parameters. To confirm this hypothesis, we developed a cross-validation framework to quantify a trained model's predictive accuracy.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 473 – 480, 2006.

2 Item Response Theory

IRT models and their corresponding parameter estimation techniques have a long history of development in the psychometrics literature. The purpose of these models is to probabilistically explain an examinee's responses to test items via a mathematical function based on his/her ability. Assessment of an examinee's ability is the first step of student modeling in an ITS because student state is a prerequisite for creating a pedagogical strategy.

The following two subsections describe the specific model and parameter estimation procedure used in our work.

2.1 Model

IRT posits a static, generative model that relates a student's ability, θ, to his/her performance on a given problem, u_i, via a characteristic curve, $f(u_i \mid \theta)$. A graphical view of this model is shown in Figure 1. Circles represent continuous variables, squares indicate discrete variables, and shaded items are observed variables.

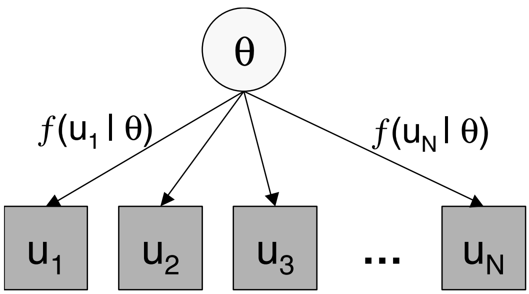


Fig. 1. Graphical depiction of an Item Response Theory model

In this work, we assume θ is drawn from a unidimensional normal distribution with mean 0 and variance 1. Experiments were also conducted with a multidimensional normal distribution, but those studies are not discussed in this paper. The random variables associated with each problem, u_i, come from a Bernoulli distribution with probability of a correct response (1) given by the following parameterized function.

$$P(u_i = \text{correct} \mid \theta) = f(u_i = 1 \mid \theta) = c_i + \frac{1-c_i}{1+\exp(-a_i(\theta - b_i))} \tag{1}$$

This is referred to as the three-parameter logistic equation, where a_i is the discrimination parameter that determines the slope, b_i is the difficulty parameter that determines the location, and c_i is the pseudo-guessing parameter that determines the lower asymptote. A plot of the function, with varying values of the discrimination parameter, is shown in Figure 2. Note that the two-parameter logistic equation is a special case of the three-parameter equation where c_i is set to 0. A more thorough description of the IRT model and the role of each of the parameters can be found in any text on the subject (i.e. [17]).

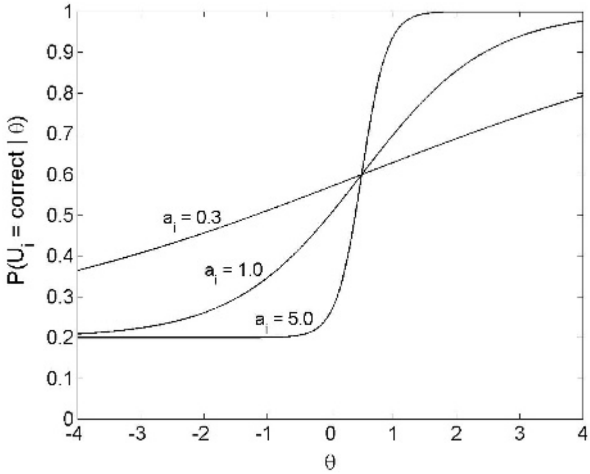


Fig. 2. Three parameter logistic function relating proficiency (θ) to the probability of a correct response. The three curves illustrate the discrimination parameter's effect, $a_i = \{0.3, 1.0, 5.0\}$, while keeping the other parameters constant at $b_i = 0.5$ and $c_i = 0.2$.

2.2 Parameter Estimation

Marginal maximum likelihood estimation [8] is the most common technique used to learn the problem parameters (see [4] for a specific implementation of this algorithm). This is an instance of the expectation-maximization (EM) [11] algorithm where the hidden student variables (θ) as well as the parameters for each problem (a_i, b_i, c_i) are estimated simultaneously. The parameters are chosen to maximize the likelihood of the data.

In the most general case, the three parameters (a_i, b_i, c_i) are assumed to be constants that should be learned from the data. However, it is well known that jointly estimating parameters a_i and c_i can prove difficult. The estimates can be constrained however by assuming the parameters themselves have prior distributions. For example, the discrimination parameter, a_i, can be assumed to come from a lognormal distribution. The prior distribution assumption helps to avoid deviant parameter estimates by shrinking the values toward the specified mean of the distribution.

3 Design of Experiments

Experiments were designed to determine the effectiveness of IRT models at capturing a student's state of knowledge. Multiple experiments were conducted to find the most appropriate modeling assumptions given our dataset.

3.1 Domain and Data

The Wayang Outpost (`http://wayang.cs.umass.edu`) [2, 3] provides web-based tutoring on SAT mathematics problems. The tutor uses multimedia as a tool for engaging students and has been shown to be particularly beneficial for girls. Specifically, the tutor presents multiple choice geometry problems to students and offers them the option to seek help in solving the problems.

Data exists for 401 high school students and 70 multiple choice problems in the Wayang Tutor. Every student completed a minimum of ten problems and each problem was attempted at least thirty times. For each problem and each student, three pieces of information were recorded: number of mistakes made, number of hints requested, and the time spent. Furthermore, the order in which the students finished the problems was tracked. Problems were assigned randomly and a single problem was not given more than once to the same student (note that pairs of very similar problems do exist in the tutor). On average, a student completed 32 of the 70 available problems. The IRT assumption of static student proficiency is justified given this limited interaction with a student. Dynamic IRT models [12] or latent transition analysis models [9] that capture student learning could be used with longer data sequences.

The data was dichotomized because the relatively small sample size does not warrant using polytomous IRT models. A conservative dichotomization procedure was used: a response was labeled as correct only if the student's first action was to click on the correct answer. If the student answered incorrectly or asked for a hint, then the data point was labeled as incorrect.

3.2 Experiments

Four experiments were run with varying assumptions about the parameters in the logistic equation. The first two experiments use the two-parameter logistic equation while the last two experiments use the three-parameter equation. The first and third experiments assume a_i and b_i are constants to be estimated from the data. In the second and fourth experiments, the discrimination parameter, a_i, is assumed to come from a lognormal prior distribution with mean 1.1 and variance 0.6. The mean of 1.1 is a typical value for the discrimination parameter. These two experiments test whether constraints, in the form of prior distributions, help in estimating the parameters. Estimates for a_i that strongly deviate from the prior mean of 1.1 are penalized according to the lognormal distribution. This has the effect of shrinking a_i estimates closer to the mean of the prior distribution. Table 1 summarizes these assumptions.

The pseudo-guessing parameter, c_i, was not estimated during the parameter estimation process in Experiments 3 and 4. Given the small amount of data available, c_i was fixed at a value of 0.2 for each problem. This corresponds to an assumption of uniform guessing as there are five responses for each multiple choice problem.

Table 1. Parameter assumptions for the four experiments

Experiment	a_i	b_i	c_i
1	constant	constant	N/A
2	~ lognormal(1.1, 0.6)	constant	N/A
3	constant	constant	0.2
4	~ lognormal(1.1, 0.6)	constant	0.2

3.3 Validation Framework

Five-fold cross validation was used to evaluate the IRT models learned in each of the four experiments. This means that ~320 students were used to train the model and ~80 students were used to test the model's predictive power. This process was repeated five times by rotating the training and testing populations such that each group of roughly 80 students was used once as the testing population.

Training the model involves running EM to learn the parameters a_i, b_i, and c_i for each problem. The testing procedure involves using the trained model to estimate a student's ability given performance on previous problems, and then to use the model again to predict how the student should fare on the next problem. The predicted response is compared with the actual student response. This is described in more detail in Figure 3.

```
Input:    a_j, b_j, c_j for each problem
          Data (u) for each student in test population
Output:   ACC, MAE, MSE
for i = 1 to (# students in test population)
   // Assume u^i_j refers to the i'th student's response
   // (0 or 1) to the j'th problem he/she attempted
   for j = 2 to (max # problems student i performed)
      θ̂_i = MLE of θ given (u^i_1, a_1, b_1, c_1), ..., (u^i_{j-1}, a_{j-1}, b_{j-1}, c_{j-1})
      p = f(u = 1 | θ̂_i, a_j, b_j, c_j)
      if (p ≥ 0.5)    then û = 1
                      else û = 0
      if (u^i_j == û) then correct += 1
                      else incorrect += 1
      MAE += |u^i_j − p|
      MSE += (u^i_j − p)^2
ACC = correct / (correct + incorrect)
MAE = MAE / (correct + incorrect)
MSE = MSE / (correct + incorrect)
```

Fig. 3. Pseudocode for the cross-validation framework. Note, MLE is short for maximum likelyhood estimate.

Three metrics were evaluated during the testing phase: accuracy, mean absolute error (MAE), and mean squared error (MSE). Accuracy compares the actual response with a predicted response, whereas MAE and MSE compare the actual response with the predicted probability of a correct response. A better model results in higher accuracy and lower MAE and MSE values. MAE and MSE are error metrics that provide a more granular explanation of the model's accuracy than just the accuracy metric. To see this, consider an example where a student answers a problem correctly but the model predicted a 0.49 probability of a correct response. The accuracy metric will have an error of 1.0 whereas MAE will have an error of 0.51 (=1.0 – 0.49) and MSE will have an error of 0.26 (=0.51^2).

4 Results and Discussion

The results from the four experiments are shown in Table 2. Note that all three metrics track with one another; therefore, MAE and MSE do not provide additional insight compared to the accuracy metric for this dataset. However, it is still useful to track these metrics because they provide information on the sensitivity of the model (e.g. a MAE of 0.37 indicates the model is not predicting the probability of a correct response to be close to the extreme values of 0 and 1).

Table 2. Results averaged across the five cross-validation runs

Experiment	Accuracy	MAE	MSE
1	72%	0.37	0.19
2	72%	0.37	0.19
3	67%	0.40	0.23
4	71%	0.38	0.21

Experiments 1 and 2 produced the best average accuracy value of 72%. Both experiments used the two-parameter logistic equation. Experiment 1 assumed the parameter a_i was a constant whereas Experiment 2 assumed a_i came from a prior lognormal distribution. These two experiments were very robust to initial starting conditions for the parameters. Thus, the prior distribution (Experiment 2) did not provide additional lift over Experiment 1.

Experiments 3 and 4 resulted in accuracy values of 67% and 71% respectively. In this case, the prior distribution assumption on the discrimination parameter, a_i, had a significant effect. This occurred because several of the 70 problems had a_i estimates that either became very small (close to 0) or very large (close to the maximum allowable value of 30). The prior distribution helped to shrink some of those extreme values closer to the distribution's mean value.

The 72% accuracy from Experiments 1 and 2 can be compared with two simple baseline strategies:

1. Always predict the student answers incorrectly (i.e. the majority class label).
2. Predict based on a student's percentage of previous problems answered correctly (if percentage is ≥ 0.5, then predict a correct response).

The first strategy achieves 61% accuracy and the second strategy 67%. The 2-parameter IRT model significantly outperforms both baselines. To demonstrate significance, the

Z-statistic was used assuming correct/incorrect predictions are modeled as binomial variables with an alpha value of 0.01 ($Z = 15.11 > Z_\alpha = Z_{0.01} = 2.33$ for strategy 1 and $Z = 6.89 > Z_\alpha = Z_{0.01} = 2.33$ for strategy 2). Accuracy values from 75% to 85% are reported in [15] for training IRT models with synthetic data. However, there are two significant differences between the synthetic datasets and the actual data used for this study. One, the sample size for the synthetic datasets is much larger. Two, there is presumably no off-task behavior (i.e. students "gaming" the system, [5]) in the synthetic datasets. Given these differences, 72% accuracy is a good starting point for modeling the Wayang dataset.

Aside from the accuracy metric, we considered a more intuitive way to gauge the results of training the IRT model. The parameter b_i measures the difficulty level of a problem, where larger values correspond to a problem being more difficult. Another simple measure of difficulty that is not directly related to the IRT model is the percent of students who answered a problem incorrectly. Again, larger values of this metric indicate the problem is more difficult. The correlation between these two measures of problem difficulty across all 70 problems was $r = +0.68$ (using the b_i estimates from Experiment 1). This is a high correlation because the percentage incorrect metric does not account for the different students that did each problem, whereas the IRT model does. The EM algorithm appears to learn realistic values for the difficulty parameter b_i.

5 Conclusions

Dichotomous IRT models were used to estimate a student's proficiency in answering multiple choice questions. The results presented in this paper came from actual data of high school students using the Wayang Outpost, a SAT-style geometry tutoring system. A cross-validation framework was introduced to evaluate the predictive power of the student model.

The best results, which predicted a student's response with 72% accuracy, were achieved using the two-parameter logistic equation. Although the three-parameter equation is more expressive, there was not enough data to effectively learn the values of the parameters. The number of parameters (and thus complexity) of the student model should be determined through a cross-validation process. As more data is gathered over time, the complexity of the model can be incrementally increased. Longer sequences of data would also warrant use of dynamic IRT models that account for student learning.

Our prior research suggests that an expert-centric model must have a large amount of data to learn the parameters of a model with many hidden variables [13]. In contrast, IRT models posit a single hidden variable and a constrained function relating the hidden variable to performance. Based on this study, this data-centric model provided reliable and accurate estimates of a student's proficiency. In the future, we will investigate the relationship between expert-centric models and data-centric models given a finite amount of data from which to learn the model parameters.

We plan to implement the IRT model to estimate a student's proficiency while he/she interacts with the tutor. Different pedagogical strategies will be tested based on the student's proficiency to determine the impact on a student's gain from pretest to posttest. We are also extending the IRT model to capture a student's (unobserved and dynamic) motivation level. Intelligent tutors are in a unique position to measure engagement because they track the number of hints requested and the response time, both key variables in detecting "gaming" behavior. Several recent papers ([3], [7]) have proposed models of student engagement. However, student modeling as a whole will be enhanced by measuring proficiency and engagement in one unified model.

References

1. Anderson, J., Boyle, C., Corbett, A., and Lewis, M. Cognitive Modeling and Intelligent Tutoring. *Artificial Intelligence*, 42(1), 7-49 (1990).
2. Arroyo, I., Beal, C., Murray, T., Walles, R., and Woolf, B. Web-based Intelligent Multimedia Tutoring for High Stakes Achievement Tests. Proceedings of the Seventh International Conference on Intelligent Tutoring Systems, 468-477 (2004).
3. Arroyo, I., Murray, T., and Woolf, B. Inferring Unobservable Learning Variables from Students' Help Seeking Behavior. Proceedings of the Seventh International Conference on Intelligent Tutoring Systems (2004).
4. Baker, F. and Kim, S.-H. *Item Response Theory: Parameter Estimation Techniques*. New York, NY: Marcel Dekker, Inc. (2004).
5. Baker, R., Corbett, A., Koedinger, K., and Wagner, A. Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game the System." Proceedings of the ACM CHI 2004 Conference on Human Factors in Computing Systems, 383-390 (2004).
6. Barnes, T. The Q-Matrix Method of Fault-Tolerant Teaching in Knowledge Assessment and Data Mining. Ph.D. Dissertation. North Carolina State University (2003).
7. Beck, J. Engagement Tracing: Using Response Times to Model Student Disengagement. Proceedings of the International Conference on Artificial Intelligent and Education (2005).
8. Bock, R. and Aitkin, M. Marginal Maximum Likelihood Estimation of Item Parameters: Applications of an EM Algorithm. *Psychometrika*, 46, 443-459 (1981).
9. Collins, L. and Wugalter, S. Latent Class Models for Stage-Sequential Dynamic Latent Variables. *Multivariate Behavioral Research*, 27(1), 131-157 (1992).
10. Corbett, A. and Anderson, J. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *Journal of User Modeling and User-Adapted Interaction*, 4, 253-278 (1995).
11. Dempster, A., Laird, N., and Rubin, D. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B*, 39, 1-38 (1977).
12. Embretson, S. A Multidimensional Latent Trait Model for Measuring Learning and Change. *Psychometrika*, 56, 495-515 (1991).
13. Jonsson, A., Johns, J., Mehranian, H., Arroyo, I., Woolf, B., Barto, A., Fisher, D., and Mahadevan, S. Evaluating the Feasibility of Learning Student Models from Data. *American Association for Artificial Intelligence Workshop on Educational Data Mining* (2005).
14. Mayo, M. and Mitrovic, A. Optimising ITS Behavior with Bayesian Networks and Decision Theory. *International Journal of Artificial Intelligence in Education*, 12, 124-153 (2001).
15. Rudner, L. An Evaluation of Measurement Decision Theory. `http://edres.org/mdt/home3.asp`.
16. Thissen, D. and Wainer, H. (Eds.). *Test Scoring*. Mahwah, NJ: Lawrence Erlbaum Associates (2001).
17. van der Linden, W. and Hambleton, R. (Eds.). *Handbook of Modern Item Response Theory*. New York, NY: Springer-Verlag (1997).

Student Preferences for Editing, Persuading, and Negotiating the Open Learner Model

Andrew Mabbott and Susan Bull

Electronic, Electrical and Computer Engineering, University of Birmingham,
Edgbaston, Birmingham, B15 2TT, UK
{axm891, s.bull}@bham.ac.uk

Abstract. This paper describes a study where students were able to view an open learner model in seven formats. They were provided with tools to edit this model directly, to attempt to persuade the system to change it, or to enter into a negotiation about the model contents. Results indicate that many students are less comfortable having direct control over the content of their learner model than in situations where the system has the final say over proposed changes.

1 Introduction

Previous studies have suggested that offering a choice of views on the learner model may be beneficial [1, 2]. However, these studies focussed on the task of simply viewing the learner model, and did not consider how learners might prefer to interact with the model. Self [3] argues that opening the learner model provides an opportunity for the student to take some responsibility for its content, thus improving its accuracy. Several systems have implemented directly editable learner models e.g. [4, 5] while others have employed a mechanism for the learner to discuss and negotiate over the model contents e.g. [6, 7].

This paper describes a study where three modes of interaction were added to an existing multiple view open learner model. These provide the learner with varying degrees of control over the model, from editing (full learner control) to persuading (full system control). An intermediate level of control is possible in a negotiation mode. The study observed students' use of these interaction modes in conjunction with the views available, in an attempt to determine if the preferences observed for viewing the open learner model are also found in editing, persuading, and negotiating. Student opinions were sought on the usefulness of the three interaction modes.

2 The Flexi-OLM System

The Flexi-OLM system models a learner's understanding of basic C programming based on their answers to multiple-choice and short-answer questions. Seven presentations of the learner model are available: *hierarchy*, a logical grouping of related concepts; *lectures*, where topics are organised the same as in the related lecture course; a *concept map* showing relationships between the topics; *prerequisites,* showing possible sequences for studying topics; *index*, an alphabetical list; *ranked*,

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 481 – 490, 2006.

where topics are listed in order of proficiency; and a textual *summary*. In all views (except summary) the same four-point colour scale is used to indicate the user's understanding of a topic: white for very limited understanding, pale yellow for somewhat limited, yellow/green for moderate, and bright green for excellent understanding. Red is used to indicate topics with possible misconceptions. Clicking on a topic name in the model displays more detailed information about that topic including a breakdown of understanding of specific concepts. Fig. 1 shows this breakdown viewed alongside the concept map. Fig. 2 highlights the important differences in structure of the remaining views.

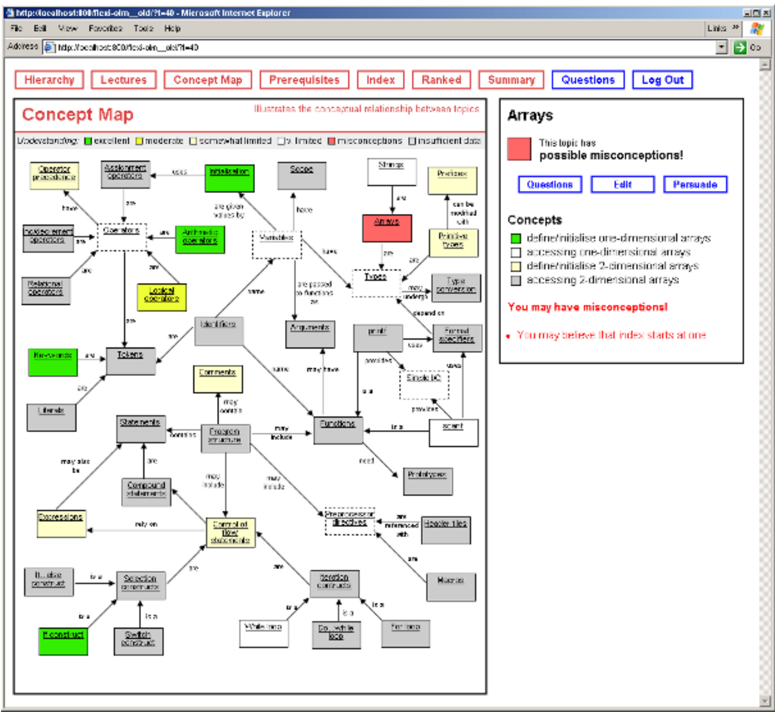


Fig. 1. Concept map view of the learner model and topic breakdown

Using the edit function (Fig. 3), learners can directly control the model of their understanding and are able to change the system's representation of their knowledge level for any concept to whatever they feel appropriate. Situations where a learner may wish to edit their model could include the following: (a) on initially accessing the system, the learner wishes to inform the system about topics they already understand (and hence do not wish to be tested on), (b) the learner suddenly grasps a concept and wants their model to reflect this without having to answer more questions, (c) the learner correctly guesses a series of answers and the system's model therefore shows a higher knowledge level than they believe they have.

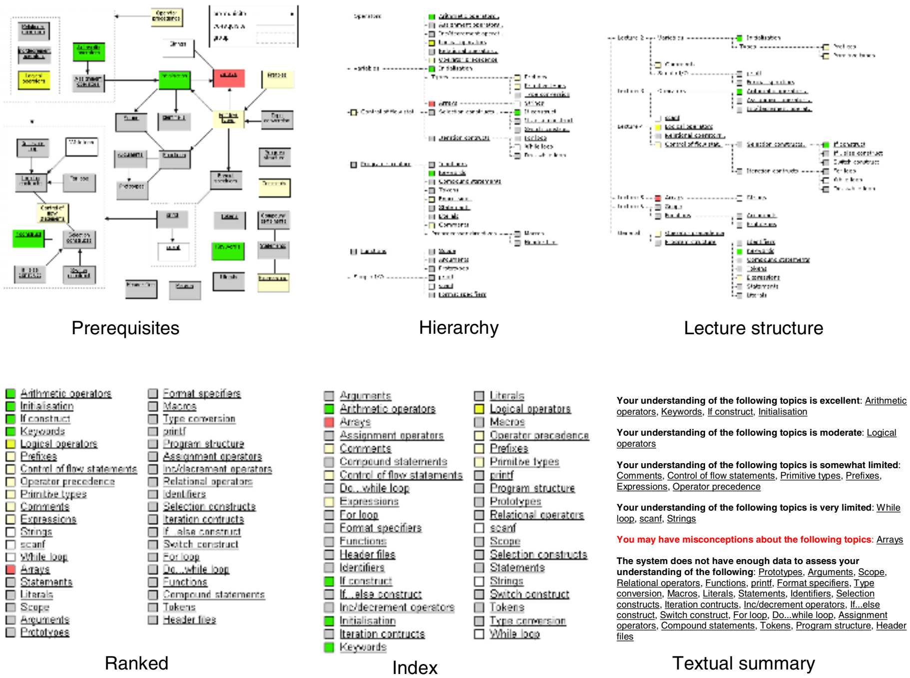


Fig. 2. The pre-requisites, hierarchy, lecture, ranked, index, and summary views

Using the persuade option (Fig. 4) allows learners to register their disagreement with the system's assessment of their level of understanding for a topic, and propose a change to a different level. The system then explains why it believes the current level to be appropriate (by summarizing the learner's understanding of the sub-concepts) and presents evidence supporting these beliefs (by providing samples of the learner's previous responses that may indicate a misconception is held, for example). If the learner still wishes to proceed, they have the opportunity to 'persuade' the system to change their model by answering a series of test questions. Situations where a learner may use the persuade mode could include the following: (a) they believe their knowledge may be higher or lower than the system asserts, but lack the confidence to edit it unchallenged, or (b) they seek the satisfaction of proving the system wrong.

In an amended version of Flexi-OLM, a chat window was added to allow users to negotiate changes to their learner model (Fig. 5) through a 'Wizard-of-Oz' style interaction simulating a chatbot. Discussion was constrained by a protocol [8] based on the negotiation model of Mr Collins [6], maintaining separate belief models for system and learner, following Baker's notion of interaction symmetry [9], ensuring that the same dialogue moves are available to both parties. Each party (a) has full control over their own beliefs, (b) can challenge the other's belief, (c) can seek justification for the other's belief (d) may request justification before changing their own beliefs, and (e) may ultimately decide to leave their belief unchanged. On the system's part, justification is provided using the learner's past responses. The system will accept the learner's suggestion if the difference between their beliefs is one level

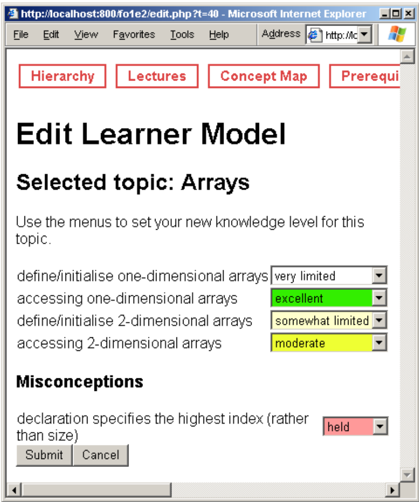


Fig. 3. Edit interface

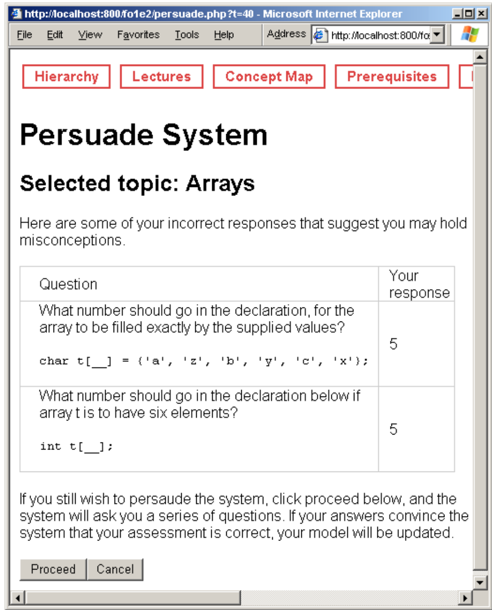


Fig. 4. Persuade interface

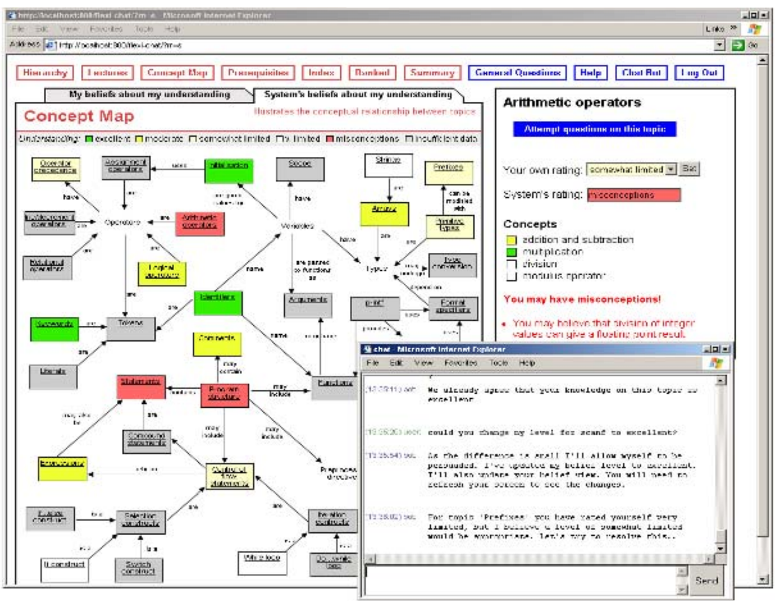


Fig. 5. Learner model and 'chatbot' window used for negotiation

(on the scale of four levels: 'very limited', 'somewhat limited', 'moderate', and 'excellent'). If the difference is two levels a compromise is offered (of changing both beliefs by one level), but if the difference is three levels the learner will be asked to support their claim by answering questions. Tabs at the top of the page allow the learner to switch between viewing their own beliefs and the system's (Fig. 5), aiding the identification of topics where there is disagreement. The aim is to reach agreement across all topics on the student's level of understanding, although the protocol allows the system to tolerate a discrepancy of a single level per topic. If greater discrepancies are present, the system will allow a short time for the learner to propose a topic for negotiation before itself selecting the topic with the greatest discrepancy and attempting resolution by initiating a negotiation.

3 The Study

A study was conducted using to compare students' use of the edit, persuade, and negotiate modes of interaction.

3.1 Participants, Materials, and Methods

Participants were 8 third-year undergraduate students from the department of Electronic, Electrical, and Computer Engineering at the University of Birmingham, who had all previously completed a course on C Programming. The students were introduced to Flexi-OLM (with edit and persuade functions) in a laboratory session, as part of a course on Interactive Learning Environments. They were asked to begin in 'test' mode, where the system attempts to fill the model as quickly as possible by asking questions on all topics. After building an initial model, they were told to explore the system more freely, spending around an hour viewing the model, answering questions, and using the edit and persuade functions as they wished. Immediately following this interaction students submitted a questionnaire concerning their use of the views and the edit/persuade functions.

The negotiated learner model version of Flexi-OLM was conducted as a 'Wizard-of-Oz' [10] style experiment, with human experimenters performing the role of the system for the negotiation parts – a fact not revealed to the students until afterwards (see [8] for further details). The system's initial belief model was constructed using the student's responses from the first part of the study, while the student's own model was elicited from a self-assessment completed at the start of the interaction. Participants were given a summary of the dialogue moves available in negotiation and asked to spend at least 20 minutes interacting with the system and their learner model, before completing a second questionnaire.

3.2 Results

Table 1 indicates the number of edit, persuade, and negotiation 'episodes' performed by each user. An edit episode involves the user adjusting their knowledge level for one or more concepts or misconceptions within a particular topic, while a persuade episode describes any situation where the user challenges the system over the model (regardless of whether they successfully effect a change). A negotiation episode begins when either system or student propose a topic for discussion, and ends with the student's last utterance on that topic.

All users except one edited their model at least once, with the maximum number of edits performed being 6. The number of persuade episodes for each user varied between 1 and 14, and the number of negotiation episodes between 2 and 9. Two users did not attempt to initiate a negotiation, while the largest number of user-initiated discussions was 7.

Table 1. Number of episodes of editing, persuading, and negotiating experienced by users

User	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Edit	6	4	4	2	2	1	1	0
Persuade	2	4	5	14	1	4	3	3
Negotiate (user initiated)	7	4	4	1	2	5	0	0
Negotiate (system initiated)	2	0	0	1	1	0	4	3
Negotiate (total)	9	4	4	2	3	5	4	3

Figure 6 indicates the number of times each user inspected each of the seven views across both parts of the experiment. Of the 8 users, only 2 (users 5 and 3) made similar use of all the views. One user (user 6) appeared to use a single view almost exclusively, while for the rest, usage was more varied. Importantly, the view receiving the most use varied from individual to individual with 5 views (all except the index and summary) being favoured at least once.

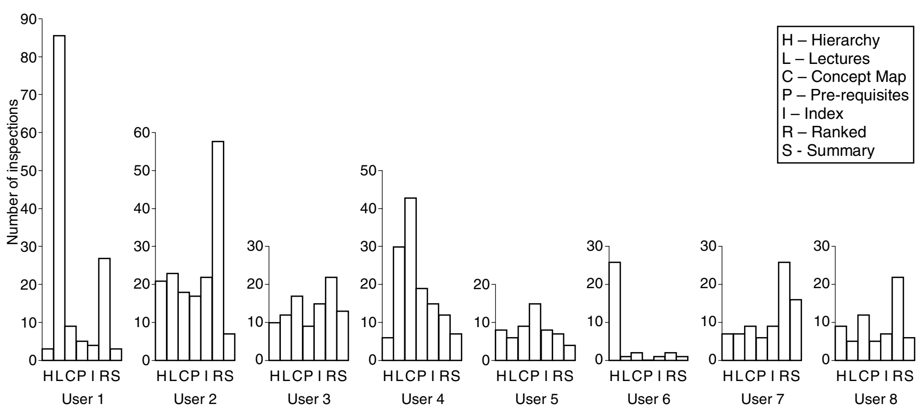


Fig. 6. The number of inspections of each view made by each participant

Table 2 shows the students' responses to questionnaire statements. A five-point scale was used, with 5 indicating strong agreement and 1 indicating strong disagreement. Each participant's responses are shown in a separate column, with the average shown in the final column. All students found viewing, persuading, and negotiating the model to be useful activities. Responses regarding editing were mixed, with 2 students finding it useful, 3 indifferent, and 2 not finding it useful. All students would be likely to persuade or negotiate in situations where they disagree with the model content, but only 2 would be likely to edit.

Table 2. Questionnaire statements and student responses

Part A									**avg**
Viewing my learner model was useful	4	5	5	4	4	5	5	5	4.6
Editing my learner model was useful	3	2	4	5	2	2	3	5	3.3
I would always edit my learner model if I disagreed with it	3	5	2	5	1	1	3	3	2.9
Attempting to persuade the system to change my learner model was useful	4	4	5	5	5	5	5	5	4.8
I would always challenge my learner model (try to persuade the system) if I disagreed with it	4	5	4	5	5	5	5	5	4.8
Part B									**avg**
Negotiating with Flexi-OLM about my learner model was useful	5	4	4	4	4	5	5	4	4.4
I would always negotiate about my learner model if I disagreed with the system.	5	5	5	5	5	5	5	5	5

Table 3. Selected comments from questionnaires

Editing
a) "Overriding what the system is telling you without proving yourself to be right seems to be missing the point of a learner model"
b) "allowed a way of cheating to improve your model"
c) "I think editing should only be decremental and not incremental as I may get some questions right by fluke, but if I know the subject matter there should be no reason why I should get the question wrong"
d) "The edit function in my opinion could be abused... Though when used sensibly saved me the bother of covering topics I know I can do"
e) "This may mean certain users will just edit their models so that they are as good as, or better than, their peers. This will not help their learning and will fool them into thinking they are doing well. "
Persuading
f) "I found the persuade function useful and liked the fact it would test me before changing my knowledge level"
g) "was extremely useful in forcing me to answer further questions and realise that I do lack knowledge in that topic even if I thought I knew it well"
h) "The persuasion function definitely improved my learning as it allowed me to keep persuading until I understood the topic"
i) "The way the system allows you to persuade it to change your level of knowledge is also useful, because it helps point out what is wrong for you a little more which facilitates the user to understand where they are going wrong and derive the correct answer"
Negotiating
j) "Negotiation was a good feature to challenge and learn whilst gain confidence in my ability as well"
k) "Using this [negotiation] version of Flexi-OLM was far less frustrating than using the previous version... I could defend myself much more easily and quickly using chatbot"
l) "I preferred interacting with the chatbot because while I am learning I would prefer to be guided on where my weak areas are. Although that could be done using the views but sometimes you get carried away doing questions not realising you are familiar with that topic, so the chatbot pops up to remind you"
m) "Useful but could sometimes become annoying, too many interruptions"

The comments in Table 3 illustrate some of the reasons given by students for their preferences in Table 2. For editing and persuading, these also include comments from students who did not participate in the negotiation part of the study, and so whose data has otherwise not been included in this paper.

The comments about editing suggest many students view the system as an assessment tool where understanding must be demonstrated and directly editing the model subverts this process (e.g. comments a, b, and c). Despite the fact it is not possible for tutors or peers to access an individual's learner model, some students were still concerned with their peers being able to set an artificially high knowledge level (e.g. d and e). Interestingly, even students who could identify legitimate reasons for editing their own model (e.g. d) did not always believe others would be so responsible, and even suggested restricting functionality (e.g. c) to prevent abuse.

In contrast, students reacted much more positively to persuasion, where all changes to the model must be justified by answering questions, and are thus considered to have been 'earned' (e.g. comment f). Two interesting uses for persuasion are revealed. Firstly students may have insufficient confidence in their own assessment, possibly exacerbated by failed attempts to persuade the system (e.g. g), and are actually attempting to persuade *themselves* that the system's assessment is correct, rather than persuade the system to change its assessment. Secondly, are situations where students do not disagree with the system at all, but are using the persuade facility as a way to improve their understanding through practice while reviewing the evidence presented by the system (e.g. h and i).

Similarly, comments regarding negotiation were mostly positive, for example identifying it as a confidence building tool (e.g. comment j) and in one case finding it less frustrating than persuading (k). The fact that the system could initiate negotiation (in contrast to editing and persuasion which are user initiated) was viewed as useful by some users (e.g. l) and annoying by others (e.g. m).

3.3 Discussion

Overall, students found viewing the learner model to be useful, with all giving agreement scores of either 4 or 5 (Table 1). The interaction data (Fig. 6) supports previous findings [1, 2] that learners have preferences for views in the learner model, and the high scores in the questionnaires for usefulness of viewing the model suggest no difficulty in selecting appropriate views to use.

Only 3 of the 8 participants responded positively to the question of whether editing the model was useful, with comments suggesting many users perceive Flexi-OLM as an assessment tool and hence view editing as 'cheating'. One reason for this could be that aside from presenting an individualised learner model, Flexi-OLM' adaptivity is limited to selecting appropriate topics to question the user on (if they select 'test mode'). In contrast to a full intelligent tutoring system with teaching material and strategies, Flexi-OLM represents an open learner model in isolation, having the aim of helping learners to identify their problems so that they can work on them themselves. Hence, students may not view maintaining an accurate model as important in shaping the interaction. If the model content was viewed as more crucial to the interaction (i.e. in a system that also performed tutoring) students may have more need for a means of making direct changes to the model (for example, to prevent the system tutoring them on a topic they already understand).

In contrast to editing, all participants agreed (6 strongly) that persuading was useful, and all agreed (3 strongly) that negotiating was useful, suggesting that students prefer situations where changes to the model must be justified. The fact that in negotiation the system will accept a very small change without justification or will offer a compromise on a moderate discrepancy could account for the small difference between the perceived usefulness of negotiating and persuading.

Persuasion and negotiation also appear to have benefits for self-assessment by making it possible for the user to prove the system wrong, and also in fostering a reflective learning cycle where the user can learn by practising questions and continually reviewing the evidence about their understanding.

Negotiation of the model illustrates a dimension of learner control beyond control of the model content – the issue of who is controlling the interaction. While editing and persuading are entirely initiated by the learner, either party can initiate negotiation, potentially allowing the system to encourage users to reflect upon their model where they would not otherwise have done so (see comment l, Table 3). The fact that one user could justify themselves "more easily and quickly" using the 'chatbot' illustrates the flexibility of negotiation compared to persuasion – in negotiation a user can attempt to convince the system to change the model immediately, whereas in persuasion the system will always justify itself first.

Questionnaire responses suggest that students would be very likely to persuade or negotiate with the system if they disagreed with their model, but much less likely to edit (though two users still claimed they would always edit), a consequence of the fact that editing was considered less useful than persuading or negotiating.

4 Summary

Students were provided with means to influence their learner model other than simply answering test questions. Despite arguments for increased learner control brought by editable learner models, our results suggest learners were most comfortable in the situation with little direct control (i.e. persuading, where they could propose changes to the model but had to demonstrate their level of understanding before these were accepted) and relatively comfortable with a small amount of control (i.e. negotiation, where small changes could be accepted by the system without challenge), but not comfortable with full direct control (editing).

Results support previous findings [1, 2] that learners find it useful to be given a choice of how to view the learner model, and have different preferences for which view they find most useful.

Of course, there are important limitations to these results. Firstly, the sample size (8 people) means a larger scale study would be required before strong conclusions can be drawn. Secondly, the participants were relatively computer-literate, so may have less difficulty navigating the multiple-view interface and negotiating with the system than students with less computer experience. Thirdly, the study does not take account of what may happen to an individual's preferences over a longer timescale. Although students were able to explore all the views thoroughly before editing, persuading, or negotiating, they only had time to perform a small number of more complex interactions (for example, some students achieved only 2 or 3 negotiation episodes). Finally, the use of self-report data relies on students' abilities to know what is beneficial to them. Even though they appear to find persuading and negotiating useful, this does not necessarily mean these features actually help them learn. However, at the very least, these results suggest that further studies contrasting editing, persuading and negotiation of the learner model are worthwhile.

Acknowledgement

The first author's research is funded by the Engineering and Physical Sciences Research Council (EPSRC).

References

1. Mabbott, A. and Bull, S.: Alternative Views on Knowledge: Presentation of Open Learner Models, in J.C. Lester, R.M. Vicari & F. Paraguacu (eds), 7th International Conference of Intelligent Tutoring Systems, Springer-Verlag, Berlin Heidelberg (2004) 689-698
2. Papanikolaou, K.A., Mabbott, A., Bull, S. & Grigoriadou, M.: Designing Learner-Controlled Educational Interactions Based on Learning/Cognitive Style and Learner Behaviour, Interacting with Computers: The Interdisciplinary Journal of Human-Computer Interaction (in press)
3. Self, J. A.: Bypassing the Intractable Problem of Student Modelling, in C. Frasson and G. Gauthier (eds.), Intelligent Tutoring Systems: at the Crossroads of Artificial Intelligence and Education, Norwood, NJ (1990) 107-23
4. Kay, J.: Learner Know Thyself: Student Models to Give Learner Control and Responsibility, International Conference on Computers in Education, AACE (1997) 17-24
5. Morales, R, Pain, H. & Conlon, T: From behaviour to understandable presentation of learner models: a case study. Proceedings of the Workshop on Open, Interactive, and other Overt Approaches to Learner Modelling, 9th International Conference on Artificial Intelligence in Education, Le Mans, France (1999)
6. Bull, S. & Pain, H.: Did I say what I think I said, and do you agree with me? Inspecting and Questioning the Student Model, in J. Greer (ed), Proceedings of World Conference on Artificial Intelligence in Education, Association for the Advancement of Computing in Education (1995) 501-508
7. Dimitrova, V.: StyLE-OLM: Interactive Open Learner Modelling. International Journal of Artificial Intelligence in Education, Vol 13 (2003) 35-78
8. Kerly, A. and Bull, S.: The Potential for Chatbots in Negotiated Learner Modelling: A Wizard-of-Oz Study, 8th International Conference on Intelligent Tutoring Systems (in press).
9. Baker, M. J.: Negotiated Tutoring. An Approach to Interaction in Intelligent Tutoring Systems. Unpublished PhD thesis. Open University, Milton Keynes, UK (1990)
10. Kelley, J. F.: An iterative design methodology for user-friendly natural language office information applications, ACM Transactions on Information Systems (TOIS), v.2 n.1, (1984) 26-41

Student Modeling with Atomic Bayesian Networks

Fang Wei and Glenn D. Blank

Computer Science & Engineering, Lehigh University
19 Memorial Drive West, Bethlehem, PA 18015, USA
(001-610)-758-4867, (001-610)-758-4605
faw2@lehigh.edu, glennblank@gmail.com

Abstract. Atomic Bayesian Networks (ABNs) combine several valuable features in student models: prerequisite relationships, concept to solution step relationships, and real time responsiveness. Recent work addresses some of these features but have not combined them, which we believe is necessary in an ITS that helps students learn in a complex domain, in our case, object-oriented design. A refined representation of prerequisite relationships considers relationships between concepts as explicit knowledge units. Theorems show how to reduce the number of parameters required to a small constant, so that each ABN can guarantee a real time response. We evaluated ABN-based student models with 240 simulated students, investigating their behavior for different types of students and different slip and guess values. Holding slip and guess to equal, small values, ABNs are able to produce accurate diagnostic rates for student knowledge states.

1 Introduction

An Intelligent Tutoring System (ITS) that individualizes its feedback can provide more useful help for each student [7][14][16]. Adaptive tutoring needs to model how well the student knows each concept in the problem domain (knowledge level) currently, not just the likely intention of the student's next step in solving a problem. Modeling a student's knowledge level will help the tutoring system provide feedback that clarifies fundamental knowledge such as concepts rather than point out errors in procedural knowledge. An ITS increases learning when it provides real-time feedback along the way where students solve a procedural problem [16]. Real time feedback helps students make progress and avoid hacking, which leads to confusion and leaving the learning environment prematurely.

Student modeling with Bayesian networks can provide information about a student's knowledge level for each granular piece of conceptual knowledge [4][7][15]. But the number of parameters and the updating time for Bayesian networks is in the order of exponential [12][13]. "Probability is not really about numbers; it is about the structure of reasoning" [12]. When researchers used probabilistic reasoning to model students' knowledge state, fewer of them precisely investigated the relationship among causes that explain student solution steps.

Table 1 compares recent work on student modeling using Bayesian networks, focusing on three features. The first two features, *diagnose concepts* and *prerequisites*, indicate whether a system diagnoses knowledge level of concepts or prerequisite

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 491 – 502, 2006.

concepts, from students' performance. The third feature, *real time*, shows whether a system runs in real time in response to student work. Check marks indicate that the authors implemented a feature successfully, while a dash indicates a partial solution.

Table 1. Comparisons between different systems

Author	Context	Diagnose concepts	Prerequi-sites	Real time
Murray (1998)	Desktop associate			√
Vanlehn et al.(2001)	Solve physics problems			√
Millan et al.(2002)	CAT system	√		
Butz et al. (2004)	Web-based learning programming lecture	√	–	
Millan et al.(2005)	CAT system	√	–	
Our work (2006)	OO-design	√	√	√

Murray used a simplified Bayesian network algorithm to run in real time in a desktop associate system [11]. This system selects appropriate tasks for a user instead of helping the user to learn desktop knowledge. It did not diagnose any concepts in the domain, instead modeling one skill at a time, such as how to open an IE browser. Nor did it model any relationship between skills, such as prerequisites.

VanLehn *et al.* modeled multiple rules in a solution graph in ANDES, which is a system that guides students to solve physics problems [1][15][16][17]. In ANDES, the student model predicts a student's next solution step and diagnoses unmastered *rules*, instead of unknown *concepts*. Concepts represent knowledge at a finer granularity than rules. For example, one ANDES rule says that if the velocity of an object is constant, then its acceleration is zero, rather than representing the concepts velocity or acceleration. Furthermore, ANDES does not model prerequisite relationships between rules. The successful real-time feedback in ANDES is to diagnose any error in the current step and to guide a student to the next correct solution step. It does not tell the underlying concepts that the student needs to learn to avoid errors.

Millán *et al.* modeled a relationship that multiple concepts cause students' answers in a Computer Adaptive Test (CAT) system [7]. The system diagnoses students' knowledge level from a list of problems chosen by a computer with either random or adaptive criterion. Millán *et al.* used 60 random problems to get the knowledge level of 14 concepts at the correctly diagnosing rate of 90%. Assuming each problem takes 5 seconds for a student, their system needs about 5 minutes to compute the students' current knowledge state. Later, Carmona *et al.* modeled prerequisite relationships [4]; using 40 problems, their new system attained a correctly diagnosing rate of 91% for 14 concepts. Run-times for diagnosis and update are still quite long.

Butz *et al.* proposed modeling prerequisite relationships using pre-knowledge from a final-exam pool [3]. Their system called BITS is a web-based system that teaches C++ programming lectures. Their student model provides learning sequences adaptive to each student through the lecture materials. The authors did not indicate the accuracy of their results. Furthermore, gathering pre-knowledge from a final exam increases the cost of knowledge acquisition, and may be idiosyncratic to the particular set of students who took the final exam.

As Table 1 shows, researchers are recognizing the importance of prerequisite relationships. But none provide a student model that accurately diagnoses concepts and prerequisites in real time. Our work considers both prerequisite and concept-to-solution-step relationships, and shows how to do so in real time, so that a student model can determine where students need help, as they solve a problem.

In this paper we present a student model to diagnose students' knowledge level in CIMEL-ITS, an intelligent tutoring system that helps beginners learn object oriented design in a CS1 course [2][9][10][18]. This student model provides a refined representation of prerequisite relationships, adds prerequisites to estimate the current students' knowledge level, and guarantees real-time responsiveness using an atomic Bayesian network (ABN). Evaluation results using 240 simulated students show that the ABN can diagnose each student' knowledge level quickly and accurately. It has correctly diagnosed over 93% of 38 concepts after 38 randomly picked distinct problems in the novice object-oriented design domain.

The organization of this paper is as follows: Section 2 describes the knowledge representation for our student model; Section 3 provides a formal definition of an ABN, with theorems limiting the number of parameters to a small constant, and explains the advantages of this approach; Section 4 presents evaluation results for the accuracy of student models using ABNs; and Section 5 outlines our conclusion and future work.

2 Knowledge Representation

According to cognitive science theory, a sound knowledge state should show a highly connected and well-defined structure [1][5]. Students need not only knowledge of individual concepts, but also the relationships between concepts, such as similarity, difference, usage and a-part-of, to build up a sound knowledge state. Our knowledge scheme represents these relationships as explicit nodes in a network.

We use a pair (*ku*,*au*) to model the causal relationship between an immediate concept or knowledge unit and an action step that a student takes to solve a problem. A *ku* is a knowledge unit, which means the knowledge that students need to learn. There are two kinds of *ku*: concept and relation between concepts. For example, relation between concepts can be Attribute_Parameter, which models understanding the difference between concepts Attribute and Parameter (a common confusion for novices). We have observed from preliminary results that students frequently struggle to understand relationships between concepts, such as the difference between Attribute and Parameter (and when to use which), or between integer and double, etc. An *au* is an action unit, which is a single step in a student's solution, e.g. writing a name for an attribute. From the definition of the pair (*ku*,*au*), *ku* directly causes *au*.

As shown in Figure 1, the knowledge units that students need to understand to solve the object oriented problems are modeled in a Curriculum Information Network (CIN) for the student model. All the knowledge units are connected by the prerequisite links. By convention, a prerequisite is a concept that a student needs to understand before understanding another concept. Different teachers may use different curricula which results to a different CIN. So more broadly, any concept one needs to teach before introducing a new concept is also a prerequisite. *Immediate* prerequisites are those concepts that strongly relate to a concept and play the most important role in understanding the concept. The concept Class in our CIN is not the aggregation of concepts Attribute and Method. Instead it represents a category of objects. A student understands the concept of Class if he can identify correct class names.

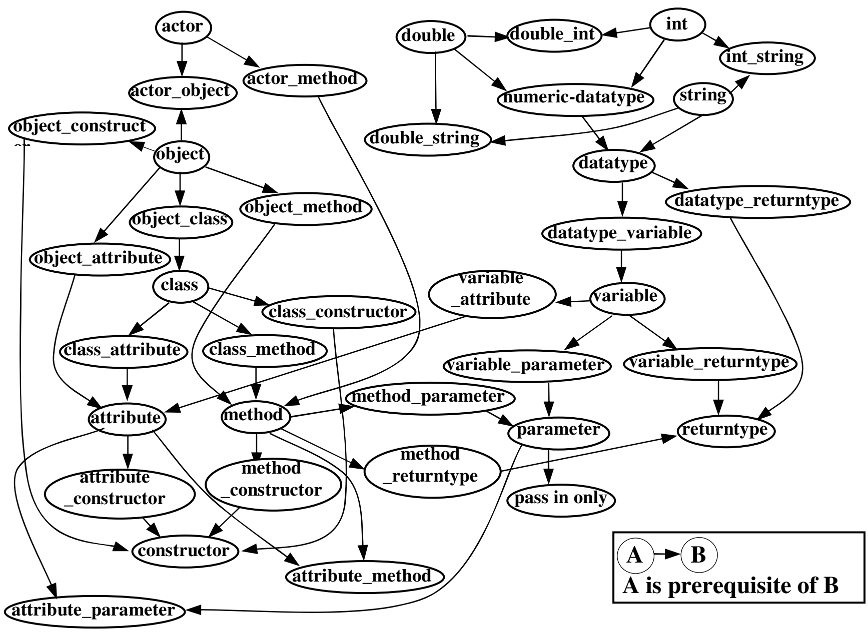


Fig. 1. Curriculum information network (CIN) for the student model

3 Atomic Bayesian Network

An Atomic Bayesian Network (ABN) focuses on just one concept, its immediate prerequisites, and its relationship to a solution step. An ABN models both the causal association between a student's solution and the most relevant concepts, and models prerequisite relationships among these concepts. It indicates that mastery of those concepts causes that whether the student makes the current solution step correctly or not. In other words, an ABN models two kinds of relationships: 1) the student needs to understand a concept at the center of an ABN before he can make the current solution step correctly, and 2) the student needs to understand all of the immediate prerequisites of the center concept before he is ready to understand the center concept.

3.1 Definition of an Atomic Bayesian Network (ABN)

As Figure 2 shows, an ABN is a directed graph composed of one edge (*ku*, *au*) and multiple edges (immediate-prerequisite(*ku*), *ku*), in which *ku* and *au* make a pair (*ku*, *au*). Immediate-prerequisite(*ku*) represents the knowledge units that must be taught right before teaching *ku*. A noisy-and relationship is enforced among all edges (immediate-prerequisite(*ku*), *ku*). Noisy-and is an uncertain relationship which is generalized from the logical AND [7][12]. Whereas logical AND requires that a student understand *all* immediate prerequisites of *ku* before he understands *ku*, noisy-and allows for uncertainty about the knowledge of immediate prerequisites. It assumes that allowing each immediate prerequisite is independent of allowing the others. For example, the concept Numeric-datatype has two immediate prerequisites: Int and Double. Noisy-and assumes that the joint events (numeric-datatype, int) and (numeric-datatype, double) are mutually independent. Given this independence, parameters in the conditional probability table for a noisy-and relationship take the product of the conditional probability values of each parent. For more details please see the proof for equation (1).

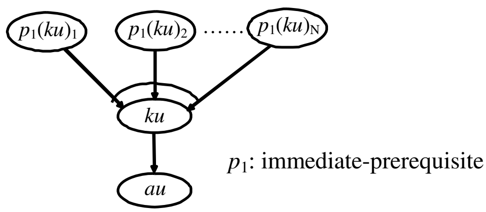


Fig. 2. Atomic Bayesian Network (ABN)

All of the variables (nodes) in an ABN have binary values, true or false. They are defined as follows:

- The variable *au* (the leaf node) represents how a student makes a solution step in a constructive exercise, such as an OO design problem. The value of true means the student makes a correct step, while false means a wrong step.
- The variable *ku* in the center represents if the student knows the most relevant concept for the current solution step. The value is true when the student understands the concept and false otherwise.
- The variable $p_1(ku)$ (the root nodes) represents if a student knows the immediate prerequisites of the center concept. The value is true when the student understands the prerequisite and false otherwise.

When a student understands *ku*, he might still make a wrong step because of a slip or unintentional mistake. Or, when a student does not know *ku*, he might guess the solution correctly. Let's also consider the possibility of errors deriving the center concept from its prerequisites. Even if the student knows all the immediate prerequisites, a student might not understand the center concept. Or a student might guess the correct meaning of the center concept even if he doesn't understand any immediate prerequisite. These characteristics in student learning can be applied to find out the conditional probability tables in an ABN.

Four variables to determine the conditional probability tables in an ABN are formally defined as follows:

- $slip_e$ is the probability a student makes a wrong step when he knows *ku*, [8][11][16][17] where *e* means an evidence: $P(au{=}\text{false}|ku{=}\text{true}){=}slip_e$ or $P(au{=}\text{true}|ku{=}\text{true}){=}1{-}slip_e$
- $guess_e$ is the probability that a student makes a correct step when he doesn't understand *ku* [8][11][16][17]: $P\ (au = \text{true}\ |\ ku = \text{false}) = guess_e$

- $slip_p$ is the probability a student fails to understand the center concept when he knows one immediate prerequisite, where p means a prerequisite (i.e., a slip in the causal relationship from a prerequisite to center concept).
- $guess_p$ is the probability that a student understands the center concept when he doesn't know any immediate prerequisites.

A conditional probability table between the nodes of *au* and *ku* can be calculated from $slip_e$ and $guess_e$ from their above definitions. A conditional probability table between the nodes of immediate-prerequisite (*ku*) and *ku* can be calculated by the definition of $slip_p$, $guess_p$ and noisy-and as

$$P(ku=\text{true} \mid p_1(ku)_i=\text{true},\ldots p_1(ku)_j=\text{false})= \prod_{i\in K}(1\text{-}slip_p)\prod_{j\in \overline{K}} guess_p \quad (1)$$

where if $i\in K$, $j\in \overline{K}$, then $p_1(ku)_i$ =true, $p_1(ku)_j$ =false, $K \cup \overline{K}$ is a set of all immediate prerequisites of the center concept, and $K=\{\ p_1(ku)_i = \text{true} \mid i \in [1,n]\ \}$, a set of immediate prerequisites that the student knows, while $\overline{k}$ is a set of immediate prerequisites that the student does not know.

Proof:
From the definition of conditional probability (t means true, f means false):

$$P(ku = \text{t} \mid p_1(ku)_i = \text{t}, \ldots p_1(ku)_j = \text{f}) = \frac{P(ku = t, p_1(ku)_i = t,\ldots p_1(ku)_j = f)}{P(p_1(ku)_i = t,\ldots p_1(ku)_j = f)}$$

Because the events of knowing immediate prerequisites of a concept are mutually independent, and because noisy-and assumes that joint events of knowing an immediate prerequisite and knowing the concept are also mutually independent, then the conditional probability of a knowledge unit given its prerequisites becomes:

$$\frac{P(ku = t, p_1(ku)_i = t)}{P(p_1(ku)_i = t)} * \ldots\ldots * \frac{P(ku = t, p_1(ku)_j = f)}{P(p_1(ku)_j = f)}$$

Applying the definition of conditional probability again, the conditional probability of an ABN, taking slip and guess into account, becomes:

$$P(ku=\text{t}|p_1(ku)_i=\text{t}) * \ldots * P(ku=\text{t}|p_1(ku)_j=\text{f})=(1\text{-}slip_p) * \ldots * guess_p=\prod_{i\in \boldsymbol{K}}(1\text{-}slip_p)\prod_{j\in \overline{K}} guess_p$$

Every solution step correlates to an ABN which stores the updated value for the ABN of next solution step. If each knowledge unit has at most k immediate parents, and altogether there are n knowledge units in the domain, an ABN will:

- Need O(1) running time instead of $O(2^n)$ for each solution step because it has a small bounded number of immediate parents.
- Update O(1) nodes instead of $O(n)$ nodes for each solution step.
- Determine 4 parameters instead of $O(nk)$ parameters in a noisy-and relationship.

Using an ABN reduces the running time for each step from exponential for a complete Bayesian network to constant time because the ABN only considers its immediate parents, which is a small bounded number for any knowledge domain. The number of conditional parameters drops to 4—two pairs of guess and slip. The number of nodes that must be updated for each step drops to the number of parents. In any domain, the number of immediate parents is much smaller than the number of total variables.

Other tutoring systems using Bayesian networks have resorted to approximate algorithms in order to avoid exponential running time [7][16]. However, approximate algorithms are arbitrary with respect to how much of the network they consider. An ABN is more efficient, and it is sufficient, because the relationship between a center concept and ancestor prerequisites is tenuous at best. For example, from the CIN in Figure 1, Actor is a prerequisite of Actor_Method (a method is a service that an Actor can use), which in turn is a prerequisite of Method. From the solution step focusing on Method (give a meaningful name for the method), an ABN only updates Actor_Method, not Actor, because the prerequisite relation between Actor and Method is intuitively tenuous; it is sufficient just to update the immediate prerequisite. As we shall see, simulation results preliminarily support our claim about the sufficiency of ABNs (experiments with real students to validate ABNs' sufficiency will be performed in our future research).

The use of ABN accelerates the diagnosis in a student's knowledge state because the ABN has a sufficiently accurate model of the student's conceptual reasoning. Table 2 compares the student modeling approaches between considering and not considering prerequisites assuming A and B are prerequisites of C, and S is a student's solution step. These two approaches all start from the initial prior probability values of 0.5 [7]. The table shows that when prerequisites are considered, four out of six posterior probabilities are further away from 0.5, which means our student model is less possible to end with undiagnosed states.

Table 2. Comparison between different student modeling approaches. A, B, C are the related concepts to S, a solution step. A and B are prerequisites of C.

	P(A|S=t)	P(B|S=t)	P(C|S=t)	P(A|S=f)	P(B|S=f)	P(C|S=f)
Millan *et al.*(2002)	0.56	0.60	0.65	0.36	0.28	0.17
Our work (2006)	0.83	0.83	0.75	0.36	0.36	0.04

3.2 Theoretic Framework for ABN

We propose a theory that will reduce the number of parameters that an ABN needs. It does so by discovering useful relationships between $slip_e$ and $guess_e$ and between $slip_p$ and $guess_p$. Theorem 1 and 2 are intended for exercises not involving multiple choice questions.

ABN Theorem 1: Let *ku* be a knowledge unit, *au* be *ku*'s evidence. If the initial prior probabilities for *ku* and *au* are P(*ku*)=0.5 and P(*au*)=0.5 when there is no information available in the domain, then

$$P\,(au = \text{true} \mid ku = \text{true}) + P(au = \text{true} \mid ku = \text{false}) = 1 \quad (2)$$

Both *ku* and *au* have binary value of true or false. A initial prior probability of 0.5 means that initially it is equally likely that the variables take a value of true or false [7][17]. To avoid any spurious bias to the domain, we choose the initial prior probabilities P(*ku*) = 0.5 and P(*au*) = 0.5 when initially we know nothing about *ku* and *au*, whether the student knows the knowledge and whether he can make the solution step correctly or not. After gaining more evidences in the domain, updated posterior probabilities from evidences will change but the conditional probability table of this Bayesian network will not change, i.e. equation (2) still holds.

Proof:
Figure 3 shows the causal relationship between *ku* and *au*. From the process of marginalization [13], the marginal probability of *au* is $P(au = \mathrm{t}) = \sum_{ku} P(au = \mathrm{t}, ku)$

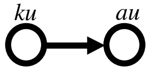


Fig. 3. Causal relationship between ku and au

From the conditioning rule [13],

$$P(au{=}\mathrm{t}){=}\sum_{ku} P(au{=}\mathrm{t}|ku) * P(ku){=}P(au{=}\mathrm{t}|ku{=}\mathrm{t}) * P(ku{=}\mathrm{t}){+}P(au{=}\mathrm{t}|ku{=}\mathrm{f}) * P(ku{=}\mathrm{f})$$

$\because$ $P(ku{=}\mathrm{t}){=}0.5$, $P(ku{=}\mathrm{f}){=}0.5$, *and* $P(au{=}\mathrm{t}){=}0.5$

$\therefore$ $0.5{=}P(au{=}\mathrm{t}|ku{=}\mathrm{t}) * 0.5{+}P(au{=}\mathrm{t}|ku{=}\mathrm{f}) * 0.5$ $\therefore$ $P(au{=}\mathrm{t}|ku{=}\mathrm{t}){+}P(au{=}\mathrm{t}|ku{=}\mathrm{f}) = 1$

ABN Theorem 2: By definition of $slip_e$, P(*au*=false|*ku*=true)=1-$slip_e$, and by definition of $guess_e$, P(*au*=true|*ku*=false)=$guess_e$, then $slip_e{=}guess_e$, where $0 \le slip_e$ or $guess_e \le 1$.

Proof:
$\because$ P(au=t|ku=t)+P(au=t|ku=f)=1 $\therefore$ 1-$\mathrm{slip_e}$+$\mathrm{guess_e}$=1 $\therefore$ $\mathrm{slip_e}$= $\mathrm{guess_e}$

ABN Theorem 3: Let *ku*, $p_1(ku)$, …, $p_k(ku)$ be a concept and its immediate prerequisite, …, and *k* levels of immediate prerequisite. There is only one immediate prerequisite at each level. If the initial prior probabilities of *ku*, $p_1(ku)$, …, and $p_k(ku)$ are P(*ku* = true) = P($p_1(ku)$ = true) = … P($p_k(ku)$ = true) = 0.5, when there is no information available in the domain, then

$$\mathrm{P}(p_k(ku)){=}\mathrm{true}|\, p_{k\text{-}1}(ku)){=}\mathrm{true}){+}\mathrm{P}(p_k(ku)){=}\mathrm{true}|\, p_{k\text{-}1}(ku)){=}\mathrm{false}){=}1 \quad (3)$$

Proof:
Figure 4 shows the causal relationship among *ku* and its ancestors. From the process of marginalization [13], the marginal probability of a particular *ku is* a summation of all other *ku*s in the network, both preceding and following. For example, for node $p_a(ku)$ ($k \le a \le 1$),

$$P(p_a(ku){=}\mathrm{t}){=} \sum_{p_k(ku), p_{k-1}(ku), \ldots p_{a+1}(ku), p_{a-1}(ku), \ldots p_1(ku), ku} P(p_k(ku), p_{k-1}(ku), \ldots p_{a+1}(ku), p_a(ku) = t, p_{a-1}(ku), \ldots ku)$$

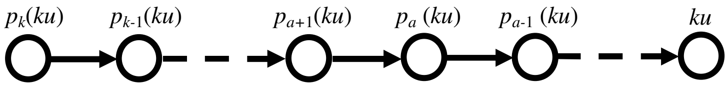


Fig. 4. Relationship between ku and its ancestors

This summation shows how to compute the probability of one node from the entire network. However, producing the above joint probability is computationally hard. So we replace the joint probability above with conditional probabilities for each node, using the Bayesian network formula [13]. Then the summation is equivalent to:

$$\sum_{p_k(ku),p_{k-1}(ku),\ldots p_{a+1}(ku),p_{a-1}(ku),\ldots p_1(ku),ku} P(p_k(ku))P(p_{k-1}(ku)\mid p_k(ku))\ldots P(p_a(ku)\mid p_{a+1}(ku))\ldots P(ku\mid p_1(ku))$$

Next, we can separate the *ku* variables for calculation, as a set of simpler summations inside the summation. Each simpler summation covers fewer *ku* variables. So the expression is equivalent to:

$$\sum_{p_{k-1}(ku),\ldots p_{a+2}(ku),p_{a-1}(ku),\ldots p_1(ku)} \{P(p_{k-1}(ku)\mid p_k(ku))\ldots P(p_{a-1}(ku)\mid p_a(ku)=t)\ldots P(p_1(ku)\mid p_2(ku))$$

$$* \sum_{p_k(ku)} P(p_{k-1}(ku)|p_k(ku))P(p_k(ku)) * \sum_{p_{a+1}(ku)} P(p_a(ku)=t\ |p_{a+1}(ku)) * \sum_{ku} P(ku|p_1(ku))\}$$

Since $\sum_{ku} P(ku|p_1(ku))=1$ no matter what is the value of the variable $p_1(ku)$, and since the conditioning rule that represents marginalization by conditional probabilities, $\sum_{p_k(ku)} P(p_{k-1}(ku)|p_k(ku))P(p_k(ku)) = P(p_{k-1}(ku))$.

We repeat the procedure of separating *ku* variables to reduce the big summation for $P(p_a(ku)=true)$ to a summation of two products, of a conditional probability and a prior probability:$P(p_a(ku)=\mathrm{t}|p_{a+1}(ku)=\mathrm{t})*P(p_{a+1}(ku)=\mathrm{t})+P(p_a(ku)=\mathrm{t}|p_{a+1}(ku)=\mathrm{f})*P(p_{a+1}(ku)=\mathrm{f})$

If the prior probability for each *ku* variable is 0.5, then

$$P(p_k(ku))=\mathrm{t}|p_{k-1}(ku))=\mathrm{t})+P(p_k(ku))=\mathrm{t}|p_{k-1}(ku))=\mathrm{f})=1$$

ABN Theorem 4: By definition of $slip_p$, $\mathrm{P}(p_k(ku)) = true \mid p_{k-1}(ku)) = true) = 1\text{-}slip_p$, and by definition of $guess_p$, $\mathrm{P}(p_k(ku)) = true \mid p_{k-1}(ku)) = false) = guess_p$, then $slip_p= guess_p$, where $0\leq slip_p, guess_p\leq 1$.

The proof of ABN Theorem 4 is similar to proving the Theorem 2.

4 Evaluation

Experimenting with human subjects is expensive and a lot of unexpected factors can make results deviate from what is hypothesized. Large amount of data are needed to prove that the student model really works. Using simulated students avoids problems with limited human subject resources, at low cost, when testing the student model, until it appears to be ready for testing with human subjects. Several researchers apply simulated students to evaluate their student models [4][6][7][17].

One problem with simulated students is how confident can we be that they represent real students. Millán et al. found that adding more pre-knowledge to simulated student improves the evaluation results [7]. Their work shows that analyzing real students is necessary for designing simulated students and can increase confidence in simulation results. The probability that a real student knows a concept will be very small when the student does not know any prerequisites. Following this intuitive rule from observing real students, a simulated student will not know a concept without knowing any prerequisites. We generated 240 students of 6 types for the simulation. Students in the 6 types understand 5, 10, 15, 20, 25 and 30 concepts in the CIN respectively. If a simulated student understands a concept, the knowledge level for this concept is 1, otherwise 0.

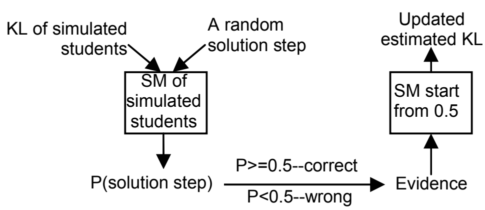


Fig. 5. Illustration of evaluation procedure

Figure 5 illustrates the procedure of our simulation. We distinguish between simulated students, on the left, with pre-determined knowledge levels, and updated student models, on the right, developed as a result of each student step. Following the arrows, the simulation starts from a randomly picked solution step and the pre-defined knowledge state of a simulated student. Using the pre-defined knowledge state as prior probabilities, an ABN calculates the probability of correctly making the solution step. If P(solution step) is greater than 0.5, the model determines that the simulated student made a correct step, and if P(solution step) is less than 0.5, the model infers an error. Thus the model gathers evidence for updating the simulated student's estimated knowledge level. The initial value of an estimated knowledge level for any concept is 0.5. The posterior probability values for concepts in the ABN are stored in a database and are used as prior probabilities for updating another ABN for the next solution step. Our experiment repeats this procedure for 38 randomly picked solution steps, for each simulated student.

There are three diagnosis results for a knowledge unit (*ku*) in the evaluation, known, unknown and undiagnosed. A posterior probability from 0.7 to 1 means the *ku* is known, a posterior probability from 0.3 to 0.7 means undiagnosed, and a probability from 0 to 0.3 means unknown [7]. A *ku* is correctly diagnosed when the updated *ku* is known and the corresponding simulated student does know it, or when the *ku* is diagnosed as unknown and the corresponding simulated student does not know it, by pre-definition.

According to our ABN theorems, $slip_e$=$guess_e$ and $slip_p$=$guess_p$. We evaluated 16 situations to investigate the influence of slip and guess values on the accuracy of the student model. Our hypotheses were 1) that pre-setting slip and guess values could lead to a reliable student model and 2) that varying slip and guess values would affect the accuracy of the student model. Evaluation results are shown in Table 3, where the correct diagnostic rate is the percentage of concepts diagnosed correctly for all simulated students.

The table shows that correct diagnostic rates are higher when $slip_p$, $guess_p$ and $slip_e$, $guess_e$ take the same value. A simulated student model that presets slip and guess values to the same (relatively small e.g. <=0.1) values thus produces estimated knowledge levels with accuracy of at least 93%, confirming our first hypothesis. With respect to the second hypothesis, there is no significant difference when the probability is changed from 0.1 to 0.001. Varying the value does not affect the behavior of the student model, so long as the slip and guess values are relatively small.

Table 3. Correct diagnostic rates with different values of slip and guess

$slip_p$, $guess_p$		0.2	0.1	0.01	0.001
$slip_e$, $guess_e$	0.2	*84.76%*	81.31%	80.40%	79.08%
	0.1	84.43%	*93.18%*	86.22%	83.51%
	0.01	86.01%	88.85%	*94.28%*	91.15%
	0.001	86.04%	88.88%	89.1%	*94.31%*

5 Conclusions and Future Work

Atomic Bayesian Networks are a new approach to student modeling. Each ABN considers a small number of immediate prerequisites, a center concept and an action step. A student model implemented with ABNs uses prerequisites to estimate the current students' knowledge level. ABN theorems show how to reduce the number of parameters for an ABN, which is that if the initial probability for any concepts is 0.5 then $slip_e = guess_e$ and $slip_p = guess_p$ where e represents evidence and p represents prerequisite. By limiting the number of parameters that an ABN needs to combine, it can guarantee real-time responsiveness.

We evaluated the sufficiency of ABNs for student models with 240 simulated students, and also investigated the influence of different slip and guess values. The results show that correct diagnostic rates are higher when $slip_p$, $guess_p$ and $slip_e$, $guess_e$ take the same value. There is no significant difference when slip and guess values are different and relatively small. The high correct diagnostic rates indicate that the student model can correctly diagnose students' knowledge states.

In future work, we will evaluate our approach with human subjects, beginning in the fall of 2006. In addition, ABNs will be enhanced to consider the effects of feedback and historical knowledge, with the goal of a supporting an ITS that evolves in response to student work, in real time.

References

1. Albacete P.L., VanLehn, K.: The conceptual helper: An intelligent tutoring system for teaching fundamental physics concepts. In G. Gauthier, C. Frasson & K. VanLehn (Eds), Intelligent Tutoring Systems: 5th International Conference.: Lecture Notes in Computer Science, Vol. 1839, Springer-Verlag, Berlin Heidelberg (2000) 564-573
2. Blank, G. D., Parvez, S., Wei, F., Moritz, S: A Web-Based ITS for OO Design. Poster for 12th International Conference on AIED, Workshop of Adaptive Systems for Web-Based Education: Tools and reusability, Amsterdam, The Netherlands, June (2005)
3. Butz, C.J., Hua, S., Maguire, R.B.: A Web-based Intelligent Tutoring System for Computer Programming. Proceedings of the IEEE/WIC/ACM Conference on Web Intelligence, (2004) 159-165
4. Carmona, C., Millán, E., Pérez-de-la-Cruz, J.L., Trella, M., Conejo, R.: Introducing Prerequisite Relations in a Multi-layered Bayesian Student Model. User Modeling: Proceedings of Tenth International conference, UM 2005. Edinburgh, UK (2005) 347-356
5. Chi, M.T.H., Koeske, R.D.: Network Representation of a Child's Dinosaur Knowledge. Developmental Psychology 19(1), (1983) 29-39
6. Collins, J.A., Greer, J.E., Huang, S.H.: Adaptive assessment using granularity hierarchies and Bayesian nets. Proceedings of the 3th ITS'96 Conference, Springer-Verlag, Berlin Heidelberg (1996) 569-577
7. Millán, E., Pérez-de-la-Cruz, J. L.: A Bayesian Diagnostic Algorithm for Student Modeling and its Evaluation. User Modeling and User-Adapted Interaction, 12, (2002) 281-230
8. Millán, E., Agosta, J.M., Pérez-de-la-Cruz, J.L.: Bayesian Student Modeling and the Problem of Parameter Specification. Brithish Journal of Educational Technology, 32(2) (2001) 171-181
9. Moritz, S., Blank, G.D.: A Design-First Curriculum for Teaching Java in a CS1 Course. ACM SIGCSE Bulletin archive. Volume 37 , Issue 2 (June 2005) 89-93
10. Moritz, S., Wei, F., Parvez, S., and Blank, G. D. (2005), From Objects-First to Design-First with Multimedia and Intelligent Tutoring. The Tenth ITiCSE, Monte da Caparica, Portugal, June (2005)
11. Murray, W.R.: A Practical Approach to Bayesian Student Modeling. Proceedings of the 4th ITS'98 Conference, Springer-Verlag, Berlin Heidelberg (1998) 424-433
12. Peral, J.: Probabilistic Reasoning in Intelligence Systems: Networks of Plausible Inference. San Mateo, CA: Morgan-Kaufmann (1988)
13. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Upper Saddle River, NJ: Prentice Hall (1995)
14. Shute, V.J.: SMART: Student Modeling Approach for Responsive Tutoring. User Modeling and User-Adapted Interaction, 5, (1995) 1-44
15. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes Physics Tutoring System: Lessons Learned. International Journal of Artificial Intelligence and Education, 15 (3) (2005)
16. VanLehn, K., Niu, Z.: Bayesian student modeling, user interfaces and feedback: A sensitivity analysis. International Journal of Artificial Intelligence in Education, 12(2) (2001) 154-184
17. VanLehn, K., Niu, Z., Siler, S., and Gertner, A.S.: Student modeling from conventional test data: A Bayesian approach without priors. Proceedings of the 4th ITS'98 Conference, Springer-Verlag, Berlin Heidelberg (1998) 434-443
18. Wei, F., Moritz, S., Parvez, S., and Blank, G. D.: A Student Model for Object-Oriented Design and Programming. The Journal of Computing Sciences in Colleges (CCSC), Vol. 20, (2005) 260-273

A Ubiquitous Agent for Unrestricted Vocabulary Learning in Noisy Digital Environments

David Wible[1], Chin-Hwa Kuo[2], Meng-Chang Chen[3],
Nai-Lung Tsao[3], and Chong-Fu Hong[2]

[1] Department of English, Tamkang University, 151 Ying-chuan Road Tamsui,
Taipei County Taiwan
wible45@yahoo.com
[2] Computer and Network Lab, Tamkang University, 151 Ying-chuan Road Tamsui,
Taipei County Taiwan
chkuo@mail.tku.edu.tw, kidd@mail.iwillnow.org
[3] Institute of Information Science, Academia Sinica, 128 Academia Road,
Section 2, Nankang Taipei 115, Taiwan
mcc@iis.sinica.edu.tw, beaktsao@mail2000.com.tw

Abstract. One of the most persistently difficult aspects of vocabulary for foreign language learners is collocation. This paper describes a browser-based agent that assists learners in acquiring collocations in context during their unrestricted Web browsing. The agent overcomes the limitations imposed by learner models in traditional ITS. Its capacity to function in noisy unscripted contexts derives from a well-understood theory of lexical knowledge that attributes a word's identity to its contextual features. Collocations constitute a central feature type, and we extract these features computationally from a 20-million-word portion of BNC. These we are able to detect and highlight in real time for learners in the noisy Web environments they freely browse. Our learner model, derived by semi-automatic techniques from our 3-million word corpus of learner English, maps detected collocations onto corresponding collocation errors produced by this learner population, alerting learners to the non-substitutability of words within the target collocations. A notebook offers a push function for individualized repeated exposure to examples of these collocations in context.

1 Purpose and Motivation

One of the most serious limitations in ITS is the fragility of learner models. A common consequence of a lack of robust learner models for a particular learning domain is that intelligent systems typically impose tight restrictions upon the learners. Only with such restrictions can the learner's behavior become predictable enough to enable the system to respond intelligently (and appropriately) within the scope the system's limited expertise. This is not only the case where learner models are meager or poorly articulated. Often equally limiting are highly articulated learner models because these correspondingly require highly articulated scripts to guarantee that this model can derive the needed inferences from the learner's behavior. Thus, quite generally, learner

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 503 – 512, 2006.

freedom and flexibility are often sacrificed as a prerequisite for expressing the system's intelligence.

Such tight restrictions are especially regrettable in foreign language learning, where the goal is to gain competence in using language to express personal meanings and to understand the meanings expressed by others in a range of contexts. Moreover, one of the richest contexts where learners can be exposed to the target language used for such authentic communication is the Web. Such rich exposure to language input offered by the Web also addresses one of the persistent limitations of traditional classroom foreign language learning: underexposure to the target language in authentic contexts. Unfortunately for system designers, the Web is correspondingly noisy and the sorts of language and contexts that the users may encounter are unpredictable. In earlier work we have referred to this environment as the "digital wild" [13]. The purpose of our recent research has been to develop an approach to designing digital tools sufficiently robust to provide context-sensitive personalized help to language learners in such environments. Here we describe and illustrate here a ubiquitous agent that provides this sort of help for vocabulary learning.

2 Approach

We refer to the overall research framework and infrastructure that we have developed under this browser-based approach with the acronym UWiLL (ubiquitous web-based interactive language learning). The tools reported in this paper build upon the infrastructure of browser-based language tools we recently developed under UWiLL [13]. In this paper we address the limitation that imperfect learner models typically translate into restrictions on the learners. We describe and illustrate an approach which retains both the learners' freedom and the system's responsiveness to them. Here our approach relies upon two fundamental ingredients: (1) a highly articulated yet computationally tractable theory of the target knowledge domain and (2) a correspondingly tractable theory of what it takes to acquire this knowledge. Within our way of framing the problem, once these two key ingredients are in place, the burden on the learner model eases dramatically, to the point where personalization can be achieved in these same noisy conditions with the addition of a relatively simple, straightforward learner model. In what follows we show how this is so.

3 A Computationally Tractable Domain Knowledge Model

A universal assumption in second language acquisition (SLA) research is that exposure to the target language is the *sine qua non* of language acquisition. Yet learners must eventually glean from this exposure a mastery of the target language. Thus, exposure to target language is useful to the extent that the learner can distill from this experience the features of the language that must be mastered, for example, to distill from exposure to English the fact that English requires verbs to agree with their subjects in finite clauses [5][8]. Accordingly, our agent, to function in the noise of the unrestricted Web, must be able to detect in real time within this noise whatever salient linguistic features it is designed to help the learner acquire. Our agent is viable for two reasons. We have an explicit theory of these linguistic features and we have computational tools that can extract them from noisy texts in real time.

4 Collocation and a Theory of Contextual Features

The specific domain of language learning that we target here is vocabulary learning. Thus the purpose of our agent is to help learners increase their mastery of the target language vocabulary in noisy unscripted contexts. Accordingly, our approach, sketched above, requires that our agent be able to detect within these unrestricted contexts precisely those linguistic features that govern the mastery of the target vocabulary. To achieve this, we need a machine-tractable theory of these features.

For this, we subscribe to a contextual view of words. This view assumes that words are, by their very nature, contextual creatures and that mastery of a word consists essentially in mastering the contextual features that govern that word's felicitous use. One of the most widely exploited types of contextual features of words in the computational linguistics literature is collocation (for example,[3][15]; inter alia). Thus, the salient contextual features of the target word that we exploit are the collocating words (or collocates) of that word. This underlying assumption is captured in the famous quote of British linguist J.R. Firth: "You shall know a word by the company it keeps" [4]. Essentially, the collocates of a word are "the company it keeps," that is, a word's collocates are the other words which it conventionally co-occurs with.

Here we motivate the notion of collocation as a fundamental dimension of the contextual features that make up a word's identity. Along the way we show a word's collocates to be (1) features that the learner must eventually master as a key aspect of vocabulary learning and (2) features that we can extract computationally in real time and detect under noisy condition for the learner's attention.

Following Manning and Schütze [7], we refer to the target word of interest as the *focal word* and to the words that this target word selects for its contextualized use as the *collocates* of that focal word. Hence a collocation comprises a pair of words: a focal word and one of its collocates. Part of mastering a noun, for example, is to master its collocates. Lack of this mastery leads learners to produce expressions such as *big rain*, *big wind*, *big respect* ("I have big respect for that coach"). These errors arise from inadequate collocation knowledge. Each of these three nouns, taken as different focal words, imposes different requirements on the selection of its collocates; they each require a different adjective to express the intended meaning: *heavy rain*, *strong wind*, *great respect*. Collocational knowledge is heavily idiomatic. That is, it does not readily generalize (e.g., *heavy rain* but not *heavy wind*; *strong wind* but not *strong rain*). Again, on our view, collocates as contextual features are not secondary aspects of word knowledge; they constitute the very heart of a focal word's identity. We have mentioned a central motivation of our work to be learners' need for adequate exposure to the target language as a means of mastering the features of the target language. Here we can frame this motivation for the particular issue of learning collocations. Specifically, second language learners require sufficient exposure to vocabulary words in context in order to detect and internalize the collocates of these words.

Wang [9] provides empirical evidence that in the particular case of collocation learning in a foreign language, exposure to examples of the target collocation is an effective strategy in helping learners acquire collocations. In fact, exposure to positive examples of a target collocation was dramatically more effective than teacher corrections and comments in helping learners acquire a collocation they had misused. Thus, the central pedagogical strategy of our ubiquitous agent is to draw the learner's attention to collocations detected in their web browsing and then to supplement this highlighting with numerous example sentences containing the detected collocation.

5 The Design of the Ubiquitous Agent: Collocator

We refer to our ubiquitous agent as Collocator. The design of Collocator exploits the free Web browsing of learners to provide the intensive exposure to collocations that is required if those collocations are to be acquired. To do this, the agent detects collocations that occur in the Web pages that the learner freely browses and then, at the request of the learner, highlights any of these detected collocations in their context on the Web page. The rest of the agent's functionality are described in what follows.

There are two versions of Collocator that operate simultaneously: *G-Collocator* (G for Greedy or General Collocator) and *P-Collocator* (P for Picky or Personalized Collocator). G-Collocator runs on an algorithm (to be described below) that detects any word pairs that exhibit a sufficiently strong word association score and treats these pairs as collocations. P-Collocator is more selective, containing a learner model that indicates which collocations have been misused by this learner population and thus require special attention. The design architecture of both G- and P- Collocator are described in what follows.

5.1 Components of the Browser-Based Agent

The schema in Figure 1 represents the components of Collocator, the browser-based agent.

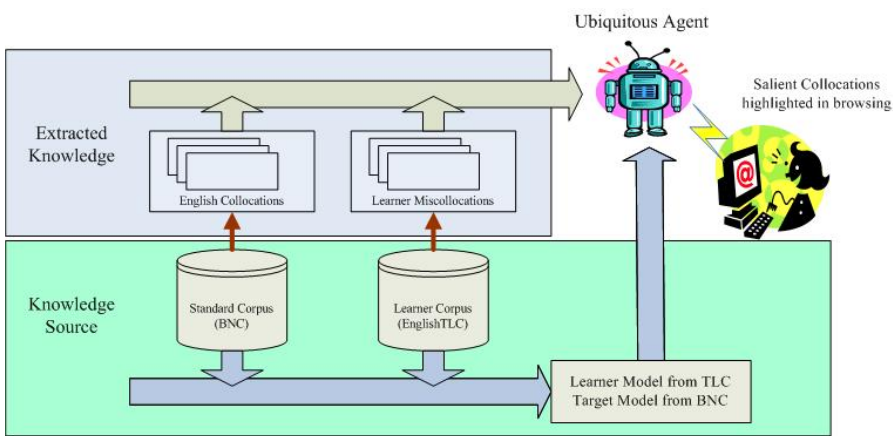


Fig. 1.

Here we describe this architecture schematically and then in the following section provide details of each component and its relation to the overall system.

The schema shows two levels of knowledge. The lower level contains two sources of knowledge that feed the agent; the upper level contains two counterpart sorts of knowledge extracted from these sources. The two knowledge sources represented on the lower level are (1) a standard English corpus (a 20-million-word portion of the British National Corpus, which we have re-indexed for efficient real-time collocation extraction) and (2) our corpus of learner English—*EnglishTLC* (3 million words of English running text produced by Taiwan learners).

From this lower level of knowledge sources we derive the upper level—extracted knowledge of two sorts. The first sort of extracted knowledge is our domain knowledge model consisting of English collocations. These are extracted from BNC through statistical word association measures. We use a traditional mutual information (MI) measure combined with our own variation of this which detects collocations that traditional MI underextracts (See [14]). The extracted collocations then serve as the target knowledge model—standard collocations. These are the collocations detected by G-Collocator. It has no particular learner model, but provides exposure to any collocations detected in the Web pages the user browses. Hence, G in G-Collocator suggests Greedy or General collocation detection.

The second archive of extracted knowledge represented on the upper level of the schema is the relevant learner model used for P-Collocator, for Personalized or Picky collocation detection. This model, derived from our 3-million-word learner corpus, consists of attested miscollocations produced by our population of learners. Miscollocations are errors such as *pay time* (rather than the correct *spend time*) or *learn knowledge* (rather than the acceptable *gain knowledge* or *acquire knowledge*). We use two techniques for mapping these miscollocations attested in our learner model onto the corresponding acceptable collocations found in the domain knowledge model that can be used in their place (for example, mapping the miscollocation *eat medicine* onto the correct collocation *take medicin*e). This mapping is the core knowledge deployed by P-Collocator. The specific target or correct collocations identified by this mapping are what we refer to in the schema as 'salient collocations'. In what follows, we describe the functionality of this agent as it accompanies learners in the context of their unrestricted Web browsing.

5.2 G-Collocator (GC)

GC detects every valuable collocation in browsed web pages. Thus, it is greedy or general. The collocation-extracting scheme is part-of-speech sensitive, which means we have to know the part-of-speech information of each word in browsed web pages. We train a Markov Model-based POS tagger [1] and use British National Corpus (BNC)[1] as our training data. The internal evaluation shows this tagger has 93% precision including identifying unknown words. After part-of-speech tagging, the agent uses the following equation from [14] to measure the word association score:

[1] http://www.natcorp.ox.ac.uk/

$$normMI(x, y) = \log_2 \frac{P(x, y)}{\left(P(x)/sn(x)\right) \cdot \left(P(y)/sn(y)\right)}$$

where x, y mean the word with specific part-of-speech and *sn* means the number of distinct senses for that word listed in WordNet. This is our adaptation of traditional mutual information (MI) in which we take into account the polysemy of the words x and y. In other words our formulation of MI is normalized for the number of senses of x and y. This formulation helps overcome traditional MI's underextraction of collocations with high frequency words. For example, our normalized MI detects *take* as one of the top collocating verbs with the noun *temperature* (as in *The nurse took the patient's temperature*) whereas traditional MI would not detect *take* as a collocate of this noun. (See [14] for details). These word probabilities are extracted from BNC.

All possible pairings of POS-specific words (x and y above) in which the two words appear within a five-word window of each other in our 20 million-words of the BNC are taken as collocation candidates, with each x,y ordered pair yielding a word association score using the above measure. A minimum score threshold is used to select which of the candidate word pairs constitute collocations. This threshold can be lowered or raised to adjust the agent's precision and recall. The collocation knowledge thus extracted from our POS-tagged BNC feeds our browser-based G-Collocator, enabling the agent to detect and highlight collocations that appear in the web pages that the user browses. Figure 2 shows a sample interface with the display of collocations detected by G-Collocator on a specific browsed web page. The detected collocations

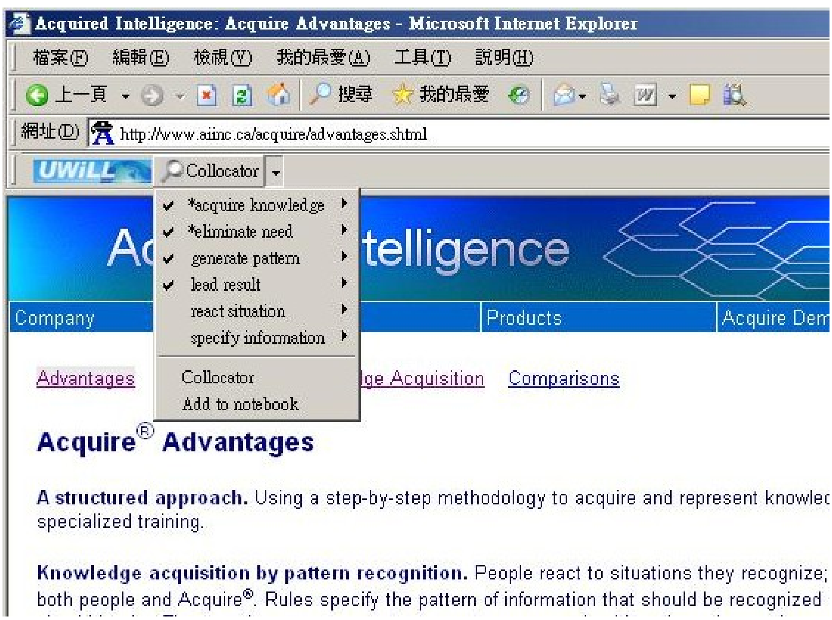


Fig. 2.

are listed on a dropdown menu. Each of these listed collocations then links to further examples of the same collocation from BNC and to a highlight option, which triggers the highlighting of the collocation within the current webpage for the learner's convenience. The check mark to the left of a collocation on the dropdown list indicates the collocations that the user has requested to be highlighted within the web page text.

5.3 P-Collocator (PC)

Collocations typically entail two dimensions of knowledge: (1) knowledge that the two words in a collocation are a conventional pairing, such as *heavy rain* or *strong wind*; and (2) knowledge that the collocate (*heavy* in *heavy rain* and *strong* in *strong wind*) is not freely substitutable, that is, knowledge that the collocation <u>*strong*</u> *wind* can not be paraphrased as *heavy wind* or <u>*big*</u> *wind*. This second aspect of collocation knowledge is sometimes referred to as non-substitutability. We have anecdotal evidence that the learners who grasp the first dimension of a collocation do not necessarily grasp its second negative dimension. Specifically, Wang [9] found in her pretests of her foreign language learners' that a substantial portion of subjects who were able to supply the correct collocate for a specific focal word in a production task also incorrectly judged the counterpart miscollocation to be acceptable as well. The weakness of G-Collocator is that it addresses only the first dimension of collocation knowledge. It conveys to learner that *take medicine* is a collocation whenever this pair is encountered in browsing, but not that *eat medicine* is an unacceptable alternative.

On the assumption that learners require both dimensions of collocation knowledge, the motivation for P-Collocator is to add to G-Collocator the second dimension: relevant unacceptable miscollocations associated with detected collocations indicating the non-substitutability that is not apparent from positive examples alone. To do this, we need a learner model, and for this learner model we need an additional knowledge source: knowledge of the miscollocations that the target learner population produces. On the basis of these attested miscollocations, P-Collocator provides personalized or picky collocation detection (hence, the P-). It does this by piggy-backing on G-collocator's results, adding our learner model and a mapping between the learner model of attested miscollocations and pinpoint domain knowledge, that is, the corresponding correct collocations.

The learner model consists of an archive of attested collocation errors found in our 3-million-word corpus of English produced by learners in Taiwan (called English Taiwan Learner Corpus or EnglishTLC). The pinpoint mapping between this learner model and specific target domain knowledge consists of pairings between each of the collocation errors in the learner model on the one hand and its counterpart correct collocation (or collocations) on the other. Piggy-backing on the collocations that G-Collocator detects, P-Collocator thus is able to determine which of these collocations that have been detected in the current webpage map back to attested miscollocations in the learner model. For example, G-Collocator will detect in a current webpage that *acquire knowledge* constitutes a collocation. P-Collocator can map this collocation onto the learner model and detect that *learn knowledge* and *get knowledge* are attested miscollocations that learners have produced instead of the correct *acquire knowledge.* Next, we describe the design of these main components in this P-Collocator's architecture.

The two main components of P-Collocator knowledge are the learner model and the mapping between standard collocations and their corresponding miscollocations produced by learners. As mentioned above, the learner model is derived from our EnglishTLC learner corpus. We use two methods to extract miscollocations from EnglishTLC. First, since the learner corpus has been partially error-tagged by teachers (See [11]), miscollocations thus tagged serve as one source of the LM. Second, using semi-automatic techniques we bootstrap from these tagged miscollocations to uncover additional, untagged, miscollocations in the learner corpus [6][12].

The second component is the mapping between each of these attested collocations in the LM to the corresponding correct collocations. An example of this would be the miscollocation *pay time* on the one hand and the correct version *spend time* on the other. A portion of these pairings have been provided by hand and designed into regular expression rules that detect and correct learner miscollocations at a 96% precision rate [6]. To supplement these, Wible et al [10] proposed a computational tool called Lexical Assistant which takes as its input attested miscollocations from EnglishTLC and returns an acceptable collocation. They hypothesize that the correct alternative to a miscollocate is likely to be found among the synonyms of that miscollocate or among other semantically similar expressions. For example, for the mistaken expression "Did you <u>eat</u> your <u>medicine</u> yet?" the correct counterpart for 'eat' here, that is, 'take', is indeed a synonym of 'eat' in one sense of 'eat' and in one sense of 'take'. In this respect, the very nature of collocation errors suggests that a valuable source for their correction is the synonym set of the wrong word. Lexical Assistant exploits the data structures of WordNet. Since, WordNet encodes other lexical semantic relations in addition to synonymy, we are able to search WordNet not only for the synonyms of the miscollocate but also for its hyponyms and hypernyms as well in order to systematically expand the set of candidate corrections for the miscollocate.

With this LM of mappings from miscollocations to their correct collocations, P-Collocator not only detects collocations in the noise of the unrestricted Web, it also points out to the user that this particular detected collocation is the correct one that should be used instead of a particular common miscollocation often produced by this population of learner. For example, upon detecting and highlighting *acquire knowledge*, P-Collocator also points out that this is the correct version of the common error *learn knowledge*.

The interface for P-Collocator is illustrated in Figure 3. The entire set of detected collocations detected by G-Collocator appears on a dropdown list from the toolbar. The subset of these detected by P-Collocator is indicated on this same list by the addition of an asterisk * (for example, the top two collocations on the dropdown list in Figure 3—*acquire knowledge* and *eliminate need*). By clicking on any of these asterisked collocations, the user can display P-Collocator's matching of this collocation to the incorrect one often used by this population of learners.

5.4 Notebook Function

The experience of web browsing is notoriously fleeting an ephemeral. In order to allow the collocation input provided by Collocator to take root in the learner's second language competence, there is a need to supplement the exposure that Collocator provides to these collocations that they encounter during browsing on the Web. It is widely acknowledged in the second language vocabulary research that repeated exposure is one of the fundamental requirements that must be met if vocabulary item is to be acquired (See [2] for a review of this literature). To create opportunities for repeated exposure from the fleeting contact with collocations on the Web, we add a notebook function for each learner. The notebook can be displayed on the left of the browser interface. It allows users to store and organize any of the collocations Collocator highlights or the additional example sentences that Collocator provides. It also supports searches for other collocations not encountered during browsing. In addition, as the key to repeated exposure, it offers a "push" function that enables the user to request repeated exposure to a particular collocation over subsequent days.

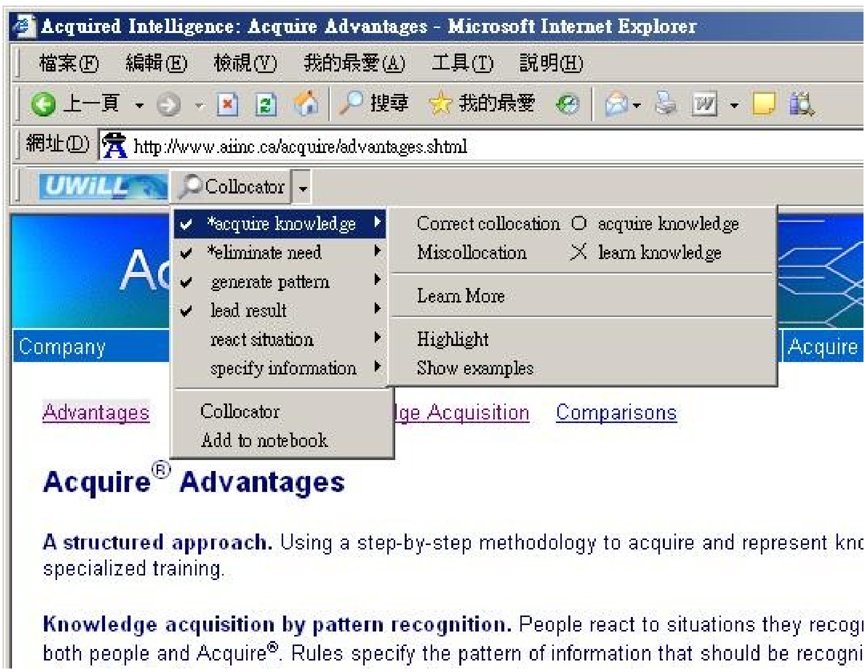


Fig. 3.

6 Future Directions

Collocator supports increased personalization of the LM by referring not only to the aggregate learner corpus for miscollocations, but also to an archive of English written by the individual user to detect errors produced by that learner. This is possible because Collocator is incorporated within the architecture of a larger online platform, IWiLL (See [11]), which automatically archives the writings that learners produce on the platform, for example as writing assignments turned in to a teacher on the platform or writings the learner has posted to any of the discussion boards on the platform. One current limitation of the personalization approach is that the small amount of individual learners' written production causes low recall of that learners' collocation problems. With these individual archives of written production currently in place within the system architecture, however, the effectiveness of this personalization of the LM will grow as individual learners' written production accrues over time.

References

1. Thorsten Brants, "TnT-A statistical part-of-speech tagger," in *Proceedings of ANLP-2000*, Seattle, Washington, (2000).
2. Judie Feng-Yi Chien, *A Study of Input and Second Language Lexical Acquisition*, Master Thesis, Department of English, Tamkang University, Taiwan, (2003).
3. Kenneth Church, William Gale, Patrick Hanks and Donald Hindle, "Using statistics in lexical analysis," in *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, edited by Uri Zernik, 115-164. Lawrence Erlbaum Associates, (1991).
4. J. R. Firth, "A Synopsis of Linguistic Theory 1930-1955" in Studies in Linguistic Analysis, Philological Society, Oxford; reprinted in Palmer, F., (ed. 1968), Selected Papers of J.R. Firth, Longman, Harlow, (1957).
5. Stephen Krashen, *Principles and Practice in Second Language Acquisition*, Cambridge: Pergamon Press (1982).
6. Anne Li-Er Liu, *A Corpus-based Lexical Semantic Investigation of Verb-Noun Miscollocations in Taiwan learners'English*, Master Thesis, Department of English, Tamkang University, Taiwan, (2002).
7. Christopher D. Manning and Hinrich Schutze, *Foundations of Statistical Natural Language Processing*, The MIT Press, (1999).
8. R. Schmidt, "Attention" in P. Robinson (ed) *Cognition and Second Language Instruction,* Cambridge: Cambridge University Press, (2001).
9. Wanju Sonia Wang, *The Effects of Degrees of Explicitness of Automated Feedback on English Learners' Acquisition of Collocations*, Master Thesis. Department of English, Tamkang, Taiwan, (2005).
10. David Wible and Anne Liu, "A Syntax-Lexical Semantics Interface Analysis of Collocation Errors," *PacSLRF (Pacific Second Language Research Forum)*, October , University of Hawaii, Manoa, Hawaii, (2001).
11. David Wible, Chin-Hwa Kuo, Feng-yi Chien, Anne Liu, and Nai-Lung Tsao, "A Web-based EFL Writing Environment: Exploiting Information for Learners, Teachers, and Researchers," in *Computers and Education* vol. 37, pp. 297-315, (2002).
12. David Wible, Chin-Hwa Kuo, Nai-Lung Tsao, Anne Liu, Hsiu-Ling Lin, "Bootstrapping in a Language Learning Environment," in *Journal of Computer-Assisted Learning*, vol 19 #1, pp. 90-102, (2003).
13. David Wible, Chin-Hwa Kuo, and Nai-Lung Tsao, "Contextualizing Language Learning in the Digital Wild: Tools and a Framework," in *Proceedings of IEEE International Conference on Advanced Learning Technologies (ICALT)*, Joensuu, Finland, (2004).
14. David Wible, Chin-Hwa Kuo, and Nai-Lung Tsao, "Improving the Extraction of Collocations with High Frequency Words," in *Proceedings of International Conference on Language Resources and Evaluation (LREC)*, Lisbon, Portugal, (2004).
15. David Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages189–196, (1995).

Learning Styles Diagnosis Based on User Interface Behaviors for the Customization of Learning Interfaces in an Intelligent Tutoring System

Hyun Jin Cha[1], Yong Se Kim[1], Seon Hee Park[1], Tae Bok Yoon[1], Young Mo Jung[1], and Jee-Hyong Lee[2]

[1] Creative Design & Intelligent Tutoring Systems (CREDITS) Research Center
[2] School of Information & Communication Engineering
Sungkyunkwan University, Suwon, Korea
{lois6934, yskim, Seonhp2, tbyoon, ewmhb, john}@skku.edu
http://credits.skku.edu

Abstract. Each learner has different preferences and needs. Therefore, it is very crucial to provide the different styles of learners with different learning environments that are more preferred and more efficient to them. This paper reports a study of the intelligent learning environment where the learner's preferences are diagnosed, and then user interfaces are customized in an adaptive manner to accommodate the preferences. A learning system with a specific interface has been devised based on the learning-style model by Felder & Silverman, so that different learner preferences are revealed through user interactions with the system. Using this interface, learning styles are diagnosed from learner behavior patterns on the interface using Decision Tree and Hidden Markov Model approaches.

1 Introduction

Individual learners have different preferences and learning styles, and these preferences are related with learner behaviors on user interface of learning environments. Thus interfaces that can adapt to each individual's specific preferences would be desired in intelligent learning environments [1].

The objective of this research is to develop an intelligent tutoring system that can diagnose individual's learning styles through learner's behavior patterns on the user interface, and customize its user interface to fit the individual's specific preferences and styles. Felder & Silverman [2] have already performed research on classification of students, development of different tutoring strategies, and the evaluation of learning strategies. By using the learning-style model by Felder & Silverman which is more comprehensive, this study demonstrated a case of learning environment where the learning styles are diagnosed using learner models and the learner's behaviors, and customized user interfaces can be, finally, reconfigured in an adaptive manner to accommodate the learning styles.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 513 – 524, 2006.

2 Learner Model

Chen and Mizoguchi [3] emphasize that a learning system is considered to be “intelligent” if it can adapt its tasks to the learning content based on a learner model, so the learner model is a very important part in intelligent learning systems. Learner model is to be updated according to the analysis in a dynamic manner to provide an adaptive learning environment tailored to each learner. In this research, learner model has been designed: (i) it can provide the tutoring system with all relevant learner information, (ii) it will help in designing a tutoring system which can respond to the learner’s various activities and situations, and (iii) for learning interface adaptation, which is the focus of this paper, it provides a capability to look through the learner’s information and activities, and then extract the appropriate learner aspects for designing the behavior-based user interface customization.

3 Learning Styles and Hypothesized User-Interface Behaviors

The *Index of Learning Style* (ILS) in a learning-style model by Felder & Silverman was adopted in this research as an appropriate category for designing the behavior-based learner diagnosis in that each learning style can be classified into two distinctive preferences [4]. The ILS has four dimensions; Global (G) vs. Sequential (Q) in terms of understanding process of information, Visual (V) vs. Auditory (A) in terms of information input, and Sensory (S) vs. Intuitive (N) in terms of information perception, and Active (C) vs. Reflective (R) in terms of information processing.

The distinctive characteristics in each dimension suggested by Felder & Silverman are described in Table 1. Among them, by using some of the characteristics which can be reflected on user interfaces, learner behavior patterns on learning interfaces were hypothesized for this research.

Table 1. Characteristics of ILS (The characteristics incorporated are annotated with brackets)

Global	Sequential
Jumping directly [G1]	Steady progression [Q1]
Big picture [G2]	Partial materials [Q2]
Visual	**Auditory**
Pictures & Demonstrations [V1]	Words & Explanation [A1]
Sensing	**Intuitive**
Patient with details, Careful but slow [S1]	Bored by details, Quick but careless [N1]
Active	**Reflective**
Work in groups [C1]	Work alone [R1]
Brief discussion or problem-solving activities, Practical [C2]	Occasional pauses for thought, Fundamental [R2]

3.1 Global vs. Sequential (G vs. Q)

The ILS work states that the instructor should offer "the big picture or goal of a lesson" before presenting the learning steps (G2). From this viewpoint, it has been hypothesized that if a learner wants to look through the overview of the contents at the beginning, they may be Global learners. Thus, as seen in Fig. 1, the overview buttons are located on the "table of content" screen for learners themselves to determine to look over the big picture of the learning contents.

Furthermore, Global learners may want to jump to the section (G1) they are interested in by clicking the section hyperlinks on the "table of content" screen rather than following the sequential order that may be preferred by Sequential learners. In main content areas (Fig. 2), Sequential style learners may study in a steady order (Q1) by clicking the "next/previous" buttons, while Global learners may jump to select the content that they want by choosing the "section name" buttons directly.

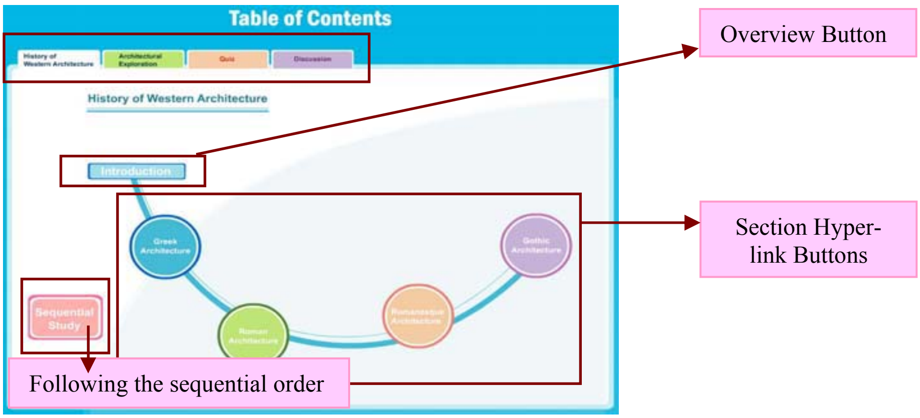


Fig. 1. Interface Layout for Table of Contents

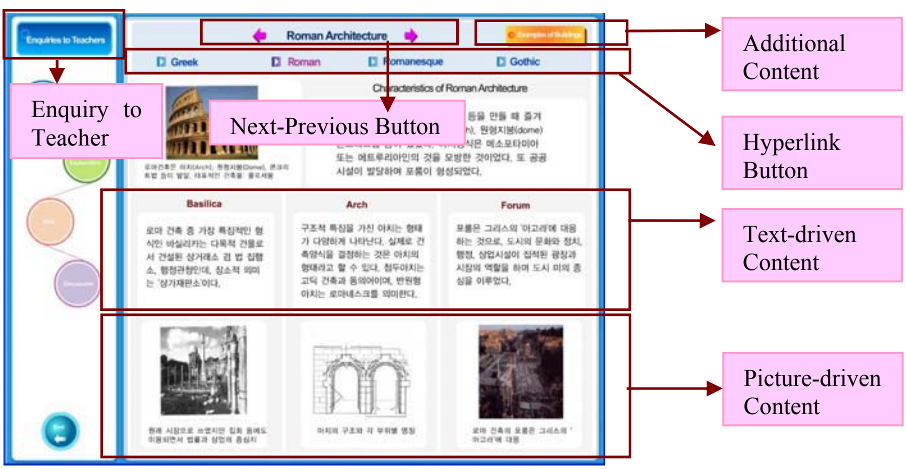


Fig. 2. Interface Layout for Main Learning Contents

3.2 Visual vs. Auditory (V vs. A)

Felder & Silverman discuss that Visual style learners may prefer images or demos (V1) on the screen, while Auditory learners may prefer written texts or explanations (A1). Thus, the second interface layout in Fig. 2 has content areas configured by both images and text with similar contents information. The learners can choose either picture-driven or text-driven areas according to their preferences. In the picture-driven area, the detailed explanations are mainly led by images in order to help the learners establish an understanding of the learning contents. On the other hand, the text-driven area is led by written texts. Therefore, comparison in button clicks and durations between text-driven and picture-driven contents can be led to diagnosing the learner's learning style in the V vs. A dimension.

3.3 Sensing vs. Intuitive (S vs. N)

An interface design has been devised to determine whether Sensory learners are patient with the additional materials and spend more time on studying the details of the references when additional contents or examples are given for reference learning materials as illustrated in Fig. 2.

A button for "Additional material" on the top right in Fig. 2 can be chosen by learners who want to study the detailed examples for the Architectural style (i.e. Roman Architecture in Fig. 2). If they clicked the button, the additional contents also have many different examples and details, so the students also go into deeper information if they are interested. It is based on that ILS regards Sensors as having "attentiveness to details (S1)" and Intuitors as being "bored by details (N1)".

Furthermore, a quiz section was designed as a problem-solving situation where learners have to select and insert a correct piece into a correct place on the problem.

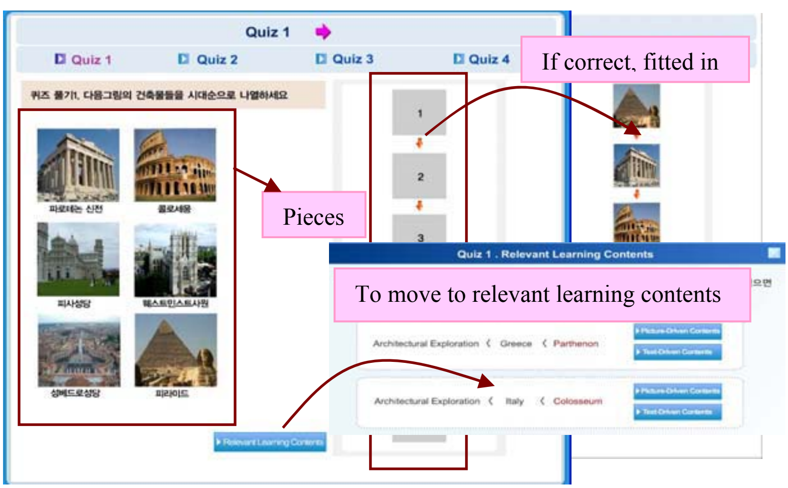


Fig. 3. Interface Layout for Problem Solving Situation

This has been suggested in that Felder & Silverman mention that while Sensory type learners are careful but may be slow (S1), Intuitive learners are quick but may be careless (N1), so the interface design shown in Fig. 3 has been devised to verify the assumption. The user interface works in the way that as soon as the students drag and drop a piece on the answer section, if correct, the piece is fitted in, but if wrong, it goes back to the original place.

To illustrate, if students are careful to choose the answer and move to the puzzle section, they may have low trials and high correctness and completeness, but if they try it out without care, they may have high trials and low correctness and completeness. Therefore, the learner's learning style in Sensing vs. Intuitive dimension can be recognized from the differences in the number of trials and the correctness on the user interface of the quiz section.

3.4 Active vs. Reflective (C vs. R)

Felder & Silverman point out that an Active learner is someone who feels more comfortable with active experimentation, and enjoys brief discussion and problem-solving activities (C2). Conversely, Reflective learners process information reflectively and want to have intervals to think about what others have told (R2). From this viewpoint, Active learners may want to participate in activities such as quiz, chatting (C1), and brief discussion, whereas reflective learners may be more interested in reviewing other learners' and professional opinions (R1) rather than doing real activities. In the discussion section, different buttons are provided based on the different characteristics.

4 Experiment

Based on these interface guidelines, a learning content was developed in the architecture domain with Macromedia Flash [4] in order to collect and verify the hypothesized behaviors. Systems concerned with user modeling for the automatic adaptation of interfaces focus on monitoring behaviors collected from the interface [5]. In this research, the learner's behaviors for the interface were also monitored in order to derive the learner's learning style preferences from the interface events, instead of using the ILS questionnaire for assessing learning preferences as in [2].

Table 2. Data Collection for Learning Styles Diagnosis

	G	Q	V	A	S	N	C	R
Number of data with low LoP [1-3]	22	20	14	8	22	12	30	17
Number of used data	20	8	42	7	27	9	14	9

As the result of ILS questionnaire, we can get *Level of Preference* (LoP) which is the mark of how much the learners belong to the specific learning style. It is represented by odd numbers from 1 to 11 and a bigger number means a stronger preference. We used data with higher LoP only as described in Table 2.

5 Learning Styles Diagnosis and Pattern Extraction

5.1 Behavior Pattern Extraction

The first step in providing adaptive learning interface customization seems to build a model for learning styles [6]. In order to build one, we collect the learners' behaviors from the user interface, and analyze the data with *Decision Trees* (DT) and *Hidden Markov Models* (HMM). DTs produce the rules of the classification which are visible and easy to understand for the pattern recognition and classification [7]. HMMs are a statistical method that uses probability measures to model sequential data represented by sequence of observations [8, 9].

Fig. 4 shows the detailed approach of this study in order to extract learners' hidden behavior patterns. First of all, the data was collected from the experiment where the learners with different learning styles studied the learning content of architecture domain with the hypothesized interfaces, and the learners' behaviors on the learning content were recorded in XML files. Secondly, a preprocessing of the data based on the XML files was performed in order to make the data more appropriate for mining, exploring the characteristics of DT & HMM. Based on the preprocessed data, DTs and HMMs are constructed for each learning style dimension.

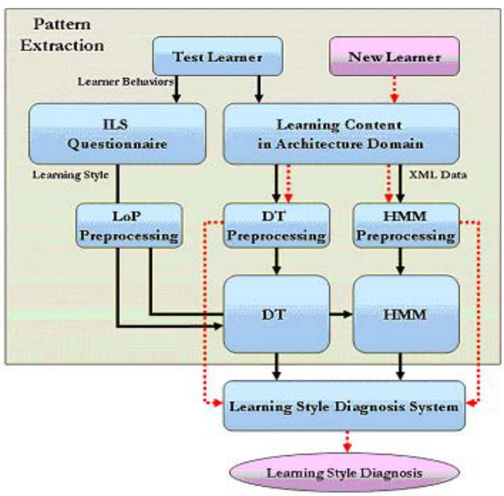


Fig. 4. Workflow of Learning Style Diagnosis for Adaptive User Interface Customization

5.2 Decision Tree

Preprocessing. The learner's data directly collected from the user interface may not be proper to use for building a user model. There can be errors or missing attributes, so some transform may be needed [10]. We removed anomalous and erroneous data, discarded the data with a LoP of 1 or 3, and transformed some data into more usable format. For instance, the actions of the same type (e.g. chatting with friends and asking to teachers) were combined into an instance, and durations in the events of the same properties (e.g. time spent on picture-driven contents) were added and put into an instance. In addition to these, it was also taken into consideration how correct the students solved the quiz or how carefully they tried to solve it. From the data, we collected the attributes for building decision trees such as the number of interface icon clicks, the durations of some activities, the correctness of solving quizzes, the number of opinions that they wrote or read, the trial rate of the quiz, and so on.

Attributes. A total of 58 attributes as shown in Table 3 has been devised so that DT based diagnosis can be constructed. For example, "GQ_7_MainCntsGlobal" in the G vs. Q dimension, represents the number of clicks for contents selection through hyper-link buttons as in Fig. 3. A global learner would move to other content with the hyper-link (MainCntsGlobal) button rather than moving in a sequential order.

Table 3. Diagnosis Attributes for DTs in G vs. S dimension

Learning Style	Attribute List	
Global vs. Sequential	GQ_1_TableofCntMenuClick GQ_2_TableofCntTabClick GQ_3_TableofCntSeqClick GQ_4_TableofCntMenuFirst GQ_5_MainMenuSymbolFst GQ_6_MainCntsSequence	GQ_7_MainCntsGlobal GQ_8_MainScreenMove GQ_9_DetailGlobal GQ_10_DetailSequence GQ_11_SubCntsGlobal GQ_12_SubCntsSequence

Training and Test. The preprocessed data were divided into two sets. Seventy percent of them were used for training and thirty percent of them were used for testing. Table 4 shows the number of data which was used in each learning style dimension. For example, 49 learners among 70 learners had a LoP larger than 3 in the V vs. A dimension. Among the 49 data, 35 were used for training (i.e. building a decision tree) and 14 for testing the built tree [11].

Table 4. Training Data and Test Data

	G vs. Q	V vs. A	S vs. N	C vs. R
Number of training set	21	35	27	17
Number of testing set	7	14	9	6

The decision tree obtained for the V vs. A dimension is shown in Fig. 5. The decision tree in the example (Fig. 5) illustrates that the root classifier is the duration on the text-driven contents by moving through the optional text button for choosing text-driven contents. In the learning system used, a user can choose either picture-driven or text-driven contents, and there are many buttons for choosing either one. If learners spent their learning time on the text-driven contents chosen by the optional button less than 332.5 seconds, they are regarded as Visual style learners.

Otherwise, the DT is to check the duration on the picture-driven contents. If the learners spent their learning time on the picture-driven contents greater than 127 seconds, they are also classified into a Visual group. If not, the next step is to count how many times the learners clicked a button for moving to the relevant picture-driven content. The DT will determine the learners to be an Auditory group if the number of the counts is less than 18.5. Lastly, depending on the number of text buttons clicks (2.5) on the table of content rather than moving through image buttons, DT will classify the learners into the Auditory group or the Visual group.

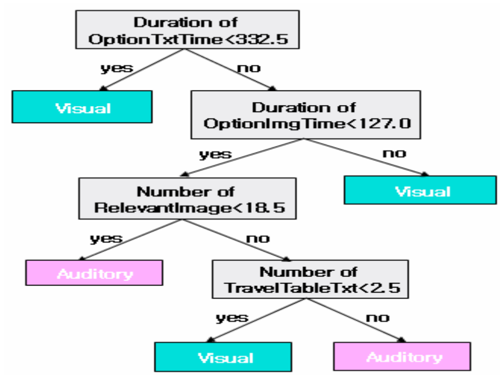


Fig. 5. An Example of the Final DTs

Those rules obtained by DT correspond to the hypothesized behaviors on the user interface. This decision tree was validated with the 14 testing data in order to test the accuracy of the trees and rules, and the error rate is 0% in the decision tree for the data. Similarly, the decision trees in Sensing vs. Intuitive (SN) and Global vs. Sequential dimensions (GQ) were also analyzed and validated with quite low error rates (SN: 22.22%, GQ: 28.57%), but Active vs. Reflective dimension had a quite high error rate (33.33%). Table 5 shows the detailed information about test sets with error rates in each dimension.

Table 5. Results of DTs (Error Rates)

DTs	G vs. Q	V vs. A	S vs. N	C vs. R
Error Rate	28.57%	0.0%	22.22%	33.33%

5.3 Hidden Markov Model

While DTs do not consider the sequence of learner's actions, HMMs do. For example, if a learner clicks button A, B and C sequentially, DTs can handle only click counters of each button, but HMMs can handle the sequence of clicks. In this viewpoint, DTs and HMMs are complementary each other.

Preprocessing. In order to train HMMs [8.9], we need sequential information. Since the learners' data collected from our learning system are the sequences of buttons or menus clicks, we can easily apply the data to HMMs. We also removed anomalous and erroneous data, and discarded the data with a number of data with low LoP (e.g.1-3). We also transformed the data. For example, to prepare the data for the G vs. S, we abstracted the menu button clicks with menu hierarchy. If there are two top-level menu buttons such as m_1 and m_2, sub-menus m_{11} and m_{12} under m_1, and m_{21} and m_{22} under m_2, and a learner sequentially clicks m_1, m_{11}, m_2, m_{21}, and m_{22}, we converted this sequence into m_1, m_1, m_2, m_2 *and* m_2. That is, we use only the highest level information. On the other hand, we do not perform this kind of transformation for the V vs. A because button clicks for the V vs. A would be meaningful.

Attributes. The collected data from the experiment consist of a variety of learning activities on the learning content. Therefore, in order to analyze separately the data in each dimension with HMM, learning activities were filtered and redefined into four different sequential data sets as a preprocessing step. Table 6 shows the sequential attributes for four different dimensions of learning styles. First of all, learner's sequential information on studying the history of western architecture abstracted in a high level, explained in the preprocessing section, was extracted to make the G vs. Q HMMs, consisting of 9 attributes (e.g. GQ_1_Main1: studying Greek architecture, GQ_2_Main2: studying Roman architecture, etc.).

Training and Test. We used on HMM program which is implemented in Java, JahmmViz [12]. We built two HMMs for each learning style dimension. For example, one HMM for Visual and one HMM for Auditory were trained for the V vs. A dimension. Fig. 6 shows the trained HMMs for the V and A where hidden states were used. As we did for building DTs, we divided the data into the training set and the test set with a ratio of 7:3. Table 7 shows the number of data which was used in each learning style dimension. For example, among the 42 data, 30 were used for training and 12 for testing in the Visual dimension. In order to verify the HMMs, we apply a test data to each HMM and evaluate the probabilities for each HMM to accept the data. If the probability of the Visual HMM is higher, we conclude that the learner of the data is Visual and vice versa. The Visual and the Auditory HMMs correctly classify 12 data among 14 (i.e. the error ratio is 14.28%). The HMMs for the G vs. S (GS) dimension and S vs. N (SN) dimension were also validated and showed low error rate (GS: 14.28%, SN: 22.22%). However, the C vs. R dimension showed a little high error rate (33.33%).

Table 6. Diagnosis Attributes for HMMs in G vs. S dimension

Learning Style	Attribute List	
Global vs. Sequential	GQ_1_Main1 GQ_2_Main2 GQ_3_Main3 GQ_4_Main4 GQ_5_Main1add	GQ_6_Main2add GQ_7_Main3add GQ_8_Main4add GQ_9_TopicChange

Table 7. Results of HMMs (Error Rates)

HMMs	G vs. Q	V vs. A	S vs. N	C vs. R
Error Rate	14.28%	14.28%	22.22%	33.33%

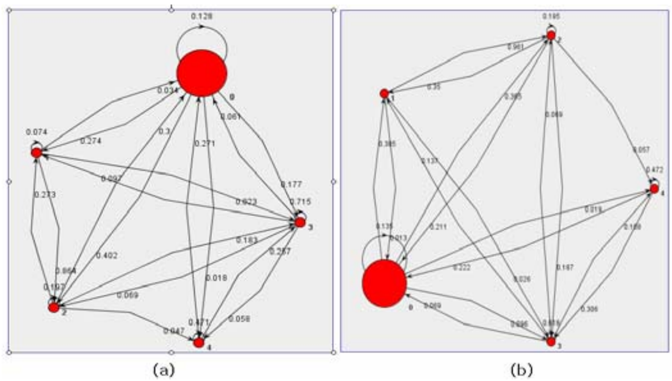


Fig. 6. (a) HMM for a Visual Learner Style and (b) HMM for an Auditory Learner Style

5.4 Result Analysis

To diagnose user's learning style, two different kinds of machine learning techniques were utilized. DT was used with focus on button click counters and durations on the hypothesized learning interface and HMM was used to analyze the sequential information of a user's learning process.

G vs. Q: DTs show 28.57% error rate and HMMs show 14.28% error rate. The sequential information is one of the essential data to extract learner's Global vs. Sequential behavior patterns, so HMMs are better for analyzing data than DTs.

V vs. A: Our methods showed 0% (DT), and 14.28% (HMM) error rates. This result illustrates that the hypothesized interfaces are well designed to classify Visual vs. Auditory learners. Since the variety of attributes, such as the number of button clicks, the time for learning, the rates of a correct answer, and trial of solving the quiz, etc. are more useful than sequence information of attributes for the V vs. A dimension, DTs show better results than HMMs. Thus, the DT technique can be utilized in order to diagnose the individual learning style in the V vs. A dimension.

S vs. N: It might be possible that both methods; DTs & HMMs are applied to identify the learner's style. Then, if the results of both methods are the same, it is obvious to determine whether she/he is sensing or intuitive. However, if not, a decision-making process is needed for the diagnosis of the learning style: (i) A gap value of probabilities derived from the S and N HMMs with "each testing data" is calculated, and then the average of the gap values with "all of the testing data" is produced. (ii) A gap value of probabilities in "a new learner's data" whose learning style is diagnosed can also be calculated by using the S and N HMMs. (iii) If the gap value from the new learner is greater than the average value, the result from the HMMs is regarded as more trustworthy. Otherwise, the result from the DTs will be chosen.

C vs. R: It seems that the error rates in both the DT and HMM methods are very high. We extracted Quiz & Discussion related button clicks from learners' data, and trained DTs and HMMs for the C vs. R dimension. In fact, most of learners tended to focus on the main learning part rather than the discussion and quiz parts due to limited time. It was statistically proved that most of learners spent on the discussion part less than 10 percent of whole experiment time. Therefore, not enough data to train DTs and HMMs were obtained from this viewpoint.

6 Adaptive Interface

As shown above, learning styles of individual learners are diagnosed based on behaviors obtained from specially designed interfaces using machine learning approaches such as DT and HMM. It means that individual learning styles can be recognized based on the user interface-based behavior patterns. Therefore, based on the learning styles diagnosis, it is also possible to develop an intelligent tutoring system that is adaptive to individual learner's learning styles and preferences. In this CREDITS center, a prototype of an intelligent learning environment that is adaptive to learning styles has been developed on the subject of heritage alive of an old temple [13].

7 Concluding Remarks

This paper describes learning styles diagnosis based on behavior patterns for user interfaces, and developing an intelligent learning system which can enhance learning efficiency and experiences by providing effective user interfaces and learning contents depending on the learner's preferences. To achieve the aim, some machine learning approaches like DT & HMM were utilized. Based on the diagnosis of learning styles with the machine learning approaches, this study also exemplified how the different learning styles can be adapted to the user interface layout in intelligent learning environments.

We are currently conducting a learning style diagnosis experiment as discussed in paper with about 600 students using an interior design-related content. We believe that a bigger pool of student data would provide more beneficial learner modeling results. Further research efforts are being made to extend beyond simple data (e.g. button clicks, the duration on a page alone, etc.) to additional data collection (e.g. different attention on either text-driven or picture-driven contents, etc.), by means of eye movements with an eye-tracker device.

For the future work, in addition to the classification methods like DT and HMM, clustering methods can be approached in order to partition the learners into 16 different learning styles groups. In that the four dimensions may have influences on one another, the learning style analysis conducted in each dimension separately needs to be extended to the combinations of those four dimensions. Furthermore, another future work will also be directed to extending the learner modeling by considering the various other kinds of learner characteristics such as emotion and motivation as well as learning mastery in providing adaptive learning support.

Acknowledgements

This research was supported by the Korean Ministry of Science & Technology through the Creative Research Initiative Program.

References

1. Karger, D. R., Quan, D.: Prerequisites for a Personalizable User Interface, Proc. of Intelligent User Interface2004 Conf., Ukita (2004)
2. Felder, R., Silverman, L.: Learning and Teaching Styles in Engineering Education, Engineering Education (1988) 78(7) 674–681
3. Chen, W., Mizoguchi, R.: Communication Content Ontology for Learner Model Agent in Multi-agent Architecture, Proc. of the 7th International Conference on Computers in Education (1999) 95-102
4. Cha, H. J., Kim, Y. S., Park, S. H., Cho, Y. J. Pashkin, M.: Adaptive Learning Interface Customization based on Learning Styles and Behaviors, Proc. International Conference on Computers in Education, Singapore (2005)
5. Branco, P., Encarnacao, L. M.: Affective Computing for Behavior-based UI Adaptation, Proc. of Intelligent User Interface2004 Conf., Ukita (2004)
6. Constantine, S., Charalampos, K., Adamantios, K.: Decision Making in Intelligent User Interface, Intelligent User Interface1997 Conf., Orlando, Florida (1997) 195-202
7. Margaret, H. D.: Data Mining Introductory and Advanced Topics, Pearson Education Inc., New Jersey (2003)
8. Lawrence, R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, Morgan Kaufmann Publishers Inc., San Francisco (1990) 267-296
9. Fok, A. W. P., Wong, R. H. S., Ip, H. H. S.: Adaptive User Interface for Personalized Education based on Hidden Markov Model, Proc. of the 9th Global Chinese Conference on Computers in Education, Hawaii (2005)
10. Mitchell, T.: Machine Learning, WCB/McGraw-Hill, USA (1997)
11. Geurtshttp, P.: Decision Tree Java Library, http://www.montefiore.ulg.ac.be/~geurts/dtapplet/dtexplication.html
12. Francois, J. M.: Jahmm Java Library, http://www.run.montefiore.ulg.ac.be/~francois/software/jahmm
13. Kim, Y. S., Kim, S. A., Cho, Y. J., Park, S. H.: Adaptive Customization of User Interface Design Based on Learning Styles and Behaviors: A Case Study of a Heritage Alive Learning System, Proc. of ASME Computers & Information in Engineering Conf., California (2005)

Comparison of Machine Learning Methods for Intelligent Tutoring Systems

Wilhelmiina Hämäläinen and Mikko Vinni

Department of Computer Science, University of Joensuu,
P.O. Box 111, FI-80101 Joensuu
Finland
{whamalai, mvinni}@cs.joensuu.fi

Abstract. To implement real intelligence or adaptivity, the models for intelligent tutoring systems should be learnt from data. However, the educational data sets are so small that machine learning methods cannot be applied directly. In this paper, we tackle this problem, and give general outlines for creating accurate classifiers for educational data. We describe our experiment, where we were able to predict course success with more than 80% accuracy in the middle of course, given only hundred rows of data.

1 Introduction

Ability to learn is often considered as one of the main characteristics of intelligence. In this sense most of the intelligent tutoring systems are far from intelligent. They are like old-fashioned expert systems, which perform mechanic reasoning according to predefined rules. They may use very intelligent-sounding methods like Bayesian networks, decision trees or fuzzy logic, but almost always these methods are used only for model representation. They determine only, how to reason in the model, but the model itself is predefined.

This situation is very surprising, compared to other fields, where machine learning methods are widely used. In modern adaptive systems both the model structure and model parameters (see terminology in Table 1) are learnt from data. However, in educational applications, it is rare that even the model parameters are learnt from data.

The main problem is the lack of data. The educational data sets are very small – the size of class, which is often only 50-100 rows. In distance learning setting, the course sizes are larger, and it is possible to pool data from several years, if the course has remained unchanged. Still we can be happy, if we get data from 200-300 students. This is really little, when we recall that most machine learning methods require thousands of rows of data.

Another problem is that the data is often mixed, and contains both numeric and categorial variables. Numeric variables can always be transformed to categorial, but the opposite is generally not possible (we can transform all variables to binary, but in the cost of model complexity). This is not necessarily a problem, because the best methods for really small data sets use categorial data. Educational data has also one advantage compared to several other domains: the data

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 525–534, 2006.

Table 1. The basic terminology used in this paper

Concept	Meaning
Model structure	Defines the variables and their relations in the model. E.g. nodes and edges in a Bayesian network, or independent and dependent variables in linear regression.
Model parameters	Assigned numerical values, which describe the variables and their relations in the given model. E.g. prior and conditional probabilities in a Bayesian network, or regression coefficients in linear regression.
Modelling paradigm	General modelling principles like definitions, assumption and techniques for constructing and using models. E.g. Bayesian networks, linear regression, neural networks, or decision trees.

sets are usually very clean, i.e. the values are correct and do not contain any noise from measuring devices.

In the following, we will tackle these problems. In Section 2, we describe the general appoach, which allows us to infer also the model structure from data. In Section 3, we give general guidelines for modelling educational data. We concentrate on constructing a classifier from real student data, alhtough some principles apply to other predicting tasks, as well. In Section 4, we report our empirical results, which demonstrate that quite accurate classifiers can be constructed from small data sets (around 100 rows data) with careful data preprocessing and selection of modelling paradigms. We compare the classification accuracy of multiple linear regression, support vector machines and three variations of naive Bayes classifiers. In Section 5, we introduce related research, and in Section 6, we draw the final conclusions.

2 Approach

Combination of descriptive and predictive modelling is often very fruitful, when we do not have enough data to learn models purely from data but on the other hand we do not want to rely on any ad hoc assumptions. The idea is to analyze the data first by descriptive techniques (classical data mining) to discover existing patterns in data. The patterns can be for example association rules, correlations or clusterings. Often the process is iterative and we have to try with different feature sets, before we find any meaningful patterns. In a successful case, we discover significant dependencies between the outcome variable and other variables and get information about the form of dependency. All this information is utilized in the predictive modelling phase (classical machine learning), which produces the actual models for prediction.

In the second phase, the system constructor should first decide the most appropriate modelling paradigm or paradigms and then restrict the set of suitable model structures. The selection of modelling paradigm depends on several factors like data size, number and type of attributes, form of dependencies (linear or non-linear). In the next section, we will give some general guidelines for the educational domain, but the most successful descriptive paradigms do also hint suitable predictive paradigms. For example, strong correlations support linear regression and strong associations support Bayesian methods.

If we have very little data, the model structure cannot be defined from data, but we can compose it according to descriptive patterns found in the first phase. If the descriptive analysis suggests several alternative model structures, it is best to test all of them, and select the one with smallest generalization error. The model parameters are always learnt from data. In small data sets it often happens that some parameters cannot be defined, because of the lack of data. A couple of missing variables can be handled by well-known heuristic tricks, but several missing variables is a sign of too complex a model.

3 Classifiers for Educational Data

The main problem in *ITS*s is classification. Before we can select any tutoring action, we should classify the situation: whether the student masters the topic or not, whether she or he will pass the course or not. However, the problem contains so many uncertain or unknown factors that we cannot classify the students deterministically into two mutual classes. Rather, we should use additional class values (e.g. the mastering level is good, average, or poor) or estimate the class probabilities. The latter approach is more recommendable, because additional variables always increase the model complexity and make it more inaccurate. On the other hand, the class probabilities can always be interpreted as intermediate class values, if needed.

Typically, the student profile contains too many attributes for building accurate classifiers, and we should select only the most influencing factors for predicting purposes. In addition, the domains of numeric attributes are large, and the data is not representative. In practice, we have to combine attributes and reduce their domains as much as possible, without losing their predictive power. As a rule of thumb, it is suggested that the data set should contain at least 10 rows of data per each model parameter. In simplest models, like naive Bayes classifiers using binary attributes, this means about $\frac{n}{20}$ independent attributes, where n is the data size.

The other consequence of this simple calculation is the model complexity. We should select modelling paradigms, which use as simple models as possible. In the following, we will suggest good candidates for both numeric and categorial data. All these classifiers are able to produce class probabilities or similar measures, instead of deterministic class decisions. We have excluded such commonly used methods like nearest neighbour classifiers, neural networks and variations of decision trees, which would require much more data to work accurately.

3.1 Classifiers for Numeric Data

The simplest predictive model for numeric data is linear regression. In multiple linear regression we can predict the dependent variable Y, given independent variables $X_1, ..., X_k$, by a linear equation $Y = \alpha_k X_k + \alpha_{k-1} X_{k-1} + ... + \alpha_1 X_1 + \alpha_0$.

Although the result is numeric value, it can be easily interpreted as a class value in binary class problem.

The main assumption in the linear regression is that the relationship should be approximately linear. However, the model can tolerate quite large deviations from linearity, if the trend is correct. On the other hand, we can use linear regression as a descriptive tool to identify, how linear or non-linear the relationship is. This is done by checking the square of multiple correlation coefficient

$$r^2 = \frac{Var(Y) - MSE}{Var(Y)},$$

where $Var(Y)$ is Y's variance and MSE is the mean of squared errors. As a rule of thumb, if $r^2 > 0.4$, the linear tendency is significant, and we can quite safely use linear regression.

The main limitations of applying linear regression in educational domain concern outliers and collinearity. The linear regression model is very sensitive to outliers, and educational data contains almost always exceptional students, who can pass the course with minimal effort or fail without any visible reason. If the data contains several clear outliers, *robust regression* [4] can be tried instead. Collinearity means strong linear dependencies between independent variables. These are very typical in educational data, where all factors are more or less related to each other. It is hard to give any exact values, when the correlations are harmless, but in our experiment the accuracy of linear regression model classifier suffered, when the correlation coefficient was $r > 0.7$. The weaker correlations did not have any significant effect.

Support vector machines (*SVM*) [7] are another good candidate for classifying educational data. The idea is to find data points ("support vectors") which define the widest linear margin between two classes. Non-linear class boundaries can be handled by two tricks: first, the data can be mapped to a higher dimension, where the boundary is linear, and second, we can define a soft margin, which allows some misclassification. To avoid overfitting but still achieving good classification accuracy, a compromise of these two approaches is selected.

Support vector machines suit especially well for small data sets, because the classification is based on only some data points, and data dimension–size ratio has no effect on model complexity. In practice, *SVM*s have produced excellent results, and they are generally considered as best classifiers. The only shortcoming is the "black-box" nature of the model. This is in contrast with the general requirement that models in *ITS* should be transparent. In addition, selecting appropriate kernel function and other parameters is difficult, and often we have to test different settings empirically.

3.2 Bayesian Classifiers for Categorial Data

The previously mentioned classifiers suit only for numeric and binary data. Now we will turn to classifiers, which use categorial data. This type of classifiers are more general, because we can always discretize numeric data into categorial with desired precision. The resulting models are simpler, more robust and generalize better, when the course content and students change.

Bayesian networks are very attractive method for educational domain, where uncertainty is always involved and we look for a transparent, easily understandable model. Unfortunately, the general Bayesian networks are too complex for small data sets, and the models overfit easily. *Naive Bayes models* avoid this problem. The network structure consist of only two layers, the class variable in the root node and all the other variables in the leaf nodes. In addition, it is assumed that all leaf nodes are conditionally independent, given the class value. In reality this so called *Naive Bayes assumption* is often unrealistic, but in practice the naive Bayes model has worked very well. One reason is that according to [1] Naive Bayes assumption is not a necessary but only sufficient condition for naive Bayes optimality. In empirical tests, naive Bayes classifiers have often outperformed more sophisticated classifiers like decision trees or general Bayesian networks, especially with small datasets (up to 1000 rows) [1].

In the educational domain, the Naive Bayes assumption is nearly always violated, because the variables are often interconnected. However, Naive Bayes classifier can tolerate surprisingly strong dependencies between independent variables. In our experiments, the model accuracy suffered only when the conditional probability between two leaf node values was $P(F = 0|E = 0) = 0.96$. The average mutual information between those variables was also high, $AMI(E, F) = 0.178$, of the same magnitude as the dependencies between class variable and leaf variables ($AMI \in [0.130, 0.300]$). The effect to classification accuracy was about the same as in linear regression model.

Tree augmented naive Bayes models (*TAN models*) [2] enlarge naive Bayes models by allowing additional dependencies. The *TAN* model structure is otherwise like in the naive Bayes model, but each leaf node can depend on another leaf node, in addition to class variable. This is often a good compromise between a naive Bayes model and a general Bayesian network: the model structure is simple enough to avoid overfitting, but strong dependencies can be taken into account. In the empirical tests by [2] the *TAN* model outperformed the standard naive Bayes, but in our experiments the improvements were not so striking.

Bayesian multinets [3] generalize the naive Bayes classifier further. In Bayesian multinets we can define a different network structure for every class value. This is especially useful, when classes have different independence assertions. For example, when we try to classify the course outcomes, it is very common that failed and passed students have different dependencies. In addition, the class of failed students is often much smaller and thus harder to recognize, because of less accurate parameter values. When we define for both classes their own model structures, the failed students can be modelled more accurately. In addition, the resulting model is often much simpler (and never more complex) than the corresponding standard naive Bayes model, which improves generalization.

Friedman & al. [2] have observed that the classification accuracy in both *TAN* and Bayesian multinet models can be further improved by certain parameter smoothing operations. In ”Dirichlet smoothing” or *standard parameterization* the conditional probability of X_i given its parent Π_{X_i} is calculated by

$$P(X_i = x_k|\Pi_{X_i} = y_j) = \frac{m(X_i = x_k, \Pi_{X_i} = y_j) + \alpha_{ijk}}{m(\Pi_{X_i} = y_j) + \alpha_{ij}},$$

where $\alpha_{ij} = \sum_k \alpha_{ijk}$ are Dirichlet hyperparameters. If $\alpha_{ijk} = 0$, the parameters reduce to relative frequencies. Other values can be used to integrate domain knowledge. When nothing else is known, a common choice is to use values $\alpha_{ijk} = 1$. This correction is known as *Laplace smoothing*. In empirical tests by [2] it produced significant improvements especially in small data sets, but in our experiments the improvements were quite insignificant.

4 Empirical Results

Our empirical tests are related to distance learning Computer Science program *ViSCoS*, in the University of Joensuu, Finland. One of the main goals in *ViSCoS* is to develop an intelligent tutoring system based on real student data. Java Programming courses have been selected as our pilot courses, because of their importance and large drop-out and failing rates. In the first phase, we have developed classifiers, which can predict the course success (either passed or failed/drop-out) as early as possible. Currently this information serves only course teachers, but the next step is to integrate intelligent tutors into course environment.

4.1 Data

Our data set consisted of only exercise points and final grades. The data hah been collected in two programming courses, (*Prog.1* and *Prog.2*), in two consecutive years 2002-2003 and 2003-2004. In *Prog.1* course, the students study to use Java in a procedural way, and object-oriented programming is studied only in *Prog.2*.

The first problem in our modelling task was feature extraction and feature selection. Each class consisted of only 50-60 students, but each student had solved exercises in 19 weeks. To increase the data size and decrease the number of attributes, we created new attributes, which abstracted away the differences between two academic years. This was relatively easy, because the course objectives had remained the same. The exercises could be divided into six categories: basic programming skills, loops and arrays, applets, object-oriented programming, graphical applications and error handling. The exercise points in each category were summed and normalized so that the maximum points were the same in both years. After that the data sets could be combined. The resulting *Prog.1* data set contained 125 rows and four attributes and *Prog.2* data set contained 88 rows and eight attributes.

The attributes, their domains and descriptions are presented in Table 2. In the original data set, all attributes were numerical, but for Bayesian classifiers we converted them to binary-valued qualitative attributes.

Table 2. Selected attributes, their numerical domain (*NDom*), binary-valued qualitative domain (*QDom*), and description

Attr.	NDom.	QDom.	Description
A	$\{0,..,12\}$	{little, lot}	Exercise points in basic programming structures.
B	$\{0,..,14\}$	{little, lot}	Exercise points in loops and arrays.
C	$\{0,..,12\}$	{little, lot}	Exercise points in applets.
D	$\{0,..,8\}$	{little, lot}	Exercise points in object-oriented programming.
E	$\{0,..,19\}$	{little, lot}	Exercise points in graphical applications.
F	$\{0,..,10\}$	{little, lot}	Exercise points in error handling.
$TP1$	$\{0,..,30\}$	{little, lot}	Total points in *Prog.1*.
$TP2$	$\{0,..,30\}$	{little, lot}	Total points in *Prog.2*.
$FR1$	$\{0,1\}$	{fail, pass}	Final result in *Prog.1*.
$FR2$	$\{0,1\}$	{fail, pass}	Final result in *Prog.2*.

4.2 Classifiers

For model construction, we performed descriptive data analysis, where we searched dependencies between numeric and categorial attributes by correlation, correlation ratio, mutual information and association rules. The analysis revealed that total points TP in both courses depended heavily on exercise points and the dependency was quite linear. Even stronger dependencies were discovered by association rules between final results FR and binary versions of exercise attributes. The dependency analysis revealed also dependencies between exercise attributes. Most of them were moderate, except the dependency between E and F. In addition, we found that final results in *Prog.2* depended strongly on *Prog.1* attributes B, C and $TP1$. B attribute affected mostly among successful students and C among failed students, but generally $TP1$ was the most affecting factor.

The goal was to predict the course final results $FR1$ and $FR2$ as early as possible, and thus we tested seven different cases:

1. $A \Rightarrow FR1$
2. $A, B \Rightarrow FR1$
3. $A, B, C \Rightarrow FR1$
4. $TP1 \Rightarrow FR2$
5. $TP1, D \Rightarrow FR2$
6. $TP1, D, E \Rightarrow FR2$
7. $TP1, D, E, F \Rightarrow FR2$

Two numeric classification methods (multiple linear regression and support vector machines) were applied to numeric data, and three versions of naive Bayes classifiers (standard *NB*, *TAN* and Bayesian multinets) to categorial data. In numerical methods, we had to learn seven different models in both paradigms, but for naive Bayes classifiers, it was enough to learn just one model for *Prog.1* and one for *Prog.2* and update the class probabilities by Bayes rule, when new exercise attributes were known.

The naive Bayes model structures are described in Figure 1. In *TAN* models, we have included the strongest dependencies between exercise attributes. The Bayesian multinet structures are quite similar, because the dependencies among failed and successful students were mostly same. The biggest difference is that in the *Prog.2* model, we use attribute B to initialize $P(FR2)$ in the succeeders'

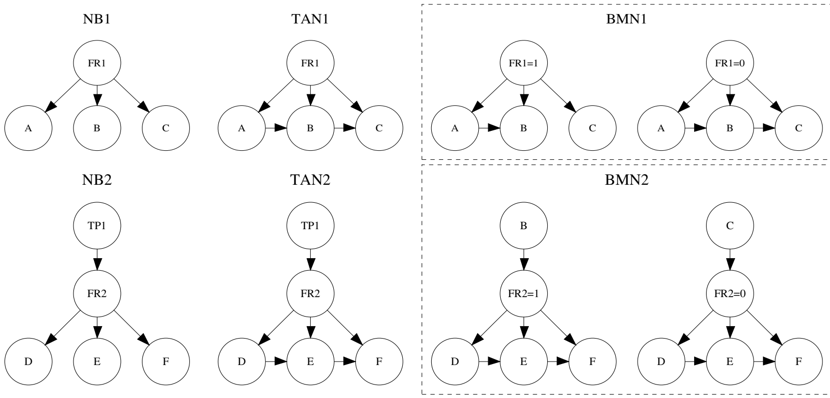


Fig. 1. Naive Bayes (*NB1, NB2*), Tree-augmented Naive Bayes (*TAN1, TAN2*), and Bayesian multinet (*BMN1, BMN2*) classifiers for predicting the final results ($FR1, FR2$) in *Prog.1* and *Prog.2* courses. A, B, C, D, E and F are exercise points. Either total points in *Prog.1* $TP1$ or exercise points in B or C categories have been used as background variables for *Prog.2* models.

network and C in the failers' network. The model parameters in *TAN* models and Bayesian networks were calculated both with and without Laplace smoothing.

4.3 Results

The models have been compared in all test cases by 10-fold cross-validation. The generalization error has been expressed as true positive and true negative rates, which tell the proportion of correctly predicted successful students from all successful students, and similarly for failed students. The results are represented in Table 3.

Table 3. Comparison of five classification methods. Linear regression (*LR*) and support vector machine (*SVM*) classifiers were trained with numeric course data, and Naive Bayes (*NB*), tree-augmented Bayesian networks (*TAN*) and Bayesian multinets (*BMN*) with categorial data. The generalization error has been estimated by 10-fold cross-validation and for each model structure and classification method true positive *TP* and true negative *TN* rates are reported.

Model	LR		SVM		NB		TAN		BMN	
structure	TP	TN	TP	TN	TP	TN	TP	TN	TP	TN
$A \Rightarrow FR1$	0.83	0.47	0.91	0.42	0.96	0.31	0.96	0.31	0.96	0.31
$A, B \Rightarrow FR1$	0.91	0.72	0.93	0.69	0.80	0.81	0.82	0.83	0.82	0.83
$A, B, C \Rightarrow FR1$	0.93	0.81	0.94	0.78	0.83	0.81	0.84	0.81	0.85	0.81
$TP1 \Rightarrow FR2$	0.70	0.68	0.88	0.59	0.96	0.51	0.96	0.51	0.75	0.38
$TP1, D \Rightarrow FR2$	0.78	0.85	0.86	0.76	0.71	0.68	0.71	0.68	0.73	0.73
$TP1, D, E \Rightarrow FR2$	0.75	0.90	0.98	0.82	0.82	0.86	0.82	0.81	0.78	0.81
$TP1, D, E, F \Rightarrow FR2$	0.70	0.92	0.94	0.82	0.86	0.86	0.84	0.81	0.78	0.81

We observe that generally the support vector machine performed best, especially in the last cases, when several attributes were included. However, in *Prog.1* data, the naive Bayes classifiers predicted the failers better than the support vector machine. Predicting potential failers/drop-outs accurately is more important, because they would need special tutoring. In addition, the real probabilities reveal more information.

When we compare different naive Bayes classifiers, we observe that the results are quite controversal. In *Prog.1*, the *TAN* and Bayesian multinet models achieve a slight improvement, but in *Prog2*, the standard naive Bayes model performs better. The results by *TAN* and Bayesian multinets are quite similar especially in *Prog.1*. This is not surprising, because the model structures are nearly the same. However, in *Prog.2*, the *TAN* model outperforms multinet model in almost all cases. This suggests that total points in *Prog.1* is a better background variable than exercise points in B or C.

We have also tried the Laplace smoothing in all Bayesian models, but it did not produce any improvement. In fact, in some cases it only increased the generalization error.

5 Related Research

So far, none of the existing *ITS*s learn the model from data, but there have been some experiments to this direction. Two studies have tackled quite a similar problem to ours, but the data sizes were much larger and the data sets contained very different features. The prediction accuracy in these studies was of the same magnitude as in our case, although in the second study, the prediction was done in the end of course.

Kotsiantis et al. [5] have compared six classification methods (Naive Bayes, decision tree, feed-forward neural network, support vector machine, 3-nearest neighbour and logistic regression) to predict drop-outs in the middle of course. The data set contained demographic data, results of the first writing assignments and participation to group meetings. The data set contained 350 students. The best classifiers, Naive Bayes and neural network, were able to predict about 80% of drop-outs.

Minaei-Bidgoli et al. [6] have compared six classifiers (quadratic Bayesian classifier, 1-nearest neighbours, k-nearest neighbours, Parzen window, feed-forward neural network, and decision tree) to predict the course final results from a learning system log data. The data contained attributes concerning each task solved and other actions like participating in the communication mechanism and reading support material. The data set contained 250 students. The best classifier, k-nearest neighbours, achieved over 80% accuracy, when the final results had only two classes (pass/fail).

6 Conclusions

In this paper, we have tackled a difficult problem of learning classifiers for *ITS* from real data. The main problem is the small size of educational data sets, and the traditional machine learning methods cannot be applied directly. However, with careful preprocessing and selection of modelling paradigms we can learn quite accurate classifiers.

We have given general outlines, how to classify successfully small data sets of both numeric and categorial data. We recommend especially variations of naive Bayes classifiers, which are robust, can handle mixed variables and produce informative results (class probabilities).

We have also reported our empirical study, where we compared five classification methods for predicting the course outcomes as early as possible. The data consisted of only exercise points and the data sets were very small (in *Prog.1* course 125 rows and in *Prog.2* course 88 rows).

For numerical data, we used multiple linear regression and support vector machine classifiers, and for categorial data, three variations of naive Bayes classifier (the standard model, *TAN* model and Bayesian multinets). All methods achieved about the same accuracy. In *Prog.1*, 80% accuracy was achieved in the middle of course and in *Prog.2* already in the beginning of course.

References

1. P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.
2. N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.
3. D. Geiger and D. Heckerman. Knowledge representation and inference in similarity networks and Bayesian multinets. *Artificial Intelligence*, 82:45–74, 1996.
4. P.J. Huber. Robust estimation of a location parameter. *Annals of mathematical statistics*, 35:73–101, 1964.
5. S.B. Kotsiantis, C.J. Pierrakeas, and P.E. Pintelas. Preventing student dropout in distance learning using machine learning techniques. In *Proceedings of 7th International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES)*, pages 267–274, 2003.
6. B. Minaei-Bidgoli, D.A. Kashy, G. Kortemeyer, and W. Punch. Predicting student performance: an application of data mining methods with an educational web-based system. In *Proceedings of 33rd Frontiers in Education Conference*, pages T2A13–T2A18, 2003.
7. V.N. Vapnik. *Statistical learning theory.* John Wiley & Sons, 1998.

Raising Confidence Levels Using Motivational Contingency Design Techniques

Declan Kelly and Stephan Weibelzahl

National College of Ireland, Dublin, Ireland
[dkelly, sweibelzahl]@ncirl.ie

Abstract. Motivation plays a key role in learning and teaching, in particular in technology enhanced learning environments. According to motivational theories, proper contingency design is an important prerequisite to motivate learners. In this paper, we demonstrate how confidence levels in an adaptive educational system can be raised using a contingency design technique. Learners that saw parts of a complete picture depending on their performance were more confident to solve the next task than learners who did not. Results suggest that it is possible to raise confidence levels of learners through appropriate contingency design and thus to automatically adapt to their motivational states.

1 Introduction

Motivation obviously plays a key role in learning and teaching. However, technology enhanced learning environments often fail to motivate learners. Teachers devote a lot of time to assess and increase their students' motivation. Experienced teachers understand that it is crucial to keep students motivated in order to achieve optimal learning results. This is underpinned by an overwhelming amount of research [6]. Students with high intrinsic motivation often outperform students with low intrinsic motivation [21], and students with high motivation engage more in learning activities and are more likely to complete a course [23]. Accordingly, successful teachers are able to detect the students' needs and preferences. They try to provide an environment that enables the students to achieve their goals. Empirical studies show that human teachers devote as much time to the achievement of students' motivational goals as to cognitive and informational goals [19].

The evident importance of motivation for learning, and the fact that learners differ in their motivational state, open promising perspectives for an adaptive educational system that adapts to these motivational states. In this paper we outline a framework for modeling the motivational states of learners. Some first empirical results indicate how parts of this framework, in particular how contingency design might be implemented in an adaptive educational system.

2 Motivational Theories

Motivation is an internal state or condition that activates behavior and gives it direction [17]. In particular, the motivation to learn is characterized by long-term,

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 535 – 544, 2006.

quality involvement in learning and commitment to the process of learning [1]. The concept of motivation (previously also called conation) has been the focus of many psychological studies. A wide spectrum of motivation theories has been developed to date. These include psychoanalytic theories [7], behavioral theories [25], humanistic theories [22], and various cognitive theories [22] [32]. In applied psychology such as organizational and educational psychology, value-expectancy theories have been shown to be fruitful.

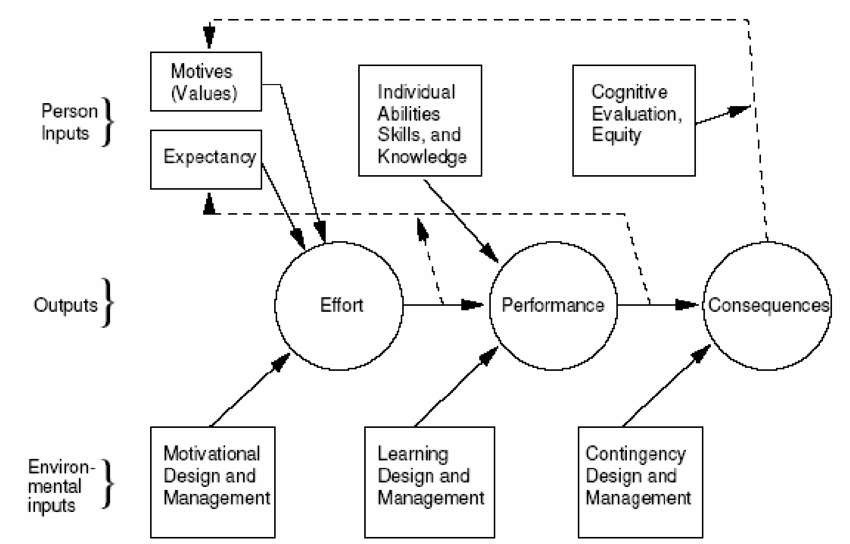


Fig. 1. Motivation theory of Keller (1983), adopted from de Vicente (2003)

One prominent example is Keller's theory of motivation in education [11]. The theory distinguishes three main outputs: effort (engaging in actions), performance (actual accomplishment) and consequences (intrinsic and extrinsic outcomes, e.g., emotional responses, social rewards, material objects). These outputs are influenced by person inputs as well as by environmental inputs (see Fig, 1). This theory provides various ways to influence students' motivation and thus their performance. The distinction between motivational design, learning design, and contingency design is very useful for the implementation of motivation strategies in technology enhanced learning systems as described below. In accordance with results presented by [30][31], we suggest that Keller's theory can serve as a framework for guiding the modeling of motivation in adaptive educational systems.

3 Adaptation to Motivational States

The adaptation to motivational states can be divided into two phases: assessment of the learner's motivational state and adaptation to these states with motivational strategies.

3.1 Assessment of Motivational States

Several different ways have been proposed so far to detect the motivational state of learners, including self-reports through sliders [30], behavioural cues in the interaction between learner and educational system [31], students' response times to tasks in combination with actual performance [3], as well as learner's attention, current task and expected time to perform [24]. Questionnaires and external standards can also be applied as external criteria of motivation [33] to validate the system's assessment.

3.2 Adaptation Strategies

Once the motivational state of the learner has been assessed correctly, there are several strategies to adapt to this state. According to Keller [12] motivation strategies can be categorized into motivational design, learning design and contingency design.

Motivational design addresses the learner's motivation directly in order to increase the effort put into a learning task. This can be done by communicating with the learner in a so called affective dialogue [27]. In particular, positive feedback and praise can have a positive impact on student motivation [29]. Motivational design might also aim at an improvement of students' self-efficacy, their attention to or perceived relevance of the topic[13].

Learning design aims at changing the content itself or selecting/recommending appropriate content according to the motivational state of the learner. This includes providing a variety of materials in order to avoid predictability and repeatability [26], involving the learners in active problem solving and divergent thinking [26], choosing activities that are meaningful and relevant to the student [4], and deciding whether the student may proceed to the next topic or not [9]. Effort behavior can also be scaffold by keeping learning activities short, using visual enhancement to support the activities, and intermingling information presentation screens with interactive screens [28]. The system might also adapt the difficulty of tasks [27] and offer help [9].

Contingency design aims at making the learner confident that effort and performance are closely coupled with consequences. This might include informing the learner about procedures (number of tasks, evaluation criteria) as well as using words and phrases that help attribute success to learner's effort and ability [28]. It has also been suggested to enhance the level of the learner's perceived control by introducing clear rules and performance criteria [20] and by offering immediate feedback [10].

The study reported in this paper focuses on the effects of contingency design in an adaptive system called EDUCE. The technique used in this study is based on gradual manifestation of pictures in dependence on performance.

4 Motivation Modeling in EDUCE

EDUCE is an adaptive intelligent educational system [14][16]that uses Gardner's theory of Multiple Intelligences (MI) as the basis for dynamically modeling learning characteristics and for designing instructional material [8]. The theory of Multiple Intelligences reflects an effort to rethink the theory of measurable intelligence embodied in intelligence testing. It is also a rich concept that offers a framework and a language for developing adaptive educational systems that supports creative, multimodal teaching [18].

The Multiple Intelligence theory states that there are eight difference intelligences that represent a different way of thinking, solving problems and learning. In EDUCE, four intelligences are used to develop instructional resources and model the learner. These are defined as the verbal/linguistic, logical/mathematical, visual/spatial and musical/rhythmic intelligences. The musical/rhythmic intelligence was chosen because it is not considered as an intelligence that can be used to deliver and inform the design of content yet the emotive power of music is widely acknowledged [5].

The motivational features of EDUCE can be described along Keller's theory as learning design and contingency design.

4.1 Learning Design in Educe

The learner's performance obviously depends on the individual abilities (Fig. 1). EDUCE recognizes the learner's MI profile using a predictive engine. The current version uses the following four intelligences in modeling the student:

- Logical/Mathematical intelligence (LM) - This consists of the ability to detect patterns, reason deductively and think logically.
- Verbal/Linguistic intelligence (VL) - This involves having a mastery of the language and includes the ability to manipulate language to express oneself.
- Visual/Spatial intelligence (VS) - This is the ability to manipulate and create mental images in order to solve problems.
- Musical/Rhythmic intelligence (MR) - This encompasses the capability to recognize and compose musical pitches, tones and rhythms.

EDUCE builds a dynamic model of the student's MI profile by observing, analyzing and recording the student's choice of MI differentiated material. Other information also stored in the student model includes the navigation history, the time spent on each learning unit, answers to interactive questions and feedback given by the student on navigation choices.

EDUCE holds a number of tutorials designed with help of subject matter experts. Each tutorial contains a set of content explaining a particular subject area. For the experiment described in this paper, Science is the subject matter. A tutorial consists of learning units that explain a particular concept. In each unit there are four different sets of learning resources, each based predominantly on one of the intelligences. The different resources explain a topic from a different angle or display the same information in a different way. Students are adaptively guided through this material based on their dynamic MI profile [15].

4.2 Contingency Design in EDUCE

Contingency design aims at making the learner confident that effort and performance are closely coupled with consequences. Each learning unit in EDUCE consists of several distinct stages. The first stage aims to attract the learner's attention (Fig. 2), the second stage provides a set of different MI resources, the third stage re-enforces the key message in the lesson and the final stage presents interactive questions on the

Fig. 2. The Awaken stage of "Opposites Attract" with four options for different resources

topic. After accessing the second stage, students may repeatedly go back and use the same or different MI resource.

The question phase offers an ideal opportunity to implement contingency design. EDUCE displays a picture which partially reveals itself based on the number of correct questions answered. In this particular tutorial a cartoon was divided into nine pieces. The meaning would not be revealed unless the learners answered all questions correctly and all nine parts are actually shown.

According to Keller, such a contingency design would increase the learners' confidence in answering questions and their performance. We designed an empirical study that aimed to demonstrate these effects.

5 Experimental Design

In order to explore the effect of Contingency Design on confidence levels and learning performance we set-up and experimental study with students. In particular the impact of the independent variable, picture strategy, on two dependent variables, learning performance and confidence levels was explored.

The picture strategy for raising confidence levels encompasses two main strategies:

1. *Show Picture*: After each question the student answers, the overall progress of the student is displayed through the use of a partially displayed picture. For example if 9 questions have been answered correctly then 4 pieces of the picture are displayed. In total there is 9 pieces in a full picture. Fig. 3 dis-plays a sample picture a student may see.
2. *Without Picture*: Here after the student answers a question no feedback is provided. The student continues to move through the tutorial without any feedback on progress.

To determine confidence levels, before each question, the student was asked the question: *Do you feel able to answer the next question?* The response to the question was gathered using a 4 point likert scale with 1 = no and 4 = yes. These answers are collated to determine confidence levels throughout the tutorial.

Learning performance is defined by the post-test score and the learning gain. To calculate the relative learning gain each student before and after a tutorial sits a pre-test and post test. The test for the pre-test and post-test is the same and consists of questions that appear during the tutorial. The questions are multi-choice question with four options.

The experiment was conducted during one session on the computer, with each student spending on average 18 minutes exploring one tutorial. The session was preceded by a pre-test and followed by a post-test. The pre-test and post-test had the same 10 multi-choice questions, which were mostly factual. Students were randomly assigned to one of the two groups defined by the picture strategy: *show picture* and *without picture*. Within each group, students explored one of two different tutorials: *static electricity* or *electricity in the home*. They were also assigned to one of two groups, one group was forced to explore all the MI informed resources, the other group was free to progress through the tutorial after looking at least at one resource.

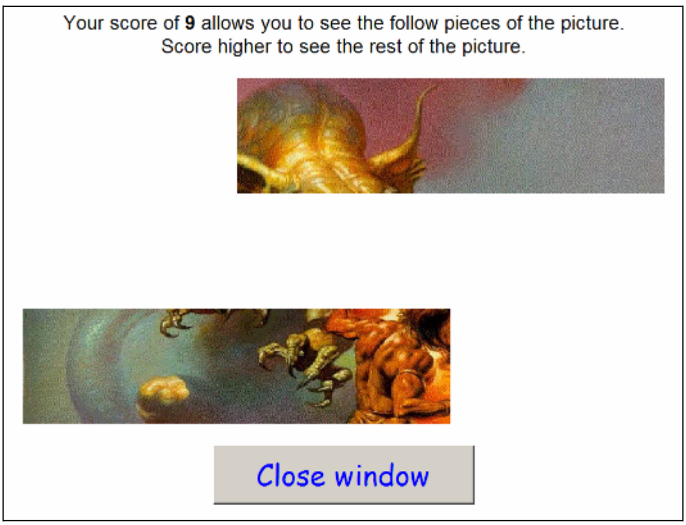


Fig. 3. Picture with fours pieces displayed

5 Results

35 boys and girls participated in the study. The ages ranged from 12 to 17, with an average age of 14. The students were randomly assigned to one of the two versions. 13 students were assigned to the show picture version and 19 students were assigned to the do not show picture version. Each student spent on average 18 minutes exploring one tutorial. The students were participating in a “Discovering University” program being run in the author’s place of work. The objective of the program was to give students the experience of third level education and to encourage them to

continue education in university. The students attending this program would primarily be from areas designated as disadvantaged in terms of the number of students who participate in third level education. The study itself was conducted in the computer laboratories in the college and took place within the 'Computer' sessions on the Discovering University program. No reward incentives were provided to the students who participated.

The results were analysed from two perspectives, the effect of the picture strategy on confidence levels and performance levels. It was expected that the show picture strategy would raise performance and confidence levels.

A one-way ANOVA was conducted to explore the impact of the picture strategy on the post-test score. Disappointingly the results were not statistically significant, however post-test score for the picture strategy (M=55.4, SD=21.45) was slightly higher that the post-test for the no picture strategy (M=54.2, SD=20.63). A similar analysis was also conducted on the relative gain score and the results were also not statistically significant. One reason may be that the assessment instrument was not sensitive enough to measure differences in performance levels.

To determine if picture strategy had impact on confidence levels, the change in confidence level between the beginning and the end of the tutorial was analysed. To determine the change in confidence, the responses to the confidence question were aggregated. For each section, the total confidence level or the sum of all responses to the confidence questions was calculated. Subsequently, the change in confidence level was calculated by finding the difference between the confidence level at the end of tutorial and the beginning of the tutorial.

A one-way ANOVA was conducted to explore the impact of the picture strategy on the change in confidence level. The results were statistically significant: $F(1, 31) = 6.613$, $p = .05$. Post-hoc comparisons using the Tukey HSD test indicated that the mean score for the show picture group (M=19.23, SD=24.33) was significantly different from the no show picture group (M=3.73, SD=16.6). Fig. 4 illustrates how the confidence level changes over time.

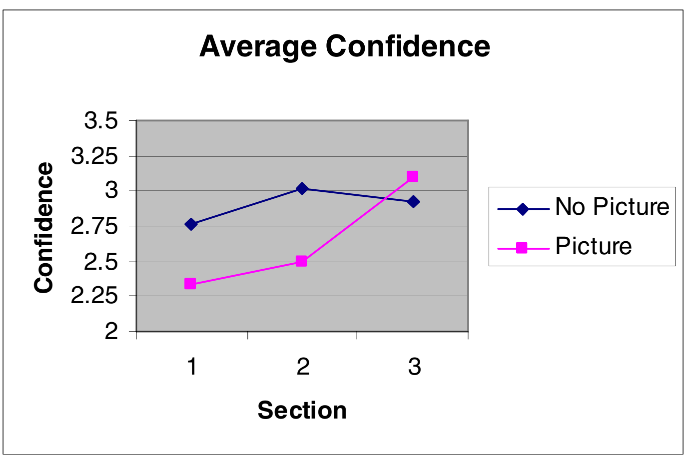


Fig. 4. Average Confidence Levels for Picture and No Picture group

The results suggest that the confidence levels of students could be raised by showing the performance level through the form of a partially displayed picture. The suggested that students could raise their confidence levels by being able to immediately link performance level to effort.

Together the results indicate that the picture strategy had an impact on the confidence level of students. However in this experiment an increase in confidence level did not translate into an increase in performance in the post-test.

6 Conclusions

This paper has described a framework for supporting motivation in an adaptive system. It has demonstrated how the EDUCE adaptive systems can be categorized according to this framework. It has also described an experiment that measures on aspect of this framework, contingency design, or the confidence that effort is linked to performance.

Results indicate that confidence levels can be raised through the use of contingency design. In EDUCE this was implemented using a picture that reveals more as the student gets more questions right. The implications for the design of adaptive systems are that it is possible to raise confidence levels through contingency design. It may be that for students with low confidence levels, an intelligent tutoring system may use adaptive strategies to support contingency design. The challenge of future work is to determine how to automatically detect confidence levels and to determine the relationship between confidence level and performance. Future work will also address other aspects of the motivational framework such as motivational design in order to increase motivation in learning environments.

References

1. Ames, C. (1992). Classroom goals, structures, and student motivation. *Journal of Educational Psychology, 84*(3), 261-271.
2. Bandura, A. (1997). *Self-efficacy: The exercise of control*. New York: W. H. Freeman.
3. Beck, J.E. (2004). Using response times to model student disengagement. *Proceedings of Workshop on Social and Emotional Intelligence in Learning Environments*. Held at International Conference on Intelligent Tutoring Systems, ITS 2004, Maceio, Brazil, August 31, 2004.
4. Beffa-Negrini, P.A., Cohen, N.L., & Miller, B. (2002). Strategies to motivate students in online learning environments. *Journal of Nutrition Education and Behavior*, 34, 334-340.
5. Carroll, K., &. (1999). Sing a Song of Science. Chicago: Zephyr Press.
6. Corno, L. (2001). Conative individual differences in learning. In: J.M. Collis & S. Messick (Eds.). *Intelligence and Personality: Bridging the gap in theory and measurement* (pp. 121-138). Mahwah: Lawrence Erlbaum
7. Freud, S. (1990). *Beyond the pleasure principle*. New York: W. W. Norton & Company.
8. Gardner H. (1983). *Frames of Mind: The theory of multiple intelligences*. New York: Basic Books.
9. Georgouli, K. (2002). The Design of a 'Motivating' Intelligent Assessment System. In: *Proceedings of the International Conference on Intelligent Tutoring Systems*, ITS 2002 (pp. 811-820), Berlin: Springer.

10. Jones, K. (1989). Running, or stumbling through, simulations. *Simulation/Games for Learning*, 19 (4), 160-167.
11. Keller, J.M. (1979). Motivation and instructional design: A theoretical perspective. *Journal of Instructional Development*, 2(4), 26-34.
12. Keller, J.M. (1983). Motivational design of instruction. In: C. M. Reigeluth (Ed.) *Instructional Theories and Models: An Overview of Their Current Status* (pp. 383 – 434). New York: Lawrence Erlbaum.
13. Keller, J.M. (1999). Using the ARCS Motivational Process in Computer-Based Instruction and Distance Education. In M. Theall (Ed.), *Motivation in Teaching and Learning: New Directions for Teaching and Learning*. San Francisco: Jossey-Bass.
14. Kelly, D. (2003). A Framework for using Multiple Intelligences in an ITS. Proceedings of EDMedia'03, World Conference on Educational Multimedia, Hypermedia & Telecommunications, Honolulu, HI.
15. Kelly, D., & Tangney, B. (2004). Predicting Learning Characteristics in a Multiple Intelligence based Tutoring System. Paper presented at the Seventh International Conference on Intelligent Tutoring Systems, ITS'04, Maceio, Brazil, 679-688.
16. Kelly, D., & Tangney, B. (2005). Matching and Mismatching Learning Characteristics with Multiple Intelligence Based Content. Paper presented at the Twelveth International Conference on Artificial Intelligence in Education, AIED'05, Amsterdam, Netherlands, 354-361.Reigeluth, C.M. (1996): A new paradigm of ISD ? Educational Technology, 36(3)
17. Kleinginna, P., & Kleinginna A. (1981). A categorized list of motivation definitions, with suggestions for a consensual definition. *Motivation and Emotion, 5*, 263-291.
18. Lazaer, D. (1999). *Eight Ways of Teaching: The Artistry of Teaching with Multiple Intelligences*, SkyLight.
19. Lepper, M. R., & Hodell, M. (1989). Intrinsic motivation in the classroom. In C. Ames & R. Ames (Eds.), *Research on motivation in education (Vol. 3)* (pp. 73-105). San Diego: Academic Press.
20. Malone, F. (1980). Toward a theory of intrinsically motivation instruction. *Cognitive Science*, 5 (4), 333–369.
21. Martens, R., Gulikers, J., & Bastaens, T. (2004). The impact of intrinsic motivation on e-learning in authentic computer tasks. *Journal of Computer Assisted Learning*, 20 (5), 368-376.
22. Maslow, A. (1954). *Motivation and personality*. New York: Harper.
23. Militiadou, M., & Savenye, W. (2003). Applying social cognitive constructs of motivation to enhance student success in online distance education. *AACE Journal*, 11(1), 78-95.
24. Qu, L. & Johnson, W.L. (2005). Detecting the Learner's Motivational States in an Interactive Learning Environment. In: C.-K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds). *Proceedings of the 12th International Conference on Artificial Intelligence in Education AIED 2005* (pages 547-554). Amsterdam: IOS Press.
25. Skinner, B.F. (1969). *Contingencies of reinforcement: A theoretical analysis*. New York: Appleton-Century-Crofts.
26. Stipek, D.J. (1993). *Motivation to learn: From Theory to practice*. Needham Heights: Allyn and Bacon.
27. del Soldato, T. (1992). Detecting and Reacting to the Learner's Motivational State. In: *Proceedings of International Conference on Intelligent Tutoring Systems*, ITS 1992 (pp. 567-574). Berlin: Springer.
28. Song, S.H., & Keller, J.M. (2001). Effectiveness of motivationally-adaptive CAI. *Educational Technology Research & Development*, 49(2), 5 - 22..

29. Terrell, S., & Rendulic, P. (1996). Using computer-managed instructional software to increase motivation and achievement in elementary school children. *Journal of Research on Computing in Education*, 28 (3), 403-414.
30. de Vicente, A., & Pain, H., (2002). Informing the detection of the students' motivational state: An empirical study. In: S.A. Cerri, G. Gouardères, & F. Paraguaçu (Eds.), *Proceedings of the International Conference on Intelligent Tutoring Systems, ITS2002* (pp. 933-943). Berlin: Springer.
31. de Vicente, A., & Pain, H. (2003) Validating the Detection of a Student's Motivational State. In: A. Mendez Vilas, J. A. Mesa Gonzalez, & J. Mesa Gonzalez (Eds.). *Proceedings of the Second International Conference on Multimedia Information & Communication Technologies in Education* (m-ICTE2003) (pp. 2004-2008).
32. Vroom, V. (1964). *Work and motivation*. New York: Wiley.
33. Waugh, R. (2002). Creating a scale to measure motivation to achieve academically: Linking attitudes and behavior using Rasch measurement. *British Journal of Educational Psychology*, 72, 65-86.

Motivating the Learner: An Empirical Evaluation

Genaro Rebolledo-Mendez[1], Benedict du Boulay[1], and Rosemary Luckin[2]

[1] IDEAS Lab, Department of Informatics, University of Sussex
Brighton, BN1 9QH, UK
{G.Rebolledo-Mendez, B.du-Boulay}@sussex.ac.uk
[2] London Knowledge Lab, Institute of Education, University of London,
London WC1N 3QS, UK
r.luckin@ioe.ac.uk

Abstract. The M-Ecolab was developed to provide motivational scaffolding via an on-screen character whose demeanour defended on modelling the learner's motivational state at interaction time. Motivational modelling was based on three variables: effort, independence and the confidence. A classroom evalu-ation was conducted to illustrate the effects of motivational scaffolding. Students had an eighty minute interaction with the M-Ecolab, divided into two sessions. The results suggested a positive effect of the motivational scaffolding, particularly for initially de-motivated students who demonstrated higher learning gains. We found that these students followed the suggestions of the on-screen character which delivered personalized feedback. They behaved in a way that was conducive to learning by being challenge-seekers and displaying an inclination to exert more effort. This paper gives a detailed account of the methodology and findings that resulted from the empirical evaluation.

1 Introduction

Can we increase students' motivation to learn? This question has shaped the nature of research on motivation in education since the 1930's and constitutes an active field of research in Artificial Intelligence in Education [1]. Motivation has been understood as a crucial factor affecting learning behaviour and is a complex phenomenon influenced by a plethora of circumstances that surround the learning experience. Intelligent Tutoring Systems (ITS) have made use of some elements proposed in Theories of Motivation: examples include the works of Lepper [2], Malone [3] and Song [4]. Our approach consists of modelling the learner's motivational state during the interaction and adapting the motivating reactions according to the model's beliefs. We have elaborated on previous work [5, 6] and have added a motivational modeller for an existing ITS, the Ecolab [7, 8]. This paper describes an evaluation of M-Ecolab, the enhanced version of Ecolab. The aims of this paper are two-fold. First, we present findings with respect to the effects of the motivational scaffolding in M-Ecolab using the methodology employed in two previous evaluations of Ecolab [8, 9]. Second, we compare and contrast the outcomes of the three evaluations.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 545 – 554, 2006.

2 Motivational Scaffolding in the Ecolab

The Ecolab [7] is an ITS designed for teaching pupils aged 9-11 years-old concepts related to food chains and webs. It consists of a simulation of an ecology laboratory where students can add and/or remove organisms to explore feeding relationships. The set of actions that pupils can perform includes move, eat, be eaten, be predator, etc. The Ecolab laboratory can be viewed from three different perspectives. The 'world view' shows plants and animals as they would look in the real world. The 'energy view' shows, using bar graphs, the levels of energy associated with the organisms in the Ecolab. Finally, the 'web view' is a diagrammatic representation of the eating relationships represented in the system. The curriculum is divided into nodes with different versions of the system imposing more or less control over the order in which the nodes are tackled. Previous evaluations of the Ecolab system have illustrated the benefits of challenging students and guiding, but not controlling, their learning [7] and of offering the learners help at the meta-cognitive level by making low-ability learners more aware of their help-seeking needs [9]. The success of previous Ecolab systems is thought to derive from modelling the learner's cognitive and meta-cognitive traits. By considering the learners' ability and collaborative support at interaction time, the Ecolab is capable of altering the system's reactions for individual learners. Ecolab provides help at four levels of quality: the deeper the level, the greater the control taken by the system and the less scope there is for the pupil to fail [10].

To shed some light onto the effect of motivating the learner we developed M-Ecolab as an extension of the Ecolab software to provide motivational scaffolding. Various approaches have been taken to assess the degree of motivation in learning environments. Song and Keller [4], for example, utilized motivational self-assessment to provide appropriate motivating techniques to the learner. Our approach, however, revolves around *modelling* three motivational traits identified as key in learning contexts [11]: effort, confidence and independence from the tutor. In our model, effort modelling considers quality and quantity of the actions within the software, and the persistence that learners display when facing errors. Independence is modelled considering the degree of help provided by the system. Confidence is modelled based on the degree of challenge-seeking that learners display during the interactions. The rationale for motivational modelling is that the system can react differently to learners in different states of motivation via a model of the learners' motivation, built by assessing their actions, cognitive and meta-cognitive states and relating them to the motivational variables previously described. Since the original Ecolab was based on a Vygotskyan model [7], an explicit "more able" partner has been incorporated in the M-Ecolab as a motivating element through the use of an on-screen character called Paul. We provide motivational scaffolding consisting of spoken feedback given at two times, pre- and post-activity. Pre-activity feedback is inevitable and informs the learner of the objectives of that learning node; post-activity feedback, on the other hand, offers comments to help learners reflect on their behaviour at that node. Since the system maintains motivational models for individual learners the feedback given by Paul at post-activity time is adjusted. The adjustment is underpinned by the motivational model and consists on alterations of Paul's voice tone and gestures. According to the model's perception of the learner's motivation/de-motivation Paul encourages the learner: to exert more effort, to be more independent or to become more confident. For example, if the motivational modelling determines a low state of motivation because the quality of the actions was poor, Paul's post-activity feedback states: "For the next node try to make fewer errors". Under these circumstances, Paul's face would reflect concern. For a more detailed description of M-Ecolab interactions refer to [12]. We also included a quiz as motivating element. It is available during the interaction via a button within the interface but constitutes just an option that learners can activate at will. If activated, the quiz asks questions related to the topic of food-chains. Wrong answers are not corrected but an indicator reflects the number of correct and incorrect answers that the learner has given during the interaction. Correct answers are praised; a maximum of three questions is allowed per learning node in order to prevent the learner concentrating more on the quiz than on the main learning activities.

3 Evaluating the M-Ecolab

To measure the influence of the motivational scaffolding on the learners' behaviour and to try to establish its impact in comparison to previous Ecolab assessments, an evaluation of the M-Ecolab was made in a local primary school during March 2005. We assessed students' knowledge of food webs and chains employing isomorphic pre- and post-tests experiment time. This test was also used in previous evaluations [8, 9]. Please note that the questions used in the pre- and post-tests were different from those of the quiz. The learners' *initial* motivation using the system was assessed via an adaptation for British primary schools of Harter's test [13]. We chose Harter's test as its reliability has been demonstrated and it is, arguably, the scale most widely used for measuring children's individual differences in motivation. There were 19 learners who employed M-Ecolab, 9 girls and 10 boys: all members of three fifth grade classes, aged 9-10. All participants had been exposed to the standard, non computer-based teaching of food-chains *prior* to the study. They were asked to complete the pre-test for 15 minutes and then Harter's test for a further five minutes. Two weeks later M-Ecolab was demonstrated with the use of a video-clip showing its functionality. At this point the researcher answered questions regarding the use of the software. One tablet PC loaded with M-Ecolab was provided for each learner. The students were then allowed to interact with it for 40 minutes. A week later a second interaction session took place for a further 40 minutes. Immediately after the second interaction the pupils were asked to complete the post-test. The participants were not taught about the topic of food chains *between* sessions.

4 Results

The previous evaluations of Ecolab looked at how two variations of the software affected participants' learning of feeding relationships according to the learners' ability (or skill) [8, 9]. The criterion employed to divide the sample was the students' results of their Science SAT (Standard Assessment Test – a national test used in the

UK) using a tertile split. We recognize that this method of analysis does not always provide the best approach to analyze differences [14]; however, we chose to follow this method in order to be consistent with precedent evaluations. We categorized children into 3 ability groups, see Table 1.

Table 1. Pre- and post-test scores according to ability

Ability	Pre-test score Mean (SD)	Post-test score Mean (SD)
Low (n = 6)	15.0 (6.03)	21.17 (6.18)
Average (n=9)	18.56 (4.95)	24.0 (4.03)
High (n = 4)	28.50 (5.92)	29.50 (2.08)

Motivation in the M-Ecolab

We wanted to explore the motivational development that learners experienced through their use of M-Ecolab. To that effect we analyzed students considering their motivational state both before and during the interaction. We acknowledge that the scales to assess motivation *before* (Harter's scale) and *during* (model of motivation specific to M-Ecolab) the interaction are different and that a better rationale should be used in future evaluations. Nevertheless, these two indications were combined to make four groups to analyze the effects of motivating techniques on learners (see Table 2):

- Group 1. Motivated students *before* and *during* the interaction (MM).
- Group 2. Motivated student *before* with low motivational *during* the interaction (MD).
- Group 3. De-motivated students *before* with high motivation *during* the interaction (DM).
- Group 4. De-motivated students *before* and *during* the interaction (DD).

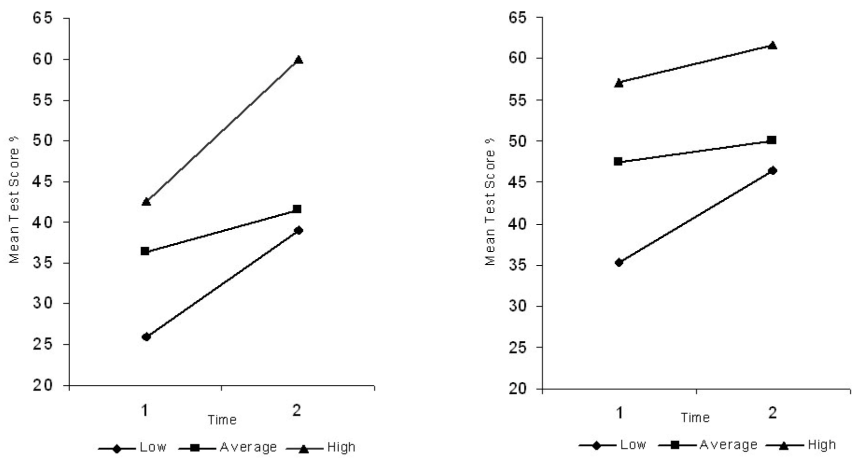
Table 2. Distribution of M-Ecolab students considering their motivational change group

Group	Population	Effort Mean (SD)	Independence Mean (SD)	Confidence Mean (SD)	Learning Gain
1	n = 6	.54 (.11)	.71 (.07)	.60 (.05)	12.87 %
2	n = 6	.38 (.09)	.42 (.09)	.65 (.14)	17.42 %
3	n = 4	.58 (.06)	.67 (.09)	.64 (.14)	40.90 %
4	n = 3	.30 (.12)	.41 (.07)	.61 (.32)	27.27 %

Table 2 reveals that the majority of students had high motivation *before* the interaction (Groups 1 and 2). Interestingly, their learning gains were the lowest for the population. By contrast, it was the students of Group 3 those with the highest percentage of learning gains. Not surprisingly members of Group 3, with three low and one average ability pupil, all had above average learning gains. We also analyzed differences in terms of the motivational variables (effort, independence and confidence) for pupils with higher learning gains (Groups 3 and 4). We found significant differences in effort (t(5)=3.932, p=.011) and independence (t(5)=4.054, p=.010) but not in their confidence. We also found that members of Group 3 *followed* the suggestions provided by Paul at post-activity feedback more than members of other groups. We speculate whether this factor could explain the differences in levels of effort and independence observed and the differences in the percentages of learning gains. The increase of motivation and learning gains observed in members of Group 3 was encouraging and led us to further investigate the interaction characteristics of learners with initial low motivation. Although the low learning gains for high motivation students is interesting, we focused our attention here on understanding the behaviours and characteristics for less motivated pupils.

Comparisons to Previous Evaluations

In Ecolab I [7] high ability students improved more than other abilities as we found a significant within-subjects pre- to post-test difference. In Ecolab II [9] low ability students benefited most from the meta-scaffolding provided (Fig. 1a & 1b).



a. Ecolab I b. Ecolab II

Fig. 1. Ability by testing time, Ecolab 1 and Ecolab II

In M-Ecolab the situation was different as it was both average and low ability students who improved their scores significantly from pre- to post-test (see Fig. 2, please note that the mean test represent percentages and not actual values as those of Table 1). One of our concerns while drawing comparisons to previous Ecolab evaluations was the difference observed, at pre-test time (T1), among children's ability at three different schools. We believe, in a similar way to [9] that, this difference is not explained by discrepancies in abilities among samples but is more likely to be an effect of shorter periods of time elapsed from the learning of the concepts of food chains in a non-computer fashion and the use of the pre-test. An analysis of variance (ANOVA) of the pre- and post-tests scores using one within-subjects variable (time: 1 = pre-test, 2 = post-test), and one between-subjects variable (ability: high, average, low) indicated the difference between the ability groups in M-Ecolab was significant ($F(2,16) = 4.022$, $R^2 = .251$, $p=.038$).

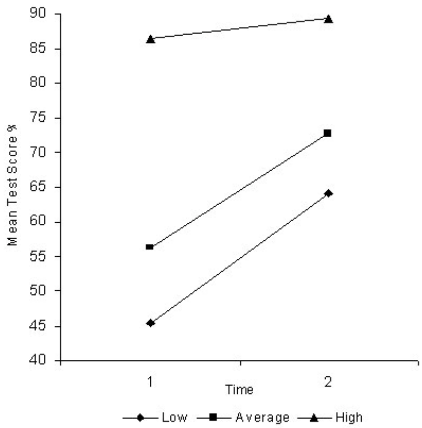


Fig. 2. Ability by testing time, M-Ecolab

We will now examine the following: What were the learners' characteristics and behaviours that may have accounted for the increase in performance observed in low and average ability students? What was the type of help received by the learners? What was the impact of the motivating techniques on the learners? And how do these compare with previous Ecolab evaluations?

Nature of M-Ecolab Interactions

To throw some light onto these questions we looked at the records of the interactions, maintained in log files kept for individual M-Ecolab learners. These records were examined to reveal both the *character* of the interactions and the *type of help* provided by the system. The relevance of using behaviours is that in combination with the students' ability, learning gains and motivation we can gather evidence of what might have constituted a fruitful interaction in M-Ecolab. To be consistent with previous evaluations we considered existing definitions of behaviours [8]:

- **Challenge-seeking** is an interaction trait associated with a learner's inclination to accept challenging activities. At the beginning of each learning node, Ecolab provides opportunities for children to select among 3 different levels of challenge. This behaviour was referred to as *exploration* in previous evaluations [9].

However, we believe that a more descriptive name should be used. The opposite behaviour is known as challenge-avoiding.

- **Busyness** is an interaction characteristic determined by measuring the number of actions of any type such as help request, adding or deleting organisms, etc. An above average number of actions categorized a student as busy or quiet otherwise.
- **Hopping** is a behaviour associated with a learner who switches frequently from one type of view to another. These interactions contain no or few series of repeated actions of the same type. The opposite of a hopping conduct is known as a persistent behaviour.

Because the essence of M-Ecolab was to provide motivating strategies to de-motivated students we were concerned about the degree of "distraction" that learners could have had during their interactions due to an excessive use of the quiz. To have an insight into how this affected students we have defined a new behaviour:

- The degree of **quiz-using** that students experienced during their interaction with M-Ecolab was considered. Learners who visited the quiz an above average number of times were considered quiz-seekers or quiz-avoiders otherwise.

In previous Ecolab evaluation the importance of the challenge-seeking behaviour has been highlighted [9]; it was found that this behaviour was present amongst students with above average learning gains (92% in Ecolab II, 82% in Ecolab I). However, there was a discrepancy between Ecolab I and Ecolab II regarding the composition of the groups of challenge-seekers: In the Ecolab II there was evidence that the group of challenge-seekers was composed of all three ability groups, as opposed to a majority of high ability students in Ecolab I. This discrepancy was thought to be the effect of meta-cognitive scaffolding provided by Ecolab II [9]. We analyzed whether the M-Ecolab produced the same effect and found that challenge-seeking was an important trait when motivating elements were present. 58% of students with above average learning gains, belonging to Groups 3 and 4, were challenge-seekers, see Table 3.

Table 3. Distribution of students considering their average learning gains and behaviours

	Challege-seekers n=10	Quiz-avoiders n = 10	Persisters n = 9
Students with above average learning gains n = 12	n = 7	n = 8	n = 8
Students with below average learning gains n = 7	n = 3	n = 2	n = 1

We found that students with above average learning, including both low and average ability, were not inclined to use the quiz. We also found that 67% of students with above average learning gains were persisters. Furthermore, when we combined behaviours to form duplets, we found that only 50% of learners with above average learning gains were both **quiz-avoiding** and **persisters**. This suggests an effect of motivating techniques in M-Ecolab which is different from both Ecolab I and II evaluations. In earlier studies it was found that the combination of busyness and challenge-seeking yielded better learning outcomes: 71% of Ecolab II and 70% of Ecolab students with above average learning gains had this behaviour.

Help-Seeking in the M-Ecolab

Two traits of the interaction from previous Ecolab evaluations were used to denote help usage in M-Ecolab:

- Students who sought an above average quantity of help were considered to have had **lots** of help, or little otherwise.
- Similarly, pupils who requested greater levels of help were contemplated as having had **deep** help, or shallow otherwise.

The study of help provision in M-Ecolab had a particular relevance for us because in a pilot study [12], where pupils used M-Ecolab for 40 minutes, we found that the motivating techniques made an impact upon the *qualities and quantities* of help selected between two experimental group. We found that M-Ecolab learners were significantly *less* independent from the system's help as they requested greater qualities and quantities of help. Our analysis showed, however, that the findings of the pilot study were not replicated. We believe that the factor that accounts for this discrepancy is the total interaction time (40 minutes for the pilot, 80 minutes for this evaluation). It may have been the case that a longer interaction had an impact on the learners' behaviours, particularly regarding help-seeking.

Nevertheless, by analyzing the help requested by low ability pupils with above average learning gains, we found that these students used *lots and shallow help* and that average ability students with above average learning gains utilized *little and deep help*. This difference in help-seeking behaviour suggests that successful students used a very different help-seeking strategy depending on their ability.

5 Discussion

Our findings suggested that by modelling motivation and adjusting the motivational reaction initially de-motivated students significantly increased their post-test scores. We also found that low and average ability students also improved their post-test performance. However, we also found that neither initially highly motivated nor high ability students increased their post-test scores; we are aware, however, that this result could be due to a "ceiling effect" (see Fig. 2). In order to find the specific causes of learning gains in de-motivated, and in low and average ability students we analyzed the learners' interactions following the approach taken by Luckin [8], and catalogued students according to behaviours. We noted that our results are consistent with previous findings [9] in the sense that *seeking for challenge* is a characteristic of learners with higher learning gains. Further analyses suggested that the combination of behaviours that yielded better learning gains for low and average ability students was that of being *quiz-avoiding and a persister* as opposed to being busy and challenge-seeker in previous Ecolab evaluations. Regarding help-seeking, we found that successful students used very different strategies depending on their ability: low ability students used lots and shallow help whereas average ability students utilized little and deep help.

Regarding initially de-motivated learners, we analyzed whether successful students (belonging to Group 3 in Table 2) varied their effort, independence or confidence during the interaction. We found that students with above average learning gains had significantly higher degrees of effort ($t(5) = 3.932$, $p = .011$) and independence ($t(5) = 4.054$, $p = .01$). Interestingly, we found that successful students *explicitly followed* the post-activity feedback provided by an on-screen character who adjusted his tone of voice and facial expressions considering individual learners' motivation. Curiously, initially motivated students did not vary their behaviour much during the interaction tending to have similar values for effort, independence and confidence during 80 minutes of interaction. We believe that the presentation of motivating techniques to motivated learners was not beneficial. This is consistent with Keller's [15] suggestions that motivating techniques should be used with care for high motivation students.

These results have suggested an effect on learning of the motivating techniques; more importantly, these effects were different depending on the students' ability and motivation. These results have also suggested an important influence of spoken feedback (adjusted considering the learners' motivational state) at post-activity time. However, we acknowledge that these results have been derived from a very small sample (n=19). We also acknowledge that our motivational modelling needs further development: we intended it to present motivating strategies to de-motivated learners only and not to all the sample as was the case. We also think that adapting the feedback and character's reactions, in conjunction with a quiz, constitute only a first step for the study of motivating techniques in ITS's and that a wider range of possibilities could be also considered in future research. But despite these drawbacks we believe that the findings reflect an interesting effect of motivating de-motivated or low and average ability students. These results also suggest general guidelines that could be used to improved students' motivation in ITS's. The results should be interpreted only as an indication of the effects of motivating learners in ITS's and as general pointers for future research on motivation.

Can we be motivated to learn? Although the topic of motivation in tutoring systems is a vast field involving both affective and cognitive states, we believe that the design of ITS's could include motivating elements that might be conducive of learning particularly for low ability students. If our ITS's are to motivate students we need to provide a means of recognizing the causes of de-motivation, particularly lack of effort or excess of dependency on the system's help, and encourage learners to improve these behaviours.

Acknowledgments

We thank Mr. M. Ayling and Greenway Primary School in Horsham, UK. This work has been partially granted by the CONACyT and Veracruzana University.

References

1. Weiner, B., *History of Motivational Research in Education.* Journal of Educational Psychology, 1990. **82**(4): p. 616-622.
2. Lepper, M.R. and R.W. Chabay, *Socializing the Intelligent Tutor*, in *Learning Issues for Intelligent Tutoring Systems*, H. Mandl and A.M. Lesgold, Editors. 1988, Springer - Verlag: New York.
3. Malone, T. and M. Lepper, *Making learning fun*, in *Aptitude, Learning and Instruction: Conative and Affective Process Analyses*, R. Snow and M. Farr, Editors. 1987, Lawrence Erlbaum.
4. Song, S.H. and J.M. Keller, *Effectiveness of Motivationlly Adaptive Computer-Assisted Instruction on the Dynamic Aspect of Motivation.* Educational Technology Research and Development, 2001. **49**(2): p. 5-22.
5. Del Soldato, T. and B. du Boulay, *Implementation of motivational tactics in tutoring systems.* International Journal of Artificial Intelligence in Education, 1995. **6**: p. 337-378.
6. De Vicente, A. and H. Pain. *Informing the Detection of the Student's Motivational State: An empirical Study*, in *Intelligent Tutoring Systems : 6th International Conference, ITS 2002*. Proc. Editors: S.A. Cerri, G. Gouardères, F. Paraguaçu. Page 933, Berlin : Springer.
7. Luckin, R. and B. du Boulay, *Ecolab: The Development and Evaluation of a Vygostskian Design Framework.* International Journal of Artificial Intelligence, 1999. **10**: p. 198-220.
8. Luckin, R., *´Ecolab´: Explorations in the Zone of Proximal Development, PhD Thesis*, in *School of Cognitive & Computer Sciences, University of Sussex.* 1998, University of Sussex: Brighton, BN1 9QH.
9. Luckin, R. and L. Hammerton, *Getting to know me: helping learners understand their own learning needs through Metacognitive scaffolding*, in *Intelligent Tutoring Systems : 6th International Conference, ITS 2002*. Proc. Editors: S.A. Cerri, G. Gouardères, F. Paraguaçu. Page 759, Berlin:Springer.
10. Wood, D.J. and H.A. Wood, *An experimental evaluation of four face to face teaching strategies.* International Journal of Behavioural Development, 1978. **1**: p. 131-147.
11. Rebolledo, G. *Motivational Modelling in a Vygotskyan ITS*, in *Artificial Intelligence in Education : 11th International Conference, AIED 2003*. Proc. Editors: U. Hoppe, F. Verdejo, J. Kay. Page 537-538, Amsterdam: IOS Press.
12. Rebolledo Mendez, G., B.d. Boulay, and R. Luckin. *"Be bold and take a challenge": Could motivational strategies improve help-seeking?* in *Artificial Intelligence in Education : 12th International Conference, AIED 2005.* Proc. Editors: CK. Looi, G. McCalla, B. Bredeweg, J. Breuker. Page 459-466. Amsterdam: IOS Press.
13. Harter, S., *A new self report scale of intrinsic versus extrinsic orientation in the classroom: motivational and informational components.* Developmental Psychology, 1981. **17**(3): p. 300-312.
14. Wright, D.B., *Comparing groups in a before-after design: when t-tests and ancova produce different results*. 2004: Brighton. p. 23.
15. Keller, J.M., *Motivational Design of Instruction.*, in *Instructional-Design theories and models: An overview of their current status*, C.M. Reigeluth, Editor. 1983, Erlbaum: Hillsdale. p. 383-434.

Approximate Modelling of the Multi-dimensional Learner*

Rafael Morales, Nicolas van Labeke, and Paul Brna

The SCRE Centre, University of Glasgow
11 Eldon Street, Glasgow G3 6NH, United Kingdom
Tel.: +44 (141) 330-3490; Fax: +44 (0)141 330 3491
{r.morales, n.vanlabeke, paul.brna}@scre.ac.uk

Abstract. This paper describes the design of the learner modelling component of the LEACTIVEMATH system, which was conceived to integrate modelling of learners' competencies in a subject domain, motivational and affective dispositions and meta-cognition. This goal has been achieved by organising learner models as stacks, with the subject domain as ground layer and competency, motivation, affect and meta-cognition as upper layers. A concept map per layer defines each layer's elements and internal structure, and beliefs are associated to the applications of elements in upper-layers to elements in lower-layers. Beliefs are represented using belief functions and organised in a network constructed as the composition of all layers' concept maps, which is used for propagation of evidence.

1 Introduction

The description given by the ADL Initiative of modern e-learning systems combines a content-based approach from computer based training with adaptive educational strategies from intelligent tutoring systems [1]. This mixture of approaches produces tensions in the design of these systems, particularly in the design of their learner modelling subsystem, which aim at supporting a wide range of adaptive educational strategies—e.g. from coarse-grain construction of e-books to tailored natural language dialogue [2]—with a general lack of what is traditionally afforded by ITS systems: painfully designed and dynamically constructed learning activities capable to provide large amounts of specific and detailed information on learner behaviour.

This type of e-learning systems make heavy use of pre-authored educational content to support learning, hoping to capitalise on the expertise of a variety of authors in producing educational materials. However, educational content is for the most part opaque to learner modelling due to the absence of domain expert subsystems to query about it. The information available is hence reduced to what is explicitly provided by

* This publication was generated in the context of the LeActiveMath project, funded under the 6th Framework Programm of the European Community - (Contract N° IST- 2003-507826). The authors are solely responsible for its content, it does not represent the opinion of the European Community and the Community is not responsible for any use that might be made of data appearing therein.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 555–564, 2006.

authors as *metadata* [3]. Unfortunately, provision of metadata is a heavy burden on authors because it amounts to work be done twice: to say something and to say what it was said. The more detailed and accurate the metadata, the more work there is to do. Consequently, metadata tends to be subjective and shallow, with a well-intentioned drive towards standardisation thwarted—from a learner modelling point of view—by the shallowness of current metadata standards such as LOM [4].

Guidance to learners through educational content in e-learning systems tends to jump in between two extremes: predefined paths and content browsing. From a learner modelling perspective, both situations are mainly equivalent, since neither of them accommodates to learner modelling needs. While in some ITS systems learner modelling can lead learner progress through the subject domain, in these type of e-learning systems it has to be *opportunistic*, taking advantage of whatever information becomes available. A learner modelling subsystem in this conditions has to do more with less, answering questions about the learner on the basis of scarce and shallow information, hopefully without pursuing blind over-generalisation.

In this paper we describe the consequences of requirements and working conditions as sketched above on the design of the Extended Learner Modeller (XLM) for LEACTIVEMATH, an e-learning system for mathematics [2]. XLM was required to model motivational and affective dispositions of learners towards a subject domain and related competencies, as well as learners' meta-cognition of their learning. Our approach to the problem can be summarised in terms of four characteristics: (i) a generic, layered and multi-dimensional modelling framework, (ii) tolerance to vague and inconsistent information, (iii) squeezing of sparse information and (iv) open learner modelling.

2 Modelling Framework

In XLM, a learner model is a collection of beliefs about the learner's states and dispositions arranged along five dimensions (figure 1): subject domain, competency, motivation, affect and meta-cognition. Each of these dimensions is described in a concept map which specifies the individual factors in the dimension that are relevant to learning and considered in learner models. The maps also specify how the different factors and attributes relate to each other. For example, in the current implementation the subject domain is a branch of mathematics known as Differential Calculus and breaks down into domain topics such as *function*, *derivative* and *chain rule* (a particular instance of *differentiation rule* that produces *derivative*s); competency is mathematical and decomposes according to the PISA framework [5]; and motivation decomposes into factors that are considered to influence learner motivation such as the interest, confidence and effort put into learning the subject domain and related competencies. The layered structure of learner models specifies how the modelled dimensions of learners interact with each other. At the bottom of the stack lies the subject domain, underlining the fact that learner modelling occurs within a subject domain, even if a learner model does not hold any belief about the subject domain *per se*. but about learner dimensions *applied to* the domain. On top of the subject domain are the layers of competency, motivation, affect and meta-cognition, each one relying on the lower layers for expressing a wide range of beliefs about the learner. For example, mathematical competencies on the

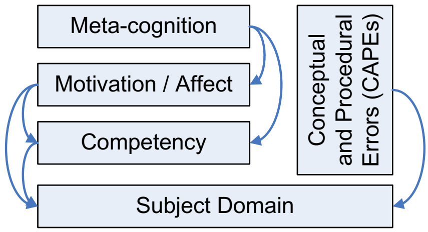


Fig. 1. A multi-dimensional and layered structure for learner models

subject domain (e.g. the learner's level of problem solving with respect to the chain rule), motivational dispositions towards the subject domain (e.g. the learner's level of confidence with respect to differential calculus) or towards competencies on the subject domain (e.g. the learner's level of effort with respect to solving problems with the chain rule).

Conceptual and procedural errors (CAPEs) is a sixth but special dimension. CAPEs are not generic as competencies, but each one specific to particular domain topics. Moreover, neither motivation nor affect nor meta-cognition apply to CAPEs, under the assumption that they are not perceived by learners.

3 Levels and Beliefs

Let us start our explanation of the type and representation of beliefs held in XLM learner models by considering a particular belief concerning the learner's *competency to posing and solving mathematical problems with derivatives*. Let us also assume that the mathematical competencies of learners can be measured in a discrete scale of four levels, from an entry level I to a top level IV— only three levels are described in [5] but LEACTIVEMATH uses four. A learner's mathematical competency is assumed to *be at* one of this levels, having achieved and passed all previous levels, hence a belief on the learner's competency to pose and solve problems with derivatives becomes a statement about *the level that the learner's competency is (more likely) at*.

In the same way, every belief in XLM is about a learner's level on something—as far as that something can be expressed as the application of upper dimensions to lower dimensions in the learner model structure (figure 1). In the current implementation, all dimensions in XLM are measured in a similar scale of four levels.

There are many ways to represent beliefs such as this, from symbolic representations using a logical formalism [6] to numeric representations such as using probability distributions [7]. In XLM, a belief in a learner model is represented and updated using a numeric formalism known as the Transferable Belief Model (TBM) [8], a variation of Dempster-Shafer Theory (DST) which is based on the notion of *belief functions* [9]. A first difference between a probability distribution and a belief functions is that, while the former assigns a number in the range of $[0,1]$ to each possible state of the world—i.e. each level the learner's competency could be at—the latter does the same but to each set of possible states of the world.

More formally, if $\Theta = \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$ is the set of all possible states of the world, then a probability distribution is a function $p : \Theta \rightarrow [0,1]$ while a belief function $b : 2^{\Theta} \rightarrow [0,1]$ maps the set of all sets of levels in Θ into $[0,1]$.

A belief in a learner model can be represented at least in three different ways, as a *mass*, a *certainty*[1] or a *plausibility* function [8, 9]. Although they are equivalent, a mass function is the easiest to manipulate computationally and is hence the one used in XLM. If s_0 is a set of levels, say $s_0 = \{\mathrm{III}, \mathrm{IV}\}$, the mass of s_0, or $m(s_0)$ can be interpreted *objectively*, as the support the evidence gives to the case that the true learner's competency level is in the set s_0 (i.e. it is either III or IV) without making any distinction between the elements of the set[2]. *Subjectively*, it can be interpreted as the part of the belief that pertains exclusively to the likelihood that the true learner's competency level is in s_0, without being any more specific towards either of the levels.

A mass function in XLM is generally required to satisfy the requirement that the sum of all its values must be one; i.e.

$$m : 2^{\Theta} \rightarrow [0,1] \quad \text{such that} \quad \sum_{s \in 2^{\Theta}} m(s) = 1. \tag{1}$$

However—in accordance with TBM and differently from DST—it is not required that the mass assigned to the empty set to be zero. Such a mass is interpreted in XLM as the amount of *conflict* in the evidence accumulated. The mass assigned to the set of all levels Θ is generally interpreted as the amount of complete *ignorance* in a belief.

4 Evidence

Evidence for learner modelling comes into XLM in the shape of *events* representing what has happened in the learners interaction with educational material and the rest of LEACTIVEMATH. Some events originate inside XLM, as is the case for events generated by its Situation Model (the component of XLM in charge of modelling the local motivational state of learners) and Open Learner Model (the graphical user interface to learner models that supports inspectability of the models and challenging of beliefs).

Events are raw evidence that needs to be *interpreted* in order to produce mass functions that can be incorporated into beliefs in learner models using a *combination rule* [10]. Currently, two categories of events are interpreted by XLM and their evidence incorporated into beliefs: *behavioural* events and *diagnostic* events. Events in the first category simply report what the learner has done or achieved, whereas events in the second category report a judgement of learner levels produced by some diagnostic component of LEACTIVEMATH.

[1] More commonly known as a *belief* function, but we use *certainty* to avoid confusion with the more general notion of belief in XLM.

[2] Imagine Theta is an international tennis team composed by Indian Papus (I), Japanese Takuji (II), Scottish Hamish (III) and French Pierre (IV). They have played against the American team and you hear on the radio than all of them, but an European one, have lost their matches. If this is all the evidence you have about who in Theta won its match, it certainly supports the case that the player is in the set {Hamish, Pierre} but does not distinguish between them.

Listing 1. Example metadata for an exercise

```
<exercise id="K3_TIMSS" for="deriv_higher/second_deriv">
  <metadata>
    <Title  xml:lang="en">Acceleration of a straight forward movement</Title>
      <depends−on><ref xref="deriv/diff_f"/></depends−on>
      < difficulty   value="easy"/>
      <competency value="think"/>
      <competency value="model"/>
      <competencylevel value="simple_conceptual"/> <!−− equiv. to II −−>
       ...
  </metadata>
   ...
</exercise>
```

4.1 Interpretation of Behavioural Events

Given an event of type ExerciseFinished, reporting that a learner has finished an exercise with some *success rate*, XLM interprets it to produce evidence for updating the learner model. The resulting evidence would be a collection of mass functions over the following set of sets of levels $2^\Theta = \{s|s \subseteq \Theta\}$. However, given the fact that levels are ranked, it makes no sense to have mass for sets that are not intervals (e.g. $\{\mathrm{I},\mathrm{III}\}$). In other words, the *focus* of m is always going to be a subset of

$$\begin{aligned}\Phi = \{&\{\mathrm{I}\},\{\mathrm{II}\},\{\mathrm{III}\},\{\mathrm{IV}\},\{\mathrm{I},\mathrm{II}\},\{\mathrm{II},\mathrm{III}\},\{\mathrm{III},\mathrm{IV}\},\\ &\{\mathrm{I},\mathrm{II},\mathrm{III}\},\{\mathrm{II},\mathrm{III},\mathrm{IV}\},\{\mathrm{I},\mathrm{II},\mathrm{III},\mathrm{IV}\}\}.\end{aligned} \tag{2}$$

For the particular case of an event of type ExerciseFinished in the current implementation of XLM, these levels are actually *competency levels*. In the general case, the nature of the levels will depend on the nature of the belief the evidence is relevant to.

In order to make the explanation easier to follow, let us consider a concrete case: interpreting an ExerciseFinished event that resulted from the learner finishing an exercise with metadata as in listing 1. An exercise of this type comes with its own additional information, including learner identifier, exercise identifier and success rate achieved by the learner in the exercise (in the range $[0,1]$).

Beliefs Addressed. By interpreting events of type ExerciseFinished, XLM generates direct evidence for beliefs grounded on the subject domain topics the exercises are related to. For example, the exercise K3_TIMSS is for a learning object with identifier deriv_higher/second_deriv, which is mapped to the domain topic *second_derivative* that represents the abstract notion of second order derivative. The metadata listed above indicates the exercise depends on the learning object deriv/ diff_f which is mapped to the topic *derivative* that stands for the abstract notion of derivative. Consequently, all direct evidence produced from an ExerciseFinished event from this exercise will be evidence for beliefs grounded on the topics *derivative* and *second_derivative*.

Metadata indicating which mathematical competencies the exercise evaluates or trains on provide further details of which beliefs should be affected by the event. For our example, this are beliefs on competencies think (think mathematically) and model (model mathematically). Therefore, the beliefs to be directly affected by our example of event would be *beliefs related to the learner's thinking or modelling mathematically with first or second derivatives*. These could be, in principle, beliefs on the learner's competencies, their motivational or affective dispositions on these competencies, or their meta-cognition on these competencies. No belief on a conceptual or procedural error is directly affected, since the event does not provide any information on CAPEs. The current implementation of XLM depends on events produced by its Open Learner Model component for updating beliefs on meta-cognition, on events produced by its Situation Model component for updating beliefs on motivational dispositions, and on events produced by LEACTIVEMATH Self-Report Tool for updating beliefs on affective dispositions. Consequently, only evidence for beliefs on learners' mathematical competencies on the subject domain are produced from events of type ExerciseFinished.

Generation of Evidence. Once the beliefs to be affected by an event have been identified, the next step is to generate the corresponding evidence: a mass function per belief over the sets of levels in Φ. Although most of the metadata for an exercise could have an impact on the evidence to be produced, for the case of an ExerciseFinished event in the current implementation of XLM only a subset of the metadata is taken into account: the relationship between the exercise and the belief addressed (i.e. whether the exercise is *for* the topic the belief is about, or only *depends on* it), the competency level of the exercise, the difficulty of the exercise and the success rate reported in the event.

The competency level of an exercise is used to determine who should find the exercise either very easy or very difficulty, and who may find it otherwise (i.e. easy, medium or difficult, the remaining terms in LEACTIVEMATH vocabulary for metadata on the difficulty of exercises). For example, being K3_TIMSS an easy exercise for competency level II means it should be a very easy exercise for any learner with competency level IV. Furthermore, we have assumed that would be the case for any exercise for competency level II, and table 1 presents the initial estimation of the difficulty of an exercise for a learner, given the competency levels of the exercise and the learner. To fill the still empty cells in table 1 we use the metadatum for difficulty of the exercise. A possible interpretation of this metadatum is given in table 2 for the case of exercises for competency level II.

At this stage, the metadata for an exercise such as K3_TIMSS supports estimates of how difficult the exercise would be for learners with different competency levels. In particular, we can see that K3_TIMSS does not discriminate between learners with

Table 1. Effect of the metadata value for competency level on the estimated difficulty of an exercise for a learner at a given competency level

Competency level of exercise	Competency level of learner			
	I	II	III	IV
I	-	-	very easy	very easy
II	-	-	-	very easy
III	very difficult	-	-	-
IV	very difficult	very difficult	-	-

Table 2. Effect of the metadata value for difficulty on the estimated difficulty of an exercise for competency level II for a learner at a given competency level

Exercise		Difficulty of the exercise for a learner at a given competency level			
Competency level	Difficulty	I	II	III	IV
	VE	M	VE	VE	VE
	E	D	E	VE	VE
II	M	VD	M	VE	VE
	D	VD	D	E	VE
	VD	VD	VD	M	VE
VE: very easy, E: easy, M: medium, D: difficult, VE: very difficult					

competency level III or IV. Hence mass should not be assigned to levels III nor IV alone in any evidence generated from it, but only to the set $\{III, IV\}$ or sets containing it.

Here is the point when we need to translate from the qualitative tags denoting difficulty to quantitative measures. In other words, we need to estimate, for every rate of success r, the probability P of achieving r given difficulty d. We need to estimate $P(r|d)$. We use scaled normal distributions

$$P(r) = \delta e^{-(r-\mu)^2/2\delta^2} \tag{3}$$

with parameters determined by difficulty as specified in table 3. They assign a 0.5 probability to being completely successful ($r = 1$) in a very easy exercise, to being moderately successful ($r = 0.75$) in an easy exercise, to being just fair ($r = 0.5$) in a medium exercise, to being unsuccessful ($r = 0.25$) in a difficult exercise and to being completely unsuccessful ($r = 0$) in a very difficult one. For example, if the success rate reported in an ExerciseFinished event for exercise K3_TIMSS is $r = 0.8$ then we get the following probabilities per competency level: 0.2730 (I), 0.4975 (II) and 0.4615 (III and IV).

Table 3. Parameters for probability assignment functions per difficulty value

Parameters	Difficulty				
	very easy	easy	medium	difficult	very difficult
μ	1	0.75	0.5	0.25	0
δ	0.5	0.5	0.5	0.5	0.5

An straightforward way of translating this probabilities into a mass function would be by normalising the probabilities obtained above and assigning them to the singletons $\{I\}$, $\{II\}$, $\{III\}$ and $\{IV\}$. However, as it was said before, exercise K3_TIMSS does not distinguish between learners with competency levels III or IV, hence it does not provide evidence for the learner having any of these levels in particular but, in any case, just of having any of them. Furthermore, what should be done if all probabilities above where the same? A possibility is to consider the exercise as unable to discriminate between the possible levels of competency of the learner, hence providing no new evidence at all. Technically, this means the mass distribution in this case should be the one corresponding to *total ignorance* (or complete lack of evidence): $m(\Theta) = 1$ and $m(s) = 0$ for all other $s \neq \Theta$.

We can generalise these two cases to an iterative method for calculating a mass functions from probabilities:

1. If $s = \{l_1, l_2, \dots, l_n\}$ is the set of levels with non zero probabilities, then make $m(s)$ equal to the smallest probability, $m(s) = \min(p(l_1), p(l_2), \dots, p(l_n))$.
2. For every level l_i in s make its probability equal to $p(l_i) - m(s)$.
3. Remove from s all levels with re-calculated probability equal to zero and start again at step (1).
4. Finally, scale all $m(s)$ uniformly, so that the total mass $\sum_{s \subseteq \Phi} m(s) = 1$.

The application of this method to the case of exercise K3_TIMSS with success rate of 0.8 and the probabilities calculated above produces the following mass function:

$$m(\{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}) = 0.549, \quad m(\{\mathrm{II}, \mathrm{III}, \mathrm{IV}\}) = 0.379, \quad m(\{\mathrm{II}\}) = 0.072,$$
$$m(s) = 0.0 \text{ for any other } s \subset \Phi.$$

In words, this is weak evidence for the learner being at competency level II (simple-conceptual) and stronger evidence for they being at a competency level in $\{\mathrm{II}, \mathrm{III}, \mathrm{IV}\}$. However, this evidence includes a fair amount of ignorance that suggest it is still plausible for the learner to be at any competency level, including level I.

Each belief that have to do with topics trained on, or evaluated by the exercise would receive such an evidence. Beliefs concerning topics the exercise depends on would receive a *discounted* evidence with increased ignorance:

$$m'(s) = d \times m(s) \text{ for all } s \neq \Theta \text{ and } m'(\Theta) = m(\Theta) + (1 - d), \tag{4}$$

where d is a discount factor in between zero and one.

4.2 Interpreting Diagnostic Events

The interpretation of diagnostic events by XLM is simpler than the interpretation of behavioural events given the fact that an estimation of the learner level is included in diagnostic events. Based on how much XLM trust the source of the event, the original estimation of the learner level is transformed into a probability distribution over the set of levels Θ. This probability distribution is then transformed into a mass function following the same procedure as explained in section 4.1.

Consider, for example, the case of LEACTIVEMATH Self-Report Tool, which is presented to learners every time they complete an exercise so that they can provide feedback on their states of liking, pride and satisfaction—which are assumed to be with respect to the exercise just finished. The values input by the learners are reported to XLM in SelfReport events. Then XLM transform a single value per factor into a probability distribution by choosing a suitable Beta distribution from the collection shown in figure 2. The mass function resulting from the interpretation of the event would constitute evidence for beliefs on the learner's affective dispositions towards the subject domain topics and competencies that result from considering the exercise metadata.

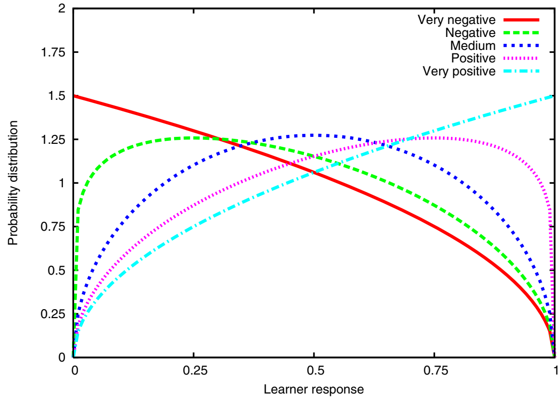


Fig. 2. Beta distributions for each value that can be reported by learners using the Self-Report Tool. The intervals $[0, 0.25]$, $[0.25, 0.50]$, $[0.50, 0.75]$ an $[0.75, 1.0]$ correspond to the levels I, II, III and IV, respectively.

4.3 Propagation of Evidence

Interpretation of events such as the ones described in sections 4.1 and 4.2 provides direct evidences for some beliefs in a learner model. These evidences are propagated to the relevant parts of the learner model following the associations between elements in the maps for each layer in the learner modelling framework (section 2). An iterative algorithm is used for belief propagation which is a simple adaptation of Shenoy-Shafer algorithm for belief-functions propagation [11]. In every iteration, all beliefs that have received updated messages (with adjusted evidences) re-calculates its own and checks whether these have changed significantly (given a predefined threshold and a method for comparing mass functions). If that is the case, and its messages are not full with ignorance (beyond another predefined threshold) then it propagates the evidence to their neighbours. The iterative process ends when no more messages have been exchanged or when a predefined maximum number of iterations have been reached.

5 Conclusions and Future Work

A learner modelling subsystem called XLM has been presented in this paper which tries to capture the multi-dimensional nature of learners. XLM uses a collection of dimensions—subject domain, conceptual and procedural errors, competency, motivation, affect and meta-cognition—defines them using concept maps and arranges them in layers. Together, the internal maps and layered framework provide a rich structure for organising beliefs about learners. Beliefs are represented using belief functions, which allow the representation of ignorance, uncertainty and conflict in evidence and beliefs. Together with a simple algorithm for propagation of evidence, XLM is the first implementation of learner models with belief functions networks we are aware of, providing in this way an alternative to Bayesian networks for learner modelling [7, 12, 13, 14].

A first complete implementation of xLM has been delivered early this year. Nevertheless, there are many issues to be revised and parameters to be adjusted before xLM reaches maturity. First, competency level and difficulty are seen as two granularities in the same scale, like metres and centimetres. This may be a misinterpretation of the nature of competency levels, which seems to represent more qualitative changes than difficulty [5]. Secondly, a core but minimum subset of metadata is taken into account while interpreting events, which could be expanded for better. Thirdly, careful analysis and evaluation of the probability assignments, probability distributions and the propagation algorithm are necessary to improve the modelling process.

References

1. Advanced Distributed Learning: Sharable Content Object Reference Model Version 1.2: The SCORM Overview. (2001)
2. LeActiveMath Consortium: Language-enhanced, user adaptive, interactive elearning for mathematics (2004) http://www.leactivemath.org.
3. National Information Standards Organization: Understanding Metadata. (2004)
4. Institute of Electrical and Electronics Engineers: IEEE 1484.12.1 Draft Standard for Learning Object Metadata. (2002)
5. Organisation for Economic Co-Operation and Development: The PISA 2003 Assessment Framework. (2003)
6. Self, J.A.: Dormorbile: A vehicle for metacognition. AAI/AI-ED Technical Report 98, Computing Department, Lancaster University, Lancaster, UK (1994)
7. Zapata-Rivera, J.D., Greer, J.E.: Inspecting and visualizing distributed bayesian student models. In Gauthier, G., Frasson, C., VanLehn, K., eds.: Intelligent Tutoring Systems: Fifth International Conference, ITS'2000. Number 1839 in Lecture Notes in Computer Science, Springer Verlag (2000) 544–553
8. Smets, P., Kennes, R.: The transferable belief model. Artificial Intelligence **66** (1994) 191–234
9. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press (1976)
10. Sentz, K., Ferson, S.: Combination of evidence in dempster-shafer theory. Sandia Report SAND2002-0835, Sandia National Laboratories (2002)
11. Shenoy, P.P., Shafer, G.: Axioms for probability in belief-function propagation. In Shachter, R.D., Levitt, T.S., Kanal, L.N., Lemmer, J.F., eds.: Proceedings of the Fourth Anual Conference on Uncertainty in Artificial Intelligence, North-Holland (1990) 169–198
12. Conati, C., Gertner, A., VanLehn, K.: Using bayesian networks to manage uncertainty in student modeling. User Modeling and User-Adapted Interaction **12** (2002) 371–417
13. Bunt, A., Conati, C.: Probabilistic student modelling to improve exploratory behaviour. User Modeling and User-Adapted Interaction **13** (2003) 269–309
14. Jameson, A.: Numerical uncertainty management in user and student modeling: An overview of systems and issues. User Modeling and User-Adapted Interaction **5** (1996) 193–251

Diagnosing Self-efficacy in Intelligent Tutoring Systems: An Empirical Study

Scott W. McQuiggan and James C. Lester

Department of Computer Science, North Carolina State University, Raleigh, NC 27695
{swmcquig, lester}@ncsu.edu

Abstract. Self-efficacy is an individual's belief about her ability to perform well in a given situation. Because self-efficacious students are effective learners, endowing intelligent tutoring systems with the ability to diagnose self-efficacy could lead to improved pedagogy. Self-efficacy is influenced by (and influences) affective state. Thus, physiological data might be used to predict a students' level of self-efficacy. This paper investigates an inductive approach to automatically constructing models of self-efficacy that can be used at runtime to inform pedagogical decisions. In an empirical study, two families of self-efficacy models were induced: a static model, learned solely from pre-test (non-intrusively collected) data, and a dynamic model, learned from both pre-test data as well as runtime physiological data collected with a biofeedback apparatus. The resulting static model is able to predict students' real-time levels of self-efficacy with reasonable accuracy, while the physiologically informed dynamic model is even more accurate.

1 Introduction

Affect has begun to play an increasingly important role in intelligent tutoring systems. Recent years have seen the emergence of work on affective student modeling [8], detecting frustration and stress [7, 21], modeling agents' emotional states [1, 11, 16], devising affectively informed models of social interaction [13, 18, 20], and detecting student motivation [24]. All of this work seeks to increase the fidelity with which affective and motivational processes are modeled in intelligent tutoring systems in an effort to increase the effectiveness of tutorial interactions and, ultimately, learning.

Self-efficacy is an affective construct that has been found to be a highly accurate predictor of students' motivational state and their learning effectiveness [25]. Defined as "the belief in one's capabilities to organize and execute the courses of action required to manage prospective situations" [2], self-efficacy has been repeatedly demonstrated to directly influence students' affective, cognitive, and motivational processes [3]. Self-efficacy holds much promise for intelligent tutoring systems (ITSs). Foundational work has begun on using models of self-efficacy for tutorial action selection [6] and investigating the impact of pedagogical agents on students' self-efficacy [5, 14]. Self-efficacy is useful for predicting what problems and sub-problems a student will select to solve, how long a student will persist on a problem, how much overall effort they will expend, as well as motivational traits such as level of engagement [22, 25]. Thus, if an ITS could increase a student's self-

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 565 – 574, 2006.

efficacy, then it could enable the student to be more actively involved in learning, expend more effort, and be more persistent; it could also enable them to successfully cope in situations where they experience learning impasses [3].

To effectively reason about a student's self-efficacy, ITSs need to accurately model self-efficacy. Self-efficacy diagnosis should satisfy three requirements. First, it should be realized in a computational mechanism that operates at runtime. Self-efficacy may vary throughout a learning episode, so pre-learning self-efficacy instruments may or may not be predictive of self-efficacy at specific junctures in a learning episode. Second, self-efficacy diagnosis should be efficient. It should satisfy the real-time demands of interactive learning. Third, self-efficacy diagnosis should avoid interrupting the learning process. A common approach to obtaining information about a student's self-efficacy is directly posing questions to them throughout a learning episode. However, periodic self-reports are disruptive.

This paper reports on the results of an experiment that investigates an inductive approach (naïve Bayes and decision tree classifications) to constructing models of self-efficacy. In the experiment, two families of self-efficacy models were induced: the model learner constructed (1) *static* models, which are based on demographic data and a validated problem-solving self-efficacy instrument [4], and (2) *dynamic* models, which extend static models by also incorporating real-time physiological data. In the experiment, 33 students provided demographic data and were given an online tutorial in the domain of genetics. Next, they were given a validated problem-solving self-efficacy instrument, and they were outfitted with a biofeedback device that measured heart rate and galvanic skin response. Physiological signals were then monitored while students were tested on concepts presented in the tutorial. After solving each problem, students rated their level of confidence in their response with a "self-efficacy slider." Both families of resulting models operate at runtime, are efficient, and do not interrupt the learning process. The static models are able to predict students' real-time levels of self-efficacy with 73% accuracy, and the resulting dynamic models are able to achieve 83% predictive accuracy. Thus, non-intrusive static models can predict self-efficacy with reasonable accuracy, and their predictive power can be increased by further enriching them with physiological data.

The paper is structured as follows. Section 2 discusses the role of self-efficacy in learning. The experimental design is presented in Section 3 (experimental method) and Section 4 (procedure), and the results are described in Section 5. Section 6 discusses the findings and their associated design implications, and Section 7 makes concluding remarks and suggests directions for future work.

2 Self-efficacy and Learning

Self-efficacy is powerful. It influences students' reasoning, their level of effort, their persistence, and how they feel; it shapes how they make choices, how much resilience they exhibit when confronted with failure, and what level of success they are likely to achieve [2, 22, 25]. While it has not been conclusively demonstrated, many conjecture that given two students of equal abilities, the one with higher self-efficacy is more likely to perform better than the other over time. Self-efficacy is intimately related to motivation, which controls the effort and persistence with which a student approaches a task [15]. Effort and persistence are themselves influenced by the belief the student has that she will be able to achieve a desired outcome [3]. Self-efficacy has been studied in many domains with significant work having been done in computer literacy [9] and mathematics education [19]. It is widely believed that self-efficacy is domain-specific, but whether it crosses domains remains an open question.

A student's self-efficacy[1] is influenced by four types of experiences [3, 25]. First, in enactive experiences, she performs actions and experiences outcomes directly. These are typically considered the most influential category. Second, in vicarious experiences, she models her beliefs based on comparisons with others. These can include peers, tutors, and teachers. Third, in verbal persuasion experiences, she experiences an outcome via a persuader's description. For example, she may be encouraged by the persuader, who may praise the student for performing well or comment on the difficulty of a problem. Her interpretation will be affected by the credibility she ascribes to the persuader. Fourth, with physiological and emotional reactions, she responds affectively to situations. These experiences, which often induce stress and anxiety and are physically manifested in physiological responses, such as increased heart rate and sweaty palms, call for emotional support and motivational feedback.

Self-efficacy holds great promise for ITSs. Self-efficacy beliefs have a stronger correlation with desired behavioral outcomes than many other motivational constructs [10], and it has been recognized in educational settings, that self-efficacy can predict both motivation and learning effectiveness [25]. Thus, if it were possible to enable ITSs to accurately model self-efficacy, they may be able to leverage it to increase students' academic performance. Two recent efforts have explored the role of self-efficacy in ITSs. One introduced techniques for incorporating knowledge of self-efficacy in pedagogical decision making [6]. Using a pre-test instrument and knowledge of problem-solving success and failure, instruction is adapted based on changes in motivational and cognitive factors. The second explored the effects of pedagogical agent design on students' traits, which included self-efficacy [5, 14]. The focus of the experiment reported in this paper is on the automated induction of self-efficacy models for runtime use by ITSs.

3 Method

3.1 Participants and Design

In a formal evaluation, data was gathered from thirty-three subjects in an Institutional Review Board (IRB) of NCSU approved user study. There were 6 female and 27 male participants varying in age, race, and marriage status. Approximately 36% of the participants were Asian, 60% were Caucasian, and 3% were of other races. 27% of the participants were married. Participants average age was 26.15 (SD=5.32).

---

[1] Self-efficacy is closely related to the popular notion of confidence. To distinguish consider the situation in which a student is very confident that she will fail at a given task. This represents high confidence but low self-efficacy, i.e., she is exhibiting a strong belief in her inability [3].

3.2 Materials and Apparatus

The pre-experiment paper-and-pencil materials for each participant consisted of a demographic survey, tutorial instructions, Bandura's Problem-solving Self-Efficacy Scale [4], and the problem-solving system directions. Post-experiment paper-and-pencil materials consisted of a general survey. The demographic survey collected basic information such as gender, age, ethnicity, and marital status. The tutorial instructions explained to participants the details of the task, such as how to navigate through the tutorial and an explanation of the target domain. Bandura's validated Problem-solving Self-Efficacy Scale [4], which was administered after they completed a tutorial in the domain of genetics, asked participants to rate how certain they were in their ability to successfully complete the upcoming problems (which they had not yet seen). The problem-solving system directions supplied detailed task direction to participants, as well as screenshots highlighting important features of the system display, such as the "self-efficacy slider".

The computerized materials consisted of an online genetics tutorial and an online genetics problem-solving system. The online genetics tutorial consisted of an illustrated 15-page web document which included some animation and whose content was drawn primarily from a middle school biology textbook [17]. The online genetics problem-solving system consisted of 20 questions, which covered material in the online genetics tutorial. The problem-solving system presented each multiple-choice question individually and required participants to rate their confidence, using a "self-efficacy slider," in their answer before proceeding to the next question.

Apparati consisted of a Gateway 7510GX laptop with a 2.4 GHz processor, 1.0 GB of RAM, 15-in. monitor and biofeedback equipment for monitoring blood volume pulse (one sensor on the left middle finger) and galvanic skin response (two sensors on the left first and third fingers). Participants' right hands were free from equipment so they could make effective use of the mouse in problem-solving activities.

4 Procedure

4.1 Participant Procedure

Each participant entered the experimental environment (a conference room) and was seated in front of the laptop computer. First, participants completed the demographic survey at their own rate. Next, participants read over the online genetics tutorial directions before proceeding to the online tutorial. On average, participants took 17.67 (SD = 2.91) minutes to read through the genetics online tutorial. Following the tutorial, participants were asked to complete the Problem-Solving Self-Efficacy Scale considering their experience with the material encountered in the genetics tutorial. The instrument asked participants to rate their level of confidence in their ability to successfully complete certain percentages of the upcoming problems in the problem-solving system. Participants did not have any additional information about the type of questions or the domain of the questions contained in forthcoming problems. Participants were then outfitted with biofeedback equipment on their left hand while the problem-solving system was loaded. Once the system was loaded, participants entered the calibration period in which they read through the problem-solving system directions. This allowed the system to obtain initial readings on the temporal attributes being monitored, in effect establishing a baseline for HR and GSR.

The problem-solving system presented randomly selected, multiple-choice questions to the participant. The participant selected an answer and then manipulated a self-efficacy slider representing the strength of their belief in their answer being correct. Participants completed 20 questions. Participants averaged 8.15 minutes (SD = 2.37) to complete the problem-solving system. Finally, participants were asked to complete the post-experiment survey at their own rate before concluding the session.

4.2 Machine Learning Procedure

The following procedure was used to induce models of self-efficacy:

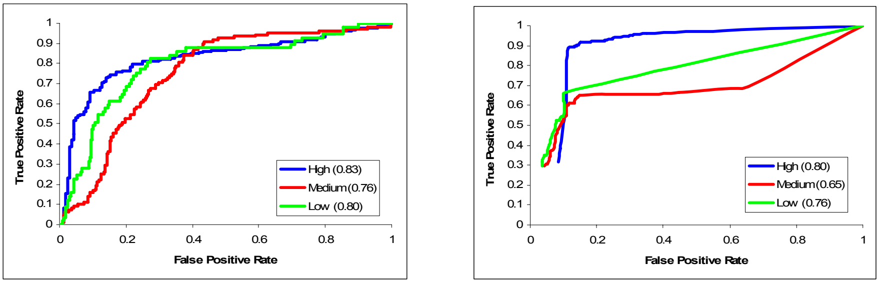
- *Data Construction:* Each session log, containing on average 14,645.42 (SD = 4,010.57) observation changes, was first translated into a full observational attribute vector. For example, BVP and GSR readings were taken nearly 30 times every second reflecting changes in both heart rate and skin conductivity.
- *Data Cleansing:* Data were converted into an attribute vector format. Then, a dataset was generated that contained only records in which the biofeedback equipment was able to successfully monitor BVP and GSR throughout the entire training session. Blood volume pulse (used for monitoring HR) readings were difficult to obtain from two participants resulting in the destruction of that data.
- *Naïve Bayes Classifier and Decision Tree Analysis:* The prepared dataset was loaded into the WEKA machine learning package [23], a naïve Bayes classifier and decision tree were learned, and tenfold cross-validation analyses were run on the resulting models. The entire dataset was used to generate several types of self-efficacy models, each predicting self-efficacy with varying degrees of granularity. These included two-level models (Low, High), three-level models, four-level models, and five-level models (Very Low, Low, Medium, High, Very High).

5 Results

Below we present the results of the naïve Bayes and decision tree classification models and provide analyses of the collected data. Various ANOVA statistics are presented for results that are statistically significant. Because the tests reported here were performed on discrete data, we report Chi-square test statistics (χ^2), including both likelihood ratio Chi-square and the Pearson Chi-square values. Fisher's Exact Test is used to find significant p-values at the 95% confidence level ($p < .05$).

5.1 Model Results

Naïve Bayes and decision tree classifiers are effective machine learning techniques for generating preliminary predictive models. Naïve Bayes classification approaches produce probability tables that can be implemented into runtime systems and used to continually update probabilities for assessing student self-efficacy levels. Decision trees provide interpretable rules that support runtime decision making. The runtime system monitors the condition of the attributes in the rules to determine when conditions



(a) (b)

Fig. 1. ROC curves for naïve Bayes (a) and decision tree (b) three-level models of self-efficacy. Areas under the curve are found in the key. Overall the naïve Bayes model correctly classified 72% of the instances while the decision tree was able to correctly classify 83%.

are met for assigning particular values of student self-efficacy. Both the naïve Bayes and decision tree machine learning classification techniques are useful for preliminary predictive model induction for large multidimensional data, such as the 144-attribute vector used in this experiment. Because it is unclear precisely which runtime variables are likely to be the most predictive, naïve Bayes and decision tree modeling provide useful analyses that can inform more expressive machine learning techniques (e.g., Bayesian networks) that also leverage domain experts' knowledge.

All models were constructed using a tenfold cross-validation scheme. In this scheme, data is decomposed into ten equal partitions, nine of which are used for training and one used for testing. The equal parts are swapped between training and testing sets until each partition has been used for both training and testing. Tenfold cross-validation is widely used for obtaining a sufficient estimate of error [23].

Cross-validated ROC curves are useful for presenting the performance of classification algorithms for two reasons. First, they represent positive classifications, included in a sample, as a percentage of the total number of positives, against negative classifications as a percentage of the total number of negatives [23]. Second, the area under ROC curves is widely accepted as a generalization of the measure of the probability of correctly classifying an instance [12].

The ROC curves (Fig. 1) above show the results of both a naïve Bayes and decision tree three-level model. Low-confidence was noted by a student self-efficacy rating lower than 33 (on a 0 to 100 scale). Medium-confidence was determined by rating between 33 and 67, while High-confidence was represented all ratings greater than 67. The smoothness of the curve in Figure 1(a) indicates that sufficient data seems to have been used for inducing naïve Bayes models. The jaggedness of the curves in Figure 1(b) indicates that more data covering the possible instances is needed. In particular, further investigation will need to consider more opportunities for students to experience instances of low self-efficacy. Despite the appearance of a lack of sufficient data, the decision tree model performed significantly better than the naïve Bayes model (likelihood ratio, $\chi^2 = 21.64$, Pearson, $\chi^2 = 21.47$, $p < .05$). The highest performing model induced from all data was the two-level decision-tree based dynamic model, which performed significantly better than the highest performing static model, which was a two-level decision tree model (likelihood ratio, $\chi^2 = 3.99$,

Table 1. Model results – area under ROC curves. Gray rows indicated static models induced from non-intrusive demographic and Problem-Solving Self-Efficacy data. The other rows represent dynamic models that were also based on physiological data.

Model	Two-level	Three-level	Four-level	Five-level
Naïve Bayes	0.85	0.72	0.75	0.64
Decision Tree	0.87	0.83	0.79	0.75
Naïve Bayes	0.82	0.70	0.69	0.63
Decision Tree	0.83	0.73	0.69	0.64

Pearson, $\chi^2 = 3.97$, $p < .05$). The three-level dynamic decision tree model was also significantly better than the static three-level decision tree (likelihood ratio, $\chi^2 = 18.26$, Pearson, $\chi^2 = 18.13$, $p < .05$). All model results are presented in Table 1.

5.2 Model Attribute Effects on Self-efficacy

Heart rate and galvanic skin response had significant effects on self-efficacy predictions (Table 2). Participants' age group was the only demographic attribute to have a significant effect on all levels of self-efficacy models (Table 3).

Table 2. Chi-squared values representing the significance of physiological signals on varying levels of dynamic self-efficacy models ($p < 0.5$). Grayed cells represent no significance.

Physiological signal	Two-level	Three-level	Four-level	Five-level
HR		9.58	15.35	12.78
GSR		9.24	17.96	14.82

Table 3. Demographic effects on self-efficacy. Chi-square values reported with $p < .05$. Grayed cells represent no statistical significance.

Demographic	Two-level	Three-level	Four-level	Five-level
Gender			18.10	11.14
Age Group	16.25	50.00	94.64	87.64
Race & Ethnicity				

6 Discussion and Future Work

Self-efficacy is closely associated with motivational and affective constructs that both influence (and are influenced by) a student's physiological state. It is therefore not unexpected that a student's physiological state can be used to more accurately predict her self-efficacy. For example, Figures 2 and 3 show the heart rates for one participant in the study over the course of solving two problems.In Figure 2, the participant reported high levels of self-efficacy, while the same participant whose heart rate progression is shown in Figure 3 reported low levels of self-efficacy. The heart rate for the high self-efficacy student gradually drops as they encounter a new question, presumably because of their confidence in their ability to successfully solve the

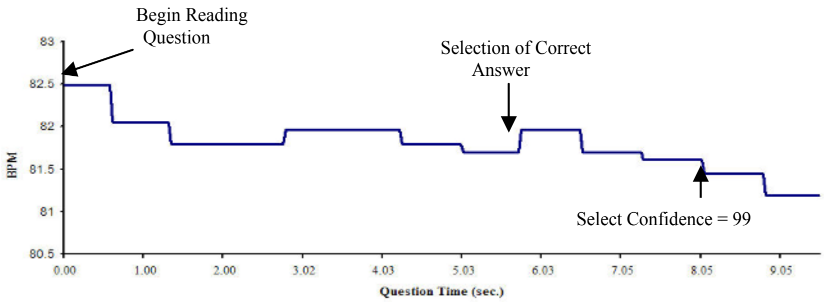


Fig. 2. Heart rate for reported high self-efficacy student

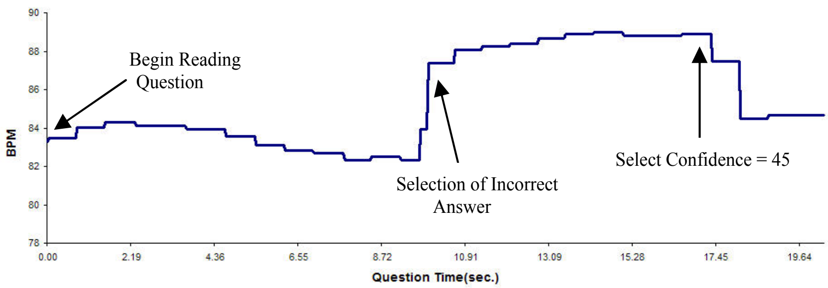


Fig. 3. Heart rate for reported low self-efficacy student

problem. In contrast, the heart rate for the low self-efficacy student spikes dramatically when the student selects an incorrect answer. This phenomenon is particularly intriguing since the students were in fact not given feedback about whether or not their responses were correct. It appears that through some combination of cognitive and affective processes the student's uneasiness with her response, even in the absence of direct feedback, was enough to bring about a significant physiologically manifested reaction. Curiously, there is a subsequent drop in heart rate after the student reports her low level of self-efficacy. In this instance, it seems that providing an opportunity to acknowledge a lack of ability and knowledge to perform may itself reduce anxiety.

The experiment has two important implications for the design of runtime self-efficacy modeling. First, even without access to physiological data, induced decision-tree models can make reasonably accurate predictions about students' self-efficacy. Sometimes physiological data is unavailable or it would be too intrusive to obtain the data. In these situations, decision-tree models that learn from demographic data and data gathered with a validated self-efficacy instrument administered prior to problem solving and learning episodes, can accurately model self-efficacy. Second, if runtime physiological data is available, it can significantly enhance self-efficacy modeling. Given access to HR and GSR, self-efficacy can be predicted more accurately.

7 Conclusion

Self-efficacy is an affective construct that may be useful for increasing the effectiveness of tutorial decision making by ITSs. It may be helpful for increasing students' level of effort, the degree of persistence with which they approach problem solving, and, ultimately, the levels of success they achieve. However, to provide accurate and useful information, self-efficacy models must be able to operate at runtime, i.e., during problem-solving episodes, they must be efficient, and they must avoid interrupting learning. A promising approach to constructing models of self-efficacy is inducing them rather then manually constructing them. In a controlled experiment, it has been demonstrated that *static* models induced from demographic data, a validated self-efficacy instrument, and information from the tutorial system can accurately predict student's self-efficacy during problem solving. It has also been empirically demonstrated that *dynamic* models enriched with physiological data can even more accurately predict student's self-efficacy during problem solving.

The findings reported here contribute to the growing body of work on affective reasoning for learning environments. They represent a first step towards a comprehensive theory of self-efficacy that can be leveraged to increase motivation and learning effectiveness. Two directions for future work are suggested by the results. First, it is important to pursue studies that investigate techniques for achieving the predictive power of dynamic models but "without the wires." Because of the invasiveness of biofeedback apparatus, it would be desirable to develop self-efficacy models that can be induced from students' actions in learning environments that perhaps can be used to infer physiological responses without actually requiring students in runtime environments to be outfitted with biofeedback sensors. Second, now that self-efficacy can be accurately modeled at runtime, the effect of specific pedagogical actions on students' self-efficacy can be investigated. Thus, it may be possible to quantitatively gauge the influence of competing tutorial strategies on students' self-efficacy, which might further increase learning effectiveness.

References

1. André, E., and Mueller, M. Learning affective behavior. In *Proceedings of the 10th International Conference on Human-Computer Interaction*. Lawrence Erlbaum, Mahwah, NJ, 2003, 512-516.
2. Bandura, A. Exercise of personal and collective efficacy in changing societies. In Bandura, A. (Ed.) *Self-efficacy in changing societies* (pp.1-45). New York, NY: Cambridge University Press, 1995.
3. Bandura, A. *Self-efficacy: The exercise of control.* New York: Freeman. 1997.
4. Bandura, A. *Guide for constructing self-efficacy scales*. Unpublished manuscript. 2005.
5. Baylor, A. and Kim, Y. Pedagogical agent design: The impact of agent realism, gender, ethnicity, and instructional role. In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, Springer-Verlag, New York, NY, 2004, 592-403.
6. Beal, C. and Lee, H. Creating a pedagogical model that uses student self reports of motivation and mood to adapt ITS instruction. In *Workshop on Motivation and Affect in Educational Software, in conjunction with the 12th International Conference on Artificial Intelligence in Education*. 2005.

7. Burleson, W. and Picard, R. Affective agents: Sustaining motivation to learn through failure and a state of stuck. In *Workshop of Social and Emotional Intelligence in Learning Environments, in conjunction with the 7th International Conference on Intelligent Tutoring Systems*. 2004.
8. Conati, C., and Mclaren, H. Data-driven refinement of a probabilistic model of user affect. In *Proceedings of the 10th International Conference on User Modeling*. Springer-Verlag, New York, NY, 2005, 40-49.
9. Delcourt, M., and Kinzie, M. Computer technologies in teacher education: the measurement of attitudes and self-efficacy. *Journal of Research and Development in Education*. 27(1):35-41, 1993.
10. Graham, S., and Weiner, B. Principles and theories of motivation. In Berliner, D., and Calfee, R. (Eds.) *Handbook of educational psychology* (pp.63-84). New York, NY: MacMillan Publishing, 1996.
11. Gratch, J., and Marsella, S. A domain-independent framework for modeling emotion. *Journal of Cognitive Systems Research*, 5(4):269-306, 2004.
12. Hanley, J. and McNeil, B. The meaning and use of the area under the Receiver Operating Characteristic (ROC) curve. *Radiology* (143):29-36, 1982.
13. Johnson, L., and Rizzo, P. Politeness in tutoring dialogs: "run the factory, that's what I'd do". In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*. Springer-Verlag, New York, NY, 2004, 67-76.
14. Kim, Y. Empathetic virtual peers enhanced learner interest and self-efficacy. In *Workshop on Motivation and Affect in Educational Software, in conjunction with the 12th International Conference on Artificial Intelligence in Education*. 2005.
15. Lepper, M., Woolverton, M., Mumme, D., & Gurtner, J. Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. In Lajoie, S. and Derry, S. (Eds.), *Computers as cognitive tools* (pp. 75-105). Hillsdale, NJ: Erlbaum. 1993.
16. Lester, J., Towns, S., and FitzGerald, P. Achieving affective impact: Visual emotive communication in lifelike pedagogical agents. *The International Journal of Artificial Intelligence in Education*, 10(3-4):278-291, 1999.
17. Padilla, M., Miaoulis, I., and Cyr, M. *Science Explorer: Cells and Heredity*. Teacher's Edition, Prentice Hall, Upper Saddle River, NJ, 2000.
18. Paiva, A., Dias, J., Sobral, D., Aylett, R., Woods, S., Hall, L., and Zoll, C. Learning by feeling: Evoking empathy with synthetic characters. *Applied Artificial Intelligence*, 19:235-266, 2005.
19. Pajares, F., and Kranzler, J. Self-Efficacy beliefs and general mental ability in mathematical problem solving. *Contemporary Educational Psychology*, 20:426-443, 1995.
20. Porayska-Pomsta, K. and Pain, H. Providing Cognitive and Affective Scaffolding through Teaching Strategies, In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*. Springer-Verlag, New York, NY, 2004, 77-86.
21. Prendinger, H., and Ishizuka, M. The empathic companion: A character-based interface that addresses users' affective states. *Applied Artificial Intelligence*, 19:267-285, 2005.
22. Schunk, D. and Pajares, F. The development of academic self-efficacy. In Wigfield, A. and Eccles, J. (Eds.), *Development of achievement motivation* (pp. 15-31). San Diego, CA: Academic Press. 2002.
23. Witten, I., and Frank, E. *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufman, San Francisco, CA, 2005.
24. de Vicente, A., and Pain, H. Informing the detection of the students' motivational state: an empirical study. In *Proceedings of the 6th International Conference on Intelligent Tutoring Systems*. Springer-Verlag, New York, NY, 2002, 933-943.
25. Zimmerman, B. Self-efficacy: an essential motive to learn. *Contemporary Educational Psychology* 25:82-91, 2000.

Using Instant Messaging to Provide an Intelligent Learning Environment

Chun-Hung Lu[1], Guey-Fa Chiou[2], Min-Yuh Day[1,3], Chorng-Shyong Ong[3], and Wen-Lian Hsu[1]

[1] Institute of Information Science, Academia Sinica, Taiwan
{enrico, myday, hsu}@iis.sinica.edu.tw
[2] Dept. of Information and Computer Education, National Taiwan Normal Univ., Taiwan
gueyfa@ice.ntnu.edu.tw
[3] Dept. of Information Management, National Taiwan University, Taiwan
ongcs@im.ntu.edu.tw

Abstract. Instant Messaging enables learners and educators to interact in an on-line environment. In this paper, we propose an intelligent ChatBot system, based on instant messaging, for student on-line coaching in an English learning environment. The proposed ChatBot facilitates synchronous communication with students by using ready reference materials including, dictionaries, authorized conversation material with speaking, and a question-answering function. The agent records and analyzes conversations so that the teacher can assess students' progress. Our contribution in this paper is that we integrate the NLP Tool and AIML into an instant messaging-based ChatBot for English as a Second Language programs.

1 Introduction

English as a Second Language (ESL) programs are run by most universities, colleges, and some businesses throughout the world. As many researchers acknowledge[4, 5, 16, 17], the best way to learn a second language is in a language environment like that of the mother tongue, but it is still a difficult task. Even though there are many off-line refresher courses, a teacher cannot interact with students anytime, anywhere. Due to the availability of the Internet, however, teachers can now use many applications to provide more assistance. When computing power is increased and Internet bandwidth is broad enough, it is possible to communicate easily with other individuals, and synchronous computer-mediated communication (CMC) becomes easier. We treat chat as a means of synchronous, real-time interaction that can be used in an online language classroom to extend the learning process well beyond the traditional four walls, and thereby make the learning process more fascinating, exciting and enriching.
Many teachers and researchers [3, 12, 16, 18, 22] have found that online chat:

- promotes learner autonomy;
- encourages collaborative learning and team work and helps develop group skills;
- promotes communication skills (carrying on a conversation, interviewing, and negotiating meaning);

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 575 – 583, 2006.

- promotes social and socialization skills and proper etiquette (greeting others, introducing oneself, leave talking, stating and reinforcing one's own ideas, interacting politely and appropriately, showing respect and being responsible, making choices, helping, coaching, etc.);
- facilitates interaction and learning with and from people of different cultures who speak different native languages;
- exposes students to speaking a language as it is used by native speakers and allows them to interact in an authentic context with those speakers;
- promotes different types of interaction: student to student, student to teacher, student to expert, and student to online resource;
- offers an appropriate way of fostering inter-peer communication in a real and meaningful environment;
- balances and increases participation among students with less involvement by teachers;
- reduces anxiety among students; and
- provides useful transcripts (chat logs) for studying the language used or for further analysis of a conversation

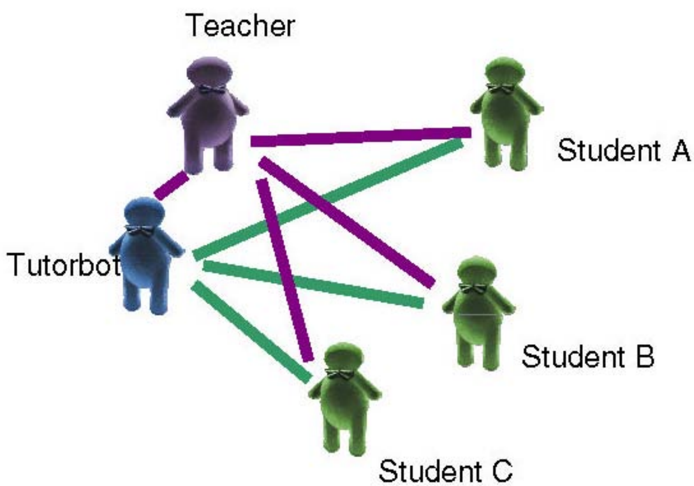


Fig. 1. TutorBot provides synchronous communication between teacher and students

In this paper, we present an enhanced Chatbot system, called TutorBot, which provides students with on-line coaching in a total English learning environment. The question in language learning programs is not whether to teach "phonics", "reading", "comprehension" or "whole language learning," but how to teach them in context —rather than in isolation — so that the learner can make connections between words, pronunciation, and meanings. A Chatbot is defined as a program that emulates human conversation and enables natural-language conversations with computers [20, 21]. The proposed TutorBot has ready reference material (dictionaries, authorized conver-sation material with speaking, and a QA function) that provides synchronous com-munication with students. TutorBot can also record conversations for analysis so that the teacher can assess students' progress.

TutorBot plays two key roles: it acts as an assistant to the teacher, and as a partner to the student. As we know, people work harder to understand material when they feel they are interacting with a partner, rather than simply receiving information passively. Figure 1 shows how TutorBot provides synchronous communication be-tween teacher and students.

2 Related Work: Psychological Reasons

In addition to the benefits mentioned earlier, chat has other advantages that make it an appropriate tool for learning a second language[5, 7, 8, 17]. Chat not only uses a communication medium that generally appeals to students, i.e., the computer, it also takes place in an innovative and exciting setting, namely, cyberspace or virtual reality. Consequently, it puts a strong emphasis on communication and authentic language like that used in the real world, and thereby generates intensive practice in some basic skills. As students are left on their own to a large extent, they have to fend for themselves. This gives them a greater sense of responsibility and a large degree of autonomy. They have to support each other, and involve themselves more intensely in the collective and collaborative construction of knowledge. Finally, the exchanges are about the 'real' world, with 'real' people, in 'real' time. No doubt, most language teachers know how difficult it is to get students to use a second language in class in a meaningful way, especially at the oral level. So, what better way than to use a tool that mirrors real-life interaction?

Given the wide-spread use of online chat by teachers and students, it is of interest to examine its advantages. The ChatBot agent encourages people to communicate via a computer because of the following factors[10, 11, 13, 14, 18, 19, 20, 21]:

- People work harder to understand material when they feel they are in a conversation with a partner, rather than simply receiving information. An agent with learning capacity can grow with the student.
- Pedagogical agents are onscreen characters that help guide the learning process during an e-learning episode.
- Skill development and expertise are strongly related to the time and efficiency of deliberate practice. An "Instant Messaging" based agent can provide practice anytime, anywhere.
- On-line learning is a collaborative endeavor in which participants learn by collaboration.

Margalit et al. [15] indicate that the majority of students and teachers believe that it is possible to learn through the medium of chat.

3 System Architecture

The system architecture of the proposed TutorBot comprises three main components or robots: (1) RRMBot, (2) ClassifyBot, and (3)AIMLBot. RRMBot provides ready reference materials (RRM), including the course dialog, idioms, and a dictionary; ClassifyBot facilitates daily conversation practice; and AIMLBot uses Artificial Intelligence Markup Language (AIML) to provide pattern-based, stimulus-response knowledge content. In addition, we use OpenNLP [2] for language analysis and a spell-check engine to improve AIML. The system architecture of TutorBot is shown in Figure 2.

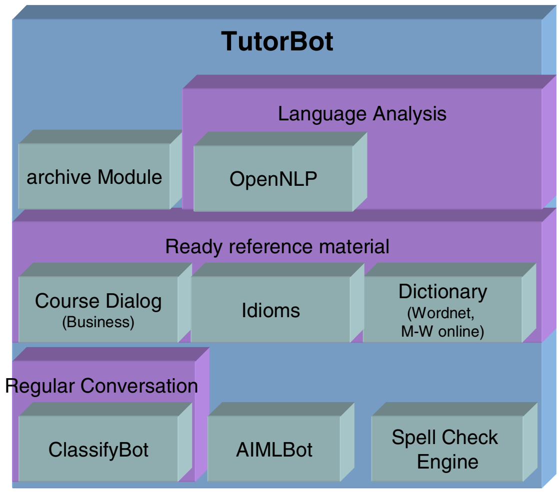


Fig. 2. The system architecture of TutorBot

In the flow chart of the conversation process, shown in Figure 3, the system is divided into a 3-tier architecture, in which every tier has a robot to find the relevant material to answer a student's questions. Initially, the system talks to the student about random topics to decide his/her proficiency level i.e., basic, medium, or advanced. Every level has ready reference (course-based) material and reference sentences.

When TutorBot receives a user's input sentence, the spell check engine processes it and corrects any errors. RRMBot then checks if the sentence exists in the ready reference materials. In addition, ClassifiyBot checks if the sentence exists in the classified conversations, while AIMLBot checks if it exists in AIML conversations. Finally, RRMBot, ClassifyBot and AIMLBot in the English environment are integrated in the conversation UI and passed to TurtorBot for interaction with the user.

3.1 RRMBot (A Robot for Ready Reference Materials)

As noted above, RRMBot provides ready reference materials (RRM). In this paper, RRM denotes a set of well-designed courses and dictionaries. We used the magazines: "Let's Talk in English", "Studio Classroom", and "Advanced," published by "Overseas Radio & Television Inc." as raw materials for course dialogs.

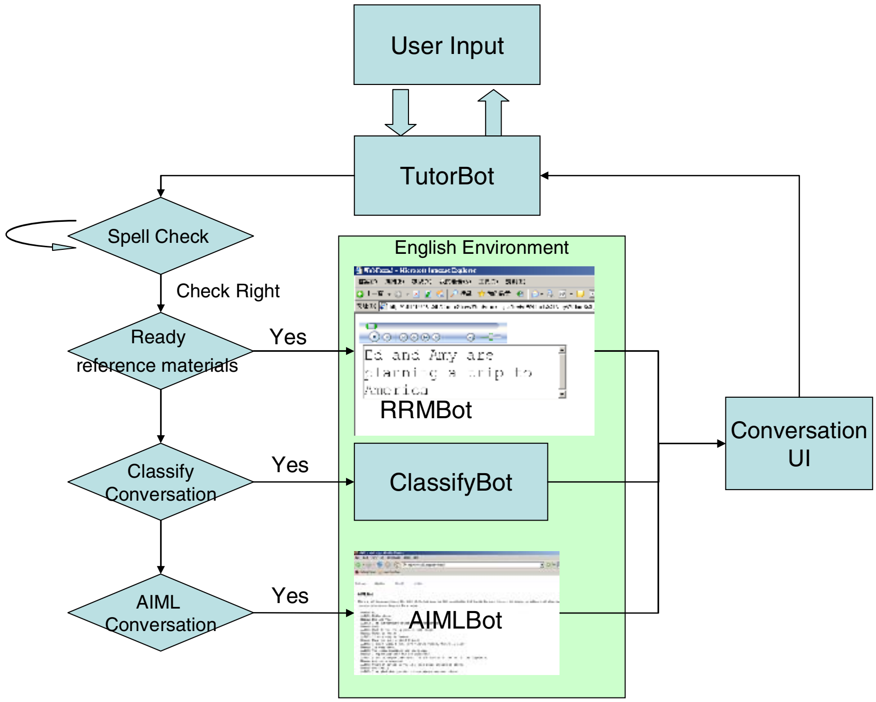


Fig. 3. Conversation flow chart

We adopt WordNet as the lexicon to provide an on-line dictionary service. When a student uses the function “Find: Term” to lookup a dictionary, TutorBot provides an explanation and a URL to retrieve an audio file from Merriam-Webster's Online Dictionary.

For example, TutorBot looks up its dictionary and performs a three-step operation: 1) it enters the new word, e.g., Find: "spectacles"; 2) it finds a description for the item "spectacles"; and 3) it provide a link so that the student can learn the term’s pronunciation.

In addition to the dictionary and the course dialogue, a series of idioms are provided for the student to practice.

3.2 ClassifyBot (A Robot for Daily Conversation Practice)

ClassifyBot provides an e-learning environment based on a corpus; therefore, we can use natural language processing technology to provide automatic item generation, which allows a learner to work with different authentic texts each time. Using a corpus integrated with a POS tagger and the OpenNLP parser, we can provide various pages containing concordances, grammar explanations, bilingual terminology, etc. A POS tagger is an algorithm that classifies words into their "Part-of-Speech" or word classes. In this paper, we use Brill’s POS tagger [6]. OpenNLP is a publicly available library that provides an organizational structure for coordinating several different projects in

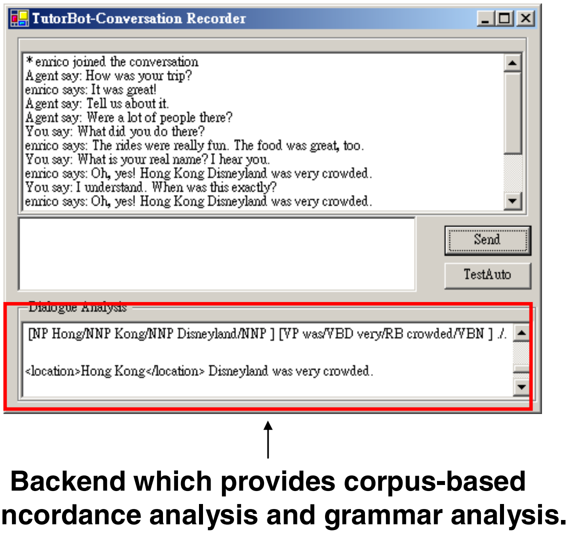


Fig. 4. A ClassifyBot that incorporates POS tagger and OpenNLP parser

NLP [2]. The OpenNLP parser module takes a string as input and produces a structured form representing that string, after which concordance analysis examines a word in the contexts in which it appears [9]. A ClassifyBot that incorporates POS tagger and OpenNLP parser is shown in Figure 4.

The parser provides several types of useful information to the system. For example, chunking/clause information enables a learner to determine whether the structure of a sentence expresses what he or she wants to say. The system can also derive thematic roles from the results of the parser. This is important for dialogues, as it helps the system avoid using material that is irrelevant to the subject.

The conversation shown in Figure 4 demonstrates a chat which user and TutorBot talk about user's trip. When a user inputs the sentence : "Oh, yes! Hong Kong Disneyland was very crowded." to ClassifyBot, the backend of ClassifyBot analyses the dialogue. The process comprises corpus-based concordance analysis and grammar analysis. In addition, the user can specify a command: Grammar "Oh, yes! Hong Kong Disneyland was very crowded." and retrieve the following corpus-based concordance analysis and grammar analysis results from the system: 1) chunk result: "[NP Hong/NNP Kong/NNP Disneyland/NNP] [VP was/VBD very/RB crowded/VBN] ./.", or 2)NER extraction result: "<location>Hong Kong</location> Disneyland was very crowded.". By using the grammar analysis, the TutorBot can find the main topic "Hong Kong" and "Disneyland". Then the processing sentence will be reformulated and expressed in terms of entities and relations in the semantic graph. The semantic graph of context is achieved and used similarity algorithms to find the nearest answer.

3.3 AIMLBot (A Robot which using Artificial Intelligence Markup Language)

Artificial Intelligence Markup Language (AIML) [1] is an XML-compliant language that facilitates the creation of Chatbots with various personalities and kinds of knowledge. The goal of AIML is to enable pattern-based, stimulus-response knowledge content to be provided, received, and processed on the Web and offline in the same manner that is possible with HTML and XML formats.

However, AIML has some problems that we must resolve. Standard AIML uses depth-first search, which does not optimize the result, as the name implies. It finds the first available solution by searching through a tree of possibilities.

The standard AIML definition of "best" does not attempt to minimize anything, but simply finds the first matching pattern, and does not test if other patterns might fit better. There are some simple ways to make the AIML search work very quickly, but they do not guarantee that any "best" equation is true. Hence, AIML just provides students with an English dialogue environment to help them practice anytime, anywhere.

Also, AIML does not include a spell-check function, although it would be very simple to include this in an efficient manner. The technique proposed here is a method for finding the best result rapidly. When AIML searches for a match to a specific word but cannot find one, the next best match is always a wild-card. Instead of matching an unknown word to an unknown group, the word should also be spell-checked and possible alternatives checked for a match with a higher priority than putting them into an unknown group. We use a spell check engine to overcome this drawback of AIML.

4 User Case Scenario and Discussion

In the following, we describe a user case scenario in TutorBot. Students first add a contact ID: msnbot@hotmail-ppe.com to the MSN Messenger contacts list. When students talk to TutorBot, the system interacts with them according to their level of English comprehension.

The proposed TutorBot has a number of advantages. TutorBot plays the role of "assistant instructor" by providing an active practice and topic tutorial for a total English learning environment. The system interacts with students according to their proficiency, since there are three proficiency levels with corresponding ready reference materials i.e., "Let's Talk in English" for beginner level, "Studio Classroom" for intermediate level, and "Advanced" for advanced level. Students can practice their conversation with TutorBot, and interact with it via authorized conversation material with speaking. The interaction is recorded in the archive module, so that the teacher can evaluate the student's learning status with the concordance and grammar analyzer.

Therefore, the proposed TutorBot gives students more opportunities to develop their English skill by allowing them to interact in an authentic context with RRM in real time without being restricted by location.

With the exception of ready reference materials, TutorBot provides a always online peer counseling service which can make students feeling more unrestrained.

5 Conclusion and Future Work

In this paper, we have proposed an intelligent Chatbot system based on instant messaging to provide students with on-line coaching in an English learning environment. The intelligent Chatbot provides a good synchronous collaboration method for language learning. TutorBot plays the role of "assistant instructor" to provide service anytime, anywhere. It facilitates synchronous communication with students by using ready reference materials, including dictionaries, authorized conversation material with speaking, and a question answering function. In addition, the agent provides records and analyzes conversations so that the teacher can assess the students' progress.

Our contribution in this paper is that we use NLP Tool and AIML to integrate several language learning components (words, sentences, sounds, and meanings) in context with an instant messaging-based Chatbot for English as a Second Language programs. Our proposed TutorBot provides the benefits of one-on-one instruction in an English learning environment in an automatic and cost effective way.

Acknowledgements. This research was supported in part by National Science Council under GRANT NSC94-2752-E-001-001-PAE.

References

1. AIML. http://www.alicebot.org/TR/2001/WD-aiml/.
2. OpenNLP. http://sourceforge.net/projects/opennlp/.
3. Abu Shawar, B. and Atwell, E.: Accessing an information system by chatting. Natural Language Processing and Information Systems, vol. 3136, (2004), 407-412.
4. Almeida D'EÇA, T.: To chat or not to chat in the EFL classroom, that is the question! Proceedings of the Language - Communication - Culture International Conference, (2002).
5. Almeida D'EÇA, T.: The Use of Chat in EFL/ESL. TESL-EJ, vol. 7, no. 1, (2003).
6. Brill, E.: Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. Computational Linguistics, vol. 21, no. 4, Dec (1995), 543-565.
7. Coghlan, M.: Finding Your Voice Online - An Inquiry into the Use of Online Voice Applications in Higher Education. The Knowledge Tree: An e-Journal of Flexible Learning in VET, vol. 5, (2004).
8. Cziko, G. A. and Park, S.: Internet Audio Communication for Second Language Learning: A Comparative Review of Six Programs. Language Learning & Technology vol. 7, no. 1, (2003), 15-27.
9. Engel, D.: Modelling Self and Other - A Hybrid Approach to the Analysis of Images of Self and Other in the Radio Addresses Delivered by the American President Before and After 9/11. University of Salzburg, (2004).
10. Farmer, R.: Instant Messaging -- Collaborative Tool or Educator's nightmare! The North American Web-based Learning Conference (NAWeb 2003) (2003).
11. Godwin-Jones, B.: Emerging Technologies: Messaging, Gaming, Peer-to-Peer Sharing: Language Learning Strategies & Tools for the Millennial Generation. Language Learning & Technology, vol. 9, no. 1, (2005), 17-22.
12. Gonzalez, D.: Teaching and Learning Through Chat: A Taxonomy of Educational Chat for EFL/ESL. Teaching English with Technology: A Journal for Teachers of English, vol. 3, no. 4, (2003).
13. Graham-Rowe, D.: Even a chatbot can turn nasty. New Scientist, vol. 188, no. 2521, Oct 15 (2005), 26-27.
14. Lenhart, A., Rainie, L., and Lewis, O.: Teenage life online: The rise of the in-stant-message generation and the Internet's impact on friendships and family relationships. Pew Internet & American Life Project Report, (2001).

15. Margalit, Z. and Sabar, N.: The use of textual chat in distance learning: Students' and teachers' attitudes toward online educational conferences in 4th-12th grades in the Israeli school system. The TCC 2003 Online Conference, (2003).
16. Martín, G. B.: Using Chats in EFL: A Tool for Collaborative Learning. The GREATA Magazine, vol. 9, no. 2, (2002).
17. Mynard, J.: Introducing EFL students to Chat Rooms. The Internet TESL Journal, vol. 8, no. 2, (2002).
18. Poole , B. J., Axmann, M., Calongne, C. M., and Cox, D.: To Chat or to Chatter: Making Online Sense of the Chat Room Environment. The TCC 2003 Online Conference, (2003).
19. Shiu, E. and Lenhart, A.: How Americans use instant messaging. Pew Internet & American Life Project Report, (2004).
20. Tatai, G., Csordas, A., Kiss, A., Szalo, A., and Laufer, L.: Happy chatbot, happy user. Intelligent Virtual Agents, vol. 2792, (2003), 5-12.
21. Tatai, G., Csordas, A., Szalo, A., and Laufer, L.: The chatbot feeling - Towards animated emotional ECAs. Progress in Artificial Intelligence, vol. 2902, (2003), 336-340.
22. Warschauer, M.: Motivational aspects of using computers for writing and communication. Telecollaboration in Foreign Language Learning: Proceedings of the Hawaii Symposium, (1996).

ArikIturri: An Automatic Question Generator Based on Corpora and NLP Techniques

Itziar Aldabe[1], Maddalen Lopez de Lacalle[1], Montse Maritxalar[1], Edurne Martinez[2], and Larraitz Uria[1]

[1] Department of Computer Languages and Systems,
Computer Engineering Faculty, University of the Basque Country
P.O. box 649, E-20080 Donostia, Spain
{jibalari,mlopezdelaca002,montse.maritxalar,larraitz}@ehu.es
[2] Computer Science Department, Udako Euskal Unibertsitatea (UEU)
ELEKA Language Engineering Enterprise
edurne@eleka.net

Abstract. Knowledge construction is expensive for Computer Assisted Assessment. When setting exercise questions, teachers use Test Makers to construct Question Banks. The addition of Automatic Generation to assessment applications decreases the time spent on constructing examination papers. In this article, we present ArikIturri, an Automatic Question Generator for Basque language test questions, which is independent from the test assessment application that uses it. The information source for this question generator consists of linguistically analysed real corpora, represented in XML mark-up language. ArikIturri makes use of NLP tools. The influence of the robustness of those tools and the used corpora is highlighted in the article. We have proved the viability of ArikIturri when constructing fill-in-the-blank, word formation, multiple choice, and error correction question types. In the evaluation of this automatic generator, we have obtained positive results as regards the generation process and its usefulness.

1 Introduction

Nowadays, it is widely recognized that test construction is really time-consuming and expensive for teachers. The use of Computer Assisted Assessment reduces considerably the time spent by teachers on constructing examination papers [11]. More specifically, e-assessment helps teachers in the task of setting tests. For example, in the eLearning Place [3] learning providers create the question bank by means of a TestMaker, a Java Virtual Machine tool. The manual construction of questions is also a fact in SIETTE[5], a web-based tool for adaptive testing. TOKA[8], a web-application for Computer Assisted Assessment, provides teachers with a platform for guided assessment in which, in addition, they construct exercises. All these tools have been used for the assessment of different subjects. However, the work we present in this article is focused on language learning. In our case, learning providers do not have to waste time preparing the questions of the exercises since they are automatically generated.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 584 – 594, 2006.

Some research on automatic generation of questions for language learning has been recently carried out. [12] includes a maker for generating question-answer exercises. [9] reports on an automatic generation tool for *Fill-in-the-Blank Questions* (FBQ) for Italian. [13] describes a method for automatic generation of *multiple choice* questions together with the Item Response Theory based testing to measure English proficiency. [10] uses different techniques such as term extraction and parsing for the same purpose. [4] also describes how to apply Natural Language Processing (NLP) techniques to create *multiple choice* cloze items. Finally, [7] reports a machine learning approach for the automatic generation of such type of questions. In the mentioned works, the authors present different methods for the automatic generation of language questions based on NLP techniques. However, in almost all of them the methods and the system architectures are only focused on a single question type. In contrast, this article proposes a NLP based system, which is able to generate four different types of questions: *FBQ, word formation, multiple choice,* and *error correction*. Moreover, we propose a system where all the inputs and outputs are in XML markup-language. Concerning to its architecture, we point out some existing differences among our system and the above mentioned ones in section four.

The system we propose provides teachers and testers with a method that reduces time and expenditure for testing Basque learners' proficiency. The user chooses which linguistic phenomena s/he wants to study, and what types of questions s/he wants to create. The system generates, automatically, the question types that the user has requested. For this, the system makes use of a real corpus and some NLP tools for Basque developed in the IXA research group[1].

In section two, we present ArikIturri, the automatic corpus-based question generator. In section three we briefly describe the question model used by this system. Section four deals with the development of the system; we talk about its architecture, as well as the use of the NLP tools and the source corpus. In section five, we comment on the evaluation of ArikIturri, and we present Makulu, the assessment application we have used for the evaluation of the generator. Finally, some conclusions and future work are outlined.

2 ArikIturri: The Question Generator

Here we present a question generator for Basque, named ArikIturri. This is a system with an open architecture (section four) to generate different types of questions. ArikIturri makes use of a data bank, which consists of morphologically and syntactically analysed sentences where phrase chunks are identified. The input of ArikIturri is represented by XML mark-up language. The outputs are question instances of a model defined also in XML mark-up language (see figure 1).

The system generates automatically the question instances. For that, it makes use of two kinds of language resources: NLP tools and specific linguistic information for question generation.

ArikIturri is independent from the assessment application, which will use the questions created by the generator. Indeed, it is the assessment application that determines

[1] http://ixa.si.ehu.es/Ixa

the type of the questions to be generated as well as the linguistic phenomena treated in those questions. For this reason, the question instances generated automatically by ArikIturri must be imported to the assessment application. This way, as showed in figure 1, different assessment applications can benefit from the question generator.

Although the knowledge representation is not a matter of this paper, we want to outline that the importation of question instances implies the matching between the concepts defined in the question model of ArikIturri and the representation of the domain of the assessment application that is using the automatic question generator.

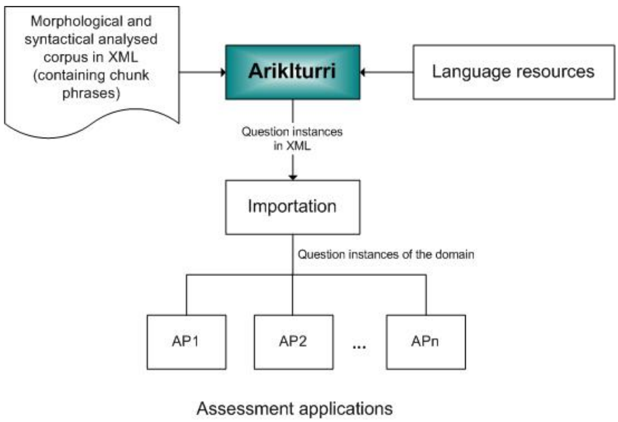


Fig. 1. ArikIturri is independent from the assessment applications

As said before, ArikIturri can generate different types of questions: *FBQ, word formation, multiple choice,* and *error correction*.

As concerns FBQ, when deciding the blanks of the question, the generator itself chooses which are the units to be removed from the text. In our approach, the system can construct questions with more than one blank, and each of them could be filled with one or more words, depending on the exercise. In order to get the blanks, the system identifies the morphosyntactic categories of the phrases in the source sentence. So far, we have experimented with two types of linguistic phenomena: morphological inflections and verb conjugation for auxiliary and synthetic forms.

Word formation questions consist of a given sentence and a word whose form must be changed in order to fit it into the sentence. To generate this question type, we use a lemmatiser for Basque [2], which gives the word (a lemma) to be changed in the blank.

In error correction questions, the aim is to correct the errors, which can be marked or not. And, in multiple choice questions types, we find a set of possible answers. Only one of them could be correct in that context, whilst the rest are incorrect answers, i.e. distractors.

In fact, in the case of multiple choice and error correction question types, the generator has to create distractors, which are the incorrect answers the system offers. The techniques used for the automatic generation of distractors can have a big influence in the results. In our case, we have defined different methods to generate distractors. As far as the generation of inflection forms is concerned, the system uses techniques of replacement and duplication of declension cases (inessive, ablative, dative…), number (singular/plural) or the inflection paradigm (finite, indefinite). As regards the verb forms' generation, the system creates the distractors by changing the subject person, the object person, the verb mode, the tense, the aspect or the verb paradigm. We have defined these techniques to create distractors as a parameter of the generator. This way, we are able to research and improve the generation of distractors depending on the results we obtain.

We also want to say that, although test correction in assessment applications for learners is not a matter of our present research, we think it is important to consider this aspect too. For the moment, and based on the source corpus, the generator provides us with only one possible correct answer in each question instance.

3 A Brief Description of the Question Model

As we have already said, the outputs of the generator are question instances of a question model defined in the XML mark-up language. A question is not an isolated concept but it is represented as a part of a whole text.

In this model, a question can have more than one answer focus. For example, FBQ can have more than one blank to fill in. Concretely, a question could have as many answer focuses as phrases are identified in the question. Of course, if the number of answer focuses was equal to the number of phrases, we would have a completely empty question with no sense. That means that the generator has to control the number of answer focuses in respect to the number of phrases of the source sentence.

Therefore, one of the main characteristics of the question model we have defined is that we find different answer focuses within the same question, i.e. more than one possible blank in a FBQ exercise. The focus is always a phrase of the sentence. The phrase, on its own, has a head, which corresponds to the words of the blank of the generated question. The head concept is defined by one possible answer, i.e. the one corresponding to the source text, any number of distractors (zero in the case of some question types), and lexical and morphosyntactic information.

Finally, it is important to stand out that the order of the phrases in the generated questions is not necessarily the same as the order in the source sentence. This is a very enriching characteristic of this question model because Basque is a free word order language. Besides, the fact that the phrases have not to be in a fixed position offers us the chance to extend the application of the method to generate new exercise types such as word order, transformation and question answering.

4 Development of the System

In this section, we explain the architecture of ArikIturri and some important aspects studied during the development of its architecture. Indeed, the source corpus and the NLP tools that the generator uses have a big influence in the quality of the system. Basically, it is important to have robust NLP tools that provide us with correct linguistic analysis. Those tools have been developed in the IXA research group at the University of the Basque Country.

4.1 The Architecture

Here we describe the main modules of ArikIturri's architecture. The generator uses as input a set of morphologically and syntactically analysed sentences (the tagged corpus), represented in the XML mark-up language, and it transforms them into the generated questions, represented in XML. The representation of the results obtained in the intermediate stages is also in XML.

As mentioned in the introduction, there are some differences among the architecture of this system and the architectures of previous works ([9],[12]). We have distinguished an *Answer Focus Identificator* module and an *Ill-formed questions rejecter* module. [13] also includes a module to reject questions, which is based on the web.

Figure 2 shows the main modules of the architecture. Depending on the parameters' specifications, *the sentence retriever* selects candidate sentences from the source tagged corpus. In a first step, it selects the sentences where the specified linguistic phenomena appear. Then, the *candidates selector* studies the percentages of the candidates in order to make random selection of sentences depending on the number of questions specified in the input parameters.

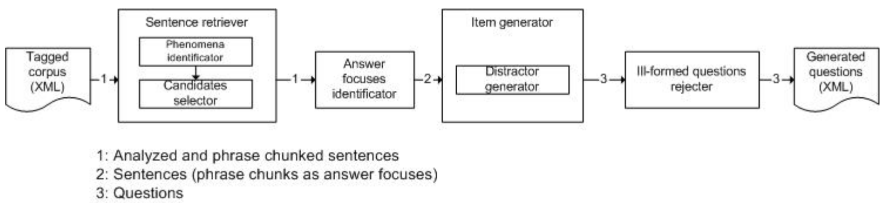


Fig. 2. The main modules

Once the sentences are selected, *the answer focuses identificator* marks out some of the chunked phrases as answer focuses depending on the morphosyntactic information of the phrases. Then, the *item generator* creates the questions depending on the specified exercise type. That is why this module contains the *distractor generator* submodule. At this moment, the system has already constructed the question instances. However, as the whole process is automatic, it is probably that some questions are ill-formed. Because of that, we have included *the ill-formed questions rejecter* in the architecture. In section 4.2 we explain how the main modules work, from the perspective of the NLP tools.

4.2 The Influence of the NLP Tools and the Source Corpus

As we foresaw at the beginning of our research, our experiments have proved that the source corpus and the NLP techniques used in the process of question generation determine the quality of the obtained questions. In the next lines, we explain the influence of these two aspects in the development of the system.

4.2.1 The NLP Tools

The results of the question generation based on corpora depend very much on the matching between the linguistic information of the answer focuses of the question and the specific linguistic phenomena that teachers want to test. When working with NLP tools, the robustness of those tools undoubtedly determines the results. The results we have obtained depend, in some way, on the quality and sophistication of the morpho-syntactic parser, the syntactic parser and the phrase chunker for Basque [1] we have made use of for the analysis of the real corpus. In some small experiments we carried out when developing ArikIturri, we studied the output of the mentioned NLP tools. Since the information given in this output did not always respond to our needs, we realised that those tools determine the results of *the sentence retriever*. As a consequence, we had to discard the study of some linguistic phenomena, such as the inflection of demonstrative pronouns or the genitive case. This problem could be solved, of course, by making some changes in the mentioned NLP tools.

The task of the *answer focuses identificator* depends on the quality of the chunked phrases. In our implementation, if the phrase has different analysis corresponding to different linguistic phenomena, the focuses' identificator do not consider that phrase as a candidate answer focus.

We have adapted the verb conjugation tool and the morphological declension tool for Basque language [6] in order to generate different well-formed words as distractors that are incorrect in a particular context, i.e. in the generated question. Sometimes, the conjugation tool and the declension tool give no output because its input parameters, automatically set by the *distractor generator,* have no sense. In these cases, the declension tool does not produce any distractor. This way, if the number of generated distractors does not match with the input parameters of ArikIturri, the *ill-formed distractors rejecter* will mark the generated question as deleted. The rejecter also controls if there are duplicated distractors in the same answer focus. This is possible, for example, because Basque sometimes uses equal inflection forms for different linguistic features.

Here we show an example of a rejected multiple choice question type. The module rejects the question because there are two identical distractors i.e. b) and c) for different inflection forms[2]. The choices of the question are *answer focuses* where the **head** of the answer is in bold.

"Dokumentua sinatu zuten________"
("They signed the document ________")

a) *alderdiaren* ***izenean*** (innesive definite singular – in the name of the political party)
b) *alderdiaren* ***izenetan*** (innesive definite plural – in the name of the political parties)
c) *alderdiaren* ***izenetan*** (innesive indefinite – in the name of certain political parties)
d) *alderdiaren* ***izen*** (lemma –name of the political party)

In respect to the word formation questions, the results make sense if some variation of the showed word matches with the answer focus. In our implementation, the word is an automatically identified lemma. That is why the correctness of the lemmatiser used when disambiguating different lemma candidates for the answer focus

[2] In this paper, we show the example as presented to the expert teacher, after the importation of the XML instances.

considerably affects the evaluation of the appropriateness of the word formation question type.

4.2.2 The Source Corpus

The language level of a text is a controversial aspect because it is difficult to define it. However, language schools are used to classify real texts according to the established language levels. In order to carry out our experiments, we have analysed the corpora classified into three different language levels. Expert teachers chose the texts. Initially, we thought that the number of instances for each linguistic phenomenon would change depending on the language level. However, the results of the analysed texts show that there is not significant difference on the rates, at least as far as morphological inflection and verb conjugation are concerned. Based on these results, we were not sure about the importance of the distinction by language levels for our experiments. And we finally decided to use only one level corpus, i.e. the high language level corpus (234 texts) for making experiments in order to define which linguistic phenomena we would use in the evaluation (section five). Using high language level corpus, we have avoided the noise that teachers would generate when discarding questions at lower levels because of the difficulty the students could find to understand the sentences.

The linguistic phenomena defined in the curricula of Basque language schools were the starting point of our experiments. Initially, we analysed the whole corpus (1303 texts - 44009 sentences) and found out that some of the linguistic phenomena taught at the language schools did not appear in that corpus. Concretely, the results of the experiments made with the *sentence retriever* showed that the number of appearances of certain verb forms was too low in the corpus. This way, we verified that the corpus limits the linguistic phenomena that can be treated in the generated questions. Moreover, the percentage of appearance of the phenomena in the corpus did not match with the importance rate that teachers gave to the learning of these linguistic contents.

As we said in section two, we focused our experiments on the study of morphological inflections and verb conjugation. With the sentence retriever, we used a sample of the corpus, i.e texts of high language level (234 texts; 10079 sentences). The results of the experiments with the *sentence retriever* determined the criteria for the evaluation of the system. More specifically, we chose five different inflection cases (sociative, inessive, dative, absolutive and ergative) and four different verb forms (present indicative, past indicative, present indicative-absolutive, present indicative-ergative) corresponding to different paradigms, modes, aspects and tenses. The sample corpus contained 16108 instances[3] of the selected inflection cases (703, 4152, 1148, 7884 and 2221, respectively) and 7954 instances of the selected verb forms (3814, 100, 2933 and 1107, respectively).

5 The Evaluation of ArikIturri

The experiments we have carried out in order to generate questions automatically have proved the viability of ArikIturri when constructing fill-in-the-blank, word

[3] 16108 inflection cases detected in 10079 sentences means that some of the cases appear more than once in the same sentence.

formation, multiple choice, and error correction questions. Although we have generated four different types of questions, for the evaluation of the system we have only taken into account the results obtained with multiple choice and error correction question types.

As said before, the NLP tools and the source corpus determine, in some way, the results of the question generator. As a consequence of the experiments carried out during the development of the system, we decided to evaluate ArikIturri taking as source data the high language level corpus and focusing on some specific linguistic phenomena, i.e. five inflection cases and four verb forms.

In this section, we present a manual evaluation of the questions created by our automatic generator. For this evaluation, we have used Makulu, a web-application developed for Computer Assisted Assessment. In this way, we demonstrate that the question instances generated by ArikIturri have been imported to the domain of the Makulu assessment application (see figure 1).

5.1 The Selected Corpus

Taking into account the few economic and human resources we had to carry out a significant evaluation, we decided to use a corpus of 1700 sentences selected from the high language level corpus. As said in section 4.2, this corpus consisted of 10079 sentences. The sentence retriever identified the selected linguistic phenomena for the evaluation, and, out of 10079 sentences we chose 1700 at random.

As the study of the automatic generation of distractors was also an interesting aspect, we limited our evaluation to multiple choice (500 sentences) and error correction (1200 sentences) question types. The reason for selecting a higher number of error correction questions was that the evaluation of this type of questions is less time-consuming.

Once we selected the analysed sample corpus, we used ArikIturri to automatically generate the questions. The *ill-formed questions rejecter* of the generator automatically rejected 58 multiple choice question instances and 292 error correction instances out of all the generated questions. This way, we obtained a sample of 1350 question instances to evaluate.

5.2 The Manual Evaluation

For the manual evaluation of ArikIturri, we have used Makulu. Makulu is a web-based assessment application for helping teachers to set test questions and assessing learners in their language proficiency. Makulu groups different questions in order to present a whole exercise. Students have two options: to make the exercises within a whole text, or to have exercises composed of grouped questions. Makulu gives to the students the results of their sessions as well as the correct answers. The task of the teachers is to set the questions that are used in learners' tests.

In order to evaluate the results of ArikIturri, we have used the questions' setting view of Makulu with an expert teacher. Makulu requests ArikIturri to generate language questions. These questions are imported to the Makulu's database. After setting the tests into Makulu, teachers can modify some components of the answer focus of the automatically generated questions. For instance, in multiple choice question type they could change the distractors, but they can never modify the correct answer, because it corresponds to the source corpus. In the next figure we can see an example of an automatically generated question. The teacher has different options: to accept it on its own, to discard it if it is not an appropriate question, or to modify it if s/he considers that, among the options, there is more than one possible correct answer. The set of rejected and modified questions give us a way to evaluate the automatically generated questions.

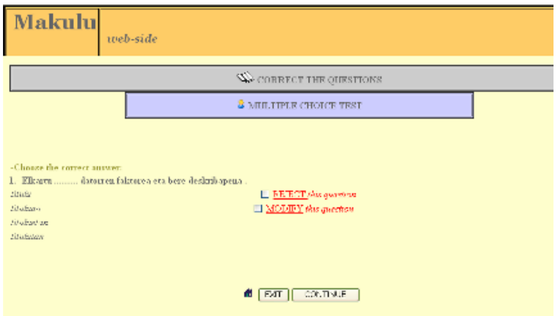


Fig. 3. Teacher's evaluation of the questions

In the next lines, we comment on the results obtained from the session with the expert language teacher. The teacher actually evaluated 1350 question instances by means of the Makulu web application. She spent 15 hours for manual evaluation. In the evaluation, we asked to the expert teacher to modify or reject questions only if they were not well formed. The expert teacher evaluated 908 questions of error correction type and 442 questions of multiple choice type. If we consider that all the questions discarded or modified by the teacher were not well generated, the results show that the percentage of the accepted questions was of %83,26 in the case of error correction questions and of %82,71 in the case of multiple choice questions. These percentages show us that the developed automatic generator obtains indeed good results. This assertion becomes even more important if we consider the time that the expert teacher needs for setting the questions. It is clear that the setting of the same number of questions with an assessment application of manual construction is more expensive and time-consuming.

Considering that, in the case of error correction questions the generator creates only one distractor per each answer focus (the error), and three different distractors in the case of multiple choice questions, the percentage of well-formed questions should be higher for error correction questions. In addition, a deeper study of the results shows that the methods used for generating distractors and the linguistic phenomena seem to have a big influence on the correctness of the generated questions. These aspects imply a more exhaustive evaluation of ArikIturri.

6 Conclusions and Future Work

The automatic generation of knowledge construction reduces considerably the time spent by teachers on constructing exercises. In this paper, we have shown the results obtained in the development of a system, ArikIturri, for automatic generation of language questions. Concretely, we have proved the viability of this system when constructing, automatically, fill-in-the-blank, word formation, multiple choice, and error correction question types.

The experiments carried out during the implementation of the system have proved that the source corpus and the NLP techniques used in the process of question generation determine the quality of the obtained questions.

ArikIturri is independent from the assessment application, which will use the questions created by the generator. Indeed, it is the assessment application that determines the type of the questions to be generated, as well as the linguistic phenomena treated in those questions. In the present work, we have experimented with Makulu, an assessment application for helping teachers to set test questions and assessing learners in their language proficiency. We have used the questions' setting view of Makulu, in order to evaluate the results of ArikIturri.

We have also presented the results of the evaluation of ArikIturri with multiple choice and error correction question types. Those results demonstrate that the automatic generator is good. In fact, the well-formed questions are more than %80. Moreover, the use of this generator is less expensive and time-consuming than manual construction of language questions. It would be interesting to evaluate the required time to create the same number of questions without ArikIturri; in this way, we would have a real precision of the time expert teachers can save with the system.

For the near future, we foresee to carry out deeper evaluations for studying the quality of the methods used to generate distractors. We also consider very important to make new evaluations with language learners and to compare the results given by different expert teachers. In addition, we are planning to make new experiments to generate new types of test questions such as question answering, word order and transformation.

References

1. Aduriz, I., Aranzabe, M., Arriola, J., Díaz de Ilarraza, A., Gojenola, K., Oronoz, M., Uria, L.,:A Cascaded Syntactic Analyser for Basque. Computational Linguistics and Intelligent Text Processing. 2945 LNCS Series. Springer Verlag. Berlin. (2004) 124-135
2. Alegria, I., Aranzabe, M., Ezeiza, A., Ezeiza, N., Urizar, R.: Robustness and customisation in an analyser/lemmatiser for Basque. LREC-2002 Customizing knowledge in NLP applications workshop (2002)
3. Boyle, A., Russell, T., Smith, S., Varga-Atkins, T.: The eLearning Place: progress report on a complete system for learning and assessment. Proceedings of the Eight International Computer Assisted Conference. M. Ashby (eds.) http://www.caaconference.com/ (2004) 71-77
4. Chao-Lin Liu, Chun-Hung Wang, Zhao-Ming Gao, Shang-Ming Huang: Applications of Lexical Information for Algorithmically Composing Multiple choice Cloze Iems. Proceedings of the 2nd Workshop on Building Educational Applications Using NLP, Ann Arbor, (2005) 1-8
5. Conejo, R., Guzmán, E., Millán, E., Trella, M., Pérez-De-La-Cruz, J.L., Ríos, A: SIETTE: A Web-Based Tool for Adaptive Testing. International Journal of Artificial Intelligence in Education, 14(1), (2004) 29-61

6. Díaz de Ilarraza, A., Maritxalar, M., Oronoz, M: IDAZKIDE: an intelligent CALL environment for second language acquisition. Proceedings of a one-day conference "Natural Language Processing in Computer-Assisted Language Learning" organised by the Centre for Computational Linguistics , UMIST, in association with EUROCALL, a special ReCALL publication. UK. (1999) 12-19
7. Hoshino, A., Nakagawa, H.: A real-time multiple-choice question generation for language testing. Proceedings of the 2nd Workshop on Building Educational Applications Using NLP, Ann Arbor. (2005) 17-20
8. Kerejeta, M., Larrañaga, M., Rueda, U., Arruarte, A., Elorriaga J.A.:A Computer Assisted Assessment Tool Integrated in a Real Use Context. In Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies. ICALT. (2005) 848-852
9. Kraift, O., Antoniadis, G., Echinard, S., Loiseau, M., Lebarbé T., Ponton C.: NLP Tools for CALL: the Simpler, the Better. Proceedings of the InSTIL/ICALL2004 – NLP and Speech Technologies in Advanced Language Learning Systems, Venice. (2004)
10. Mitkov, R., An, L.: Computer-aided generation of multiple-choice tests. Proceedings of the 1st Workshop on Building Educational Applications Using NLP. HLT-NAACL. (2003) 17-22
11. Pollock, M.J., Whittington, C.D., Doughty, G.F.: Evaluating the Costs and Benefits of Changing to CAA. Proceedings of the Fourth International Computer Assisted Conference CAA, http://www.caaconference.com/. (2000)
12. Schwartz, L., Aikawa, T., Pahud, M.: Dynamic Language Learning Tools. Proceedings of the InSTIL/ICALL2004 – NLP and Speech Technologies in Advanced Language Learning Systems, Venice. (2004)
13. Sumita, E., Sugaya, F., Yamamota, S.: Measuring Non-native Speakers' Proficiency of English by Using a Test with Automatically-Generated Fill-in-the Blank Questions. Proceedings of the 2nd Workshop on Building Educational Applications Using NLP, Ann Arbor. (2005) 61-68

Observing Lemmatization Effect in LSA Coherence and Comprehension Grading of Learner Summaries

Iraide Zipitria[1,2], Ana Arruarte[1], and Jon Ander Elorriaga[1]

[1] Language and Information Systems Department, Computer Science Faculty, University of the Basque Country, Manuel Lardizabal pasealekua, 1 20018 Donostia, Basque Country, Spain

[2] Department of Research Methods in Education (MIDE), University of the Basque Country, Tolosa etorbidea, 70 20018 Donostia, Basque Country, Spain

{iraide, arruarte, elorriaga}@si.ehu.es

Abstract. Current work in learner evaluation of Intelligent Tutoring Systems (ITSs), is moving towards open-ended educational content diagnosis. One of the main difficulties of this approach is to be able to automatically understand natural language. Our work is directed to produce automatic evaluation of learner summaries in Basque. Therefore, in addition to language comprehension, difficulties emerge from Basque morphology itself. In this work, Latent Semantic Analysis (LSA) is used to model comprehension in a language in which lemmatization has shown to be highly significant. This paper tests the influence of corpus lemmatization while performing automatic comprehension and coherence grading. Summaries graded by human judges in coherence and comprehension, have been tested against LSA based measures from source lemmatized and non-lemmatized corpora. After lemmatization, the amount of LSA known single terms was reduced in a 56% of its original number. As a result, LSA grades almost match human measures, producing no significant differences between the lemmatized and non-lemmatized approaches.

1 Introduction

A relevant task for Intelligent Tutoring Systems (ITSs) is to produce adequate diagnosis of learner domain comprehension.

Among different strategies in student learning diagnosis, one is to infer learner comprehension analysing learner responses produced in Natural Language. The main advantage of this approach is to allow learners a greater response freedom. As a result, richer information on learner comprehension is gathered, as they have no hint or boundaries in response production. Thus, learners have the whole responsibility over the produced answer.

In this context, we are leading to free text comprehension for the evaluation of summaries. Summaries are widely used as an educational diagnostic strategy to observe comprehension, or how much information from text is retained in memory [1-3]. As it happens in other educational diagnostic methods, it does not necessarily produce

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 595 – 603, 2006.

a perfect match of learner knowledge, but it produces a good approximation of the information retained in memory.

Overall grades of summaries are decided taking into account several considerations. Behind a final summary assessment mark, there are certain parameters that contribute to the final grade. But, how do they relate to the global summary grade? Are they all equivalent? Not all the variables affect final grading decisions in the same way [4]. When deciding the final summary-grade some variables have more relevance than others. But, once the summary grading context is defined, the automatic measures for each variable need to be calculated.

The tutoring system needs to comprehend the summary. Free text makes its automatic evaluation more complex and bounded to Natural Language Processing (NLP). Many NLP solutions are strongly bounded to the specific language for which they have been created. Then, when using automatic NLP related open-ended diagnostic systems, language change requires adapting its NLP related modules to the specific language. Similarly, although it is a more general approach, using language comprehension models such as Latent Semantic Analysis (LSA) [5], might require certain level of adjustments for morphologically different languages. Due to it, we have run this study to observe the behaviour of LSA understanding of summaries on an agglutinative context.

In order to make progress in this matter, discourse and content related variable measures are automatically obtained using Latent Semantic Analysis. This work studies the impact of lemmatizing an agglutinative language to automatically grade comprehension and coherence in learner summaries. The paper starts describing LSA and language, follows with human-LSA comparison experiments and finishes with conclusions.

2 Latent Semantic Analysis and Agglutinative Languages

Automatic modelling of natural language comprehension has progressed dramatically using Natural language understanding techniques such as LSA [5]. We are working towards the improvement of the automatic evaluation of text comprehension and coherence on summaries using Latent Semantic Analysis (LSA).

Latent Semantic Analysis (LSA) is a statistical corpus-based Natural Language understanding technique, which was first developed by [6] and later found to be comparable to humans by [5, 7]. It has been widely used to model human semantic similarity on a variety of contexts [1, 8-11]. It has also been successful in summary evaluation [12], and for automatic grading of comprehension and coherence [8, 12, 13].

Text comprehension is graded calculating LSA cosine of a learner text in comparison to target text. It can be the reading text, graded summaries or sections of the reading text. When comparing to sections of the text LSA provides information on text coverage [13].

Another relevant measure for discourse comprehension is text coherence, which is calculated by comparing each sentence with subsequent sentences in the text. In a similar way subject change can also be graded by comparing subsequent paragraphs [8].

Several ITS systems have already successfully measured comprehension and coherence using LSA [10, 12, 14]. But, LSA judgments do not take into account word order or syntactic information. How would it affect to an agglutinative language? Effects of modelling an agglutinative language using LSA are analyzed in the next section.

2.1 Merging Syntax and Semantics in Word Level

There is psychological foundation in syntax semantics relation. Syntax and semantic relatedness has been object of several psycholinguistics studies. From early language acquisition meaning is acquired together with syntax [15]. Further evidence for interaction between syntax and semantics in sentence comprehension has been found by [16, 17]. But, research in LSA shows that despite the lack of syntactic information or word order LSA produces human like similarity measures [18].

However, there are languages in which a high amount of syntactic information together with semantic content is carried not only at sentence level but by the word itself. This is the case of agglutinative languages where words contain postpositions carrying several forms of syntactic information.

Table 1. Effect of lemmatization on the word **etxe/casa**(house). This example clearly shows variability and word morphology difference in relation to Spanish.

Basque lemmatized	*Basque*	*Spanish lemmatized*	*Spanish*
etxe	**etxe**	**casa**	**casa**
	etxea		**casa**s
	etxeak		
	etxean		
	etxearen		
	etxeek		
	etxeen		
	etxeetako		
	etxeetan		
	etxeetara		
	etxeko		
	etxekoak		
	etxera		
	etxetatik		
	etxetik		
	etxez		

An example of this sort of languages is Basque. It is a non Indo-European agglutinative language, in which words are formed by joining morphemes together producing a greater amount of word variability –see example in Table 1–. In addition, Basque sentences tend to follow Subject-Object-Verb (SOV) structure. Nonetheless, the order of words on a sentence in Basque can vary. Due to it, Basque is considered a free word ordered language.

Then, considering how LSA works, when producing the LSA matrix in Basque LSA would not only consider word level semantic information, but also syntactic information related to it. For instance, the word **etxetik** agglutinates the information contained in the English words **from the house**. Then, the implication of using this agglutinative word in Basque would produce the same effect as using **fromthehouse** and similar form types. Each of these word constructions would be considered distinct from the word **house** when constructing the LSA matrix. Then, in the case of Basque when producing a matrix using the common LSA procedure, not only concept level but further syntactic information would be considered joined to each particular conceptual context.

Then, the approach of using lemmatization in Basque corpora would attempt to solve this difference in language morphology. The goal of this method is to develop a matrix that would resemble better other non-agglutinative language structures. By lemmatizing, word level syntactic information is filtered leaving only content relevant conceptual information. Following the previous example, instead of using words of the type of **fromthehouse** only **house** would be considered when developing the LSA semantic space. Hence, removing word level syntactic information, this approach tries to compare better to LSA matrixes of non-agglutinative languages. Results from both types of matrixes (lemmatized and non-lemmatized) are compared to human data to see which one resemble better human results.

Moreover, recent studies refer to the effect of syntax in LSA similarity judgments [19-21]. SELSA, adds syntactic neighbourhood to words to improve LSA results [21]. FLSA, widens LSA with dialogue act classification information [20]. Finally, SLSA shows that adding structural information it is possible to obtain better measures and deeper knowledge on similarity of the different parts of the sentence [19]. Then, different variations of syntactic information over semantic calculations are producing positive effect on LSA results.

Nonetheless, current methods to measure coherence and comprehension with LSA do not take into account text structure or syntactic information [8]. Hence, considering the impact of structural information agglutination in semantics, we have tested the lemmatized and non-lemmatized methods, to observe how the syntax-semantic combination affects to grading in an agglutinative and a non-agglutinative language.

In addition, the relevance of the use of adequate corpora to obtain optimum LSA results has been found to be very relevant in previous studies [22]. Moreover, LSA is known to be a general approach adaptable to a variety of languages [23]. Nevertheless, thus far LSA has mainly been tested with English corpora and non-agglutinative languages. But, do results obtained so far apply the same way to every language? Which is the effect of lemmatization and adding text structure when measuring coherence and comprehension with LSA?

3 Comparing LSA and Lemmatized-LSA to Measure Coherence and Comprehension in Basque

Previous research in LSA shows that the use of syntax can improve LSA similarity results [19-21]. Our work test different modes of LSA measures (lemmatized and non-lemmatized LSA) against human evaluation ratings to observe which method compares better to expert summary grading. More specifically, 15 teachers participate as graders.

3.1 Procedure

Five summaries were evaluated by 15 teachers to measure coherence and comprehension. The obtained results are compared to two methods of LSA matrix composition. One of them the matrix is constructed from Basque text in natural language and the other one from lemmatized Basque corpus. In a similar way, in the lemmatized LSA mode students summaries were also lemmatized to be compared to the lemmatized terms.

As part of the pre-processing, summaries were spell-checked to avoid LSA to get confused by learner spelling mistakes; this way, misspelled words would be significant for LSA when measuring comprehension and coherence.

The coherence and comprehension grading were calculated following previous measuring strategies [8, 12, 13] and tested under both types of corpora.

The main reason to observe the effect of lemmatization was the relevance of this method in agglutinative languages and more specifically in Basque. Moreover, similar work in sentence level demonstrates lemmatized corpora to be significantly closer to human judgments than the non-lemmatized one. Therefore, the goal of this work is to observe the final effect of corpus lemmatization in the final LSA based automatic comprehension and coherence grades. The cited corpora have been developed from a (Basque) book collection on economy, music, marketing, administration, fossils and the world. Corpora was lemmatized using POS tagging for Basque [24] .

LSA matrix dimensionality was chosen based on results observed on previous research with the same lemmatized and non-lemmatized corpora. LSA matrixes where created for 150 dimensions in its non-lemmatized versions and 300 dimensions in the lemmatized versions.

3.2 Results

The non-lemmatized-Basque version of the corpus, due to the agglutinative characteristic of the language, produced in the LSA semantic space 56% more term variability than its parallel non-lemmatized equivalent (see Table 2). In other words, after lemmatization, the corpus is reduced in more than half of its original size.

Table 2. Lemmatized vs. Non-lemmatized corpora effects in amount of distinct words recognized in LSA.

Corpora	LSA terms
Basque	30.111
Lemmatized Basque	13.188

LSA **comprehension** measures were computed following previous work in LSA discourse grading [12, 13]. Human and LSA grading comparison is shown in the Table 3:

Table 3. Comparison of LSA and human grades to judge reading text comprehension in learner summaries

	Human grades	LSA	Lemmatized LSA
Summary 1	6.9	5.7	5
Summary 2	4.7	4.8	4.5
Summary 3	4.2	4.6	4.1
Summary 4	9	5.1	5
Summary 5	3.3	4	4

Although, both corpora produce similar judgments, the non-lemmatized LSA method shows slightly more accuracy than the lemmatized one in relation to human grades. Nonetheless, the difference variability between them is not very significant with a standard deviation of only 0.22. The non-lemmatized version has a standard deviation of 0.89 in comparison to human grade mean and the lemmatized one 0.97. Nonetheless, none of them is greater than the human inter-rater deviation 1.55. Most of LSA grades for comprehension were very close to the mean grade provided by human graders. Moreover, when observing individual teacher grades in several summaries LSA matches human grades. Therefore, in this data deviation between LSA and humans would not be different from human judge variability in almost every summary.

The only significant variability found is the high deviation shown between LSA and human judges in *Summary 4*. Here, comprehension grading is 4 points lower for both LSA modes than the mean shown by humans. Moreover, a very significant point to highlight is that it is precisely in highly rated summaries where our graders show less deviation. For *Summary 4* graders produced an inter-rater deviation of 0.75.

A very similar effect was found when automatically grading the effect of coherence. LSA grades for **coherence,** in most of the cases, were very close to the mean grade provided by human graders. LSA coherence measures were calculated in concordance to previous research in LSA [8]. The resultant human and LSA grading comparison is shown in the Table 4:

Table 4. Comparison of LSA and human grades to judge learner summary coherence

	Human grades	LSA	Lemmatized LSA
Summary 1	5.6	5.6	5.14
Summary 2	3.2	4.7	3.94
Summary 3	3.8	4.6	4.03
Summary 4	8.9	4.8	3.69
Summary 5	4.2	5	3.14

Although, both corpora produce similar judgments, the non-lemmatized LSA method again shows a higher level of accuracy than the lemmatized one. In this case, the variability between them is a little higher with a standard deviation of only 0.67.

The non-lemmatized version has a standard deviation of 1.02 in comparison to human grade mean and the lemmatized one 1.08. Nonetheless, none of them is greater than the human inter-rater deviation 1.1. When observing some of the individual teacher grades in several summaries out of the 15 judges, LSA precisely matches human grades. Therefore, in coherence grading deviation between LSA and humans would not be different from human judge variability.

Here again, variability is found in the high deviation shown between LSA and human judges in *Summary 4*. Here, comprehension grading is 4 and 5 points lower for lemmatized and non-lemmatized LSA modes respectively. Moreover, a very significant point to highlight is that it is precisely in highly rated summaries where graders show less deviation. For *Summary 4* graders produced an inter-rater deviation of 0.53.

3.3 Discussion

Lemmatization dramatically reduced LSA term quantity. Then, the non-lemmatized Basque LSA semantic space has 56% more significant terms than its parallel non-lemmatized LSA matrix.

But the obtained coherence and comprehension grades do not significantly differ from each other. Thus, it is important to point out how such a dramatic difference in word number has so little impact in the final summary grade. Then, if the results are similar with a smaller number of words and a considerably smaller size, the lemmatized corpus would be computationally more efficient.

Another significant finding is the low level of agreement between LSA comprehension and coherence grades using both methods and human evaluation decisions in *Summary 4*. Moreover, it is precisely this summary the one in which human graders show a higher level of agreement. LSA results for summary 4 are not comparable at all to human grades and do not show the quality of the analyzed summary. Thus far, we have not found the reason behind these results but a deeper analysis of this automatic grading process need to be done to refine accuracy in high grades.

4 Conclusions

In the context of the automatic evaluation of summaries, this work has been carried out to observe LSA measures of two variables that are required to compute overall summary grades: coherence and comprehension. More specifically, this work has been done to observe the effect that the use of lemmatization has on the Basque agglutinative language. The fact of being agglutinative means that each word carries not only semantic but syntactically relevant information in most of each distinct occurrence. Then, agglutination in languages implies adding syntactic information to conceptual information in word level that non-agglutinative languages do not carry. Therefore, the impact of lemmatization in these languages tends to be greater than in the case of the non-agglutinative ones. Hence, this research observed how coherence and comprehension grading of summaries are affected by the use of lemmatized or non-lemmatized corpora as a source text.

We assumed that LSA would require more data to acquire same level of comprehension than in non-agglutinative languages. But, it turns out that after lemmatization results did not differ significantly. Therefore, we can save computational power using lemmatized corpora in an agglutinative language without having significant loss in grading accuracy. The reason is that even if the lemmatized version uses only one third of LSA terms it still performs almost as well as the non-lemmatized version. Therefore, here LSA shows that does not require syntactic information to resemble human grades in summary evaluation.

Then, LSA can be used to grade comprehension and coherence in Basque summaries using either lemmatized or non-lemmatized corpora. Results resemble human grades in most of the poor summaries with high precision in relation to our human judges. Our concern now, is to know how we could find a greater level of accuracy with those summaries with a high grades for human judges.

LSA is widely used in ITS natural language understanding on a variety of systems. So far, it has been mainly tested with non-agglutinative languages. Therefore, the result of this paper can be relevant for other systems with a similar goal or linguistic context. Moreover, dispersion between human graders and LSA automatic scores leaves an open question in the level of accuracy that LSA is able to gain in proficient summaries.

Future work will be directed to further analyse the automatic grading of other significant variables in the evaluation of summaries.

Acknowledgments

Work funded by the University of the Basque Country (UPV00141.226-T-15948/2004) and the Gipuzkoa Council in a European Union program.

References

1. Kintsch, W., Patel, V.L. and Ericsson, K.A., *The role of long-term working memory in text comprehension.* Psychologia, 1999. **42**: p. 186-198.
2. Barlett, F.C., *Remembering; a Studty in Experimental and Social Psychology*. 1932: Cambridge University Press.
3. Garner, R., *Efficient Text Summarization. Costs and Benefits. Journal of Educational Research*, 1982. **75**(5): p. 275–279.
4. Zipitria, I., Elorriaga, J.A., Arruarte, A. and Diaz de Ilarraza, A. *From Human to Automatic Summary Evaluation*. in *Proceedings of Intelligent Tutoring Systems 2004*. 2004. Maceio, Alagoas, Brasil: Springer.
5. Landauer, T.K. and Dumais, S.T., *A solution to Plato's problem: The Latent Semantic Analysis theory of acquisition, induction, and representation of knowledge.* Psychological Review, 1997. **104**: p. 211--240.
6. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R., *Indexing by Latent Semantic Analysis.* Journal of the American Society of Information Science, 1990.
7. Landauer, T.K., Foltz, P. and D., L., *Introduction to Latent Semantic Analysis.* Discourse Processes, 1998. **25**: p. 259--284.
8. Foltz, P.W., Kintsch, W. and Landauer, T.K., *The Measurement of Textual Coherence with Latent Semantic Analysis.* Discourse Processes, 1998. **25**: p. 285-307.

9. Wolfe, M.B.W., Schreiner, M.E., Rehder, B., Laham, D., Foltz, P.W., Kintsch, W. and Landauer, T.K., *Learning from text:Matching readers and texts by Latent Semantic Analysis.* Discourse Processes, 1998. **25**(309-336).
10. Graesser, A.C., Person, N.K. and Harter, D., *Teaching tactics and dialog in Autotutor.* International Journal of Artificial Intelligence in Education, 2001. **12**: p. 257-279.
11. Wiemer-Hastings, P. and Graesser, A., *Select-a-Kibitzer: A computer tool that gives meaningful feedback on student compositions.* Interactive Learning Environments, 2000. **8**(2): p. 149–169.
12. Wade-Stein, D. and Kintsch, E., *Summary Street: Interactive Computer Support for Writing.* Cognition and Instruction, 2004. **22**(3): p. 333-362.
13. Miller, T., *Essay assessment with latent semantic analysis.* Journal of Educational Computing Research, 2003. **28**.
14. Ventura, M.J., Franchescetti, D.R., Pennumatsa, P., Graesser, A.C., Jackson, G.T., Hu, X., Cai, Z. and TRG. *Combining Computational Models of Short Essay Grading for Conceptual Physics Problems*. in *Proceedings of Intelligent Tutoring Systems*. 2004. Maceio, Brasil: Springer.
15. Tomasello, M., *Constructing a Language: A Usage-Based Theory of Language Acquisition*. 2003: Harvard University Press.
16. Palolahti, M., Leino, S., Jokela, M., Kopra, K. and Paavilainen, P., *Event-related potentials suggest early interaction between syntax and semantics during on-line sentence comprehension.* Neuroscience Letters, 2005. **384**(3): p. 222.
17. Hagoort, P., *Interplay between Syntax and Semantics during Sentence Comprehension: ERP Effects of Combining Syntactic and Semantic Violations.* Journal of Cognitive Neuroscience, 2003. **15**(6): p. 883--899.
18. Landauer, T.K., Laham, D., Rehder, B. and Schreiner, M.E. *How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans*. in *19th Annual Meeting of the Cognitive Science Society*. 1997: Erlbaum.
19. Wiemer-Hastings, P. and Zipitria, I. *Rules for Syntax, Vectors for Semantics*. in *In Proceedings of the 23rd Annual Conference of the Cognitive Science Society*. 2001. Edinburgh: Erlbaum, Mahwah, NJ.
20. Serafin, R. and Eugenio, B.D. *FLSA: Extending Latent Semantic Analysis with Features for Dialogue Act Classification*. in *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*. 2004. Barcelona, Spain.
21. Kanejiya, D., Kamar, A. and Prasad, S. *Automatic Evaluation of Students' Answers using Syntactically Enhanced LSA*. in *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing*. 2003.
22. Olde, B.A., Franceschetti, D.R., Karnavat, A., Graesser, A.C. and TRG. *The right stuff: Do you need to sanitize your corpus when using latent semantic analysis?* in *24rd Annual Conference of the Cognitive Science Society*. 2002: Erlbaum.
23. Landauer, T.K. and Littman, M.L. *A statistical method for language-independent representation of the topical content of text segments*. in *Proceedings of the Sixth Annual Conference of the UW Centre for the New Oxford English Dictionary and Text Research*. 1990.
24. Aduriz, I., Aranzabe, M., Arriola, J., Ilarraza, A.D.d., Gojenola, K., M.Oronoz and Uria, L. *A Cascaded Syntactic Analyser for Basque*. in *Computational Linguistics and Intelligent Text Processing. 2945 LNCS Series*. 2004: Springer Verlag. Berlin.

Adaptable Scaffolding – A Fuzzy Approach

Selvarajah Mohanarajah, Ray Kemp, and Elizabeth Kemp

Institute of Information Sciences and Technology
Massey University
New Zealand
{S.Mohanarajah, R.Kemp, E.Kemp}@massey.ac.nz

Abstract. Employing scaffolding is not new in education. In CBL systems, scaffolding has been used with different levels of adaptability. This paper describes a novel design for the Learner Model, which handles the effects of uncertainty formally in the Scaffolding process. We have used this design in our CBL system (LOZ) for learning Object-Z specification. Learners can easily modify the scaffolding process if they wish. They can inspect the underlying fuzzy model and its processes. We use the fuzzy logic theory for dynamic prediction. A time threshold is used to avoid unnecessary dynamic prediction. The proposed design is domain independent and may be used for a wide range of CBL systems with little modification. We conducted an evaluation based on pre-experimental design and the results were very encouraging.

1 Introduction

Scaffolding in learning can be linked to the seminal educational concept known as Zone of Proximal Development [2]. Later, Collins et al included this strategy in their cognitive apprentice model [3]. There are two main steps involved in creating scaffolding lessons in CBL systems; designing suitable scaffolds (lesson plans) and designing software to realize the scaffolding process [4]. The CBL system may just provide a suitable environment, or give various levels of adaptable support for the learners (that hides the scaffolding process). Providing motivational learning environments tailored to a variety of learners has been considered a challenging task in CBL research for a long time. The challenge is mainly due to the uncertainty associated with the inadequate sources of evidence about the learner.

A typical adaptive system for procedural type domains usually performs four important actions: diagnosis, feedback selection, prediction, and curriculum advancement. The diagnosis (and later prediction also) is usually based on incomplete and noisy data. Consequently, the PAS (Pedagogical Action Selection)[1] [5] is made using those unreliable diagnose and error prone predictions. If inappropriate feedback or an untimely sequence of lessons (due to poor scaffolding process) is given to the learners recurrently, they may become confused or irritated, and, ultimately, their

[1] The term Pedagogical Action Selection first coined by Mayo refers the acts of both selecting the remedy, usually feedback after a misconception is identified, and selecting the next pedagogical action (usually curriculum sequencing, involving one or more actions: revising, moving forward, testing etc).

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 604 – 614, 2006.

motivation could be affected. In this paper, we propose a unique formal approach to reduce the impact of uncertainty on the scaffolding process. We consider the strength of pedagogical action as a fuzzy variable and use a novel technique based on both fuzzy and Bayesian theories. The proposed design discussed in this paper is domain independent and can be used for PAS in a wide range of CBL systems with minimum modification. To the knowledge of the authors, Fuzzy Logic theory has not been used in its full strength for uncertainty handling in CBL systems. We use a simple, but formal technology to handle uncertainty in macro-level assessment to aid scaffolding process. Finally, we conducted an evaluation using final year computer science students and the results were very encouraging.

2 Related Research

Besides informal heuristic approaches, there have been several formal AI techniques proposed for treating uncertainty in the diagnosis and/or prediction phase. Jameson [6] discusses various student models that use numerical AI techniques such as Bayesian Belief Networks, Fuzzy Logic and the Dempster-Shafer Theory of Evidence (DST). The usage of a Dynamic Bayesian Network for student modeling is extensively discussed in [7]. In LOZ, the micro level diagnosis task is limited (as the free-form style answering is avoided at present) by employing Multiple Choice Questions (MCQ) instead. Therefore, LOZ needs to handle uncertainty mainly for PAS (and for prediction) only.

After diagnosing a misconception, CBL systems may use ad-hoc or heuristic methods for PAS. However, especially for curriculum sequencing, some CBL systems use formal AI techniques such as Bayesian Belief Networks [8] and Fuzzy Logic (Sherlock-II [9]). Recently, statistical decision theory was used for curriculum advancement in the following systems: DT Tutor [10] and CAPIT [5].

3 Step I: Designing Lesson for Scaffolding

The first step in designing instruction based on scaffolding is identifying the appropriate scaffolding steps and designing suitable lesson material that keeps seamless continuity between those steps. First we shall see how the domain model is organized to support scaffolding.

3.1 Domain Model in LOZ

The learning material associated with the domain knowledge in LOZ is organized into several concepts [11]. Each concept is associated with a series of mental states. The term ‘mental state’ is used a lot in psychology and defined as “a mental condition in which the qualities of a state are relatively constant even though the state itself may be dynamic”[12] We use the term ‘mental state’ differently in this study. We assume that a person is in a certain ‘mental state’ if they have already constructed sufficient knowledge (and/or skills) to perform a certain task, may be mental or physical.

In our research, each sub concept is associated with a series of mental states (a basic, final and some intermediate mental states). The learning outcome of a sub-concept is attaining considerable strength in the associated final mental state. The success of scaffolding in learning depends on the selection of right mixture of support and challenge. The mental states are to be carefully selected as each of them in the series represents a scaffolding stage. After presenting the relevant material for learning a concept, learners are guided through a learning path, gradually from the basic mental state to the highest mental state, using MCQs (Multiple Choice Questions) and relevant feed-back. Figure 1 illustrates a concept and the related mental states in the domain model of LOZ.

Initially, learners are given a UML specification and asked to produce (or select) a corresponding Object-Z specification. In this way, learners are initially freed from performing abstraction and allowed to concentrate on understanding the underlying mathematical concepts. Gradually, the UML help will be withdrawn. Eventually, the learner will be able to perform abstractions as well as be able to apply mathematical notations to represent those abstractions.

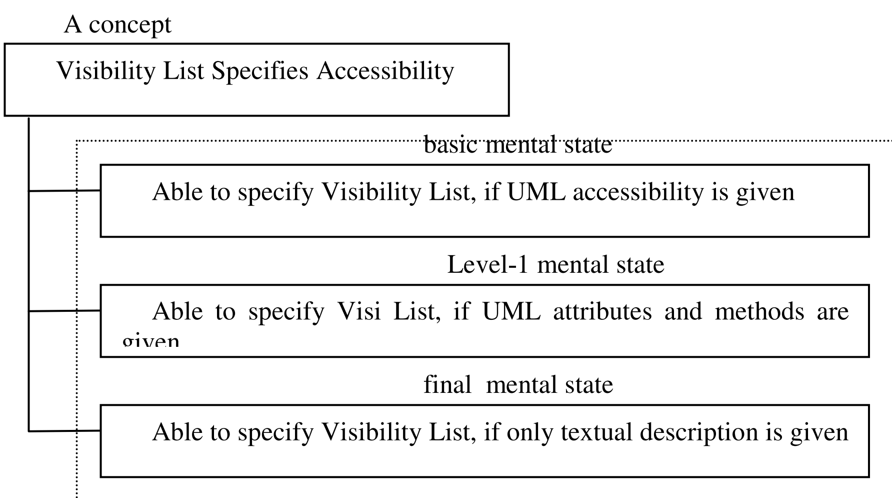


Fig. 1. Example for a Sub Concepts and related Mental States

3.2 Feedback and Removing Scaffolds

The potential of MCQ in educational testing is widely discussed in the measurement literature. Ben-Simon et al [13] state, *"Today MC tests are the most highly regarded and widely used type of objective test for the measurement of knowledge, ability, or achievement"*. MCQs plays important role in our system also. They are used here as scaffolding blocks rather than assessment units, and are carefully selected to match the relevant mental states. The inappropriate answers for each MCQ reflect some misconceptions associated with those mental states. The relevant feedback is designed to address those misconceptions. Each MCQ will also have one or more worked examples in the related topic.

After learning a concept, the learner will be given an MCQ at its basic mental state (scaffolding level 1). Learner may request worked examples on the current topic at this time. Depending on the answer, the system should provide suitable pedagogical actions tailored to the learners (feedback and next action- whether to move to the next scaffolding level or not). The factors influencing the effect of the feedback on the students were extensively analyzed by many researchers (e.g. [14]). Enthused by the Mason et al's research [15], we designed two disjoint sets of fuzzy variables for the strength of pedagogical actions (figure 2), one for the correct answer (PASc) and the other for the wrong answer (PASw). Both variables vary from one to five levels with increasing pedagogical strength. For example, the level-5 PAS for correct answer (PAScL5) indicates that the system assumes the given MCQ is very hard for the particular learner; and therefore, they need some informative feedback even if they answered it correctly (it may be a lucky guess). However, the learner will be given a problem at the same scaffolding level to prove that the previous success is not a lucky guess. Whereas, the level-5 PAS for wrong answer (PASwL5) indicates that, in addition to giving detailed feed back, the learner will be encouraged to return to the previous scaffolding level. Naturally, the boundaries of these PAS levels are not fixed, we let both measures PASw and PASc be fuzzy variables and let the relevant subsets be represented by the membership function given in figure 3a.

4 Step II: Designing Software to Implement Scaffolding

The CBL system may just provide a suitable environment for learning through scaffolding (learner has full control on selecting the level of scaffolding, questions, etc.), or give various levels of adaptable support. The learner model that realizes adaptability may be just simple and only groups the learners under different stereotypes. At the other extreme, the learner model may give micro level diagnoses during problem solving. Our learner model, which is positioned between these two approaches, provides macro level diagnoses [16] after an answer is given.

There is a basic question discussed in the CBS community for decades. Does a CBS systems need to be adaptable? If so, how far it should be adaptable? To get adaptability, it is obvious that we need to model learners. The question is, in other words, do we need an optimal Learner Model? We, like some other researchers [9], believe that the Learner Models need not be optimal. Learners are capable of adapting to a variety of teachers who make sub-optimal assumptions about students. Our student model in LOZ will not be based on rigorous mathematics to minutely differentiate learners' abilities, mental states, preferences etc. Rather it will, in general, avoid the kind of extreme system behavior (for example, system may not allow the learner to move to next scaffold level), which leads the learner to doubt the usefulness of the system for learning.

Level	**PASw - for Wrong answer**	**PASc- for Correct answer**
L1	Let them Answer-Once-Again (same MCQ) : If answered correct in second time, give another MCQ at the next Scaffolding Level (Move to next level)	Affirm (just inform that the answer is correct) Move to next level
L2	Explain Why the selected answer is Wrong : Let them Answer-Once-Again (same MCQ) : If answer correct second time give another MCQ, but at the same Scaffolding Level (Stay at the same level)	Explain why the selected answer is correct Move to next level
L3	Explain why the selected answer is Wrong & why the system's answer is correct : Give another MCQ, but at the same Scaffolding Level	Explain why the selected answer is correct & why other answers are wrong, Move to next level
L4	Explain why the selected answer is Wrong & why the system's answer is correct : Compare with Z and UML, if applicable. Give another MCQ, but at the same Scaffolding Level	Explain why the selected answer is correct & why other answers are wrong Provide topic related explanation Move to next level
L5	Explain why the selected answer is Wrong & why the system's answer is correct : Provide topic related explanation : Compare with Z and UML, if applicable: Give another MCQ, but at the next lowest Scaffolding level (Move to next-low level)	Explain why the selected answer is correct & why other answers are wrong, Provide topic related explanation Stay at the same level

Fig. 2. Different Levels of Pedagogical Actions

4.1 Adaptable Scaffolding in LOZ

During scaffolding, after an MCQ is answered, the system measures the learner's performance (P) frequently. For a normal MC test, P will be just a binary variable (correct/wrong). In the normal MC test, partial knowledge in a given test item [4] cannot be identified. As we use MCQ as scaffolding blocks, identifying partial knowledge and partial misconceptions are crucial for our system. If we allow the learner to indicate their degree of belief in each answer for their correctness (or wrongness), P may be fuzzy. Moreover, we will not consider the time taken to answer a question while measuring performance. But, in reality, a good student may answer a question quickly (and correctly).

In general, we may expect the learner with a strong mental state will answer the question correctly. But this is not certain since a lucky guess or careless mistakes may still play a role. The strategy used for PAS is critical as there is a potential for the system's trustworthiness to become questionable and eventually the user may abandon the learning process. To incorporate dynamic adaptability in scaffolding process, the Learner Model in LOZ keeps some measures including Strength on Mental States (SMS). SMS indicates a person's strength on a particular mental state (related to a scaffolding stage). Naturally, the performance (P) depends on SMS. SMS cannot take yes-or-no type of values. Therefore, we let SMS be fuzzy and take three values: Strong, Medium and Weak. The fuzzy membership function of SMS we use in this research is given in figure 3b (a score SMS=75 means, the system believes 75% that the learner is strong and 25% medium in a particular mental state). Assume that SMS will be 50 (medium-100%) for the basic mental states of each sub concept. Influenced by the locally intelligent learner model concept discussed in [16], we, at present, assume that the Mental States are mostly independent of each others (except those related to the same concept).

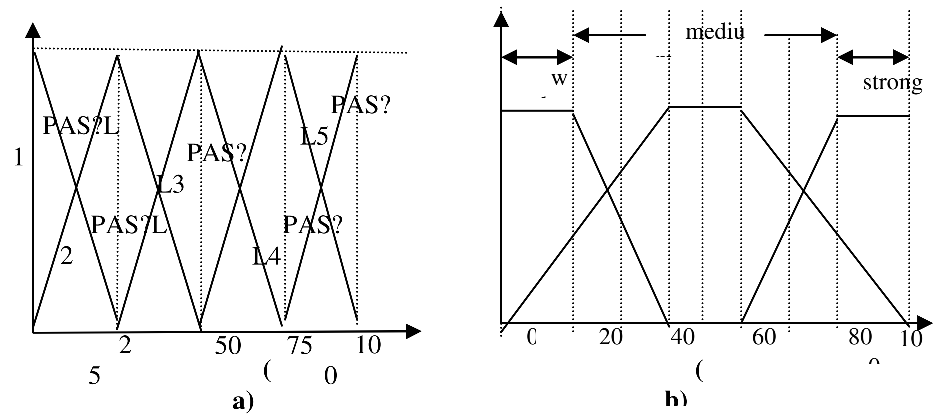
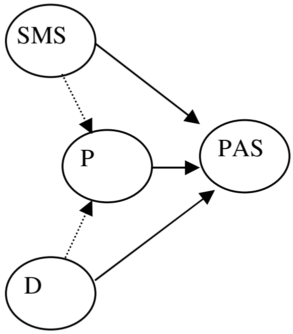


Fig. 3. Fuzzy Membership Functions for **(a)** PAS **(b)** SMS ,(D and L also)

The performance (P) of a learner on an MCQ depends not only on SMS but also on the difficulty level of MCQ. Therefore, we let another fuzzy variable D to represent the difficulty level of an MCQ, which takes three values: Hard, Moderate and Easy. Let D also assume same fuzzy membership function in figure 3a. The variable D for each MCQ will be initially proposed by domain experts, and naturally, will match closely with their scaffolding level. The required level of pedagogical action (PAS) for a learner depends on all the three variables SMS, D and P (figure 4a). Finally, system uses some fuzzy rules to provide suitable pedagogical actions (figure 4b).

4.2 Fuzzification and Defuzzification

Now we shall see how the fuzzification and defuzzification processes are performed. Assume P is binary and independent of SMS (in order to handle uncertainty – figure 4a). In the fuzzification process, the crisp inputs (Strength of a Mental State of a learner (SMS) and/or Difficulty level of a question (D)) are converted to the corresponding fuzzy values using the membership functions (figure 1(a)). Thereafter, appropriate rules (figure 4b) are applied to obtain fuzzy values for the output variable PAS.



(a)

Mental State (SMS)	Difficulty level (D)	PASw Answer Wrong	PASc Answer Right
Strong	Low	PASwL1	PAScL1
Strong	Moderate	PASwL2	PAScL2
Strong	High	PASwL3	PAScL3
Medium	Low	PASwL2	PAScL2
Medium	Moderate	PASwL3	PAScL3
Medium	High	PASwL4	PAScL4
Weak	Low	PASwL3	PAScL3
Weak	Moderate	PASwL4	PAScL4
Weak	High	PASwL5	PAScL5

(b)

Fig. 4. (a) Causal Relations (b) Fuzzy Rules for PAS

For example, let a learner be at Mental State (SMS=65) and fail a question with difficulty (D=35).

Therefore, according to the fuzzy membership function given in figure 1(a);

For Mental State (SMS); P(Strong)=0.25, P(Medium)=0.75;
And for Difficulty (D); P(Low)=0.25, P(Moderate)=0.75

According to the fuzzy rules given in figure 2(c) (apply Fuzzy AND rule, X AND Y = min(X, Y)).

SMS(Strong) & D(Low) → PAS(PASwL1) ;
P(PASwL1)= (0.25&&0.25)=0.25,
SMS(Strong) & D(Moderate) → PAS(PASwL2) ;
P(PASwL2)= (0.25&&0.75)=0.25,
SMS(Medium) & D(Low) → PAS(PASwL2) ;
P(PASwL2)= (0.75&&0.25)=0.25,
SMS(Medium) & D(Moderate)→ PAS(PASwL1) ;
P(PASwL3)= (0.75&&0.75)=0.75,

Finally, the defuzzification process determines the crisp value of PAS. Simply, in defuzzification, the PAS variable with the highest probability will be selected (here it is PASwL3). If two PAS levels have equal beliefs the PAS with the highest strength will be selected. Otherwise to find a crisp (precise) value for pedagogical action as a unique numeric representation, we could use Larsen's Product Operation Rule combined with mirror rule at extremes [1] for defuzzification process (figure 5). According to this rule, the crisp value of PASw will be given by the equation,

$$(0.75*50 +0.25*25 + 0.25*0) / ((0.75+0.25+0.25) = 35$$

Therefore, for (SMS=65), (D=35), and (P=wrong),
the resultant PAS value is (PASw = 35)

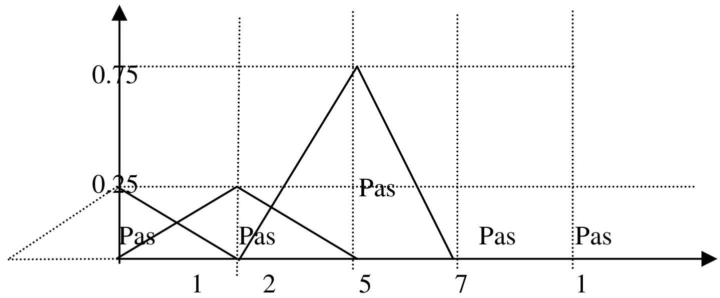
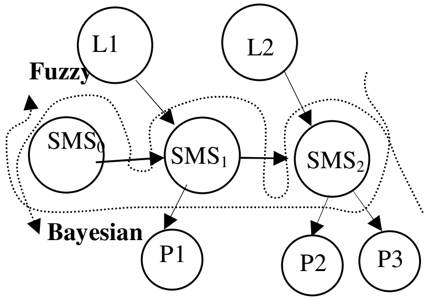


Fig. 5. Defuzzification, using Larsen's Product Rule (with mirror rule) [1]

4.3 Dynamic Prediction

Interestingly, the variable SMS is dynamic as its value changes over time and depends on its previous values. Learners may forget or learn something from other sources. To make things easy, we will assume only Markov's first order chain in this study [17] where the new value for SMS will depend only on its immediate preceded value. Nevertheless, the variable SMS will be predicted only with learner interactions rather in certain time intervals. The duration between interactions is not considered for predicting in this study though long duration may cause much forgetting or much learning from other resources. But, to measure the degree of learning from the system between two subsequent learner's interactions, we keep a fuzzy variable L explicitly. However, we employed a time threshold that restricts the unnecessary predictions when the time between interactions is negligible.

The learner interacts with the system mainly after learning a concept, after learning a feedback, or to answer an MCQ. Let the fuzzy variable L take three values: thoroughly, moderately and scarcely. Naturally, the variable L depends on the learner's general abilities and qualities. According to Vygotsky's ZPD [2], the measure SMSold (past knowledge level) cannot alone determine the new knowledge level (SMSnew). The learner's general abilities and qualities (to learn from the social interaction - here it is interaction with the system- though it is a weak comparison), and also how much time they spent actively on learning process also plays an important role. The general ability of a learner may be initialized by the level of knowledge in pre-requisites, and later, for each concept, could be updated against the final SMS value. Measuring the time a learner spent actively on an on-line task is extremely hard (in our prototype, we used minimum, maximum and average time limits for each lesson, and in extreme cases, learner's assistance is explicitly asked for). The time threshold is used here to avoid unnecessary prediction. For example, if two MCQ questions are attempted consecutively within the time threshold, SMS will be considered static and the old belief will be simply updated against the new evidence using Bayesian rules (see phase 1 in [7]). Otherwise a dynamic causal network will be considered for predicting (with fuzzy rules in figure 6b - see 'filtering' and 'prediction' in [17]). The static updating process in LOZ will not be discussed in this paper. Representing dynamic causality related to learner's mental state using a fuzzy model is more natural for human understanding.



(a)

SMS_i	Degree of Learning - L_i	SMS_{i+1}
Strong	Thoroughly	Strong
Strong	Moderately	Strong
Strong	Scarcely	Medium
Medium	Thoroughly	Strong
Medium	Moderately	Medium
Medium	Scarcely	Weak
Weak	Thoroughly	Medium
Weak	Moderately	Weak
Weak	Scarcely	Weak

(b)

Fig.6. (a) Dynamic Relations (b) Fuzzy Rules for the Dynamic Relation

5 Evaluation and Discussion

An evaluation was conducted based on pre-experimental design. There were some pre-requisites to use this system; and therefore, we required third or fourth year students to help us in this process (we managed to get ten final year students). Initially, a pre-test was given. The students were asked to learn 'Visibility List' using the prototype. Later, a post- test was given. The system also updates the log files with each interaction. A questionnaire was also administered.

The student-t test for paired samples gives very encouraging results. At 95% confidence level ($\alpha = 0.05$), the t-value for one tail at the degree of freedom 9 is 1.833 (since the alternate hypothesis is one directional, t critical one tail is used), and the estimated t-statistics is 8.51. That is, the test ascertains that learning Object-Z using LOZ is effective with 95% confidence. The power (99%) and the effect size (cohen's d= 2.03) are comparable to some other similar tests reported in the literature [18]. The accuracy of the student model was also found to be high. The Pearson product moment correlation coefficient between the post test scores and the learner model's estimate of the student is 0.827. The evaluation process and the results will be discussed later in detail.

6 Summary and Future work

To reduce the effects of uncertainty in PAS, we have designed a novel strategy using formal AI techniques. The strength of pedagogical action is considered as a fuzzy variable with suitable fuzzy membership functions and fuzzy rules. As the underlying fuzzy model and its processes are easy to explain, the effort related to opening the Learner model to the learners and mentors will be relatively simple [19]. The fuzzy logic (with a time threshold) could be effectively used in the dynamic prediction process. The proposed design is domain independent and may be used for any CBL system with few changes.

Domain model plays vital role in this design. The MC test should be enhanced to identify partial knowledge and partial misconceptions in a topic. The confident-based MC testing [20] may be a better choice. Appropriate strategies need to be accommodated to overcome the weak assumptions made in this study (e.g. time length does not affect learning). It is very difficult to measure the time spent by a learner actively during a lesson. Interacting with the learner to fix this problem also found to be cumbersome. As a remedy, the learner model may be opened to the learners [19].

Finally, we have implemented a functional prototype of our new learner model design (for our CBL system LOZ). The evaluation of this prototype revealed encouraging results. This work will be extended to a web-based environment in the near future.

References

1. Hopgood AA. Intelligent systems for engineers and scientists. Boca Raton, FL: CRC Press, 2000.
2. Vygotsky LS. Mind in Society (PP 84-91). Cambridge, MA: Harvard University Press, 1978.
3. Collins A, Brown J, S., Newman S, E. Cognitive Apprenticeship: Teaching the Crafts of Reading, Writing and Mathematics. In: Resnick L, B., editor. Knowing, Learning and Instruction: Essays on honour of Robert Glaser. Hillsdale, NJ: Lawrence Erlbaum, 1990. pp. 453-94.
4. Bradbard DA, Parker DF, Stone GL. An alternative Multiple-Choice Scoring Procedure in a Macroeconomics Course. Decision Sciences Journal of Innovative Education, USA. 2004;2 (1).
5. Mayo M, Mitrovic A. Optimising ITS behaviour with Bayesian networks and decision theory. International Journal of Artificial Intelligence in Education (IJAIED) 2001;12 (2):124-53.
6. Jameson A. Numerical Uncertainty Management in User and Student Modelling: An overview of Systems and Issues. User Modelling and User-Adapted interaction 1996;5.
7. Reye J. Student Modelling based on Belief Networks. International Journal of Artificial Intelligence in Education 2004;14 (63-96).
8. Mislevy RJ, Gitomer DH. The Role of Probability-Based Inference in an Intelligent Tutoring System. User Modelling and User-Adapted Interaction 1996;5:pp. 253-82.
9. Katz S, Lesgold A, Eggan G, Gordin M, Greenberg L. Self-Adjusting Curriculum Planning in Sherlock-II. Lecture Notes in Computer Science 1992;602:343-55.
10. Murray RC, VanLehn K, editors. DT Tutor: A decision-theoretic, dynamic approach for optimal selection of tutorial actions. ,. Montreal, Canada.: New York: Springer., 2000.
11. Mohanarajah S, Kemp R, H., Kemp E, H. Towards an Interactive Learning Environment for Object-Z. Journal of Issues in Information Sciences and Information Technology Education ISSN: 1539-3585 2004;Vol 1 (1):pgs 0993-1003.
12. Dictionary. Mental State (http://wordwebonline.com/en/MENTALSTATE). (referred on 17/05/05).
13. Ben-Simon A, Budescu DV, Nevo B. A comparartive study of measures of partial knowledge in multiple-choice tests. Applied Psychological Measurement 1997;21 (1):65-88.

14. Kulhavy RW, Stock A, W,, Hancock E, T, Swindell K, L, Hammrich P, L. Written Feedback: Response Certitude and Durability. Contemporary Educational Psychology 1990;15:319-32.
15. Mason JB, Bruning R. Providing Feedback in Computer-based Instruction: What the research tells Us. Retrieved February 17, 2004 from http://dwb.unl.edu/Edit/MB/MasonBruning.html . 1999.
16. McArthur D, Lewis M, Bishay M. The roles of AI in education: Current progress and future prospects. RAND cooperation, Santa Monica, CANADA, 1999.
17. Russell SJ, Norvig P. Artificial intelligence : a modern approach. Upper Saddle River, N.J.: Prentice Hall/Pearson Education, 2003.
18. Chin N, D. Empirical Evaluation of User Models and User-Adapted Systems. User Modeling and User-Adapted Interaction 2001;11:181-94.
19. Mohanarajah S, Kemp R, H., Kemp E, H. Unfold the scaffold & Externalizing a Fuzzy Learner Model. ED-MEDIA05 (accepted for publication). Montreal, Canada, 2005.
20. Gardner- Medwin AR. Enhancing Learning and Assessment Through Confidence-Based Learning (accepted). Enhancing Teaching and Learning through Assessment. Hong Kong, 2005. pp. 80-5.

P.A.C.T. – Scaffolding Best Practice in Home Tutoring

Orla Lahart[1], Declan Kelly[1], and Brendan Tangney[2]

[1] National College of Ireland, Dublin, Ireland
[olahart, dkelly]@ncirl.ie
[2] University of Dublin, Trinity College, Ireland
tangney@tcd.ie

Abstract: Research indicates a high correlation between parental involvement and a child's learning. The most effective parental involvement is when parents engage in learning activities with their child at home. However, parental involvement around learning activities may not occur spontaneously due to lack of domain knowledge, teaching strategies or structured support. This paper discusses how these issues can be addressed through the Parent and Child Tutor (P.A.C.T.). In particular, P.A.C.T will provide support for Suzuki parents during violin practice at home. This paper presents two studies; the first study identifies a set of best practice exemplars through lesson observations and interviews with the domain expert which informs the design of P.A.C.T. The second study validates the design of the system through analysing parent-child practice with and without the support of P.A.C.T. Results suggests that P.A.C.T. is effective in significantly increasing the use of best practice exemplars, in particular positive reinforcement and motivational games.

1 Introduction

Research suggests that there is a high correlation between parental involvement and a child's learning [1]. The most effective parental involvement is where parents engage in learning activities with their child in the home [2]. However, parental involvement around learning activities may not occur spontaneously due to lack of domain knowledge, teaching strategies or structured support. This paper discusses how these issues can be addressed through the Parent and Child Tutor (P.A.C.T.). The objective of P.A.C.T. is to provide a personalised adaptive scaffolding system which encourages best practice during home tutoring. In particular, P.A.C.T will provide support for Suzuki parents during violin practice at home. The Suzuki method was chosen as the pedagogical framework that informs the system as it is based on the premise that little can be achieved without the parent's active involvement [3]. The design of such a system involves three phases. First, it is necessary to identify a set of best practice exemplars, as these will inform the system's design. The second phase involves the design of the non-adaptive computer based support. The third phase is concerned with providing intelligent support, which can be achieved through personalised adaptive scaffolding. This paper presents two studies which address the first two phases. In the first study we identify a set of best practice exemplars through lesson observations and interviews with the domain expert. The design of P.A.C.T. is

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 615 – 624, 2006.

based on the findings from the first study. The second study focuses on the validation of this design through analysing parent-child practice with and without the support of P.A.C.T. Results suggest that P.A.C.T. is effective in significantly increasing the use of best practice exemplars, in particular positive reinforcement and motivational games. We conclude by discussing how this research can inform the future development of a personalised adaptive scaffolding system which can further promote best practice during home tutoring.

2 Background

Current research literature informs us that there is a high correlation between parental involvement and a child's academic success [1].The term parental involvement is used to describe an array of parent behaviours from attending school functions to actively tutoring their child at home [4]. There are strong indications which suggest that the most effective forms of parent involvement are those which engage parents in working directly with their children on learning activities in the home [2]. Vygotsky's research mirrors this argument when outlining that learning is a collaborative process and a child's potential development can be achieved through collaboration with a more able peer or guidance by a teacher [5]. However, further research s needed to formalise the parent's role in their child's learning.

The Suzuki Method is one such teaching strategy that places a huge emphasis on the parent's role in the learning process of the child and their importance as home tutor. The Suzuki Method is based on the premise that talent is a product of environment rather than heredity [3]. Dr. Shinichi Suzuki (1898-1998) believed that every child has talent, which can be developed if the proper environment exists. All children learn to speak their own language with relative ease and if the same natural learning process is applied in teaching other skills, these can be acquired as successfully. In the Suzuki Method, the learning process is broken down into the smallest possible steps which allow the child to achieve success at every level and learn at their own pace. The actual process of teaching the young child to play the instrument involves a trio of players: the child, the teacher and the Suzuki parent. The Suzuki parent provides daily home tutoring for the child. People have described this three-way relationship as a tripod: if one leg is missing nothing can be achieved. The Suzuki parent works at home with the student and tries to mimic the lesson environment. However, due to lack of tutoring experience Suzuki parent's can find it difficult to motivate their child to practice. Also, through lack of domain knowledge the Suzuki parent can facilitate bad habits becoming ingrained in the child's playing [6]. One solution may be to provide scaffolding to assist the parent during home tutoring.

Scaffolding facilitates the learning process by providing support for the learner. Often scaffolding is faded throughout the learning process so that the student becomes self-reliant [7]. Vygotsky's socio-culture approach and the notion of the Zone of Proximal Development (ZPD) are at the heart of the concept of scaffolding [5]. Scaffolding has been described as a direct application of Vygotsky's concept of teaching in the ZPD [8]. The ZPD is described as the distance between actual development during independent problem solving and the learner's potential development if assisted by a more able peer or adult [5]. Much of Vygotsky's work highlights the benefits for the learner of working in collaboration with a more able peer, adult or domain expert. The domain expert can be represented using intelligent software support, for example, personalised adaptive scaffolding. A number of systems exist which illustrate how technology can be used to scaffold the learning process [9, 10, 11].

Additionally, parents can provide scaffolding for their young learner's development. Technology has afforded parents a greater opportunity to get involved in their child's learning. This involvement can have a number of guises from increasing the possibility of parent, child and teacher discussions [12, 13] to providing extra resources for the child at home [14] to collaborative learning between the parent and child [15]. However, little work exists in providing support for the parent during home tutoring. Research to date has focused on providing support for the child in achieving their learning outcomes [16].

In summary, research suggests the need for parental involvement to ensure a child's academic success. In particular, the Suzuki method is one teaching philosophy that emphasizes the need for the parent. However, parents can find home tutoring difficult due to lack of pedagogical knowledge. One method of supporting parents is through the provision of scaffolding. However, despite research indicating that scaffolding can have a positive influence on the learning process little research exists on its benefits when supporting the parent during home tutoring. Consequently, this research investigates through P.A.C.T. how parents, and in turn the learning process, can benefit from scaffolding during home tutoring.

3 Study 1 – Identifying Best Practice Exemplars

P.A.C.T. will provide personalised adaptive scaffolding, which supports Suzuki parents during violin practice at home. Study 1 is concerned with the identification of best practice exemplars. The research aim of this study is to identify best practice exemplars in terms of the Suzuki method. This was necessary, as these exemplars would form the basis of the computer-based support. A number of qualitative research studies, namely, interviews with the domain expert (a qualified Suzuki teacher) and Suzuki music lesson observations were carried out.

The purpose of the interview with the domain expert was to gain an understanding of best practice and identify best practice exemplars. The interview consisted of some low-level questions such as the structure of a music lesson and home practice and more high-level open-ended questions. Consequently, it was necessary to identify how the aforementioned exemplars might be incorporated into home tutoring. This was the context for observing six music lessons of beginning Suzuki students.

3.1 Study1 – Analysis and Findings

From a qualitative analysis of the interview and lesson observations a set of best practice exemplars and their practical application was identified. Table 1 illustrates these best practice exemplars.

Table 1. Suzuki best practice exemplars

Exemplar	Description
Listening	Daily listening to music/language is necessary to develop new music/language skills.
Mastery Learning	Emphasises the need to focus on one step at a time, as it is easier for a child to master new skills when each step is mastered before the next is encountered.
Motivational Games	Parents are encouraged to use creative and imaginative strategies to increase motivation. This can be achieved through the use of playful games.
Positive Reinforcement	Parents should always encourage their child's efforts through positive affirmations.
Tutoring Variations	Involves the presentation of content using different tutoring styles to help the learner master difficult concepts.
Repetition	Emphasises the need to repeat new skills many times until they are mastered.
Review	Continuous review of previously acquired knowledge is necessary for mastery and can be used as a building block in learning new skills.

Listening is incorporated into each lesson through the child listening to the teacher playing Suzuki pieces. Suzuki tells us that the young child's rate of progress is directly dependent upon the amount of listening done [3]. Mastery learning involves new skills being taught in a step-by-step approach where each skill is mastered in turn. For example, if a child is learning a new rhythm they would practice on the E string so the left hand is not required, allowing them to focus on the rhythm. Lessons begin with the so-called Numbers game, a motivational game to help children place the violin. Motivational games are also used throughout the entire practice to help children with particular aspects of the playing. Positive Reinforcement is used to recognise the child's efforts and to encourage and motivate the child to do better. Phrases such as "Well done" or "That was very good, now, can you do better?" are used by the teacher. During lessons the teacher is very aware of the child's concentration level. If the child becomes distracted or is finding a new skill too difficult to master the teacher varies the teaching method. For example, if the child was learning a new rhythm but finding it difficult to play the teacher may ask the child to clap it. During Suzuki lessons students repeat new skills many times. Repetition is structured, phrases such as "Repeat that three times" are used. The initial part of each lesson is a review of previously learned pieces from the Suzuki repertoire. The teacher continuously stresses the importance of review. All lessons incorporate the same teaching strategies, namely listening, mastery learning, motivational games, positive reinforcement, tutoring variations, repetition and review, where appropriate, to benefit the child.

4 P.A.C.T. – The Design

The Parent and Child Tutor (P.A.C.T.) is web-based educational technology that encourages best practice by Suzuki parents when practicing at home with their child. P.A.C.T. is designed based on the set of best practice identified in study 1. P.A.C.T provides the parent with screen-based reminders of best practice and how this can be incorporated into home tutoring. The parent can first decide if it is best for the child to begin by listening to the Suzuki repertoire, playing some review pieces, practicing new skills or playing a game. It was observed from the lesson observations that practice could be subdivided into the following components: Listening, Review or Learning New Skills. Figure 1 illustrates how these options are presented to the user.

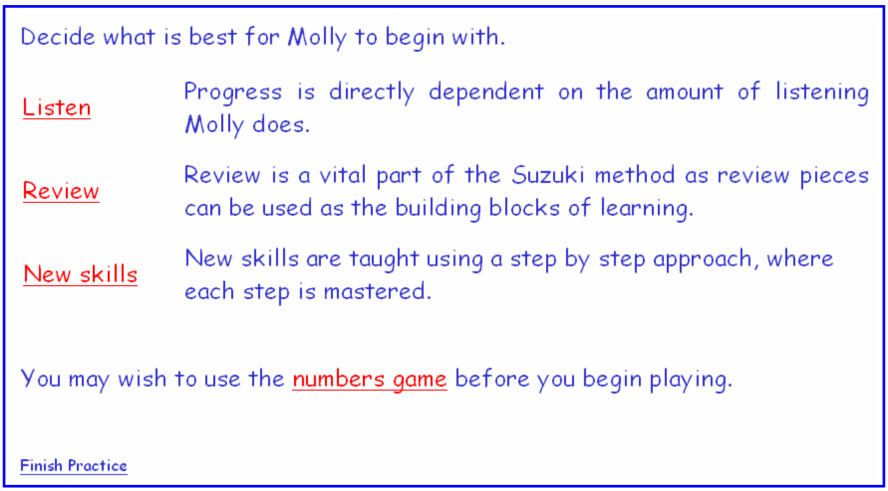


Fig. 1. Parent's Initial Choice

If the parent decided to focus on learning new skills they can focus on the melody, rhythm or the child's posture. It was important when designing P.A.C.T. to break each task into manageable steps, mastery learning being a best practice exemplar. Parent's are reminded of other best practice exemplars where appropriate based on findings from Study 1. For example, Figure 2 illustrates the teaching strategies, which could be incorporated when learning a new melody. An example of how the Listening can be applied during home tutoring is illustrated in Figures 3.

As parents may not posses may technical skills it was important to keep the design simple and straightforward. Also, it was important to remember that the parent's focus should be on their child therefore, it was important to ensure that instructions were clear and concise and did not over complicate the practice.

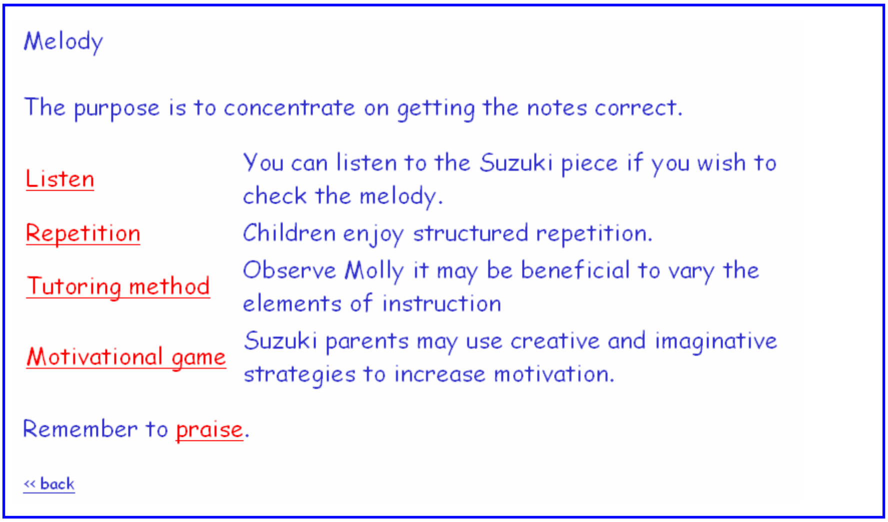


Fig. 2. Choice of Exemplars

Fig. 3. Listening Exemplar

5 Study 2 – Validation of P.A.C.T.

The purpose of Study 2 is to validate P.A.C.T. The study consists of parent-child practice observations and qualitative feedback from two domain experts. Two volunteer parent and child (aged 5 and 6 years) dyads participated in the research study. The sample was opportunistic and the only criterion was that they were beginning Suzuki students. The actions and the conversations of the participants were video recorded. The experiment design consisted of parents practicing with their child for ten minutes to give an indication of the type of tutoring the parent was providing. Following this parents were presented with an opportunity to familiarise themselves with P.A.C.T. Another ten-minute practice session followed where parents were assisted by P.A.C.T. After practicing with P.A.C.T. parents were asked a set of four questions including questions such as "What did you enjoy about using P.A.C.T?"

Qualitative feedback was also received from two domain experts (qualified Suzuki teachers). Feedback was received on the design of P.A.C.T. Also, the domain experts were given an opportunity to view the video recordings of the parent-child dyads interaction with P.A.C.T and give feedback. Feedback with the domain experts was recorded.

5.1 Study 2 – Analysis and Findings

The video data from the parent and child practice sessions was coded and analysed using the schema listed in Table 1. Two assessors independently rated the occurrences of best practice exemplars in the tutoring strategies used by the parents during practice sessions with and without P.A.C.T. The assessors were two music teachers, a Suzuki teacher and a conventional music teacher. The Pearson Correlation Coefficient[1] was used to gain a measure of the similarity between assessors' ratings, which, resulted in a correlation of 0.86, the correlation between assessors' ratings was calculated to demonstrate the similarity in their findings. The average of both assessors' ratings for each category was calculated. Table 2 lists the average occurrences of best practice exemplars with and without using P.A.C.T. It is surprising to note the dramatic increase in best practice exemplars when parents practiced with P.A.C.T. and in particular the increased use of positive reinforcement and motivational games.

When practicing without P.A.C.T. practice was unstructured and unfocused. With both dyads, the child was busy playing the Violin but there was little focus or goal to the practice. Dyad 1 spent a lot of time deciding what to practice but less time actually practicing. With Dyad 2 the practice was more fluid but the child did not learn a lot, as there wasn't an attempt to master any specific skill. On observing the practice the domain experts confirmed that this would typically be the type of practice that would take place in the home environment of both dyads. When practicing with P.A.C.T. the practice was much more structured, Dyad 1 spent less time deciding what to practice, on observing the practice the domain experts commented that this was helpful as during Dyad 1's home practice more time is spent deciding what to practice than practicing. With Dyad 2 the practice was much more structured and the parent focused on one aspect of the child's posture, a range of best practice exemplars were used to tutor the child in this specific task.

---

[1] It was not possible to use Cohen's Kappa test as assessors' were not asked to sort a number of items into a set of categories but to measure the number of occurrences of best practice exemplars.

Table 2. Average ratings of best practice exemplars

	Listening	Mastery Learning	Motivational Games	Positive Reinforcement	Tutoring Variation	Repetition	Review
Dyad 1	0	0	0	3	0	0	0.5
Dyad1 + P.A.C.T.	1	3.5	6.5	7	0	1.5	3.5
Dyad 2	0	0	0	2	0	0	2
Dyad2 + P.A.C.T.	2.5	4.5	4	9	0.5	4.5	3.5

As can be seen from Table 2 there was a dramatic increase in the use of motivational games with both dyads. It is important that parents use creative and imaginative strategies to increase motivation [6]. Both parents stated that they particularly liked the motivational games. One parent states "they gave me ideas, made it interesting for her (the child)" and "the child would definitely look forward to it if they knew they were coming up". Dyad 1 primarily used motivational games to increase motivation and to engage the child. However, once the child began to concentrate the parent focused on having a correct bow hold. Dyad 2 used the motivational games as another tutoring method to focus on the bent thumb.

There was also a dramatic increase in the use of positive reinforcement. "Very good, can you do better?" is the basic Suzuki formula [3]. When practicing without P.A.C.T. parents were quite sparing with praise. With P.A.C.T. there was quite an increase in the use of positive affirmations. This may be due to the screen prompts or because practice was more focused and goal-oriented.

The domain experts were particularly pleased with the inclusion of the listening exemplar. Their experience is that students do not listen enough to the recordings of the Suzuki repertoire, with parents usually stating explanations such as "Can not find the CD" or "The CD player isn't working". The domain experts believed that P.A.C.T. would promote listening during home practice. They were also particularly pleased with the inclusion of the repetition and review exemplars. It is their experience that parents do not encourage the child to review previously learned pieces during daily practice and this is reflective in the child's playing. Also, it is their experience that parents do not encourage repetition of tasks to develop new skills. If repetition is used it is often unfocused and unstructured. The results in Table 2 illustrate that there was an increase in the occurrences of review for both dyads when practicing with P.A.C.T. With Dyad 2 there was also an increase in repetition.

On overall impressions of P.A.C.T both dyads stated that they would prefer practicing with P.A.C.T. One parent followed this by stating "It took the blame off me". It is often difficult for parent to know how to practice with their child and they may find it difficult to be confident in their tutoring abilities [6]. The domain experts commented on this saying "If they don't know what to do the computer (P.A.C.T.) is there". The domain experts believed that P.A.C.T. would have a positive influence on home tutoring. They stated "I can see it being of huge benefit to teachers and parents", "I'm excited about it I think it will be very good" and "I think it would be very very useful". When asked did they think that the inclusion of P.A.C.T. would lengthen the practice it was stated "No, I think it would focus it (the practice) a lot better".

The main criticism of parents and domain experts was that they expected some content in P.A.C.T. which was aimed specifically at the child. For example, if P.A.C.T. contained more visual stimulus for the child in the form of graphics or animations that could be used a reward when a task was completed.

Domain experts also believed that P.A.C.T. could afford parents the opportunity to realise that the Suzuki method is not just about learning to play an instrument but the overall development of the child (e.g. building self-esteem and hand-eye co-ordination).

In summary, the research studies carried out indicate that through the provision of scaffolding through P.A.C.T., for parents during home tutoring, there is an increase in best practice.

6 Conclusions

This paper introduced the Parent and Child Tutor (P.A.C.T), a non-adaptive computer based support which facilitates best practice during home tutoring. In particular, P.A.C.T. supports the parent during Suzuki violin practice. The paper presented two studies. In the first study we identified a set of best practice exemplars through lessons observations and teacher interviews. The results from this study informed the design of PACT. The second study investigated the effectiveness of PACT in supporting best practices exemplars. Results suggest a surprisingly high increase in the use of best practice exemplars when the parent was supported by P.A.C.T. In particular, there was a dramatic increase in the occurrences of positive reinforcement and motivational games. Qualitative feedback from domain experts indicate the benefits for parents and teachers in using P.A.C.T. Future research will involve the development of P.A.C.T. to provide personalised adaptive scaffolding. P.A.C.T. will be adaptive in so far as it addresses the weakness in parent's tutoring style and in structuring home tutoring. In particular adaptive scaffolding may encourage greater use of best practice exemplars and structured tutoring. It is expected that personalised support will increase the effectiveness of home tutoring.

References

1. Greenwood, G. E., & Hickman, C. W. (1991). Research and practice in parent involvement: Implications for teacher education. The Elementary School Journal, 91, 278-288.
2. Dornbusch, S. M., & Ritter, P. L. (1988). Parents of high school students: A neglected resource. *Educational Horizons, 66*, 75-77.
3. Suzuki, S. (1986). Nurtured by Love: The Classic Approach to Talent Education 2nd edition. Suzuki Method International.
4. Hickman, C. W., Greenwood, G. E., & Miller, M. D. (1995). High school parent involvement: Relationships with achievement, grade level, SES, and gender. *Journal of Research and Development in Education, 28,* 125-134.
5. Vygotsky, L. (1978). Mind in society: the development of higher psychological processes. Cambridge MA: Harvard University Press. (Original material published in 1930, 1933 and 1935).
6. Starr, C., & Starr, W. (1983). To learn with love: a companion for Suzuki parents. Knoxville, TN: K. Ellis Press.
7. McLoughlin, C., Winnips, K., & Oliver, R. (2000). Supporting Constructivist Learning through Learner Support On-line. In Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2000 (pp. 674-680). Norfolk, VA: AACE.
8. Wells, G. (1999) Dialogic Inquiry: Towards a Sociocultural Practice and Theory of Education. New York: Cambridge University Press.
9. Koedinger, K. R., J. R. Anderson et al. (1997). Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education 8: 30-53.
10. Luckin, R. and Hammerton. L, (2002). Getting to know me: helping learners understand their own learning needs through metacognitive scaffolding. Intelligent Tutoring Systems. S.A.Cerri, G. Gouarderes and F. Paranguaca (Eds) Berline, Springer-Verlag: 759-771.
11. Jackson, S., J. Krajcik et al. (1998). The Design of Guided Learner-Adaptable Scaffolding in Interactice Learning Environments. Conference on Human Factors in Computing Systems, Los Angeles, California, United States, ACM Press/Addison-Wesley Publishing Co. New York, NY, USA.
12. National ParentNet Association Available from: http://www.parentnetassociation.org/content/resources/research.htm [Accessed 20 December, 2005]
13. IBM Research ThinkReseach: New educational software devised by IBM - Featured Concept Wired for Learning By Stephen S.Hall. Available from: http://domino.research.ibm.com/comm/wwwr_thinkresearch.nsf/pages/wired396.html [Accessed 20 December, 2005]
14. NumberWorks@Home. Available from: http://www.numberworks.com/nwathome/default.asp [Accessed 20 December, 2005]
15. O'Connor, J., KeraWalla, L., & Lickin, R. (2005). Using Discuission Prompts to Scaffold Parent-Child Collaboration Around a Computer-Based Activity. In Proceedings of AIED 2005, Amsterdam, Holland. p176 – 183.
16. The Parent Tutor. Available from: https://parenttutor.com/default.asp [Accessed 20 December, 2005]

Scaffolding Problem Solving with Annotated, Worked-Out Examples to Promote Deep Learning

Michael A. Ringenberg and Kurt VanLehn

University of Pittsburgh, Learning Research and Development Center
3939 O'Hara St., Pittsburgh PA 15260, USA
412-624-3353
mringenb@pitt.edu, vanlehn@cs.pitt.edu
http://www.pitt.edu/~mringenb/

Abstract. This study compares the relative utility of an intelligent tutoring system that uses procedure-based hints to a version that uses worked-out examples for learning college level physics. In order to test which strategy produced better gains in competence, two versions of Andes were used: one offered participants graded hints and the other offered annotated, worked-out examples in response to their help requests. We found that providing examples was at least as effective as the hint sequences and was more efficient in terms of the number of problems it took to obtain the same level of mastery.

1 Introduction

At the heart of most educational research is the search for ways to improve the instruction of novices. One strategy that has been found to be very effective is one-on-one human tutoring [1]. The economics of providing one-on-one tutoring has prompted the investigation of other techniques to boost learning. Another technique is to use intelligent tutoring systems to supplement classroom instruction and to substitute for individualized instruction. Another technique is to use embedded examples in instructional material [2], [3], [4], [5], [6]. As both paths have met with some success, it is worth comparing them and exploring ways to combine them.

Our study was done with modification of Andes, an intelligent tutoring system that aids the instruction of college-level introductory physics [7]. The main function of Andes is to present students with problems and to let the students solve them with the option of receiving adaptive scaffolding from the system. The two types of adaptive scaffolding in Andes are flag feedback and hints. Flag feedback marks the student's input as either correct or incorrect. When the student asks for help, Andes presents the student with a hint. The hint either points out what is wrong with the input or suggests a step to do next. The hint is based on the anticipated next step in solving the problem. It is designed to help the student identify and apply the missing relevant basic principles and definitions. In this

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 625–634, 2006.

way, Andes tries to link the current problem-solving step with facts the student has already been taught.

Each hint is staged in a graded fashion known as a hint sequence. The student is typically presented first with a vague suggestion to prompt self explanation of the next step or to identify and correct the current error. The student can then ask for the next level in the hint sequence if the student judges that the previous hint was insufficient. The hints become more concrete as the sequence is followed. The last level in a hint sequence typically supplies the entire anticipated next correct problem-solving step. This is referred to as the bottom-out hint. This graded structure of hints has been used in several intelligent tutoring systems (For more information on Andes, please see `http://www.andes.pitt.edu/`). Students can and do resort to "help abuse" when this form of adaptive scaffolding is offered [8]. Students can click through the hints rapidly in order to get to the bottom-out hint and will ignore the rest of the hint sequence. This strategy is a problem because it is associated with shallow learning [8].

Our basic hypothesis is that novice students will learn more effectively if we replace Andes' hint sequences with worked-out near-transfer examples. A worked-out example is a solved problem with all of the anticipated problem-solving steps explicitly stated. A near-transfer example has a deep structure similar to that of the current problem and uses the same basic principles. Several lines of evidence suggest that worked-out examples will be more effective for novices than hint sequences.

First, based on an observation from previous Andes studies, some students will find the solution to one problem through help abuse and then refer back to that solved problem when faced with a similar problem. In essence, they are using the first problem to create a worked-out example. This observation is consistent with studies showing that novices prefer to learn from examples as opposed to procedural instructions [9].

Second, we suspect that the hints provided by Andes can provide good targeted help to students who are already familiar with the subject material and have an adequate understanding of the underlying principles. However, for novices, the first hints in the graded hint sequence probably make little sense. Novices are not sufficiently familiar with the subject material for the hints to activate the reasoning needed to finish the anticipated next step, nor are they familiar enough with the problem solving structure to understand why the hint is relevant.

Third, worked-out examples have been shown to be effective instruction in some cases. In one study, worked-out examples were more effective than presenting procedural rules [3]. However, examples are more effective when they alternate with problem solving, presumably because studying large blocks of examples becomes boring [10]. By using a single example in place of a hint sequence for each problem, we can avoid the boredom of large example blocks.

On the other hand, worked-out examples are not always effective. Their usefulness requires that students self-explain the solution steps listed in the example.

A self-explanation for an example is a meaningful and correct explanation of a step in the student's own words [11]. Unfortunately, students do not tend to produce self-explanations spontaneously and many students produce ineffective self-explanations. Useful self-explanations can be categorized as either derivations or procedural explanations [12]. Derivations answer the question "Where did this step come from?" and procedural explanations answer the question "Why was this step done?"

When students do not engage in self-explanation, they do not tend to develop a deep understanding of the material. Novices tend to match surface features of a problem, like diagrams and problem statement wording, with those in a worked-out example. In contrast, experts use the principles and deep structure as criteria for matching a worked-out example to a problem [13]. The deep structure refers to a general plan or sequence of principle applications that can be followed in order to solve the problem. By providing worked-out examples with well-structured explicit steps to the solution and annotations of the relevant principles for each step, we are presenting students with examples of good self-explanations. This is expected to promote identification of the underlying problem structure and facilitate recognition of similar problem structure in different problems. Providing an annotated, worked-out example during problem solving enables a direct comparison and should encourage the student to focus on the common deep structure between the problem and the example. This will lead the students who are provided with these examples to perform better on tasks that test the deep structural understanding of the problems than those who are not provided with them.

2 Methodology

This was a hybrid study in that it was both naturalistic and experimental. The experiment was conducted during a second semester, college level physics course. As part of the graded homework for this course, students solved problems with Andes. Students who volunteered to participate in the experiment used a modified version of Andes to do this homework. The post-test for this study was administered either three or four days before the in-class exam, depending on students' regular lab sessions. The time frame of homework completion was at the students' discretion, and ranged from a few weeks before the relevant in-class exam to many weeks after. This unanticipated confound was resolved by the creation of a new category "No-Training" for participants who had not done any of their homework before this study's post-test.

The study had two experimental conditions: Examples and Hints. In the Examples condition, participants were presented with annotated, worked-out examples in response to any help request while using Andes. Each problem was mapped to a single example, but several problems were mapped to the same example if they shared the same deep structure. In the Hints condition, participants were given Andes' normal graded, step-dependent hints in response to help requests. The dependent variable for this experiment was performance on a problem matching task. Participants were asked to choose which of two problem statements would be solved most similarly to the given problem statement. This task is meant to evaluate deep learning by measuring participants' recognition of deep structure similarities.

The study participants were recruited from students already participating in the physics section of Pittsburgh Science of Learning Center LearnLab (`http://www.learnlab.org/`). The physics section was run as part of the General Physics I/II classes in the United States Naval Academy in Annapolis, Maryland. A total of forty-six volunteers were recruited from two sections of this course taught by the same professor. Participants were instructed to download the version of Andes which was modified for this study to use for the assigned homework problems on the topic of "Inductors." Because use of Andes was demonstrated in class and required for homework throughout the course, students in these sections were expected to be familiar with it. No restrictions were placed on use of the study-assigned Andes program, textbooks, professors, peers, or any other supplementary material. Due dates for Andes homework in the course were not rigidly enforced. Only the unmodified Andes version of the homework on Inductors was made available to the participants for in the Hints condition. Those in the Examples condition were assigned the same homework problems but instead were given access only to a modified version of Andes with the graded hints replaced with a worked-out example problem.

The worked-out examples were designed to be near-transfer problems where numeric values and some other surface features were changed. The solution to a homework problem requires solving for a variable in an equation while the worked-out example shows steps to solving a different variable in the same equation. For example, the equation for Ohm's law is $V = IR$ (voltage = current*resistance). If one homework problem gives values for V and R and asks the student to calculate I, and another gives values for V and I and asks for R, then the one worked-out example used for both of these questions would show steps for calculating V from given values for I and R. This relationship means that only five worked-out examples were needed for the ten homework problems. The problem solving steps in the examples were written and annotated with the principle used in each step, or with a list of the equations that were algebraically combined for a given step. The example was designed to show completed problem solving steps and solutions identical to those used in unmodified Andes problems. The principles in the annotations were linked to the appropriate Andes subject matter help pages so that the same body of problem-solving information was available to all participants.

The post-test was administered during the last lab session of the class prior to the in-class examination on this material. The test format was adapted from the similarity judgment task described by Dufresne, et. al [14]. It consisted of twenty multiple choice questions in random order, with randomly ordered answer choices, presented one at a time with thirty minutes given to complete the test. Each question contained three unsolved problems: a model problem and two comparison problems. Each of these problems consisted of a few sentences and a diagram. There were four possible types of relationship between the model problem and the two comparison problems:

I. Same surface features with different deep structure
II. Same surface features with the same deep structure
III. Different surface features with different deep structure
IV. Different surface features with the same deep structure

Only one of the comparison problems in each question had the same deep structure as the model problem (Type II and IV). The homework covered five different deep structure concepts. In the post-test, four questions were related to each deep structure concept, each with a different combination of surface feature relatedness. The theoretical strengths of this method of measuring competence include emphasis on deep structure and de-emphasis of algebraic skills [14]. The participants were given the following written instructions:

> "In the following evaluation, you will be presented with a series of problem statements. You do not have to solve the problems! Your task will be to read the first problem statement and then decided which of the following two problems would be solved most similarly to the first one."

In contrast to the format used by Dufresne, et. al [14] The model problems in this study were repeated as infrequently as possible (given the small number of variables in each equation). The present study also drew all correctly matched deep structure problems in the post-test from the assigned homework and worked-out examples.

The participants were assigned to the two experimental groups in a pairwise random fashion based on their cumulative Grade Point Average (GPA). This single criterion was used to balance the two groups in terms of previous performance without regard for other variables such as class section, gender, academic major, or age. Ten specific homework problems from the fourteen Inductance problems available in Andes were assigned to the class.

3 Results

Of the twenty-three participants assigned to each condition, only nine from the Examples condition and twelve from the Hints condition had asked for help from Andes to solve at least one of the homework problems before the study's post-test. Twenty other participants had not started working on the assignment and two of these did not complete the post-test either. Five participants had solved at least one homework problem in Andes without using the any of the available help. Only those participants who completed the post-test were included in the final analysis. Of those who did any of the homework, only those who asked for help at least once were exposed to the manipulated variable, and so the five participants who did not ask for help were excluded.

There were no significant differences in performance on circuit questions among the three conditions on the in-class examination administered before

Variable Averages	No-Training	Hints	Examples	p
In-Class Circuit Exam	187 ± 21	178 ± 34	201 ± 38	0.5684
Total Training Time (s)	-	7942 ± 3681	4189 ± 2407	0.0735
Time per Problem (s)	-	672 ± 238	508 ± 162	0.2540
# of Problems Solved	-	11 ± 1.53	8.1 ± 2.8	**0.0427**
Weighted Post-Test	3.56 ± 0.49	4.30 ± 0.37	4.88 ± 0.56	0.0010
Problem Efficiency	-	0.413 ± 0.076	0.711 ± 0.215	**0.0034**

Fig. 1. Results are reported as mean±95% confidence limits

this study. This suggests that even though the participants who did not do homework self-selected the No-Training condition, all three conditions ended up with equivalently competent students. (see Fig. 1: In-Class Circuit Exam; $F_{(2,36)} = 0.57$, $p = 0.5684$). There was a notable but not statistically significant difference in the total time participants chose to spend solving homework problems between the Examples and Hints groups, with the Hints group spending more time on problems (see Fig. 1: Total Training Time; $t_{17.6\ corrected} = 1.90$, $p = 0.0735$). In contrast, the average time spent per problem (see Fig 1: Time per Problem; $t_{19} = 1.18$, $p = 0.2540$) was more consistent between the two groups. There was a significant difference in the average number of problems attempted between the two groups, with the Hints groups working on more problems than the Examples group (see Fig. 1: # of Problems Solved; $t_{19} = 2.17$, $p = 0.0427$).

By construction, the post-test questions varied considerably in their difficulty; for instance, it should be easier to identify similar deep structure when the surface features are also similar, and harder to identify them when the surface features are different. To more accurately measure competence, a weighted score was used. The post-test questions were weighted according to their difficulty as determined by the performance of the No-Training participants on each of the questions. The weight given to each question was $1 - \frac{\#correct}{18}$, where $\#correct$ was the number of participants from the No-Training condition who answered the given question correctly The calculated weights on the problems were in agreement with *a priori* expected performance differences on the different problem types.

When an ANOVA model was fit using the weighted post-test scores (see Fig. 1: Weighted Post-Test), a statistically significant difference among the three groups was detected ($F_{(2,36)} = 8.49$, $p = 0.0010$). With the Tukey-Kramer adjustment for multiple comparisons, it was found that the participants in the Examples condition did significantly better on the post-test than those in the No-Training condition ($t = 3.98$, $p = 0.0009$). The Hints condition also did better than the No-Training condition ($t = 2.45$, $p = 0.0496$). However, it was not possible to distinguish a difference between the Hints condition and the Examples condition based solely on the weighted post-test score ($t = 1.61$, $p = 0.2525$). Other dependent variables, such as GPA and in-class examination scores, were not found to be significant factors in any ANCOVA models.

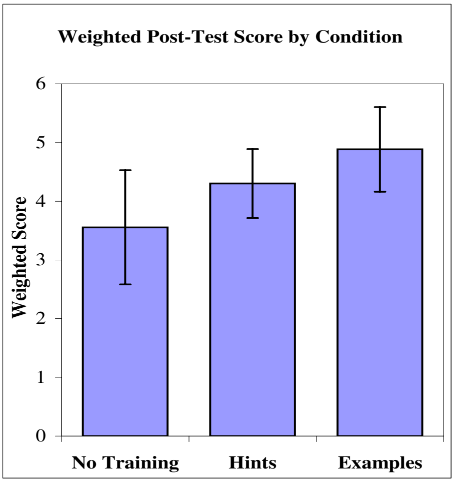


Fig. 2. Results are reported as means with error bars showing the ±95% confidence limits. Either form of training is better than none, but the difference in weighted post-test scores of the Hints and Examples conditions are not statistically significant.

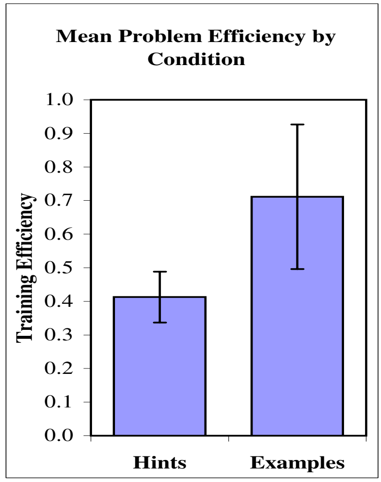


Fig. 3. Problem efficiency is defined as by condition where training efficiency is the weighted post-test score divided by the number of problems solved. Results are reported as means with error bars showing the ±95% confidence limits. Training problems with examples were more efficient at raising post-test scores than training problems with hints.

Problem efficiency was also calculated, that is, the increase in weighted post-test score per training problem done. The Examples and Hints conditions had weighted post-test scores that were not significantly different from each other but the participants in the Examples condition chose to do fewer training problems. The problem efficiency for the Examples condition was significantly higher than for the Hints condition (see Fig. 1:Problem Efficiency and 3; $t = 3.34$ $p = 0.0034$).

An ANCOVA model was fit using the weighted post-test scores with the number of problems attempted as the covariate and the Hints or Examples condition as the categorical variable (see Fig. 4. It was determined that the interaction effect between the condition and the number of problems was not significant ($p = 0.8290$). When the number of training problems was controlled by estimating the mean weighted post-test score at the overall mean number of training problems ($\mu_* = 9.76$), the difference in scores for the two training conditions condition was significant ($\mu_{Examples} = 4.88 \pm 0.46$; $\mu_{Hints} = 4.14 \pm 0.50$; $t = 2.30$, $p = 0.0338$). This was consistent with the Problem Efficiency results and demonstrates that given the same number of training problems, participants in the Examples condition performed better on the post-test than participants in the Hints condition.

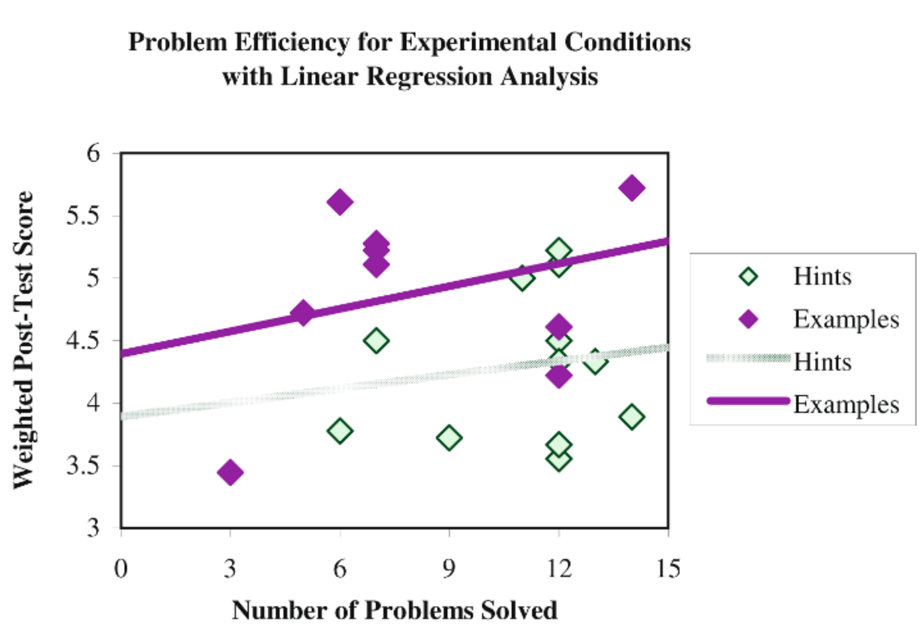


Fig. 4. Weighted post-test score versus number of problems solved with a fitted regression lines for the Hints and Examples conditions

4 Discussion

The results of this study demonstrate the value of working on Andes training problems to improve competence, whether with worked-out examples or graded hints. Although students in the No-Training condition were self-selected, they showed no significant difference in competence with basic circuit concepts prior to the study (as measured by scores on an in-class exam). One important difference between the two training conditions was the time on task, measured by the amount of time spent on the training homework. Participants in the Example condition chose to solve fewer training problems on average than the participants in the Hints condition. This was not due to the participants in the examples condition taking longer to solve problems, as the average time to solve each problem was not significantly different, but due to participants in the Hints condition choosing to spend more time working on more problems. Though participants in the Examples condition solved fewer problems on average than those in the Hints condition, they did at least as well on the post-test. This evidence supports the hypothesis that worked-out examples are a more efficient form of problem-solving help than graded hints. Due to the small number of participants involved in this study, aptitude treatment interactions could not be examined. A larger study might reveal an expertise reversal effect, where worked-out examples are more effective than graded hints for novices and less effective than graded hints for experts [15].

While previous studies have shown that providing worked-out examples can lead to shallow learning [13], this study indicates that worked-out examples may in fact be more efficient at promoting deep learning than graded hints. This has implications for tutoring system design in that examples may be a valuable addition to intelligent tutoring systems. Moreover, adding worked-out examples to an intelligent tutoring system should be fairly easy. The examples used in this study were easily added to Andes, mostly because there was no "intelligence" that needed to be designed to implement this strategy. One possible disadvantage of graded hint sequences is that they may be too rigid to accommodate the individual thought processes of different students. If the intelligent tutoring system that provides graded hints is good at assessing the participant's thought process, then the hints it can provide are likely to be effective. If the system can't identify and provide feedback relevant to a student's thought process, the hints will probably seem close to meaningless. If this happens too often, the student may decide that the hints are useless.

Worked-out examples can be a way of making sure that the system can provide a shared context for the student. They may be of particular value when the system has not collected enough data to evaluate the student effectively or if communication seems to have failed. One way to integrate worked-out examples into a graded hint system is to replace the bottom-out hint with the relevant worked-out example. This change may be particularly useful in addressing the help abuse [8]. It would be informative to see whether this strategy would reduce help abuse or provide incentive for the participants to click through a hint sequence rapidly just to see the worked-out example at the end. Worked-out examples could be integrated into more complicated intelligent tutoring systems that can assess the utility of different actions and provide these examples when appropriate [16]. Increasing the diversity of possible useful actions in such a system could only improve its performance.

References

1. Bloom, B.S.: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher **13** (1984) 4–16
2. Sweller, J., Cooper, G.A.: The use of worked examples as a substitute for problem solving in learning algebra. Cognition and Instruction **2** (1985) 59–89
3. Cooper, G., Sweller, J.: Effects of schema acquisition and rule automation on mathematical problem-solving transfer. Journal of Educational Psychology **79** (1987) 347–362
4. Brown, D.E.: Using examples and analogies to remediate misconceptions in physics: Factors influencing conceptual change. Journal of Research in Science Teaching **29** (1992) 17–34
5. Catrambone, R.: Aiding subgoal learning - effects on transfer. Journal of Educational Psychology **87** (1995) 5–17
6. Koehler, M.J.: Designing case-based hypermedia for developing understanding children's mathematical reasoning. Cognition and Instruction **20** (2002) 151–195

7. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: Andes physics tutoring system: Five years of evaluations. In Looi, G.M.C.K., ed.: Proceedings of the 12th International Conference on Artificial Intelligence in Education, Amsterdam, IOS Press (2005)
8. Aleven, V., Koedinger, K.R.: Investigations into help seeking and learning with a cognitive tutor. Papers of the AIED-2001 Workshop on Help Provision and Help Seeking in Interactive Learning Environments (2001) 47–58
9. LeFevre, J.A., Dixon, P.: Do written instructions need examples? Cognition and Instruction **3** (1986) 1–30
10. Trafton, J.G., Reiser, B.J.: The contribution of studying examples and solving problems to skill acquisition. In: Proceedings of the 15th Annual Conference of the Cognitive Science Society, Hillsdale: Lawrence Erlbaum Associates, Inc. (1993) 1017–1022
11. Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P., Glaser, R.: Self-explanations: How students study and use examples in learning to solve problems. Cognitive Science **13** (1989) 145–182
12. Chi, M.T.H., VanLehn, K.A.: The content of physics self-explanations. Journal of the Learning Sciences **1** (1991) 69–105
13. VanLehn, K., Johns, R.M.: Better learners use analogical problem solving sparingly. In Utgoff, P.E., ed.: Machine Learning: Proceedings of the Tenth Annual Conference, San Mateo, CA, Morgan Kaufmann Publishers (1993) 338–345
14. Dufresne, R.J., Gerace, W.J., Hardiman, P.T., Mestre, J.P.: Constraining novices to perform expertlike problem analyses: Effects on schema acquisition. Journal of the Learning Sciences **2** (1992) 307–331
15. Kalyuga, S., Ayres, P., Chandler, P., Sweller, J.: The expertise reversal effect. Educational Psychologist **38** (2003) 23–31
16. Murray, R.C., VanLehn, K.: Dt tutor: A decision-theoretic, dynamic approach for optimal selection of tutorial actions. In Gauthier, Frasson, VanLehn, eds.: Intelligent Tutoring Systems: 5th International Conference. Volume 1839 of Lecture Notes in Computer Science., Montreal, Canada, Berlin: Springer (2000) 153–162

A Appendix

– For examples of the training problems; the annotated, worked-out examples; and the post-test problems, visit `http://www.pitt.edu/~mringenb/AWOE/`.
– For more information about Andes, visit `http://www.andes.pitt.edu/`.
– For more information about LearnLab, visit `http://www.learnlab.org/`.

Scaffolding vs. Hints in the Assistment System

Leena Razzaq and Neil T. Heffernan

Computer Science Department
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01602
508 831-5006
{leenar, nth}@cs.wpi.edu

Abstract. Razzaq et al, 2005 reported that the Assistment system was causing students to learn at the computer but we were not sure if that was simply due to students getting practice or more due to the "intelligent tutoring" that we created and force students to do if they get an item wrong. Our survey indicated that some students found being forced to do scaffolding sometimes frustrating. We were not sure if all of the time we invested into these "fancy" scaffolding questions was worth it. We conducted a simple experiment to see if students learned on a set of 4 items, if they were given the scaffolds compared with just being given hints that tried to TELL them the same information that the scaffolding questions tried to ASK from them. Our results show that students that were given the scaffolds performed better although the results were not always statistically significant.

1 Introduction

Early evidence that the Assistment system was causing students to learn was reported by Razzaq et al, 2005 [9]. The Assistment system, a web-based system that aims to blend assisting students and assessing their knowledge, was causing students to learn 8th grade math at the computer, but we were uncertain if that was simply due to students getting more practice on math problems or more due to the "intelligent tutoring" that we created and force students to participate in if they got an item wrong. Our survey indicated that some students found being forced to do scaffolding sometimes frustrating. We were not sure if all of the time we invested into these "fancy" scaffolding questions was worth it. We conducted a simple experiment to see if students learned on a set of 4 items if they were forced to do the scaffolding questions, which would ASK them to complete each step required to solve a problem, compared with being given hints, which would TELL them the same information without expecting an answer to each step. In our study, our “scaffolding” condition represents a more interactive learning experience than the "hints" condition. Several studies in the literature have argued that more interactivity will lead to better learning.

Studies indicate that experienced human tutors provide the most effective form of instruction known [2]. They raise the mean performance about two standard deviations compared to students taught in classrooms. Intelligent tutoring systems can offer excellent instruction, but not as good as human tutors. The best ones raise performance about one standard deviation above classroom instruction [6].

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 635 – 644, 2006.

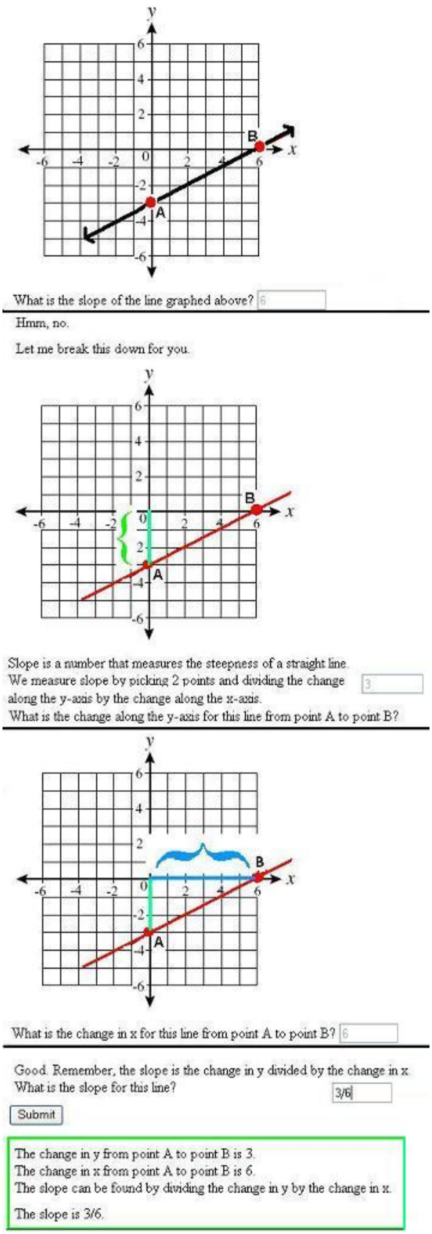


Fig. 1. An Assistment in progress

In studying what makes a tutoring session successful, VanLehn, Siler and Murray (1998) [11] identified principles for effective teaching. One important principle was that tutors should not offer strong hints or apply rules to problems themselves when students make mistakes. Students miss the opportunity to learn how to solve a problem when they are given an answer and are not allowed to reason for themselves.

Merrill, Reiser, Ranney and Trafton (1992) [7] compared the effectiveness of human tutors and intelligent tutoring systems. They concluded that a major reason that human tutors are more effective is that they let the students do most of the work in overcoming impasses, while at the same time provided as much assistance as necessary. [5] argues that the main thing human tutors do is to keep students on track and prevent them from following "garden paths" of reasoning that are unproductive and unlikely to lead to learning. [5] pointed to the large number of remarks made by tutors that helped keep students on track while learning Lisp programming. Modeling, coaching, and scaffolding are described by Collins, Brown and Hollum (1991) [3] as the heart of cognitive apprenticeship, which they claim "help students acquire an integrated set of skills through processes of observation and guided practice." An important part of scaffolding is fading, which entails progressively removing the support of scaffolding as the student demonstrates proficiency [3].

VanLehn et al (2005) [10] reviews several studies that hypothesize that the relationship between interactivity and learning exists, as well as a few studies that failed to find evidence for this relationship. [10] found that when students found text to be too difficult, tutoring was more effective than having the students read an explanation of how to solve a problem. We believe that our results show that this was true for one of the problems in our experiment which proved to be very difficult for the students.

This experiment would show that it MIGHT be beneficial to have this scaffolding, but the experiment would consciously confound on time as students being forced to do the scaffolding questions would take longer. If this experiment worked we would follow up with an experiment that controlled for time on task. Our results showed that students that were given the scaffolds performed better with an effect size of 0.3. Our survey results seem in line with this result in that students that said they tried to get through difficult problems as quickly as possible were negatively correlated with learning during the course of the year according to Feng et al (2005) [4]. We now plan a follow up study to see if it is worth the extra time.

In this paper, we will present a brief introduction of the Assistment system, how an experiment is executed and our experimental design followed by our results and discussion.

2 The Assistment System

Two years ago, Heffernan and his colleague Ken Koedinger received funding[1] to develop a web-based assessment system, designed to collect formative assessment data on student math skills. Since the assessment is delivered online, students can be tutored on items that they get incorrect. We are currently working with teams of paid and volunteer Worcester Polytechnic Institute (WPI) and Carnegie Mellon students and teacher volunteers to create the Assistment website, which is reported on in [9].

2.1 What Is an Assistment?

Once students log into the system they are presented with math items taken from one of the Massachusetts Comprehensive Assessment System (MCAS) tests for math given in previous years. The MCAS test is a state test given to all public school students in Massachusetts. Figure 1 shows a screenshot of an Assistment for an MCAS problem from 2003. If the student had answered correctly, she would have moved on to a new item. The screen shot shows that she incorrectly typed 6 and that the system responded with, "Hmm, no. Let me break this down for you" and followed that up with a question isolating the first step for finding slope, finding the rise. Once she answered that question correctly, she was asked a question focusing on the second step, finding the run. After successfully identifying rise and run, the student was asked to divide these two values and find the slope, repeating the *original question* (we use this term to distinguish it from the other questions we call *scaffolding questions* that help break the problem into pieces). We see that the student then asked for a hint and was told, "The change in y from point A to point B is 3. The change in

[1] This research was made possible by the US Dept of Education, Institute of Education Science, "Effective Mathematics Education Research" program grant #R305K03140, the Office of Naval Research grant # N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. Razzaq was funded by the National Science Foundation under Grant No. 0231773. All the opinions in this article are those of the authors, and not those of any of the funders.

x from point A to point B is 6. The slope can be found by dividing the change in y by the change in x." This student asked for a second hint and received "The slope is 3/6."

2.2 Reporting in the Assistment System

Teachers think highly of the Assistment system not only because their students can get instructional assistance in the form of scaffolding questions and hint messages while working on real MCAS items, but also because they can get online, live reports on students' progress while students are using the system in the classroom.

Student Name	Total time before (min)	Time spent today (min)	Original Items					Scaffolding + Orig. Items				Most Difficult MA. Stan
			# Done	# Correct	% Corr.	MCAS Score*	Perf. Level	# Done	# Correct	% Correct	# Hint Req.	
Tom	34	0	15	3	20%	200	Failing	30	16	53%	15	N.1.8-understanding-number representations (Error times:
Dick	32	0	38	26	68%	242	Proficient	81	56	69%	4	P.1.8-understanding-pattern times: 2/6)
Harry	33	0	20	9	45%	220	Needs improv	63	28	44%	63	P.1.8-understanding-pattern times: 8/10)

Fig. 2. The Grade book

The "Grade Book", shown in Figure 2, is the most frequently used report by teachers. Each row in the report represents information for one student, including how many minutes the student has worked on the Assistments, how many minutes he has worked on the Assistments today, how many problems he has done and his percent correct, our prediction of his MCAS score and his performance level.

2.3 Experiments in the Assistment System

The Assistment System allows randomized controlled experiments to be carried out [8] fairly easily. Problems are arranged in curriculums in the system. The curriculum can be conceptually subdivided into two main pieces: the *curriculum* itself, and *sections*. The *curriculum* is composed of one or more *sections*, with each *section* containing *problems* or other *sections*. This recursive structure allows for a rich hierarchy of different types of *sections* and *problems*.

The *section* component is an abstraction for a particular listing of problems. This abstraction has been extended to implement our current *section* types, and allows for future expansion of the *curriculum* unit. Currently existing *section* types include "Linear" (*problems* or sub-*sections* are presented in linear order), "Random" (*problems* or sub-*sections* are presented in a pseudo-random order), and "Experiment" (a single *problem* or sub-*section* is selected pseudo-randomly from a list, the others are ignored).

When an experiment has been carried out, the Experiment Analysis tool can be used to extract the data from the experiment. This tool, developed by Shane Gibbons and Emilia Holban at WPI, allows a researcher to enter a curriculum number, which is a unique identifier, and returns a list for every section in the curriculum. The list contains students who completed problems in the section and whether they got the item correct or incorrect and how much time they spent on each problem. The Experiment Analysis tool is also able to automatically compare performance on particular items or sections.

3 Experimental Design

An experiment carried out in 2004 tested to see whether scaffolding in the Assistment system get students to learn more than hints. In that experiment, 11 MCAS items on probability were presented to 8th grade students in Worcester, Massachusetts. We will refer to this as the Probability Experiment. Some students received the scaffold version of the item while others received the hint version. In the scaffold condition, the computer broke each item down into 2-4 steps (or scaffolds) if a student got the original item wrong. In the hints condition, if students made an error they simply got hints upon demand. The number of items was controlled for. When students completed all 11 items, they saw a few items that were very similar to test if they could do "close"-transfer problems.

The results of the statistical analysis showed a large gain for those students that got the scaffolding questions, but it was discovered that there was a selection-bias. There were about 20% less students in the scaffolding condition that finished the curriculum, and those students that finished were probably the better students, thus invalidating the results. This selection bias was possible due to a peculiarity of the system that presents a list of assignments to students. The students are asked to do the assignments in order, but many students choose not to, thus introducing this bias. This will be easy to correct by forcing students to finish a curriculum once they have started it. Another reason for this bias could be due to fact that students in the hint condition can finish problems faster than students in the scaffold condition. We tried to address both of these issues in the new experiment.

For the new experiment, we chose to focus on items that involved slope and intercept, which according to data from within the Assistment system, students found difficult. We will refer to this experiment as the Slope Experiment. Four MCAS items were chosen for the experiment and four more were chosen as transfer items to test whether the students had learned how to do slope problems. Two of the transfer items were also presented at the beginning of the experiment to serve as pre-test items. Students who got both pretest items correct did not participate in the experiment as they probably had already mastered the material. Students who got a pre-test item wrong were not told the answer or given any tutoring on the item. They were shown a message that told them that they would come back to this problem at the end of class.

To make sure that all of the students had the opportunity to complete the transfer items, we timed the students during the Slope Experiment. The students were given 20 minutes to work on a curriculum containing the two pretest items and four experiment items. They were then given 15 minutes to complete another curriculum containing the 4 transfer items. Unlike the Probability experiment, students had to complete the curriculums before proceeding to any other assignment. This procedure also ensured that students would work on the transfer items regardless of which condition they were in.

Figure 3 shows a slope item used in the experiment. The item on the left, in the scaffolding condition, shows that a student has answered incorrectly and is immediately presented with a scaffolding question. The item on the right, in the hints condition, shows that a student got the item wrong and received the buggy message, outlined in red, of "That is incorrect". The hint shown outlined in green appears when the student requests a hint by pressing the Hint button. We tried to make the hints in the hints condition similar to the scaffolding questions so that the scaffolding condition did not have an unfair advantage. However, the hints tended to be shorter than scaffolding questions. The difference is that the students in the scaffolding condition were forced to give answers to the individual steps in the problem. We hypothesize that if there is a difference between scaffolding and hints in this

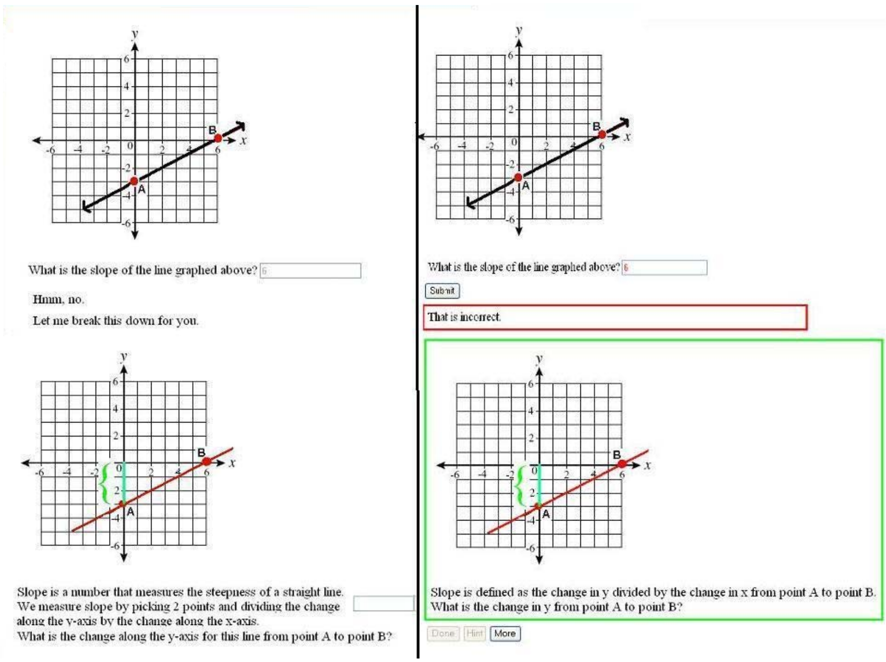


Fig. 3. A scaffold item in the experiment is shown on the left. A hint item is shown on the right.

experiment it will be due to forcing students to work actively to solve a problem, i.e. learning by doing, rather than allowing them to be passive.

174 students from 3 middle schools in Worcester participated in the Slope Experiment. 25 students were excluded for getting both pretest items correct, 11 in the scaffold condition and 14 in the hints condition. Another 5 students were excluded because they had not completed any transfer items, 2 in the scaffold condition and 3 in the hints condition. After these exclusions, there were 75 students in the scaffold condition and 69 students in the hints condition.

4 Results

We first ran an ANOVA to test whether the two conditions differed by pre-test. The result was not statistically significant so we were able to conclude that the groups were fairly balanced in incoming knowledge. We did know that of the two pretest items given, one of them was much harder than the other; 18% of the students got the first pretest item correct as opposed to 45% who got the second pretest item correct. The first pretest item concerned finding the y-intercept from an equation (What is the y-intercept in this equation: $y = 3/4x - 2$?). The second pretest item presented the student with 3 points and asked them to choose the graph that contained the points.

We report two different ways of analyzing our data, as we did not know ahead of time which method would be more likely to detect an effect. The first method, Analysis #1, takes into account 4 items on the posttest, while the second method, Analysis #2, only uses a single item, but has the advantage of being able to use performance on the pretest. Is it more important to have more items on your test, or is it more important to use information from the pretest? We did not know. In Analysis #1, we compared the two groups' average posttest/transfer scores but ignored pretest scores, while in Analysis #2 we looked at differing performance on the harder of the two pretest items that was repeated in the posttest/transfer section. In both analyses, we report the p-values and the effect sizes. We also report the confidence intervals on the effect sizes.

Inclusion criteria: not1aveonpre from onerowdatawith0insteadofnulls

	DF	Sum of Squares	Mean Square	F-Value	P-Value	Lambda	Power
condition	1	.213	.213	2.483	.1173	2.483	.329
Residual	146	12.517	.086				

Means Table for avetrans
Effect: condition
Inclusion criteria: not1aveonpre from onerowdatawith0insteadofnulls

	Count	Mean	Std. Dev.	Std. Err.
1	71	.359	.276	.033
2	77	.435	.307	.035

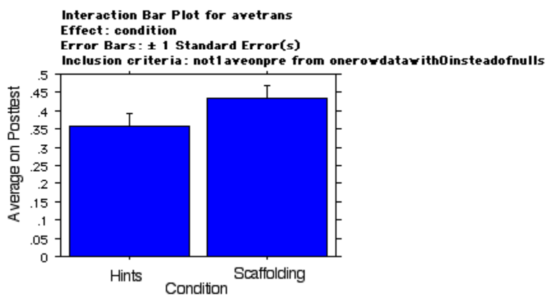


Fig. 4. Results for average on posttest items by condition

4.1 Analysis #1

For Analysis #1, we ran an ANOVA on the average scores on the transfer items by condition. We remind the reader that there were 4 posttest/transfer items so the scores were either 0%, 25%, 50%, 75% or 100%.

The result showed a p-value of 0.117 with an effect size of 0.3 (See Figure 4). We also calculated the 95% confidence interval for this effect size of .3 and got [-0.03, 0.6]. Because zero is included in this interval, we do not have 95% confidence that the effect size is real. We wanted to get a sense of the significance of this effect size so we calculated the 90% confidence interval and found the range to be [0.01, 0.56]. This implied that the effect size was great than .01 with 90% confidence. We interpret this as somewhat weak evidence in support of the hypothesis that students learned more in the scaffolding condition.

4.2 Analysis #2

We also looked at scores on the transfer items that students had seen as pretest items. For the first pre-test item, which concerned finding the y-intercept from an equation, the ANOVA showed a statistically significant p-value of 0.005 with an effect size of 0.85 (See Figure 5). The 95% confidence interval of the effect size of 0.85 is [0.5, 1.2], meaning that we are 95% confident that the effect size is somewhere between 0.5 and 1.2, implying that the effect size seems to be at least greater than 0.5, which is a very respectable effect size.

ANOVA Table for trans1com
Inclusion criteria: not1onpre1 from onerowdatawith0insteadofnulls

	DF	Sum of Squares	Mean Square	F-Value	P-Value	Lambda	Power
condition	1	.748	.748	8.212	.0048	8.212	.827
Residual	139	12.657	.091				

Means Table for trans1com
Effect: condition
Inclusion criteria: not1onpre1 from onerowdatawith0insteadofnulls

	Count	Mean	Std. Dev.	Std. Err.
1	67	.030	.171	.021
2	74	.176	.383	.045

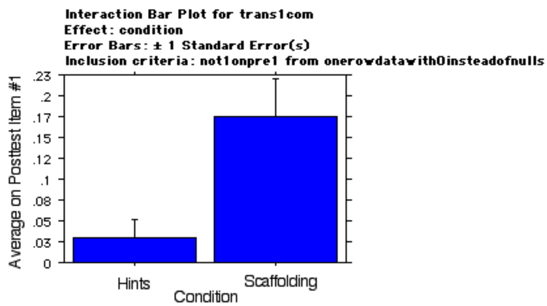


Fig. 5. Results on the transfer item for the first pre-test item by condition

For the second pre-test item, the scaffold condition did better on the transfer item than the hint condition, but the result was not statistically significant.

5 Discussion

In the previous section, we did two different analyses to look for effects of learning. Before we did the analyses we were not sure which was the better way of detecting differences. The results seem to show that there is more learning with scaffolding than with hints, although the difference was not always significant between the two conditions.

The first pretest item on finding the y-intercept from an equation proved to be a difficult problem for all of the students and scaffolding helped significantly. Perhaps the scaffolding had a greater positive effect on learning for the first pretest item because it was much more difficult for the students than the second pretest item. We cannot prove that yet, but would like to study the link between the difficulty of an item and the effectiveness of scaffolding.

In May, 2004, we gave students who were using the Assistment system a survey. 324 students participated in the survey where they were asked their opinions on the Assistment system and math in general. Students who said they tried to get through difficult problems as quickly as possible were negatively correlated with learning [4] during the course of the year. We believe that this falls in line with the results to the Slope Experiment in that students who were in the hint condition could finish items faster. Students who were in the scaffolding condition were forced to spend more time doing scaffolding and ended up learning more. Students who thought that breaking a question down into smaller steps did not help them understand how to solve similar problems was negatively correlated with MCAS scores. Over 60% of the students surveyed thought that the Assistment system helped them prepare for the MCAS. Students who liked using the Assistment system better than normal classroom activity were positively correlated to MCAS scores.

For future work, we plan a follow up study to see if scaffolding is worth the extra time where we will control for time.

Acknowledgements

This research was made possible by the US Dept of Education, Institute of Education Science, "Effective Mathematics Education Research" program grant #R305K03140, the Office of Naval Research grant #N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. All of the opinions in this article are those of the authors, and not those of any of the funders.

This work would not have been possible without the assistance of the 2004-2005 WPI/CMU ASSISTment Team including Brian Junker at CMU, Andrea Knight, Abraao Lourenco, Michael Macasek, Goss Nuzzo-Jones, Kai Rasmussen, Steven Ritter at Carnegie Learning, Carolyn Rose at CMU, Terrence Turner, Ruta Upalekar, and Jason Walonoski.

References

1. Baker, R.S., Corbett, A.T., Koedinger, K.R. Detecting Student Misuse of Intelligent Tutoring Systems. Proceedings of the 7th International Conference on Intelligent Tutoring Systems, Maceio, Brazil (2004)
2. Bloom, B. S. The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-one Tutoring. Educational Researcher, 13, (1984) 4-16.
3. Collins, A., Brown, J. S., Holum, A. Cognitive Apprenticeship: Making Thinking Visible, American Educator, 6(11), (1991) 38-46
4. Feng, M., Heffernan, N. T., Koedinger, K. Predicting State Test Scores Better with Intelligent Tutoring Systems: Developing Metrics to Measure Assistance Required. Submitted to the 8th International Conference on Intelligent Tutoring Systems (2005) http://nth.wpi.edu/pubs_and_grants/ITS2006/Submissons/Ming/feng.doc

5. Graesser, A. C., Person, N., Magliano, J. Collaborative Dialog Patterns in Naturalistic One-on-One Tutoring. Applied Cognitive Psychology, 9, (1995) 359-387
6. Koedinger, K. R., Anderson, J. R., Hadley, W. H. & Mark, M. A. Intelligent tutoring goes to school in the big city. In Proceedings of the 7th World Conference on Artificial Intelligence in Education, pp. 421-428. Charlottesville, VA: Association for the Advancement of Computing in Education. (1995)
7. Merrill, D., Reiser, B. Ranney, M., and Trafton, J. Effective Tutoring Techniques: A Comparison of Human Tutors and Intelligent Tutoring Systems. Journal of the Learning Sciences 2(3): 277-305. (1992)
8. Nuzzo-Jones, G., Walonoski, J.A., Heffernan, N.T., Livak, T. The eXtensible Tutor Architecture: A New Foundation for ITS. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, Amsterdam: ISO Press (2005) 902-904
9. Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R, Walonoski, J.A., Macasek. M.A., Rasmussen, K.P. The Assistment Project: Blending Assessment and Assisting. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, Amsterdam: ISO Press (2005) 555-562
10. VanLehn, K., Graesser, A. C., Jackson, G. T., Jordan, P., Olney, A., & Rose, C. P. When is reading just as effective as one-on-one interactive human tutoring? In Proceedings of the 27th Annual Meeting of the Cognitive Science Society (pp. 2259-2264). Mahwah, NJ: Erlbaum. (2005) 2259-2264
11. VanLehn, Kurt, Siler, Stephanie, Murray, Charles. What Makes a Tutorial Event Effective? Proceedings of the Twenty-first Annual Conference of the Cognitive Science Society (1998).

An Approach to Intelligent Training on a Robotic Simulator Using an Innovative Path-Planner

Roger Nkambou[1], Khaled Belghith[2], and Froduald Kabanza[2]

[1] Université du Québec à Montréal
Montréal, Québec H3C 3P8, Canada
nkambou.roger@uqam.ca
[2] Université de Sherbrooke,
Sherbrooke, Québec J1K 2R1, Canada
khaled.belghith@usherbrooke.ca, kabanza@usherbrooke.ca

Abstract. In this paper, we describe the open knowledge structure of *Roman Tutor*, a simulation-based intelligent tutoring system we are developing to teach astronauts how to manipulate the Space Station Remote Manipulator (SSRMS), known as "Canadarm II", on the International Space Station (ISS). We show that by representing the complex ISS-related knowledge in the form of a three-layered architecture with different levels of abstraction, and by using a new approach for robot path planning called FADPRM, it is no longer necessary to plan in advance what feedback to give to the learner or to explicitly create a complex task graph to support the tutoring process.

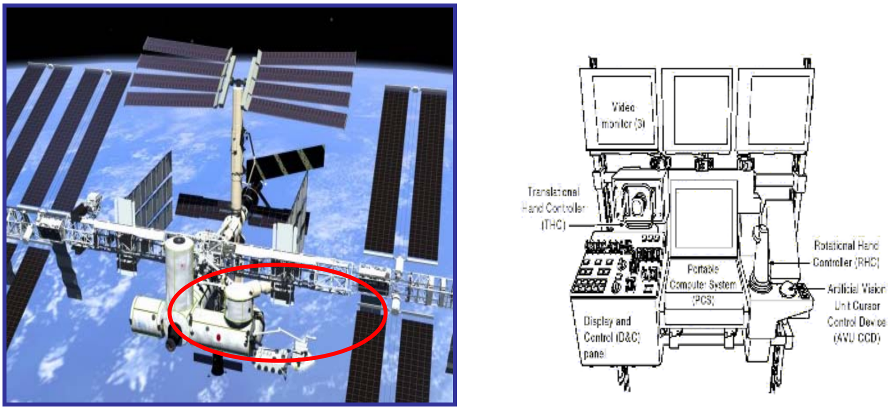
1 Introduction

This paper presents *Roman Tutor*, a simulation-based tutoring system to support astronauts in learning how to operate the Space Station Remote Manipulator (SSRMS), an articulated robot arm mounted on the international space station (ISS). Fig. 1-a illustrates a snapshot of the SSRMS on ISS. Astronauts operate the SSRMS through a workstation located inside one of the ISS compartments. As illustrated on Fig. 1-b, the workstation has an interface with three monitors, each connected to a camera placed at a strategic location of the ISS. There are a total of 14 cameras on the ISS, but only three of them are seen at a time through the workstation. A good choice of the camera on each of the three monitors is essential for a correct and safe operation of the robot.

SSRMS can be involved in various tasks on the ISS, ranging from moving a load from one place of the station to another to inspect the ISS structure (using a camera on the arm's end effector) and making repairs. These tasks must be carried out very carefully to avoid collision with the ISS structure and to maintain safety-operating constraints on SSRMS (such as avoiding collisions with itself and singularities). At different phases of a given manipulation such as moving a payload using the arm, the astronaut must choose a setting of cameras that provides him with the best visibility while keeping a good appreciation of his evolution in the task. Thus astronauts are trained not only to manipulate the arm per se, but also to recognize visual cues on the station that are crucial in mentally reconstructing the actual working environment

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 645 – 654, 2006.

from just three monitors each giving a partial and restricted view, and to remember and be able to select cameras depending on the task and other parameters.



(a) **(b)**

Fig. 1. ISS with SSRMS (a) and the workstation (b)

One challenge in developing a good training simulator is of course to build it so that one can reason on it. This is even more important when the simulator is built for training purpose [1]. Up until now, Simulation-Based Tutoring is possible only if there is an explicit problem space associated with tasks that are carried out during training to be able to track student actions and to generate relevant tutoring feedbacks [2,3]. Knowledge and model tracing are only possible in these conditions [4]. However, it is not always possible to explicitly develop a well-informed task structure in some complex domains, especially in domains where spatial knowledge is used, as there are many possibilities to solve a given problem. This paper proposes a solution to this issue through a system called *RomanTutor* which uses a path planner to support spatial reasoning on a simulator and make it possible model tracing tutoring without an explicit task structure. We developed a simulator of the ISS and SSRMS with quite realistic rendering and kinematics constraints as with the real environment. The Simulator knowledge structure will be described later. Fig. 1a also illustrates a snapshot of the simulator with SSRMS moving a payload from one place to another on the ISS. This simulates an operation that actually took place during the construction of the ISS.

As most complex tasks deal in one way or another with moving the SSRMS and for the simulator to be able to understand students' operations in order to provide feedback, it must itself be aware of the space constraints and be able to move the arm by itself. A path-planner that calculates arm's moves without collision and consistent with best available cameras views is the key training resource on which other resources and abstract tutoring processes hinge.

After a brief description of the path planner, we outline the different components of *Roman Tutor* and show how the path planner is used to provide amazingly relevant tutoring feedback to the learner.

2 The FADPRM Path-Planner

In the literature, several approaches dealing with the path-planning problem for robots in constrained environments were found [7-9]. Several implementations were carried out on the basis of these various approaches and much of them are relatively effective and precise. The fact is that none of these techniques deals with the problem of restricted sight we are dealing with in our case [10].

That's why we designed and implemented FADPRM [11] a new flexible and efficient approach for robot path planning in constrained environments. In more of the obstacles the robot must avoid, our approach holds account of desired and non-desired (or dangerous) zones. This will make it possible to take into account the disposition of cameras on the station. Thus, our planner will try to bring the robot in zones offering the best possible visibility of the progression while trying to avoid zones with reduced visibility.

FADPRM [11] allows us to put in the environment different zones with arbitrary geometrical forms. A degree of desirability *dd*, a real in [0 1] is assigned to each zone. The *dd* of a desired zone is then near 1, and the more it approaches 1, the more the zone is desired; the same for a non-desired zone where the *dd* is in [0 0.5]. On the international Space Station, the number, the form and the placement of zones reflect the disposition of cameras on the station. A zone covering the field of vision of a camera will be assigned a high *dd* (near 1) and will take a shape which resembles that of a cone; whereas a zone that is not visible by any camera from those present on the station will be considered as an non-desired zone with a *dd* near to 0 and will take an arbitrary polygonal shape.

The ISS environment is then preprocessed into a roadmap of collision-free robot motions in regions with highest desirability degree. More precisely, the roadmap is a graph such that every node n is labeled with its corresponding robot configuration $n.q$ and its degree of desirability $n.dd$, which is the average of *dd*s of zones overlapping with $n.q$. An edge (n,n') connecting two nodes is also assigned a *dd* equal to the average of *dd* of configurations in the path-segment $(n.q,n'.q)$. The *dd* of a path (i.e., a sequence of nodes) is an average of *dd* of its edges.

Following probabilistic roadmap methods (PRM) [12], we build the roadmap by picking robot configurations probabilistically, with a probability that is biased by the density of obstacles. A path is then a sequence of collision free edges in the roadmap, connecting the initial and goal configurations.

Following the Anytime Dynamic A* (AD*) approach [13], to get new paths when the conditions defining safe zones have dynamically changed, we can quickly re-plan by exploiting the previous roadmap. On the other hand, paths are computed through incremental improvements so that the planner can be called at anytime to provide a collision-free path and the more time it is given, the better the path optimizes moves through desirable zones. Therefore, our planner is a combination of the traditional PRM approach [12] and AD* [13] and it is flexible in that it takes into account zones with degrees of desirability. This explains why we called it Flexible Anytime Dynamic PRM (FADPRM).

We implemented FADPRM as an extension to the Motion Planning Kit (MPK)[12] by changing the definition of PRM to include zones with degrees of desirability and changing the algorithm for searching the PRM with FADPRM. The calculation of a configuration's *dd* and a path's *dd* is a straightforward extension of collision checking for configurations and path segments. For this, we customized the Proximity Query Package (PQP)[14]. In the next section, we show how FADPRM is used as a tutoring resource within *Roman Tutor*.

3 Roman Tutor

3.1 Roman Tutor Architecture

Roman Tutor's architecture contains six main components (Fig. 2): the simulator, the FADPRM path planner, the movie generator, the task editor, the student model and the tutoring sub-system.

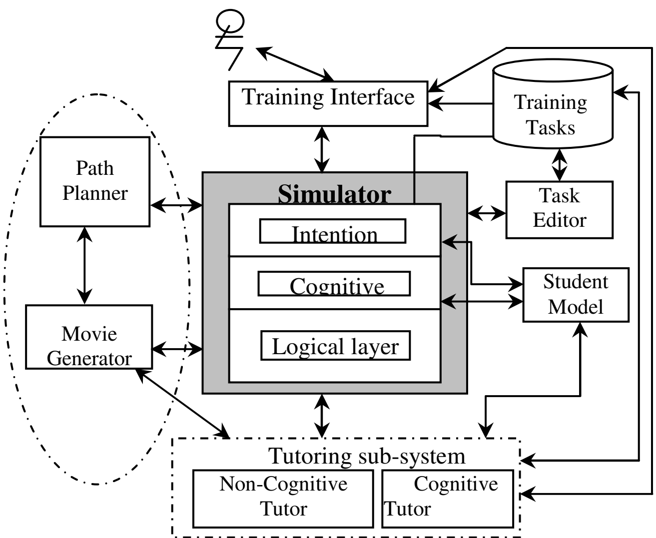


Fig. 2. Roman Tutor Architecture

As most knowledge related to the simulation-based robotic manipulation in *Roman Tutor* is mainly consistent with spatial reasoning, an appropriate structure is needed for the simulator. We equipped our system with a path planner, FADPRM, which as we said before will provide a framework to support the reasoning process within the simulator. However, this reasoning base won't be sufficient to bring useful tutoring explanations to guide and orient the astronaut during his manipulations. The level of explanation that could be given here remains very limited because the planner is connected at the logical level (level 1) of the simulator which is made up essentially of physical components in the environment such as the robot, the obstacles, the cameras and the zones, and some related low-level parameters and variables such as the configuration of the robot, the degree of desirability and whether there is a collision or not. This level is equivalent to the structure proposed by Forbus [15]. As we already said, a useful way to better support spatial reasoning is to extract and explicitly represent qualitative descriptions of shape and space. This is also known as *spatial aggregation* [5,6]. We did this by adding 2 other levels within the whole architecture for an elaborate cognitive support: the cognitive level and the intentional level.

The cognitive level (level 2) corresponds to an aggregation of the logical level in terms of zones and corridors of safe operation annotated by different domain knowledge elements. Corridors define portions of path the learner could raise during the manipulation of the robot from one place to another on the station. They are defined generally according to the geometry of the environment, the density and location of obstacles and to the choice of the spatial orientation predefined in the ISS environment (Which axis defines going down or up for example? The x-axis? The z-axis?). These corridors are annotated with high-level data providing an elaborated basis for the explanation process as said before, such as visibility through cameras and some environment-related knowledge (proximity to main obstacles, etc.). Zones define precise places of the station with particular characteristics we have to raise during the explanation process in order to draw the learner's attention on some specific knowledge in the environment such as narrow passages, or very small devices to which we have to pay a particular attention like antennas for example. The whole environment is thus aggregated into various areas annotated with appropriate knowledge in order to get more semantic richness in guiding the astronaut during displacements of the arm (Fig. 3).

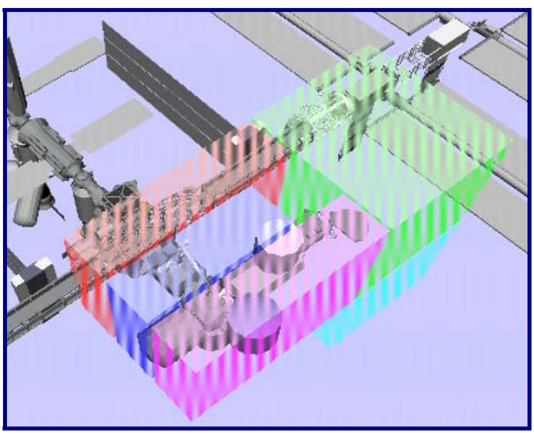


Fig. 3. Space aggregation into corridors and zones

The intentional level (level 3) contains structures of predefined tasks. The task's structure is a problem space made up a set of corridors and zones defined in level 2. The intentional level makes it possible to better follow the evolution of the astronaut in the execution of the selected task. More specifically, a task's problem space consists of a set of sequences of corridors the astronaut could undertake and a set of zones that could be related to these robot's displacements. For example, in a task for moving a load from one place to another, the learner could make the robot go through corridor 1 until reaching a specific element of the ISS, then follow corridor 2 until the robot becomes visible through the camera on the right corner of the Truss and reach the goal. Another path to execute the same task could be just going throw corridor 3 until reaching the final configuration while paying attention to zone 3 containing a tiny antenna.

The student model makes it possible to keep track of the learner's domain knowledge acquisition during trainings in order to better adapt the interaction. In addition to keeping a record on the performances of the learner, this model also stores environment-related knowledge the student acquired/understood and information on what procedures or concepts he used. It thus refers to levels 2 and 3.

The expert designs and publishes new tasks by accessing the Task Editor menu. The creation of a new task automatically requires the creation of a related abstract knowledge structure. The latter is defined at level 3 and refers to elements of level 2. We however left the possibility of creating new tasks directly in level 2 (without explicit task structures). Such a task does not allow a tutorial reasoning at the 3rd level of the simulator but is very suitable in the free practice mode where the astronaut could generate new tasks if needed.

The movie generator is a very important tutoring resource in the context of our simulator. It takes a plan computed by the path planner according to the demonstration needs and generates a movie illustrating the path proposed through appropriate cameras.

The tutoring sub-system consists of a cognitive tutor and a non-cognitive tutor. The non-cognitive tutor uses the FADPRM path planner to reason at level 1 of the simulator, whereas the cognitive tutor reasons starting from level 3 and allows thorough explanations during task execution. The rest of the paper focuses on the non-cognitive tutor.

3.2 Roman Tutor User Interface

The *Roman Tutor* interface (Fig. 4) resembles that of the robotic workstation on which the astronaut operates to manipulate the SSRMS. We have three monitors each of them connected to one of the fourteen cameras present on the station. On each monitor, we have buttons and functionalities to move the corresponding camera: Tilt, Pan and Zoom.

SSRMS can be manipulated in two modes: the For mode or the Joint-by-Joint mode. In the For mode, the learner moves the robot starting from his end-effector's position and by commanding him to go forward or backward, left or right and up or down. In the Joint-by-Joint mode, the learner selects a joint in the robot and moves it according to the link assigned to it. In the two modes, the manipulation is done incrementally. While manipulating the robot, the astronaut can choose and change the camera in each monitor to have a better sight of the zone he is working in.

Windows at the bottom of the *Roman tutor* interface contain the trace done so far by the astronaut. Every operation done by the learner is posted on this window: the selection of a new camera in a monitor, the displacement of a camera and the manipulation of the robot in the For/Joint-by-Joint mode. This trace contains also all information about the current state: if there is a collision or not, the coordinates of the End-Effector, the position and the orientation of the cameras.

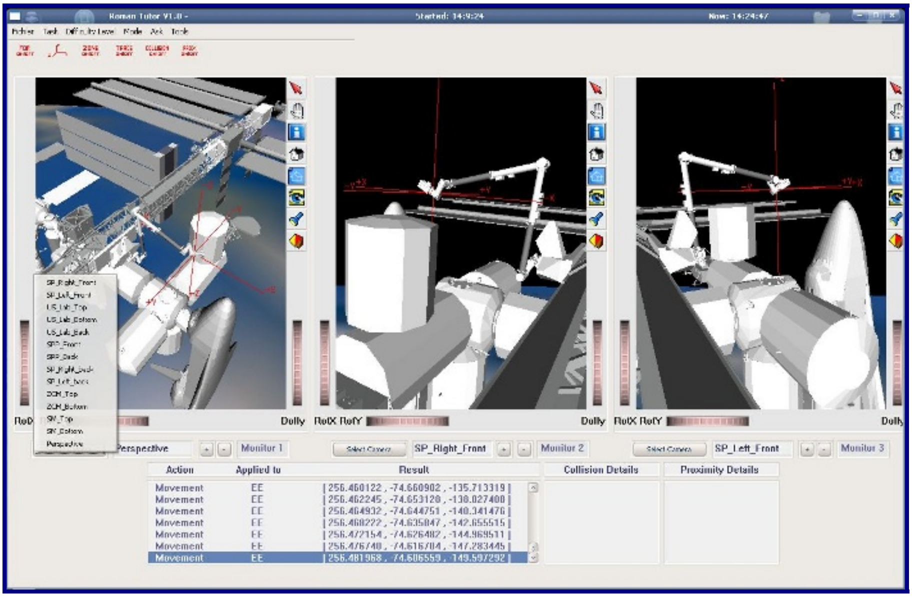


Fig. 4. Roman Tutor User Interface

The Trace window keeps then a continuous track of all operations done so far by the learner. So if he had done a mistake (a collision for example), by inspecting the Trace window, he will find out to what this was due to and he will see what to do to get out of it.

Roman Tutor's interface contains four main menus: Task, Mode, Ask and Tools. From the menu 'Task', the learner chooses one of the several tasks he wants to work on. From the second menu 'Mode', the learner chooses between two different modes: Free and Assisted. In the 'Assisted' mode, the non-cognitive tutor intervenes when needed to support the learner by guiding him or by giving him an illustrated video of the task he has to do. In the 'Free' mode, the learner relies only on the Trace window to carry on his task. In the two modes, the 'Ask' menu allows the learner to ask different types of questions while executing a task. In the last menu 'Tools', the expert is provided with three different tools that will help him design and validate new tasks to add into the system. These different tools are: the FADPRM Path-Planner, the Movie Generator and the Task Generator.

4 Using FADPRM Path-Planner for the Tutoring Assistance

One of the main goals of an intelligent tutoring system is to actively provide relevant feedback to the student in problem solving situations [3]. This kind of support becomes very difficult when an explicit representation of the training task is not available. This is the case in the ISS environment where the problem space associated with a given task consists of an infinite number of paths. Moreover, there is a need to generate new tasks on the fly without any cognitive structure. *Roman Tutor* brings a solution to these issues by using FADPRM as main resource for the tutoring feedback.

4.1 Training Tasks in Roman Tutor

Roman Tutor includes four different types of tasks on which we may train astronauts: Spatial Awareness, GoTo, Inspect and Repair.

The 'Spatial Awareness' task improves the learner's knowledge about the space station's environment by providing him with some exercises such as naming and locating ISS elements, zones and cameras. This is a very important activity since astronauts don't have a complete view of the station while manipulating the robot and must memorize a spatial model of the ISS in order to execute the different tasks. In the 'Assisted' mode, the non-cognitive tutor invokes this type of activity when it notices a lack of understanding in the student profile about some environment-related knowledge during the displacements of the robot.

In the 'GoTo' task (Fig. 5), the learner has to move the SSRMS, carrying a load or not, from one position to another different on the ISS.

Inspect and Repair tasks are variants of the 'GoTo' task. In 'Inspect', the astronaut is trained on how to go towards an element or a zone in the station and how to inspect it at several different points. In the 'Repair' task, the astronaut is trained on how to go towards an element of the station and how to execute some repairs on it at several points using the manipulator.

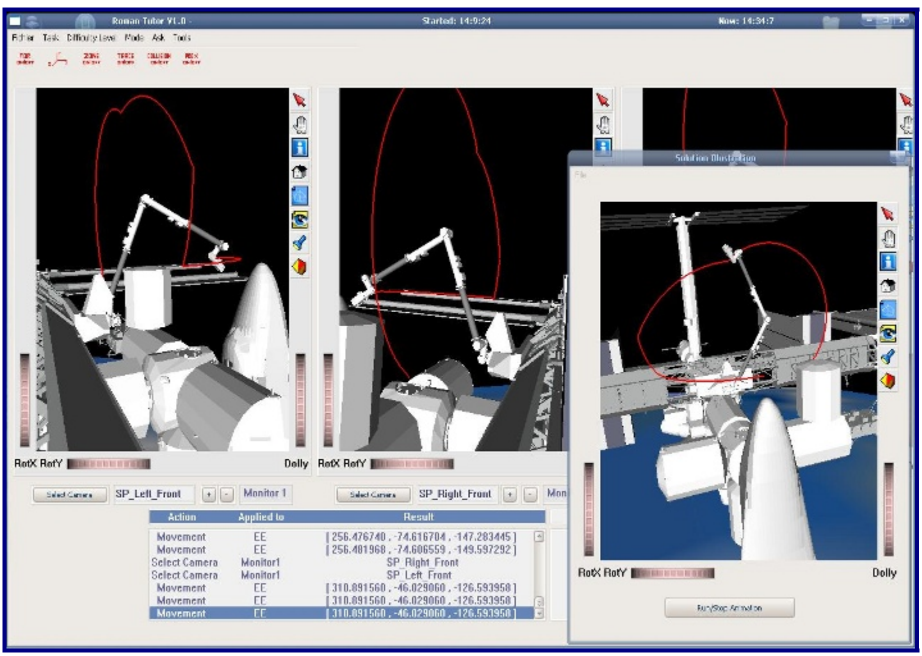


Fig. 5. 'GoTo' Task in Roman Tutor

4.2 Continuous Tutoring Assistance in Roman Tutor

As we said in the previous section, the astronaut can choose between two modes: ‘Free’ and ‘Assisted’. In the assisted mode, the non-cognitive tutor continuously assists the learner during his progression in the task.

In ‘GoTo’ tasks for example, the non-cognitive *Tutor* uses the anytime capability of the FADPRM path planner to validate incrementally student’s action or sequence of actions, give information about the next relevant action or sequence of actions, and generate relevant task demonstration resources using the movie generator. In Fig. 5, the learner is shown an illustration of the task he’s working on.

In the ‘Inspect’ Task, the learner has to reach several points on different positions of the station. The non-cognitive tutor calls the FADPRM Planner incrementally between these different positions and provides the learner with different indications that will help him follow the plan linking all these different points.

An analogue scheme is also used (both in the ‘Free’ and ‘Assisted’ mode) with the different questions in the ‘Ask’ menu. These questions may be of three different forms: How To, What if and Why Not. Several extensions are associated to these different questions. For example, with ‘How to’, one could have: How to Go To, How to avoid obstacle, How to go through zone. *Roman Tutor* answers How-To questions by generating a path consistent with the best cameras views using FADPRM and by calling the movie generator to build an interactive animation that follows that path. The incremental planning capability of FADPRM is used by *Roman Tutor* to bring answers to the What-If and Why-Not questions. In both cases, *Roman Tutor* provides the learner with relevant explanations given that his action or sequence of actions is out of scope of the generated plan or may bring him to a dead end.

We see here the importance of having FADPRM as a planner in our system to guide the evolution of the astronaut. By taking into account the disposition of the cameras on the station, we are assured that the plan the learner is following passes through zones that are visible from at least one of the cameras placed on the environment.

5 Conclusion

In this paper, we described how a new approach for robot path planning called FADPRM could play an important role in providing tutoring feedback to a learner during training on a robot simulator. The capability of the path-planner built within the simulator’s architecture to predict and to determine what manipulations might achieve a desired effect makes it a useful component in training systems involving the ‘physical’ world. We also, detailed the architecture of the intelligent tutoring system *Roman Tutor* in which FADPRM is integrated.

This constitutes a very important contribution especially in the field of intelligent tutoring systems. In fact, we have shown that it is not necessary to plan in advance what feedback to give to the learner or to explicitly create a complex task graph to support the tutoring process.

Many extensions are under study to improve the performance of our intelligent tutoring simulator. The most important one will be to implement and test the cognitive tutor which exploits levels 2 and 3 of the simulator’s structure. This will allow us to validate the usefulness of these layers in cognitive tutoring. We will also improve the expressiveness of domain knowledge structures under SSRMS within the three layers of the simulator. This will lead to a better problem diagnosis process during training.

Acknowledgment. We would like to thank the Canadian Space Agency, The Infotel Group and the Natural Sciences and Engineering Research Council of Canada (NSERC) for their logistic and financial support. Special thanks to Mahie Khan, Daniel Dubois, Patrick Hohmeyer and Kaufmann Meudja for their contribution. Special thanks also to all members of J.C. Latombe's Laboratory at the University of Stanford for providing us with MPK.

References

1. Forbus, K.: Articulate software for science and engineering education. In Smart machines in education: The coming revolution in educational technology. AAAI Press (2001)
2. Richard Angros, W. Lewis Johnson, Jeff Rickel, Andrew Scholer: Learning domain knowledge for teaching procedural skills. AAMAS, 1372-1378 (2002)
3. VanLehn, K.: The advantages of Explicity Representing Problem Spaces. User Modeling, Springer Verlag LNAI 2702:3. (2003)
4. Crowley R.S., Medvedeva, O., Jukic, D.: SlideTutor - A model-tracing Intelligent Tutoring System for teaching microscopic diagnosis. Proceedings of Int. Conf. on Artificial Intelligence in Education (2003)
5. Bailey-Kellogg, C., Zhao, F.: Qualitative Spatial Reasoning Extracting and Reasoning with Spatial Aggregates. AI Magazine, 24(4), 47-60, AAAI (2003)
6. Kuipers, B.: The Spatial Semantic Hierarchy. Artificial Intelligence 119: 191-233 (2000)
7. Latombe, J.C.: Robot motion planning. Kluwer Academic Publishers, Boston, MA, (1991)
8. Overmars, M.H.: Recent developments in motion planning. Computational Science - ICCS 2002, Part III, Springer-Verlag, LNCS 2331, (2002) 3-13
9. LaValle, S. M., Branicky, M. S., Lindemann, S.R.: On the relationship between classical grid search and probabilistic roadmaps. Int. Journal of Robotics Research 23: 673–692, (2004)
10. Roy, J., Nkambou, R., Kabanza, F.: Supporting spatial awareness in training on a telemanipulator in space. Intelligent Tutoring Systems, LNCS 3220, 860-863, Springer-Verlag (2004)
11. Belghith, K., Kabanza, F., Hartman, L., Nkambou, R.: Anytime Dynamic Path-Planning with Flexible Probabilistic Roadmaps. Proc. IEEE Int. Conf. on Robotics and Automation (2006)
12. Sanchez, G., Latombe, J.C.: A single-query bi-directional probabilistic roadmap planner with lazy collision checking. Int. Symposium on Robotics Research (ISRR'01). Springer Tracts in Advanced Robotics, Springer, 403-417 (2001)
13. Likhachev, M., Ferguson, D., Stentz, A., Thrun, S.: Anytime Dynamic A*: An Anytime Replanning Algorithme. Proc. of Int. Conf. on Automated Planning and Scheduling, 2005
14. Larsen, E., Gottschalk, S., Lin, M., Manocha, D.: Fast Distance Queries using Rectangular Swept Sphere Volumes. In Proc. of IEEE Int. Conf. on Robotics and Automation, 4:24-28, (2000)
15. Forbus, K.: Using qualitative physics to create articulate educational software. IEEE Expert, 32-41 (1997).

Robust Simulator: A Method of Simulating Learners' Erroneous Equations for Making Error-Based Simulation

Tomoya Horiguchi[1] and Tsukasa Hirashima[2]

[1] Faculty of Maritime Sciences, Kobe University, 5-1-1, Fukaeminami, Higashinada, Kobe, Hyogo, 658-0022 Japan
horiguti@maritime.kobe-u.ac.jp
[2] Department of Information Engineering, Hiroshima University, 1-4-1. Kagamiyama, Higashihiroshima, Hiroshima 739-8527 Japan
tsukasa@isl.hiroshima-u.ac.jp

Abstract. Error-based Simulation (EBS) is a framework for assisting a learner to become aware of his errors. It makes a simulation based on his erroneous hypothesis to show *what unreasonable phenomena would occur if his hypothesis were correct*, which has been proved effective in causing cognitive conflict. In making EBS, it is necessary (1) to make a simulation by dealing with a set of inconsistent constraints because erroneous hypotheses often contradict the correct knowledge, and (2) to estimate the 'unreasonableness' of phenomena in a simulation because it must be recognized as 'unreasonable.' Since the method used in previous EBS-systems was very domain-dependent, this paper describes a method for making EBS based on any inconsistent simultaneous equations by using TMS. It also describes a set of general heuristics to estimate the 'unreasonableness' of physical phenomena. By using these, a prototype EBS-system was implemented and examples of how it works are described.

1 Introduction

The critical issue in assisting constructivist learning is to provide a learner with feedback which causes cognitive conflict when he makes errors [9]. We call this 'the assistance of error-awareness,' and think there are two kinds of methods for it. The one is to show the correct solution and to explain how it is derived. The other is to show *an unreasonable result which would be derived if his erroneous idea/solution were correct*. We call the former 'indirect error-awareness,' and the latter 'direct error-awareness.' [5]

Usual simulation-based learning environments (SLEs, for short) [11-13] give the assistance of indirect error-awareness because they always provide the correct solution (i.e., correct phenomena) a learner should accept finally. The understanding by such assistance is, however, 'extrinsic' because they show only physically correct phenomena whatever erroneous idea a learner has. In addition, in usual SLEs, a learner must translate his (erroneous) hypothesis into the input which doesn't violate the constraints used by a simulator. This makes it difficult to identify what kind of phenomena a learner predicts. It is, therefore, difficult to estimate the 'seriousness' of the difference between the correct phenomena and his prediction.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 655 – 665, 2006.

Error-based Simulation (EBS, for short) [2, 3] is a framework for assisting such direct error-awareness in SLEs. It makes simulations based on the erroneous ideas/solutions externalized by a learner (we call them 'erroneous hypotheses'), which results in unreasonable (unacceptable) phenomena and makes him be aware of his errors. EBS can make the understanding 'intrinsic' because a learner is shown the 'unreasonableness' of his hypothesis as physically impossible phenomena. In addition, he can input his (erroneous) hypothesis as it is, which makes it possible to control the 'unreasonableness' of the phenomena. It has been proved that EBSs cause strong cognitive conflict and lead learners to a deeper understanding [6-8].

In designing SLEs with EBS, two issues must be addressed. (1) The representation of erroneous hypotheses often contradicts the constraints necessary for making a simulation (i.e., the correct knowledge of the domain). (2) The result of a simulation must be recognized to be 'unreasonable' by a learner. Therefore, two mechanisms are necessary: the one for making a simulation by dealing with a set of constraints which may include a contradiction, and the other for estimating the 'unreasonableness' of phenomena in simulation by using explicit criteria.

We have developed a few EBS-systems for mechanics problems in which a learner is asked to set up equations of mechanical systems [4, 6, 7]. The technique used in them, however, could deal with only a limited class of errors in simultaneous equations, and used domain-specific heuristics to avoid contradiction in calculation. Because such implicitness of knowledge in dealing with constraints made it difficult to predict what kind of physical phenomena would occur in a simulation, the criteria for the 'unreasonableness' of them were given externally and empirically. In other words, it was difficult to apply to other domains.

In this paper, therefore, we propose a technique which can deal with any erroneous simultaneous equations/inequalities to make EBS. It is called 'Partial Constraint Analysis (PCA, for short),' which detects and eliminates contradictions in a set of constraints given by simultaneous equations/inequalities. We also propose a set of heuristics to estimate the 'unreasonableness' of phenomena in EBS in a general way. It describes the meaning of typical equations/inequalities of physical systems and is used for predicting what kind of physical phenomena would occur if they were violated. We think these methods are useful because they are domain-independent and because the problem in setting up equations of physical systems is important in learning science.

2 Requisites for EBS-Systems and the Previous Method

2.1 Requisites for EBS-Systems

The simulators in usual SLEs are designed to do calculation by using a set of constraints which represents correct knowledge of the domain. A learner must interact with the environments within these correct constraints. In EBS-systems, on the other hand, a learner is allowed to externalize his hypothesis without this limitation. That is, when a hypothesis is erroneous, it may violate the correct constraints and the simulator can't do the calculation. In making EBS, therefore, it is necessary to analyze the union of the correct constraints and the constraint which is the representation of a learner's erroneous hypothesis. If a contradiction is detected, some of the correct constraints are relaxed (i.e., deleted) to make the rest consistent (i.e., EBS shows that if a learner's erroneous hypothesis were correct, it would be inevitable for some correct constraints to be violated). The module which deals with such inconsistency is called the 'robust simulator.'

In addition, phenomena in EBS must be recognized as 'unreasonable' by a learner. While they are physically impossible phenomena because they include an erroneous hypothesis (and/or some correct constraints are deleted), a learner who has only incomplete knowledge of the domain doesn't always recognize their 'unreasonableness.' It is, therefore, necessary to estimate the 'unreasonableness' of the phenomena in EBS. When some correct constraints are deleted, the physical meaning of them provide useful information for estimation. When no correct constraints are deleted (i.e., the erroneous hypothesis doesn't contradict them), the criteria for estimation must be given externally (in this case, the erroneous hypothesis contradicts the correct knowledge which isn't represented explicitly in the simulator (e.g., correct prediction about the behavior of the system)).

2.2 The Previous Method and Its Limitation

We have developed a few EBS-systems for the mechanics problem in which a learner is asked to set up equations of mechanical systems [4, 6, 7]. They, however, can deal with only a limited class of errors in simultaneous equations, and since their robust simulators don't have explicit knowledge for dealing with constraints, the criteria for 'unreasonableness' were given externally and empirically. In this section, we describe the method used in these systems and discuss its limitation.

In previous EBS-systems, a learner inputs a set of equations of motion each of which corresponds to each object (i.e., particle) in a physical system (this is his hypothesis). In the simulators, the correct equations of motion are described with the values of constants and the domains of variables in them. In order to deal with the frequent errors in this domain efficiently, it is assumed that a learner inputs the equations of motion for all of the objects, and that only one of them is erroneous. Under this condition, it is necessary to make simulations whether the simultaneous equations are consistent or inconsistent.

The procedure [2, 3] is as follows: First, one variable is chosen from the variables in the erroneous equation as 'ER-Attribute,' which reflects the error. After that, the values of the other variables in the simultaneous equations are calculated by using the correct equations (as for the erroneous equation, the corresponding correct one is used). Then, these values are substituted for the variables in the erroneous equation to calculate the value of ER-Attribute (this is called 'ER-Value'). Thus, the EBS is made in which the erroneous behavior of the object which has ER-Attribute (this is called 'ER-Object') reflects a learner's error. Though this ER-Value contradicts the values of the other variables mathematically when the ER-Attribute (or the variable which depends on ER-Attribute) is in other equations[1], it is regarded as appropriate reflection of a learner's error in this problem (i.e., he isn't able to think about the behavior of the ER-Object consistently with those of the other objects). In fact, when

1 It is because while the values of other variables are calculated not to contradict the 'correct value' of ER-Attribute, ER-Value (i.e., the 'erroneous value' of ER-Attribute) is calculated by using them.

ER-Attribute is the position/velocity/acceleration of ER-Object, this procedure means deleting the constraints on the relative position/velocity/acceleration between some objects (they prevent the objects from overlapping and/or balance the internal force between them).

In such simulations, the phenomena often occur in which the constraints on the relative position/velocity/acceleration between objects are violated (i.e., the objects overlaps and/or the internal force between objects doesn't balance). These constraints, however, weren't explicitly represented in previous EBS-systems, that is, the knowledge of constraints the robust simulator deletes to deal with contradiction is implicit. Instead, the criteria which check whether the phenomena in EBS qualitatively differ from the correct phenomena are given externally to estimate the 'unreasonableness' of phenomena in EBS.

3 PCA: Partial Constraint Analysis

In previous EBS-systems, the procedure for avoiding contradiction (i.e., the constraint to be deleted) was specified before calculation by utilizing the assumption of the domain (i.e., domain-dependent). Under general conditions, however, it is necessary to detect the cause of contradiction in a set of constraints explicitly and to eliminate it appropriately. In this section, we propose PCA as a method for doing such a calculation. PCA deals with simultaneous equations/inequalities which may include contradiction. The reasons it specializes in simultaneous equations/inequalities are as follows:

- Simultaneous equations/inequalities are one of the most popular form for representing and understanding (natural) phenomena. They often become an objective in learning by themselves.
- They can represent the concepts and relations of the domain elaborately, and can be used for doing quantitative calculation.
- Because they represent relatively strong constraints (i.e., global and/or quantitative ones), they easily contradict each other when they are erroneous.
- As for the problem in which a learner is asked to set up equations of (physical) systems, the assistance of direct error-awareness becomes important because indirect error-awareness is often unhelpful. That is, in such a problem, while a learner can often predict the correct phenomena, he has difficulty in setting up the equations to describe them. It is no use showing him the simulation of correct phenomena (by using correct equations).

3.1 The Algorithm

PCA can deal with a set of constraints as follows:

- All of the constraints are represented in the form of equation/inequality.
- It is possible to solve each equation/inequality for an arbitrary variable in it symbolically.

We, hereafter, assume the constraints include only equations because the generality isn't lost[2]. PCA first constructs the 'Constraint Network' (CN, for short) of given

[2] While an equation gives a constraint on the value of a variable, an inequality gives a constraint on the range of a variable. From this viewpoint, they can be regarded as equivalent in the following algorithm.

simultaneous equations *S*, then searches for the consistent part of it (we call it 'Partial Constraint Network,' PCN, for short). CN is a graph structure which represents the dependency between equations and variables in *S*. It has two kinds of nodes: equation node and variable node (they are called e-node and v-node, respectively). An e-node stands for an equation and a v-node stands for a variable in *S*. Each v-node is linked to the e-nodes in which it is included. There are two kinds of variables: endogeneous variable and exogenous variable. The values of endogeneous variables are determined by the internal mechanism of the system, while the values of exogenous variables are given externally. The v-node which stands for the latter is called ex-v-node. In CN, an e-node is regarded as a calculator. That is, an e-node with *n* links calculates the value of one v-node linked to it by being supplied *n-1* values from the other links. A v-node with *n* links gets its value from one e-node linked to it and supplies the value to the other links. When *S* is consistent, the value of each v-node is uniquely calculated by just one e-node[3] (assuming all the values of ex-v-nodes are given externally). That is, for each v-node, the unique path is determined which propagates the value(s) of ex-v-node(s) to it (If CN has a loop, the values of v-nodes in the loop are determined simultaneously by solving the simultaneous equations for it).

When *S* is inconsistent, the following irregularities occur:

- (under-constraint) There are some e-nodes in each of which the sufficient values of (*n-1*) v-nodes aren't supplied to calculate the value of a v-node. In other words, there are some v-nodes the values of which can't be determined by any path.
- (over-constraint) There are some v-nodes each of which has more than one path to determine its value. In other words, there are some e-nodes which have no simultaneous solution.

Taking an ex-v-node (or a v-node which is given its temporary value) as an initial PCN, PCA extends it by adding the nodes step by step to which the value can be regularly propagated. To each v-node in PCN, the method for calculating its value is attached symbolically using the values of ex-v-nodes and e-nodes on the path of the propagation (it is called 'calc-method,' for short). When PCA meets irregularities, it resolves them as follows:

- (under-constraint) When the values of some v-nodes (necessary for calculation) aren't supplied to an e-node, PCA gives them temporary values and continues the calculation by using the values (a v-node given its temporary value is called a 'dummy').
- (over-constraint) PCA deletes one of the e-nodes responsible for the contradiction from PCN to make the rest consistent.

The procedure above is continued until no propagation of values is possible any more. If PCA meets a loop, it tries to solve the simultaneous equations which consist of the e-nodes in the loop (i.e., to determine one (final) value of the dummies included in the simultaneous equations). In order to detect a contradiction and identify the e-nodes responsible for it, PCA must have nonmonotonic reasoning ability. Cooperation

---

3 When an equation has more than one solution (e.g., an equation of the second degree gives two values of a variable *x*), the search by PCA is continued for all of them in parallel (CN is duplicated by the number of the solution).

with TMS (Truth Maintenance System) [1] is a promising approach to realize PCA, because TMS provides the efficient function for maintaining the dependency network of propagation of constraints. We, therefore, adopt this approach.

By tracing the dependency network, PCA can also explain of why the simultaneous equations aren't (or are) solvable, which is the reason it propagates constraints instead of searching the solvable subset(s) of the equations in turn.

3.2 The Mechanism of Contradiction Handling

In this section, we elaborate on the mechanism of detecting and eliminating a contradiction (i.e., over-constraint). In order to identify the e-nodes responsible for a contradiction when it is detected, we make a TMS maintain the justification of calc-method of each v-node (i.e., the e-node and other v-nodes used for determining its calc-method). By introducing a TMS, all the possible PCNs can be obtained independently of the choice of the initial PCN. In this paper, we describe the algorithm with a basic JTMS (Justification-based TMS) [1]. In the following, the correspondences between the nodes in CN and the TMS nodes, the justifications, the detection and handling of a contradiction are explained in this order.

Since the value (i.e., calc-method) of an ex-v-node is given externally, a premise node is assigned to it. A simple node (i.e., neither premise nor assumption) is assigned to a v-node, which is made 'in' by a justification when its calc-method is determined, and is made 'out' otherwise. When a v-node n is made a dummy, the corresponding assumption node *dum-n* is created and enabled, which justifies (the simple node of) n. That is, the calc-method of a dummy is made temporarily determined. An assumption node is assigned to an e-node, which is enabled when it is used in determining the calc-method of a v-node, and is retracted when the calc-method is cancelled. When the calc-method of a v-node n_s is determined by using an e-node n_e and the determined v-nodes which are linked to n_e, an assumption node J_S (which stands for this calculation) is created and enabled, and (the simple node of) n_s is justified by J_S, n_e and (the TMS nodes of) the v-nodes[4].

A contradiction is detected when all the v-nodes which are linked to an e-node n_e are determined and the constraint of n_e (it is called 'c-equation*') can't be satisfied. In this case, a contradiction node is justified by n_e (which is enabled) and (the TMS nodes of) the v-nodes. Then, the contradiction handler is called to retract one of the assumption nodes underlying the contradiction (it is marked 'unsimulatable' and deleted from PCN). The e-nodes which are used in determining the calc-methods of the v-nodes which are made 'out' by this retraction and n_e are also retracted (if not deleted) and become the objects of search again (their corresponding assumption nodes of calculation J_Ss are also retracted).

When the c-equation* can be satisfied, the calc-method of a dummy n_d in it is (finally) determined. In this case, all the calc-methods of v-nodes that include n_d are replaced by substituting the solution of c-equation* for them (let the v-nodes x_ps ($p = 1, ..., t$)). In addition, the justifications of (the simple nodes of) x_ps must be also replaced. There are two kinds of replacing of justification: (1) backward-replacing

[4] J_S (and J_L which is defined later) is used for cancelling the justification of (the simple node of) n_s when its calc-method is cancelled afterwards.

and (2) loop-replacing. The former occurs when there is only one v-node which is linked to n_e (where the c-equation* arose) and the calc-method of which includes n_d. In this case, the directions of all justifications between n_e and n_d are reversed. The latter occurs when there is more than one such v-node, which corresponds to a loop. In this case, a simple node *Loop* (which stands for the solution of simultaneous equations in the loop) is created. It is justified by the e-nodes which were used in determining the previous calc-methods of x_ps, (the nodes of) the v-nodes which are linked to these e-nodes (and the calc-methods of which don't include n_d) and an assumption node J_L which is created and enabled (which stands for this calculation). The x_ps are justified by the *Loop* (when the *Loop* is made 'out' afterwards, J_L is also retracted). In both cases, all the assumption nodes that stand for the previous calculations are retracted.

3.3 An Example of How PCA Works

In Fig.1a, assume that the simultaneous equations in the loop (eq_1, eq_2, eq_3 and eq_4) have a solution. When PCA begins a search taking { *e1* } as an initial PCN, it first meets eq_1 which is linked to more than one undetermined v-nodes. Assume that *z* is made the dummy and the calc-method of *x* is determined by using *dum-z*, *e1* (which is an ex-v-node) and eq_1. Then, after determining the calc-methods of *y* (by *x* and eq_2) and *w* (by *y* and eq_3) (eq_5 is pushed into the stack), PCA meets eq_4. Since all the v-nodes that are linked to eq_4 are determined, it solves the c-equation* for the (only) dummy *z* and substitute the solution for the calc-methods of *x*, *y* and *w*. Fig.1b and 1c show the dependency networks of TMS nodes before and after the solution of the loop (all the nodes for calculations (J_Ss) are omitted). In them, after *dum-z* (which justified *z*) was retracted, the replaced calc-methods of *x*, *y*, *w* and *z* are justified by the *Loop* (the *Loop* is justified by all the TMS nodes which were concerned with the solution of c-equation* (i.e., eq_1, eq_2, eq_3, eq_4 and *e1*)). Then, eq_5 is popped out of the stack and the calc-method of *v* is determined (by *w* and eq_5). Next, PCA meets eq_6. Since all the v-nodes that are linked to eq_6 are determined and the calc-methods of which don't include dummies, this constraint is unsolvable. Therefore, it justifies a contradiction node by using *v*, eq_6 and *e2*, and the TMS shows the assumption nodes underlying the contradiction eq_1, eq_2, eq_3, eq_4, eq_5 and eq_6). When eq_2 is retracted (i.e., deleted from PCN) for example, *Loop* loses its support (i.e., it becomes 'out') and *x*, *y*, *z*, *w* and *v* become 'out.' Therefore, the e-nodes which were used for determining their calc-methods (i.e., eq_1, eq_3, eq_4, eq_5 and eq_6) are also retracted and pushed into the stack again. When eq_5 is retracted to eliminate the contradiction, only *v* becomes 'out.' In this case, only the calc-method of *v* is redetermined (by using *e2* and eq_6).

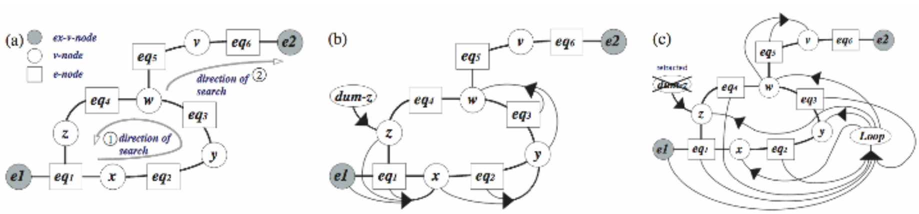


Fig. 1. An example of how PCA works

4 Heuristics for Estimating the 'Unreasonableness'

4.1 Four Heuristics for Deleting Constraints

When erroneous simultaneous equations[5] are unsolvable, PCA outputs a PCN in which a (set of) equation(s) is deleted from them. In general, there exists a multiple choice of (sets of) equations to be deleted to make PCN consistent, and the choice considerably influences what kind of physical phenomena would occur in simulation. In other words, by describing the meaning of typical equations/inequalities in physical systems (i.e., what kind of constraints they represent) in advance, it becomes possible to estimate the 'unreasonableness' of physical phenomena in EBS. This can be used as the criteria when a robust simulator makes a choice from (sets of) equations to be deleted.

In this section, from the viewpoint of the assistance of direct error-awareness, we discuss the criteria for making EBS 'unreasonable' as much as possible, to propose four domain-independent heuristics.

- *(H1) Don't delete the erroneous equations.* The purpose of EBS is to show '*if a learner's erroneous hypothesis were true, some 'unreasonable' phenomena would occur*.' An erroneous equation which reflects a learner's error, therefore, must not be deleted. Only when there is more than one erroneous equation and they contradict each other, may some of them be deleted.
- *(H2) Delete the equations which represent the topic of learning (e.g., physical law/principle) prior to others.* When the topic of learning is a relation between some physical constants/variables (e.g., physical law/principle), it is useful to delete the equation which represents it because the phenomena in which the relation doesn't hold much focus on the topic. For example, when a learner is learning 'the law of conservation of energy,' it facilitates his error-awareness to show 'if his erroneous equations were true, the total energy of the system wouldn't be conserved.'
- *(H3) Delete the equations/inequalities which describe the values of physical constants or the domains of physical variables prior to others.* The equations/inequalities which describe the values of physical constants or the domains of physical variables often represent the most basic constraint of the domain, such as the meaning of the constants/variables, the conditions of existence of physical objects/processes which have the constants/variables as attributes, or the conditions for the equations/inequalities to be valid. The phenomena in which these constraints are violated, therefore, are easily recognized as 'unreasonable.' For example, an inequality which describes a coefficient of friction as nonnegative, an equation/inequality which describes the free/blocked space of a mechanism, and an inequality which describes a Reynolds number as not more than *2300* (i.e., $R_e. <= 2300$) to assume a steady and laminar flow in a Bernoulli equation.
- *(H4) Delete the fundamental circuit equations and cut set (incidence) equations of the system prior to others.* In physical systems, fundamental circuit equations of across variables and cut set (incidence) equations of through variables [10]

[5] Also in this section, the word 'equations' is used as including inequalities. Only when it is necessary to emphasize inequalities, it is described as 'equations/inequalities.'

often represent the most basic constraints of the domain, such as the conservation of basic physical amounts, or the relations between components of the system to be held. The phenomena in which these constraints are violated, therefore, are easily recognized as 'unreasonable.' For example, cut set (incidence) equations of fluid systems or electric circuits represent the conservation of the total amount of substance which flow through components (e.g., equation of supply and demand, Kirchhoff's first law), and fundamental circuit equations of mechanical systems represent the relative velocities between components to be held. If they are deleted, the phenomena may occur in which a substance appears/disappears without its cause, or rigid objects overlap each other in spatiotemporal space.

4.2 Examples of Making EBS by Using the Heuristics

We implemented a prototype robust simulator which makes EBS by using PCA and the heuristics above. Examples in elementary electric circuits and mechanics are also prepared in which a learner is asked to set up equations for a system. In this section, we describe its implementation and illustrate how it works.

Generation of explanation. By tracing the dependency network of justifications made by PCA, the system can make an explanation of why the simultaneous equations aren't (or are) solvable. By using the heuristics above, it can also explain how unnatural phenomena will occur in the EBS. A dependency network and the meaning of a deleted equation are translated into quasi-natural language by using a simple template of explanation.

An example in an electric circuit. As for the electric circuit shown in Fig.2a, assume that a learner set up the (erroneous) equations shown in Fig.2b. These simultaneous equations are unsolvable because the two loops in their constraint network (i.e., the loop with variables v_1, v_2, i_1 and i_2, and the loop with variables v_2, i_2 and i_3) are simultaneously unsolvable (Fig.2c). The robust simulator, therefore, tries to delete some of the equations in these loops (i.e., equations (1), (2), (3), (4) and (5')). According to *(H1)*, equation (5') is not an option. Since equations (1), (3) and (4) are fundamental circuit equations and equation (2) is an incidence equation, equation (2) is deleted according to *(H4)* (in this case, *(H2)* and *(H3)* are inapplicable). The calculation by using the rest (i.e., equations (1), (3), (4) and (5')) yields an 'unreasonable' phenomenon in which the total amount of electric current at node A isn't conserved (i.e., $i_1 = 2.25(A)$, $i_2 = -0.5(A)$ and $i_3 = 1.5(A)$). Fig.2d shows the explanation made by the system.

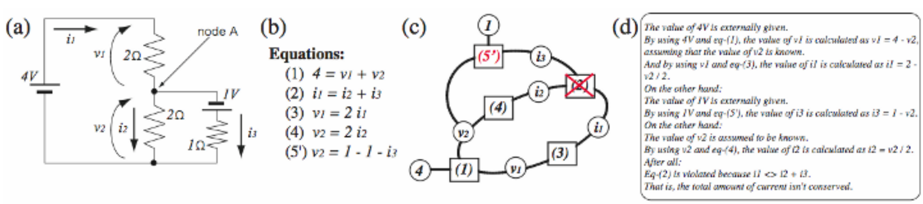


Fig. 2. An example in electric circuit

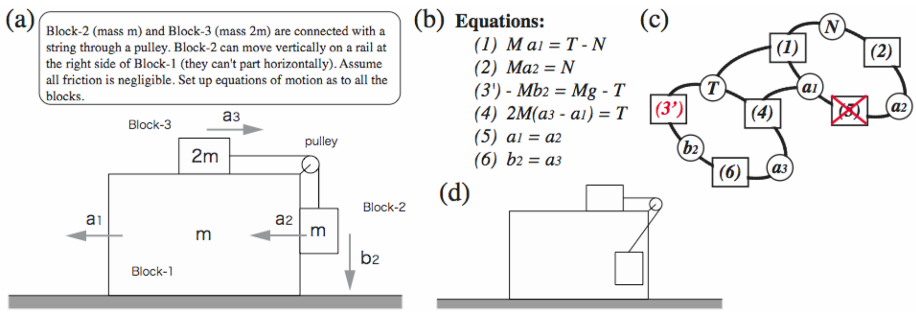


Fig. 3. An example in electric mechanics

An example in mechanics. As for the mechanical system shown in Fig.3a, assume that a learner set up the (erroneous) equations shown in Fig.3b. These simultaneous equations are unsolvable because the two loops in their constraint network (i.e., the loop with variables a_3, b_2 and T, and the loop with variables a_1, a_2 and N) are simultaneously unsolvable (Fig.3c). The robust simulator, therefore, tries to delete some of the equations on these loops (i.e., equations (1), (2), (3'), (4), (5) and (6)). According to *(H1)*, equation (3') is not an option. Since equations (1), (2) and (4) are incidence equations and equations (5) and (6) are fundamental circuit equations, equation (5) is, for example, deleted according to *(H4)* (in this case, *(H2)* and *(H3)* are inapplicable). The calculation by using the rest (i.e., equations (1), (2), (3'), (4) and (6)) yields an 'unreasonable' phenomenon in which the relative velocity between Block-1 and Block-2 isn't held (i.e., $a_1 = g/4$, $a_2 = 9g/4$, $a_3 = b_2 = 3g/2$, $T = 5Mg/2$ and $N = 9Mg/4$), that is, these blocks overlap each other (Fig.3d).

5 Conclusion

In this paper, we presented a method of simulating learners' any erroneous equations for making EBS. By this extension, it becomes possible to utilize EBS in various domains (in which setting up equations is important) for assisting direct error-awareness. We think our method is useful to cause cognitive conflict when a learner makes errors in SLEs.

References

1. Forbus, K.D., deKleer, J.: Building Problem Solvers. The MIT Press (1993)
2. Hirashima, T., Horiguchi, T., Kashihara, A., Toyoda, J.: Error-Visualization by Error-Based Simulation and Its Management. Proc. of AIED97 (1997) 450–457
3. Hirashima, T., Horiguchi, T., Kashihara, A., Toyoda, J.: Error-based Simulation for Error-Visualization and Its Management. Int. J. of Artificial Intelligence in Education, **9** (1998) 17–31
4. Hirashima, T., Horiguchi, T.: Evaluating of Error-Based Simulation by Using Qualitative Reasoning Techniques. Proc. of QR-01 (2001) 128–133

5. Hirashima, T., Horiguchi, T.: Error-Visualization for Learning from Mistakes. Transactions of Japanese Sciety for Information and Systems in Education, **21** (2004) 178–186 (in Japanese)
6. Horiguchi, T., Hirashima, T., Kashihara, A., Toyoda, J.: Error-Visualization by Error-Based Simulation Considering Its Effectiveness -Introducing Two Viewpoints-. Proc. of AIED99 (1999) 421–428
7. Horiguchi, T., Hirashima, T.: A Method of Creating Counterexamples by Using Error-Based Simulation. Proc. of ICCE2000 (2000) 619–627
8. Horiguchi, T., Hirashima, T., Okamoto, M.: Conceptual Changes in Learning Mechanics by Error-based Simulation. Proc. of ICCE2005 (2005) 138–145
9. Perkinson, H.J.: Learning From Our Mistakes: Reinterpretation of Twentieth Century Educational Theory. Greenwood Press (1984)
10. Shearer, J.L., Murphy, A.T., Richardson, H.H.: Introduction to System Dynamics. Addison-Wesley Publishing Company (1971)
11. Towne, D., de Jong, T., Spada, H. (Eds): Simulation-based experiential learning. Berlin/New York: Springer (1993)
12. Towne, D.M.: Learning and Instruction in Simulation Environments. Educational Technology Publications, Englewood Cliffs, New Jersey (1995)
13. Wenger, E.: Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge. Morgan Kaufmann (1990)

Evaluating the Effectiveness of Tutorial Dialogue Instruction in an Exploratory Learning Context

Rohit Kumar, Carolyn Rosé, Vincent Aleven, Ana Iglesias, and Allen Robinson

Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213
{rohitk, cp3a, va0e, alr}@andrew.cmu.edu

Abstract. In this paper we evaluate the instructional effectiveness of tutorial dialogue agents in an exploratory learning setting. We hypothesize that the creative nature of an exploratory learning environment creates an opportunity for the benefits of tutorial dialogue to be more clearly evidenced than in previously published studies. In a previous study we showed an advantage for tutorial dialogue support in an exploratory learning environment where that support was administered by human tutors [9]. Here, using a similar experimental setup and materials, we evaluate the effectiveness of tutorial dialogue agents modeled after the human tutors from that study. The results from this study provide evidence of a significant learning benefit of the dialogue agents.

1 Introduction

In this paper we evaluate the instructional value of an implemented tutorial dialogue system integrated with an exploratory simulation-based learning environment. Tutorial dialogue has long been argued to hold a great potential for improving the effectiveness of instruction that can be offered by intelligent tutoring systems. This claim is largely based on evidence from famous studies of expert human tutoring, where it was demonstrated to beat classroom instruction by two standard deviations [2,3]. Dialogue offers the potential for eliciting a high degree of cognitive engagement from students and offers tutors a great deal of flexibility in adapting the presentation of material to the individual needs of students.

While tutorial dialogue holds the potential for many benefits for the student, it also comes with a cost in terms of both time and energy. Vanlehn et al. (2005) present a review of a series of experiments comparing human tutoring to non-interactive control conditions. Surprisingly, the advantages of human tutoring are not consistently demonstrated across studies. In order for the benefits of tutorial dialogue to be demonstrated conclusively, the benefits must outweigh the cost. One potential explanation for the inconsistency in the pattern of results from previous comparisons of human tutoring to non-interactive alternatives is that some studies were conducted in environments that did not take advantage of the potential benefits of dialogue to a great enough extent for the benefits experienced by students to clearly outweigh the cost. Consequently, our work is motivated more by the question of where tutorial dialogue might have the greatest impact on learning rather than evaluating whether dialogue is always a more effective form of instruction than an alternative. Thus, the research goal of the CycleTalk project, which forms the context for the work presented in this paper, is to evaluate the benefits of tutorial dialogue in an exploratory learning

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 666 – 674, 2006.

context where we hypothesize that the creative nature of the task will create an environment in which the benefits of tutorial dialogue will be more clearly evidenced than in previously published comparisons.

2 Motivation

The study presented in this paper builds on results from a previous study in which the students performed the same task with the same simulation environment but interacted with a human tutor rather than a tutorial dialogue system [9]. In that study, we compared three goal level conditions: The first condition was a script based learning condition (S) in which students worked through a set of written instructions that explained step-by-step how to move through the simulation space. The second condition was a slightly more exploratory problem solving condition (PS) where students were given their instructions at a higher level in terms of problem solving goals and would need to use means-ends analysis to derive the set of low level steps required to satisfy those goals. However, they were provided with reference material that contained all of the same information about how to achieve those goals as the students in the S condition. Thus, for all practical purposes, the only difference between the instructional materials provided to the PS condition students and those provided to the S condition students was the insertion of some extra section divisions and the labeling of the section headers. Furthermore, they used an augmented version of the simulation environment that allowed them to request hints during their problem solving. In a final, more exploratory condition, rather than being presented with an exact ordering of problem solving goals, students were provided with the same set of goals but told that they were free to address those goals in whatever order served their instructional objectives best. They were to negotiate the ordering with a human tutor who was there to support them. Thus, we referred to this condition as the Negotiated Problem Solving Goals (NPSG) condition. Because the students were able to interact with a human tutor, they used the original version of the simulation software that did not include the help button that the PS students had access to. The students in the NPSG condition, which was the only condition with dialogue-based support, learned the most out of the three conditions. In particular, they learned significantly more than students in the PS condition ($p < .05$) and marginally more than the students in the S condition ($p < .1$).

We consider these experimental results to contribute to the line of research investigating the trade-offs between human tutoring and non-dialogue control conditions, although the experimental setup is different in important ways from that used in previous comparisons. Consider the following series of empirical investigations. First, an evaluation of the AutoTutor system, a tutorial dialogue system in the domain of computer literacy, showed an advantage over re-reading of a textbook of about 0.5 standard deviations [8]. The textbook re-reading condition itself was no better than a no-treatment control condition. Similarly, a recent evaluation of WHY-AutoTutor, a system based on the same architecture as the original AutoTutor but applied to the domain of qualitative physics, demonstrates a significant advantage of this system over a textbook reading control [6]. However, in a different experiment the learning results obtained with WHY-AutoTutor were no worse than *a human tutoring condition* and yet not better than those in a control condition in which students read targeted "mini-lessons," short texts that covered the same content as that presented in the dialogue [5]. In [9] as discussed in the previous paragraph, again we evaluated the merits of human tutoring (the NPSG condition) in comparison to two non-dialogue control conditions (the S and PS conditions). But note that the setup was different in important ways. First, students in all conditions in our study were presented with informationally equivalent reading materials. Rather than replacing the reading materials as in [5], the role of the human tutors in our study was to help students navigate and understand the materials. Secondly, the reading materials were neither as brief nor targeted to the test as the "minilessons" used in [5] nor were they as extensive as a text-book. Thus, the key difference is that because decisions about how to navigate the materials were required, there was a potential benefit to be gained from support in this navigation from the negotiation with the tutor that would result in appropriate tailoring of the material.

The purpose of the study presented in this paper is to evaluate the first implementation of the NPSG approach to instructional support in a simulation-based learning environment.

3 The CycleTalk System

We are conducting our research in the domain of thermodynamics, using as a foundation the CyclePad articulate simulator [4]. CyclePad was developed with the intention of allowing students to engage in design activities earlier in their education than was possible previously. Our explorations of CyclePad use focus on design and optimization of thermodynamic cycles, specifically Rankine cycles. A thermodynamic cycle processes energy by transforming a working fluid within a system of networked components (condensers, turbines, pumps, and such). Power plants, engines, and refrigerators are all examples of thermodynamic cycles. Rankine cycles are a type of heat engine that forms the foundation for the design of the majority of steam based power plants that create the majority of the electricity used in the United States. There are three typical paradigms for design of Rankine cycles, namely the Simple Rankine Cycle, Rankine Cycle with Reheat, and Rankine Cycle with Regeneration. As students work with CyclePad on design and optimization of Rankine Cycles, they start with these basic ideas and combine them into novel designs.

We have constructed a cognitive task analysis describing how students use CyclePad to improve a design of a thermodynamic cycle [10]. Students begin by laying out the initial topology of a cycle using the widgets provided by CyclePad. For example, they may choose to construct the topology for a Simple Rankine cycle, which consists of a heater, a turbine, a condenser, and a pump. Students must next set values for key parameters associated with each widget until the cycle's state is fully defined. At that point, the student can explore the relationships between cycle parameters by doing what are called sensitivity analyses, which allow the student to observe how a dependent variable's value varies as an independent variable's value is manipulated. Students may experiment with a number of alternative designs. Based on their experience they can plan strategies for constructing cycle designs with higher efficiency. Making adjustments to improve cycle efficiency is called optimization. As part of this optimization process, students may reflect upon their understanding of how thermodynamic cycles work.

As a foundation for a tutorial dialogue system, we constructed a tutoring system backbone to integrate with CyclePad. The purpose of this tutoring system backbone was to introduce the capability of tracing the student's path through their exploration through the simulation space as well as to provide the capability of offering students hints along the way in the style of model tracing tutors. We used a tool set called the Cognitive Tutor Authoring Tools (CTAT) [7,1] to develop this backbone tutor. The Cognitive Tutor Authoring Tools (CTAT) support the development of so-called Pseudo Tutors, which can be created without programming, namely, by demonstrating correct and incorrect solutions to tutor problems, which are then stored in a representation referred to as a Behavior Graph, which is then used to trace the solution paths students follow as they are working with the Pseudo Tutors at run time. Each node in the Behavior Graph represents an action a student may make. We integrated tutorial dialogue with the psuedotutors by allowing dialogues to be launched either in response to hint requests or as success messages when students complete the action associated with a node in the Behavior Graph. We build on the foundation of our prior work building and evaluating Knowledge Construction Dialogues (KCDs)[11].

4 Content Development for Tutorial Dialogues

Using an exploratory corpus analysis tool called InfoMagnets that was developed in our research group, a thermodynamics domain expert constructed a topic analysis of a corpus of human tutoring dialogues collected during the previous classroom study involving CyclePad mentioned earlier [9]. Altogether each student's protocol was divided into between 10 and 25 segments such that the entire corpus was divided into approximately 379 topic segments altogether. The resulting analysis consisted of 15 distinct topics, with each student covering between 4 and 11 of these topics either once or multiple times throughout their interaction with the tutor. In support of the instructional value of these discussion topics, we found a strong and reliable correlation between topic coverage (i.e., percentage of topics discussed at least once) and post-test score, with pre-test score used as a covariate (R^2=.715, N=21, $p < .05$). The topic analysis of the corpus gives us a way of quickly getting a sense of how tutors divided their instructional time between different topics of conversation. Based on our topic analysis of the human-tutoring corpus, we designed 12 KCDs, 7 of which were focused on the specifics of optimization and 5 of which were more conceptual.

Here is an example collected during our study of a student conversing with a KCD about manipulating the maximum temperature of the cycle.

> **Tutor:** Let's stop for a minute and reflect on what we have seen. Having seen how changing T-Max (i.e., the Maximum Temperature at the Heater Outlet) affects the efficiency of the cycle, do you think it is a good idea in general for the Maximum Temperature to be increased and kept high?
> **Student:** *yes*
> **Tutor:** Okay. Let's think now about one important consequence of keeping T-Max high. What happens to the steam quality at the turbine exit if you increase the maximum temperature of the cycle?
> **Student:** *Increases*

Tutor: Right. As increasing the temperature makes the efficiency and steam quality better, we should increase the maximum temperature of the cycle. But can we keep on increasing the maximum temperature at the heater outlet indefinitely, or are there limitations to keep in mind?
Student: *there are limitations such as material constraints*
Tutor: Great. we should try to keep T-Max (i.e., the maximum temperature at the Heater outlet of the cycle) as high as possible without risking safety concerns or exceeding the maximum temperature the material can withstand. Keeping T-Max high increases efficiency and keeps the steam quality high, which are both important. This is an important principle to keep in mind while optimizing the efficiency of your cycle.

5 Method

The purpose of our study was to test whether we can achieve the same benefit with an implemented tutorial dialogue system as we obtained with the presence of a human tutor in the NPSG condition from [9].

Experimental procedure common to all conditions. The study consisted of a 3 hour lab session. We strictly controlled for time between conditions. The 3-hour lab session was divided into 9 segments: (1) After completing the consent form, students were given 15 minutes to work through an introductory exercise to familiarize themselves with the CyclePad software. (2) Students then had 15 minutes to work through a 50 point pre-test consisting of short answer and multiple choice questions covering basic concepts related to Rankine cycles, with a heavy emphasis on understanding dependencies between cycle parameters. (3) Students then spent 15 minutes reading an 11 page overview of basic concepts of Rankine cycles. (4) Next they spent 40 minutes working through the first of three focused materials covering the Basic Rankine Cycle. The materials included readings, suggested problem solving goals, and analyses to help in meeting those goals. (5) Next they spent 20 minutes working through the second set of focused materials, this time focused on Rankine Cycles with Reheat. (6) They then spent 20 minutes through the third set of focused materials, this time focusing on Rankine Cycles with Regeneration. (7) They then spent 10 minutes on each of two Free Exploration exercises, one of which was designed to test whether students learned how to fully define a Rankine cycle, and one of which was designed to test the student's ability to optimize a fully defined cycle. (8) They then spent 20 minutes taking a post-test that was identical to the pretest. (9) Finally, they filled out the questionnaire. The experimental manipulation took place during steps (4)-(6).

Experimental design. Our experimental manipulation consisted of 3 conditions. The only difference between conditions during the experimental manipulation was the version of the software the students used. In the control condition, students used the original CyclePad system. This was a replication of the script condition (S) from [9]. In the first experimental condition, students used a version of CyclePad augmented with feedback and help that were integrated with CyclePad using psuedotutors (PSHELP). The PSHELP condition was similar to the problem solving condition (PS) from [9] except that in addition to typical hints and feedback messages, dialogues in the form of Knowledge Construction Dialogues (KCDs) were attached to nodes related to KCD topics in such a way that if a student asked for help on that node, they would get the dialogue as the hint message. Thus, students only saw dialogues when they asked for help on nodes that had KCDs attached to them. In a second experimental condition, the same KCDs were attached to success messages on the same nodes so that students got the dialogues after they successfully completed an action or if they asked for help on that action (PSSUCCESS). In both experimental conditions, students only viewed each unique KCD once. If additional opportunities to view the same KCD came up, students instead were presented with a hint summarizing the message of the KCD.

Outcome Measures. We evaluated the effect of our experimental manipulation on three outcome measures of instructional effectiveness. One outcome measure was assessed by means of a Pre/Post test containing 32 multiple choice and short answer questions that test analytical knowledge of Rankine cycles, including relationships between cycle parameters. A domain expert associated each question on the test with the set of concepts related to the 12 authored KCDs discussed in Section 4 that the student would need to have a grasp on in order to correctly answer the problem. Using this topic analysis of the test, we can compute a concept specific score for each student on each test, and thus measure concept specific knowledge gain. Next there were two separate measures of practical knowledge, based on success at the two Free Exploration exercises from step 7 of the experimental procedure. For the Free Exploration 1 exercise where students were charged with the task of fully defining a Rankine cycle, they received a 1 if they were successful and 0 otherwise. For Free Exploration 2, the students were evaluated on their ability to optimize a fully defined cycle. Thus, their score for that exercise was the efficiency they achieved, as measured by the CyclePad simulator.

Participants. 31 students from a sophomore Thermodynamics course at Carnegie Mellon University participated in the study in order to earn extra credit. The study took place one and a half weeks after Rankine cycles were introduced in the lecture portion of their class. The study took place over two days, with two lab sessions on each day.

6 Results

The goal of our evaluation was to measure the value added of dialogue to the CycleTalk system. Our two experimental conditions present two different approaches to integrating dialogues with a version of CyclePad that was augmented with an intelligent tutoring framework that allowed students to ask for hints. Students had the opportunity to encounter two different types of dialogues. In particular, 5 dialogues covered basic knowledge about the concept of Reheat, the concept of Regeneration, and some basic knowledge about properly initializing cycle parameters prior to optimization. 7 additional KCDs covered specific topics related to interpreting sensitivity analyses and doing optimization based on the results.

As mentioned, the difference between the PSHELP condition and the PSSUCCESS condition was that students in the PSHELP condition only received KCDs in response to help requests whereas students in the PSSUCCESS condition also received KCDs as success messages after successfully completing a sensitivity analysis. Thus, the PSSUCCESS condition included more paths where students had the opportunity to encounter KCDs, although the system ensured that each full KCD was never viewed by the same student more than once. Students in the PSSUCCESS condition were significantly more likely to see each KCD than students in the PSHELP condition, as computed from logfile data using a binary logistic regression with an observation for each KCD (i.e., whether the student viewed that KCD or not during their experience with CyclePad) for each student in the two experimental conditions ($p < .05$). Students in the PSHELP condition only viewed a KCD when they asked for a hint. And in practice, students in the two experimental conditions did not ask for help frequently. Specifically, only about 14% of the problem solving actions of students were help requests. On average, students in the PSHELP condition viewed 1.8 KCDs (st. dev .837) whereas students in the PSSUCCESS condition saw 2.7 (st. dev. 1.9). The difference in coverage of KCDs between conditions was mainly on the KCDs related to sensitivity analyses. Only 1 of 7 KCDs focusing on interpreting sensitivity analyses was viewed by any student in the PSHELP condition, whereas in the PSSUCCESS condition all but one of these KCDs was viewed by at least one student. The difference between experimental conditions is interesting from the standpoint of evaluating the contribution of manipulating the number of KCDs viewed on learning. Nevertheless, it is a concern that so few of the authored KCDs were viewed by students on average even in the condition where they were viewed most frequently, and further increasing the number of opportunities for students to view KCDs is one of the goals of our continued work.

As mentioned above, the study took place over two days, with two lab sessions on each day. Two lab sessions on day 1 were assigned to the control condition (S). The first lab session on the second day was assigned to the first experimental condition (PSHELP). The final experimental condition took place during the second lab session on the second day (PSSUCCESS). We learned after the experiment was in progress that a quiz on Rankine cycles was administered to the class in between the lab sessions on the first day and the lab sessions on the second day. Thus, presumably because students were studying the day before the quiz, on average pretest scores increased from lab session to lab session such that there was a weak but significant correlation between lab session number and pretest score (R-squared = .14, $p < .05$, N=17). We expect that students on the second day when the experimental conditions took place were less motivated to learn the material than students on the first day since the quiz had already been given. Furthermore, since they had already studied, any learning that would take place would necessarily need to be on topics that remained difficult for the students even after studying. Finally, attendance in the first lab on the second day was lower than in the other conditions. Because of the interference of the quiz between the lab sessions where the control condition was conducted and the lab sessions where the two experimental conditions were conducted, we disregard the comparison between the control condition and the two experimental conditions and focus only on the difference between the two experimental conditions, although a summary of results from all three conditions is displayed in Table 1.

There was a significant effect of test phase $F(1,60) = 44.98$, $p < .001$, with no significant interaction with condition. Thus, students in all three conditions learned. It is impressive that the lab sessions on the second day when the experimental conditions

Table 1. Summary of results from all three conditions

Condition	Pretest Average Total	Posttest Average Total	FreeExplore 1 Success Rate	FreeExplore 2 Average Efficiency
S	20.64 (5.56)	31.39 (5.86)	23%	38.14 (10.97)
PSHELP	20.67 (3.56)	27.83 (6.02)	0%	38.09 (13.12)
PSSUCCESS	24.86 (4.10)	32.45 (4.06)	20%	34.09 (14.17)

were conducted yielded significant learning gains even though they were conducted on the same day as the quiz, which took place that morning. Because the difference in presentation of KCDs between the two experimental conditions is subtle, in order to increase the statistical power of the comparison, we evaluated the significance of the difference in learning between conditions using a repeated measures ANOVA, with a separate observation for each of the concepts the pre/post test was designed to test. The effect of condition on Concept Posttest Score with Concept Pretest Score used as a covariate and Concept as a fixed factor demonstrated a significant effect both of Concept, $F(10, 327) = 15.55$, $p < .001$, and of Condition, $F(2, 327) = 3.25$, $p < .05$, with no significant interaction. Thus students learned more about some concepts than others consistently across conditions. A pairwise Tukey test comparing learning between the PSHELP and PSSUCCESS conditions demonstrated significantly more learning in the PSSUCCESS condition, which is the condition where more KCDs were viewed ($P < .05$), effect size .35 standard deviations. Thus, manipulating the number of KCDs viewed had a significant positive effect on student learning, although there were no significant effects of condition on either of the FreeExploration exercises.

In a follow-up study with the same materials conducted at the US Naval Academy where we contrasted the S and PSSUCCSS conditions, we confirmed a significant effect in favor of the PSSUCCESS condition $F(1,86) = 5.57$, $p < .05$, effect size .25 standard deviations.

7 Discussion and Conclusions

In this paper we presented results from a study that demonstrate the instructional effectiveness of Knowledge Construction Dialogues (KCDs) modelled after the human tutors from the study previously published in [9]. Nevertheless, the system we evaluated in this study still falls far short of a full implementation of the NPSG condition from that study. In our current work we are exploring ways to increase the similarity between our implemented tutorial dialogue system and the behavior of the human tutors from the NPSG condition. In particular, the number of KCDs students view during their experience with the system still need to be increased by a factor of 2 or 3 to bring it more in line with the number of topics covered in discussions with the human tutors from the human tutoring study. Furthermore, while the content development for the KCDs evaluated in this study were motivated by an analysis of the human tutoring corpus from the previous study, they played more of a role of eliciting reflection from students rather than assisting with navigation to the same extent that the human tutors did.

Acknowledgements

This project is supported by ONR Cognitive and Neural Sciences Division, Grant number N000140410107.

References

1. Aleven, V, & Rosé, C. P. (2005). Authoring plug-in tutor agents by demonstration: Rapid rapid tutor development, *Proceedings of AI in Education '05.*
2. Bloom, B. S. (1984). The 2 Sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher*, 13:4–16.
3. Cohen, P. A., Kulik, J. A., and Kulik, C. C. (1982). Educational outcomes of tutoring: A meta-analysis of findings. *American Educational Research Journal*, 19:237–248.
4. Forbus, K. D., Whalley, P. B., Evrett, J. O., Ureel, L., Brokowski, M., Baher, J., Kuehne, S. E. (1999). CyclePad: An Articulate Virtual Laboratory for Engineering Thermodynamics. *Artificial Intelligence* 114(1-2): 297-347.
5. Graesser, A., VanLehn, K., the TRG, & the NLT. (2002). *Why2 Report: Evaluation of Why/Atlas, Why/AutoTutor, and Accomplished Human Tutors on Learning Gains for Qualitative Physics Problems and Explanations*, LRDC Tech Report, (2002) University of Pittsburgh.
6. Graesser, A. C., Jackson, G. T., Mathews, E. C., Mitchell, H. H., Olney, A., Ventura, M., Chipman, P., Franceschetti, D., Hu, X., Louwerse, M. M., Person, N. K., and the Tutoring Research Group, (2003). Why/AutoTutor: A Test of Learning Gains from a Physics Tutor with Natural Language Dialog. *Proceedings of the Cognitive Science Society*.
7. Koedinger, K. R., Aleven, V., Heffernan, N., McLaren, B., & Hockenberry, M. (2004). Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. In J. C. Lester, R. M. Vicario, & F. Paraguaçu (Eds.), *Proceedings of Seventh International Conference on Intelligent Tutoring Systems, ITS 2004* (pp. 162-174). Berlin: Springer Verlag.
8. Person, N., Bautista, L., Graesser, A., Mathews, E., & The Tutoring Research Group (2001). In J. D. Moore, C. L. Redfield, & W. L. Johnson (Eds.), *Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future, Proceedings of AI-ED 2001* (pp. 286-293). Amsterdam, IOS Press.
9. Rosé, C. P., Aleven, V., Carey, R., Robinson, A., Wu, C. (2005). A First Evaluation of the Instructional Value of Negotiatble Problem Solving Goals on the Exploratory Learning Continuum, *Proceedings of AI in Eduction '05.*
10. Rosé, C. P., Torrey, C., Aleven, V., Robinson, A., Wu, C. & Forbus, K. (2004). CycleTalk: Towards a Dialogue Agent that Guides Design with an Articulate Simulator, *Proceedings of the Intelligent Tutoring Systems Conference*.
11. Rosé, C. P., Jordan, P., Ringenberg, M., Siler, S., VanLehn, K., Weinstein, A. (2001). Interactive Conceptual Tutoring in Atlas-Andes, Proceedings of AI in Education 2001
12. VanLehn, K., Graesser, A., Tanner, J., Jordan, P., Olney, A. & Rosé, C. P. (2005). When is reading just as effective as one-on-one interactive tutoring? *Proceedings of the Annual Meeting of the Cognitive Science Society.*

Narrative-Centered Tutorial Planning for Inquiry-Based Learning Environments

Bradford W. Mott and James C. Lester

Department of Computer Science
North Carolina State University
Raleigh, NC 27695 USA
{bwmott, lester}@ncsu.edu

Abstract. Recent years have seen growing interest in narrative-centered learning environments. Leveraging the inherent structure of narrative, narrative-centered learning environments offer significant potential for inquiry-based learning in which students actively participate in engaging story-based problem-solving. A key challenge posed by narrative-centered learning is orchestrating all of the events in the unfolding story to motivate students and promote effective learning. In this paper we present a narrative-centered tutorial planning architecture that integrates narrative planning and pedagogical control. The architecture continually constructs and updates narrative plans to support the hypothesis-generation-testing cycles that form the basis for inquiry-based learning. It is being used to implement a prototype narrative-centered inquiry-based learning environment for the domain of microbiology. The planner dynamically balances narrative and pedagogical goals while at the same time satisfying the real-time constraints of highly interactive learning environments.

1 Introduction

Narrative is central to human cognition. Because of the motivational force of narrative, it has long been believed that story-based education can be both engaging and effective. Much educational software has been devised for story-based learning. These systems include both research prototypes and a long line of commercially available software. However, this software relied on scripted forms of narrative: they employed either predefined linear plot structures or simple branching storylines. In contrast, one can imagine a much richer form of narrative learning environment that dynamically crafts customized stories for individual students at runtime. Recent years have seen the emergence of a growing body of work on dynamic narrative generation [4, 21, 23], and narrative has begun to play an increasingly important role in intelligent tutoring systems [10, 20].

Narrative offers significant potential for inquiry-based learning. In inquiry-based learning, the student iterates through cycles of questioning, hypothesis generation, data collection, and hypothesis testing. In a narrative-centered inquiry-based learning environment, the student could be featured as the central character in a dynamically generated story. She would be presented with problems to solve, and the plot would be shaped in such a way that she would at pedagogically appropriate times "discover" evidence confirming or disconfirming her hypotheses.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 675 – 684, 2006.

Narrative-centered learning environments for inquiry-based learning should satisfy three requirements. First, they should strike a delicate balance between advancing the plot and achieving tutorial goals. The former cannot be ignored without making the narrative less engaging and coherent; the latter cannot be ignored without reducing pedagogical effectiveness. Second, narratives should be customized for individual students. Plots driven by students' problem-solving activities should be tightly coupled to hypothesis-generation-testing cycles to create the best possible learning outcomes. Third, narrative generation must interleave planning and execution to satisfy the real-time requirements of highly interactive learning environments. Because of the complexities of narrative planning and the dynamic tutorial state on which it depends, narratives must be planned incrementally, plans must be monitored, and they must be revised as the storyworld and tutorial state change.

This paper introduces a narrative-centered tutorial planning architecture for inquiry-based learning environments. The architecture integrates narrative planning and tutorial control via a hierarchical task network (HTN) planner [6] that operates in two coordinated planning spaces. In the tutorial planning space, the planner constructs tutorial plans to achieve pedagogical goals such as topic sequencing, problem introduction, problem solving, and advice generation. In the narrative planning space, the planner constructs narrative plans to achieve story goals such as how to direct the characters' actions, how to devise coherent plots, and how to create engaging experiences for the student. The dual planning space approach achieves modularity for authoring and maintenance of plan operators, and it enables the planner to guide the student's actions at both the pedagogical and narrative levels. The architecture is being used to implement CRYSTAL ISLAND, a prototype inquiry-based learning environment for the domain of microbiology. Preliminary experience with the narrative planner and the learning environment suggests that the architecture can effectively balance pedagogical and narrative goals, create customized narratives for individual students, and interleave planning and execution.

2 Narrative-Centered Inquiry-Based Learning

Narrative experiences are powerful. In his work on cognitive processes in narrative comprehension, Gerrig identifies two properties that readers of narrative experience [5]. First, readers are transported, i.e., they are somehow taken to another place and time in a manner that is so compelling it seems real. Second, they perform the narrative. Like actors in a play, they actively draw inferences and experience emotions as if their experiences were somehow real. It is becoming apparent that narrative can be used as an effective tool for exploring the structure and process of "meaning making." For example, narrative analysis is being adopted by those seeking to extend the foundations of psychology [3] and film theory [2].

Learning environments may utilize narrative to their advantage. One can imagine narrative-centered curricula that leverage a student's innate metacognitive apparatus for understanding and crafting stories. This insight has led educators to recognize the potential of contextualizing all learning within narrative [25]. Because of the active nature of narrative, by immersing learners in a captivating world populated by intriguing characters, narrative-centered learning environments can enable learners to participate in the construction of narratives, to engage in active problem solving, and to reflect on narrative experiences [15]. These activities are particularly relevant to inquiry-based learning. *Inquiry-based learning* emphasizes the student's role in the learning process via concept building [27] and hypothesis formation, data collection, and testing [7]. For example, a narrative-centered inquiry-based learning environment for science education could foster an in-depth understanding of how real-world science plays out by featuring science mysteries whose plots are dynamically created for individual students.

Historically, learning effectiveness has functioned as the sole metric by which learning environments are gauged. However, from a practical perspective, it has become clear that educational software that fails to engage students will go unused. In Malone's classic work on motivation in computer games and educational software [11], he distinguishes between game playing experiences (and educational experiences) that are extrinsically motivating and those that are intrinsically motivating. In contrast to extrinsic motivation, intrinsic motivation stems from the desire to undertake activities sheerly for the immediate pleasure to be derived from them. By emphasizing qualities such as challenge, curiosity, and fantasy, Malone argues that learning environments can create intrinsically motivating experiences [12]. Hence, in addition to their potential cognitive benefits, narrative-centered learning also offers significant potential for providing intrinsic motivation.

Several projects have begun to explore narrative-centered learning environments. Some have begun to devise powerful models of ITSs that can be informed by narrative [20]. Of particular interest here are approaches that enable children to be creative storytellers in collaborative, play-oriented environments [10]. Narrative-centered learning environments have also been employed in the service of creative writing [22], story creation [14], second language learning for training applications [8], individualized video-based lesson planning customized to particular students [9], social behavior education [1], and problem-solving for health education [13]. The narrative-centered tutorial planning architecture proposed in this paper seeks to bridge the gap between tutorial planning and narrative planning in order to provide story-based problem-solving experiences that complement those explored in other efforts in narrative-centered learning.

3 Narrative-Centered Tutorial Planning Architecture

The narrative-centered tutorial planning architecture directs all of the core activities of the learning environment (Figure 1). All student activities are mediated through the interface manager for the virtual environment. The interface manager interacts with the world model, which houses the 3D object and character models, the properties of manipulable objects, and the scene geometries. The world model drives both the rendering and sound engines. The planner consists of three components: a tutorial planner, a narrative planner, and a plan executor and monitor. The tutorial planner operates in the tutorial planning space. It utilizes domain knowledge, curriculum constraints, tutorial strategies, and concept difficulty annotations to make its decisions. The narrative planner operates in the narrative planning space. It utilizes a library of plot elements, a library of character behaviors, a set of world event catego-

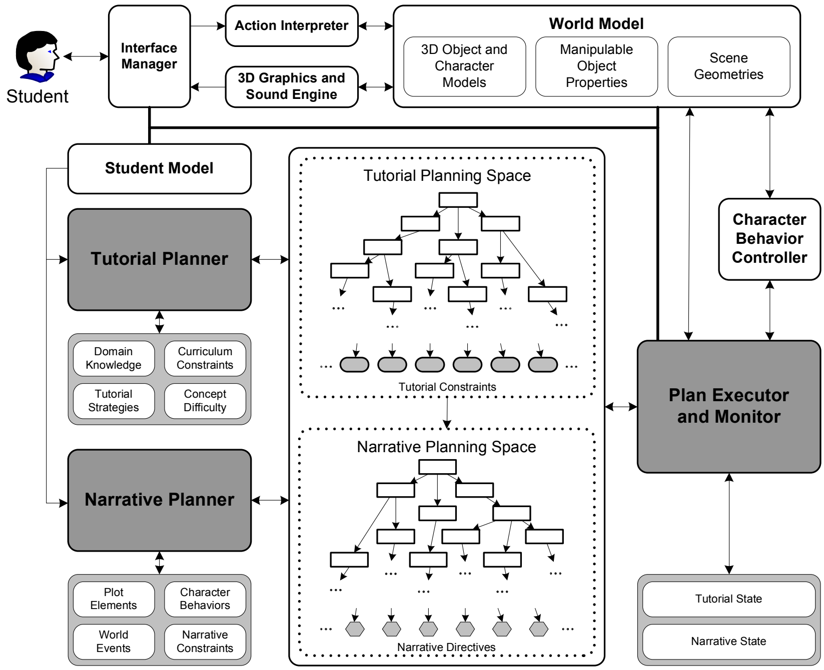


Fig. 1. Narrative-Centered Tutorial Planning Architecture

ries, and narrative constraints on possible stories to make its decisions. The plan executor and monitor interact with both the tutorial and narrative planning spaces. It sends directives to the character behavior controller and the world model. All three planning components are influenced by the student model, and the plan executor and monitor also reads from (and updates) the tutorial and narrative states.

The narrative-centered tutorial planning architecture provides all of the functionalities that classic tutorial planners provide, as well as the functionalities that narrative planners provide. With regard to tutorial planning, it selects and presents problems, sequences content from the curriculum, provides timely and context-specific advice and explanations, manages the initiative, and selects and executes tutorial strategies [17, 19, 24, 26]. To address the requirements of inquiry-based learning, its tutorial strategies support question formation, hypothesis generation, data collection, and hypothesis testing. With regard to narrative planning, it generates all plot elements, sequences plot elements into coherent and engaging stories, and directs characters' actions and storyworld events to achieve tutorial and narrative goals.

Dynamically reasoning about narrative-centered tutorial strategies is inherently a planning problem. Planning has been used for tutorial strategy formation and execution [19] and for narrative generation [4, 21]. A multitude of formalisms have been developed for automatically constructing plans to achieve a given set of goals or tasks [6]. However, planning is a challenging problem: in the worst case it is worse than NP-complete. While many prototypes have been devised over the decades, it has been challenging to create planners that function well in practical applications. Hierarchical task network (HTN) planners [6] have found a much broader use in fielded systems than competing approaches. If one has application-specific knowledge about a problem, then it can be incorporated into an HTN planner. HTN planners employ a problem reduction approach that supports reasoning about constraints, resolving interactions, and backtracking to alternate decompositions if necessary [6]. Because HTN planning can be very efficient with sufficient application knowledge – this is the case with both tutorial planning and narrative planning – the narrative-centered tutorial planning architecture utilizes an HTN planning system based on SHOP2 [18].

In addition to being expressive and efficient, it is critical that tutorial planning and narrative planning support one another: they cannot be permitted to diverge. All (or most) tutorial goals should be realized through plot elements, and all (or most) plot elements should be generated in support of tutorial goals. Although some learning might occur through non-narrative means, e.g., providing textual and animated explanations external to the story, the student should remain immersed in the story to the greatest extent possible. Moreover, although engaging story events could be created that served no tutorial purpose, the interests of pedagogy must drive the student's experience.

Two distinct approaches can be taken to reasoning about tutorial and narrative planning in a manner that ensure that tutorial and narrative planning are mutually supportive. One approach uses a single planning space and the other uses two planning spaces. In the *single planning space* approach, tutorial methods, operators, preconditions, and effects are scattered throughout one planning space. In this approach, a single method can have preconditions on both tutorial goals and narrative goals, and the effects of operators can be on both tutorial states and narrative states. However, such an approach requires the construction of methods and operators that are difficult to author and maintain. By intermixing tutorial and narrative predicates and effects throughout a single planning space, modularity is violated and expanding HTN libraries become increasingly difficult as domain complexity grows.

In the *dual planning space* approach, one planning space is allocated to tutorial planning and a second is allocated to narrative planning. This approach offers the advantage of modularity: narrative planning issues can be considered separately from tutorial planning issues. However, for the two planners to work in concert, they must effectively coordinate their actions, which will result in a single stream of events occurring in the virtual storyworld. To this end, the tutorial planner posts goals in the tutorial planning space that are achieved by operators in the narrative planning space. Thus, appropriate customized narratives can be generated in the service of pedagogical objectives.

One can distinguish three alternate coordination models for communication between the two planning spaces. First, in the *parallel* model, the two planning spaces could operate "side-by-side." Tutorial goals could be posted and achieved by the tutorial planner while narrative goals could be posted and achieved by the narrative planner. However, resolving inconsistencies between the sequences of actions suggested by the two planners would be very challenging and would not scale to larger, more complex domains. Second, in the *narrative-driven* model, the narrative planner could post goals in the narrative planning space that would be achieved by operators in the tutorial planning space. While such an approach might produce coherent and engaging narratives, these would at times be produced at the cost of effective learning. Third, in the *tutorial-driven* dual planning space model, the tutorial planner could post goals in the tutorial planning space that would be achieved by operators in the narrative planning space. Here, appropriate customized narratives would be generated in the service of pedagogical objectives. The architecture thus adopts the tutorial-driven model.

The *tutorial planning space* houses all concepts, goals, methods, and operators for reasoning about the student's learning experience, as well as the tutorial state. These encode domain (subject matter) knowledge, curriculum sequencing constraints represented as a partial order on concepts, the student model (the current implementation uses a simple overlay student model), and difficulty annotations on concepts. They also encode inquiry-based learning strategies that guide hypothesis-generation-testing cycles. HTN methods represent decompositions of higher level tutorial goals to lower level tutorial goals. All HTNs eventually bottom out in tutorial constraints, which collectively guide narrative planning and focus it on the most relevant regions of the narrative planning space that are consistent with the current tutorial plan.

The *narrative planning space* houses all goals, methods, and operators for reasoning about the storyworld. These encode plot construction knowledge, character behaviors, storyworld event categories, and narrative constraints, including coherence and flow constraints. All narrative HTNs eventually bottom out in primitive narrative events, which play out in the storyworld. These primitive events are directives that will be physically interpreted in the virtual environment.

In narrative-centered tutorial planning, tutorial and narrative plans can be very complex. Moreover, the specifics of plans are highly dependent on the current tutorial and narrative state, which are highly dynamic and are themselves highly dependent on the actions of the student in the storyworld. It would therefore be infeasible for the planner to attempt to construct fully specified tutorial and narrative plans. Planning and execution must be interleaved at runtime to permit replanning as needed. Planning is initiated when top-level tutorial goals are posted. It operates in four highly interleaved phases of operation:

- *Plan construction*: During construction, HTN methods, operators, preconditions, and effects are instantiated, and the methods recursively invoke lower-level methods. Construction in the tutorial space builds a full set of conceptual and inquiry-based problem-solving constraints. Construction in the narrative space creates the plot and character behaviors.
- *Plan execution*: During execution, narrative operators drive events in the storyworld. (Tutorial operators are not directly executed *per se*; rather, the tutorial constraints are used to guide the selection of narrative HTNs and their instantiation.)
- *Plan monitoring*: During monitoring, the planner tracks all activities in the world. It checks for unanticipated violations of preconditions brought about by changes in the tutorial and narrative states. These changes result from actions taken by the student in the storyworld.
- *Replanning*: During replanning, the planner uses the current tutorial and narrative states to modify the current plan so that the preconditions of upcoming methods and operators (i.e., methods and operators that are as of yet unexecuted) will be reestablished. Sometimes replanning in the narrative space causes a cascading of replanning in the tutorial space.

The HTN-based narrative-centered tutorial planning architecture operates in the four phases, incrementally constructing and executing plans while continuously monitoring the tutorial and narrative states and replanning as necessary until all of the tutorial goals have been achieved and student completes the interactive story.

4 An Implemented Narrative-Centered Tutorial Planner

The narrative-centered tutorial planning architecture is being used to implement CRYSTAL ISLAND, a prototype inquiry-based learning environment for the domain of microbiology being constructed for middle school students (Figure 2). CRYSTAL ISLAND features a science mystery set on a recently discovered volcanic island where a research station has been established to study the unique flora and fauna. The user plays the protagonist attempting to discover the origins of an unidentified infectious disease at the research station. The story opens by introducing her to the island and the members of the research team for which her father serves as the lead scientist. As members of the research team fall ill, it is her task to discover the cause of the outbreak. She is free to explore the world and interact with other characters while forming questions, generating hypotheses, collecting data, and testing her hypotheses. Through the course of her adventure she must gather enough evidence to correctly choose among candidate diagnoses including botulism, cholera, giardiasis, paralytic shellfish poisoning, salmonellosis, and tick paralysis as well as identify the source of the disease.

The narrative-centered tutorial planning architecture has been implemented with an HTN planner that is based on the SHOP2 planner [18] and was constructed in our laboratory to meet the specific needs of narrative and tutorial planning. For efficiency, it was designed as an embeddable C++ library to facilitate its integration into high-performance 3D gaming engines. The implementations of the tutorial planner and the narrative planner were both built with the custom HTN planner. Their respective planning spaces have well defined APIs that support appropriate accesses but establish modularity. The virtual world of CRYSTAL ISLAND, the semi-autonomous characters that inhabit it, and the user interface were implemented with Valve Software's Source™ engine, the 3D game platform for Half-Life 2. The Source engine also provides much of the low-level (reactive) character behavior control. The character behaviors and artifacts in the storyworld are the subject of continued work. The tutorial and narrative planners are fully implemented, a decision-theoretic "director" agent based on dynamic decision networks has been implemented to guide the narrative in the face of uncertain student actions [16], and the method and operator libraries for the microbiology domain are currently being built out.

To illustrate the behavior of the narrative-centered tutorial planning architecture within the CRYSTAL ISLAND learning environment, consider the following situation. A student has been interacting within the storyworld and learning about infectious diseases and related topics. In the course of having members of her research team become ill, she has learned that an infectious disease is an illness that can be transmitted from one organism to another. As she concludes her introduction to infectious diseases, she learns from the camp nurse that the mystery illness seems to be salmonellosis and that the source of the disease must be identified. The narrative planner has de-

Fig. 2. CRYSTAL ISLAND Learning Environment

cided to have her pursue satisfying the tutorial constraints associated with the *Spread-of-Infectious-Diseases* topic by constructing a plan which has the unfolding story involve the spread of a disease by means of contaminated food. Specifically, it chooses salmonellosis as the illness and contaminated eggs as the source of the bacterial infection. Although, the student has made steady progress while learning about infectious diseases, the task of identifying the source of the illness has left her wandering aimlessly around the storyworld to locate the source. As the execution and monitoring components of the system assess the unfolding story, it determines that the student's progress towards identifying the origins of the illness is lagging. To address this, the narrative planner updates the narrative plan to include a hint action realized in the camp nurse revealing that she believes the source of the disease is something that the victims ate.

Experience to date suggests that tutorial and narrative planning libraries are relatively straightforward to construct. First, as others have found, HTN-based plan methods and operators seem to be intuitively authored, even with fairly complex planning tasks. We anticipate that the modularity inherent in HTN's "recipes" will support the complexities to be faced in building out the domain. Second, separating the activities of the architecture into two distinct but coordinated spaces seems to be essential for efficient authoring; other approaches that were considered early in the project were discarded because of the resulting complexity. Third, the practicalities of real-time planning call for a planning formalism (and system) that could easily be integrated with other components and operate with high efficiency. HTN planners can more easily incorporate domain-specific knowledge than classic STRIPS-style planners. This property can be leveraged by narrative-centered learning environments such as CRYSTAL ISLAND, which are highly interactive.

5 Conclusion

The HTN-based narrative-centered tutorial planning architecture addresses the three requirements set forth above for inquiry-based learning environments. First, it balances plot advancement and tutorial goal achievement seamlessly by the built-in coordination of the two planning spaces via the lower-level tutorial constraints and the upper-level narrative goals. Second, it customizes narratives for individual students by basing both tutorial and narrative planning on the student model and tutorial state. Third, it interleaves planning and execution by operating in the four phases of construction, execution, monitoring, and replanning; it satisfies the real-time performance requirements through the efficiency provided by HTN planning.

The architecture represents a first step towards inquiry-based learning environments that offer students effective and engaging problem-solving experiences in rich, interactive storyworlds. It suggests several lines of promising work. Of particular interest here are investigating approaches for incorporating models of affect that support students, both with respect to their interactions with characters in the virtual world and their problem-solving activities *per se*. It will also be important to incorporate much more expressive student modeling techniques that can be used for plan recognition in "narrative diagnosis," i.e., how to most accurately predict a student's current goals given the openness of the learning environment but exploiting the model of narrative on which the unfolding story is based. Finally, it will be important to develop a precise, empirically grounded understanding of the kinds of problem-solving interactions that narrative can most effectively promote in inquiry-based learning.

References

1. Aylett, R., Louchart, S., Dias, J., Paiva, A., and Vala, M. 2005. FearNot! – An Experiment in Emergent Narrative. In *Proceedings of the Fifth International Conference on Intelligent Virtual Agents*, 305-316. Kos, Greece.
2. Branigan, E. 1992. *Narrative Comprehension and Film. London*: Routledge.
3. Bruner, J. 1990. *Acts of Meaning*. Cambridge, MA: Harvard University Press.
4. Cavazza, M., Charles, F., and Mead, S. 2002. Interacting with Virtual Characters in Interactive Storytelling. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 318-325. Bologna, Italy.
5. Gerrig, R. 1993. *Experiencing Narrative Worlds: On the Psychological Activities of Reading*. New Haven: Yale University Press.
6. Ghallab, M., Nau, D., and Traverso, P. 2004. *Automated Planning: Theory and Practice*. Morgan Kaufmann.
7. Glaser, R., Schauble, L., Raghavan, K., and Zeitz, C. 1992. Scientific reasoning across different domains. In E. De Corte, M. C. Linn, H. Mandle, and L. Verschaffel (eds.), *Computer-Based Learning Environments and Problem Solving*, 345-373. Berlin: Springer.
8. Johnson, L., Beal, C., Fowles-Winkler, A., Narayanan, S., Papachristou, D., Marsella, S., and Vilhjálmsson, H. 2004. Tactical Language Training System: An Interim Report. In *Proceedings of the Seventh International Conference on Intelligent Tutoring Systems*, 336-345. Maceió, Alagoas, Brazil.
9. Luckin, R., Underwood, J., du Boulay, B., Holmberg, J., and Tunley, H. 2004. Coherence Compilation: Applying AIED Techniques to the Reuse of Educational Resources. In *Proceedings of the Seventh International Conference on Intelligent Tutoring Systems*, 98-107. Maceió, Alagoas, Brazil.
10. Machado, I., Brna, P., and Paiva, A. 2001. Learning by Playing: Supporting and Guiding Story-Creation Activities. In *Proceedings of the Tenth International Conference on Artificial Intelligence in Education*, 334-342. San Antonio, Texas.

11. Malone, T. 1981a. Toward a Theory of Intrinsically Motivating Instruction. *Cognitive Science* 5(4):333-369.
12. Malone, T. 1981b. What makes computer games fun? *Byte* 6:258-277.
13. Marsella, S., Johnson, L., and LaBore, C. 2000. Interactive Pedagogical Drama. In *Proceedings of the Fourth International Conference on Autonomous Agents*, 301-308. Barcelona, Spain.
14. Marshall, P., Rogers, Y. and Scaife, M. 2002. PUPPET: A Virtual Environment for Children to Act and Direct Interactive Narratives. In *Proceedings of the Second International Workshop on Narrative and Interactive Learning Environments*. Edinburgh, Scotland.
15. Mott, B., Callaway, C., Zettlemoyer, L., Lee, S., and Lester, J. 1999. Towards Narrative-Centered Learning Environments. In *Proceedings of the 1999 Fall Symposium on Narrative Intelligence*, 78-82. Cape Cod, MA.
16. Mott, B. and Lester, J. 2006. U-DIRECTOR: A Decision-Theoretic Narrative Planning Architecture for Storytelling Environments. To appear in *Proceedings of the Fifth International Conference on Autonomous Agents and Multi-Agent Systems*, Hakodate, Japan.
17. Murray, W. 1989. Control for Intelligent Tutoring Systems: A Blackboard-based Dynamic Instructional Planner. In *Proceedings of the Fourth International Conference on Artificial Intelligence and Education*, 150-168. Amsterdam, Netherlands.
18. Nau, D., Muñoz-Avila, H., Cao, Y., Lotem, A., and Mitchell, S. 2001. Total-Order Planning with Partially Ordered Subtasks. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*. Seattle, WA.
19. Peachey, D. and McCalla, G. 1986. Using Planning Techniques in Intelligent Tutoring Systems. *The International Journal of Man-Machine Studies* 24(1):77-98.
20. Riedl, M., Lane, H., Hill, R., and Swartout, W. 2005. Automated Story Direction and Intelligent Tutoring: Towards a Unifying Architecture. In *Proceedings of the Workshop on Narrative Learning Environments at the Twelfth International Conference on Artificial Intelligence in Education*, 23-30. Amsterdam, Netherlands.
21. Riedl, M. and Young, M. 2004. An Intent-Driven Planner for Multi-Agent Story Generation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 186-193. New York.
22. Robertson, J. and Good, J. 2003. Ghostwriter: A narrative virtual environment for children. In *Proceedings of the 2003 International Conference on Interaction Design and Children*, 85-91. Preston, England.
23. Si, M., Marsella, S., and Pynadath, D. 2005. Thespian: Using Multi-Agent Fitting to Craft Interactive Drama. In *Proceedings of the Fourth International Conference on Autonomous Agents and Multi-Agent Systems*, 21-28. Utrecht, Netherlands.
24. Vassileva, J. and Wasson, B. 1996. Instructional Planning Approaches: from Tutoring towards Free Learning. In *Proceedings of the European Conference on Artificial Intelligence in Education*, 1-8. Lisbon, Portugal.
25. Wells, C. 1986. *The Meaning Makers: Children Learning Language and Using Language to Learn*. Portsmouth, NH: Heinemann.
26. Woolf, B. and McDonald, D. 1984. Building a Computer Tutor: Design Issues. *IEEE Computer* 17(9):61-73.
27. Zachos, P., Hick, L., Doane, W., and Sargent, C. 2000. Setting Theoretical and Empirical Foundations for Assessing Scientific and Discovery in Educational Programs. *Journal of Research in Science Teaching* 37(9):938-962.

Using Simulated Students for the Assessment of Authentic Document Retrieval

Jonathan Brown and Maxine Eskenazi

Language Technologies Institute, Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, PA 15213
jonbrown@alumni.cmu.edu, max@cs.cmu.edu

Abstract. In the REAP system, users are automatically provided with texts to read that are targeted to their individual reading abilities and needs. To assess such a system, students with different abilities use it, and then researchers measure how well it addresses their needs. In this paper, we describe an approach using simulated students to perform this assessment. This enables researchers to determine if the system functions well enough for the students to learn the curriculum and how factors such as corpus size and retrieval criteria affect performance. We discuss how we have used simulated students to assess the REAP system and to prepare for an upcoming study, as well as future work.

1 Introduction

The CMU Language Technologies Institute REAP project has created software to find appropriate authentic documents for students learning to read their native language or a new language. In this paper, we focus on system assessment using simulated students, artificial models of student behavior consistent with real-world human students.

Simulated students have primarily been used by teachers practicing tutoring, students working with simulated peers, and developers testing products to get precise feedback without human subjects [1]. REAP falls into the third category. The major difference between our simulated students and those in the literature concerns student modeling techniques. Many simulated student implementations rely on production rule and machine learning techniques, focusing on modeling student knowledge [1]. System performance is measured by analyzing that knowledge for completeness and correctness. Although we place importance on correctly modeling student knowledge, evaluating the system's effects on student curriculum progression is more central. Our simulated students determine, given a curriculum and various operating parameters, how far students can progress within that curriculum in a given amount of time. In REAP there are several factors which affect a student's progress, such as document retrieval criteria and document corpus size. Although these factors are specific to REAP, we believe our student simulation methods are also applicable to other domains. We begin by describing REAP and the various factors affecting student progress. We then describe why we need simulated students, their implementation, their use in preparing for an upcoming study, and future work.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 685 – 688, 2006.

2 The REAP System

In today's classrooms, students often read prepared texts. Since these texts are not authentic, students are not exposed to the real language used in everyday written communication. With prepared texts, all students read the same text. Students who are having trouble have no remediation texts, and those who are ahead cannot advance more quickly. REAP supplies Web texts that are both authentic and personalized to individual reading abilities. Because mastery of core vocabulary is essential to developing more complex reading comprehension skills [2][3], students' abilities are presently modeled at the lexical level. REAP lets students quickly acquire the desired level of lexical mastery by automatically assessing un-mastered vocabulary [4] and updating the student knowledge representation before choosing the next document. This enables REAP to have two goals. First, REAP is a framework that presents students with texts matched to their own reading levels. Second, REAP enables researchers to test hypotheses on improving the vocabulary skills of language learners. Thus the system supports specific criteria for the documents retrieved: grade level, as determined by the readability component developed by Collins-Thompson and Callan [5]; amount of certain terms unknown to the student, as determined by our student modeling [6][7].

A number of factors affect progression through the curriculum. Concerning corpus size, the larger the corpus, the more selective REAP can be during document selection. For document selection criteria, such as grade level and length, requiring documents at 7th grade level results in fewer than also allowing 6th grade documents. REAP also supports document ranking criteria. It can choose documents containing words for which a question was recently answered incorrectly. Finally, question responses inform the student model, and determine progress and what text to select next.

3 Simulating Student Performance

Given the factors that affect student progress in the curriculum, we want to know what happens if we add documents to the corpus or if administrators change restrictions. For example, do we have enough documents to complete a given curriculum with a given set of criteria? This can be answered by using student simulation, using data from prior student use to make predictions about future use of the proposed system.

3.1 Implementation

In order to simulate student activity, REAP needs to predict student response to the automatically generated vocabulary questions. To predict that response, we implemented all-correct (AC), percent-wrong (PW), and conditional-probability (CP) simulated students. The AC simulated student simply gets all questions correct. Although this is not realistic, since few students perform this way, this simulated student is very useful. A student who always answers correctly may not read as many documents as a less-accurate student, because, at any point in time, REAP has less choice of un-mastered words to focus on. In our studies, AC represents the worst case.

The PW simulated student randomly chooses when a question will be answered correctly, such that the total percentage of questions incorrect is approximately equal to the percentage incorrect in past studies. PW simulations are more accurate than AC simulations but are still not realistic models of human behavior. The CP student addresses PW student shortcomings by conditioning performance on other features, such as the number of times a question has already been answered about this word. The likelihood of getting a word wrong after previously answering one or more questions about it is different than for no preceding questions. Like PW, CP's response is trained on student data and randomized according to this probability. The degree to which a student's response can be conditioned depends on the amount of existing data. In case of sparse data, CP students use averages for a desired feature. Other features we could use are whether the word was previously answered correctly and the likelihood of a specific student answering correctly. We have not yet conditioned on a specific student's performance since students change from one study to the next. CP more realistically mimics human behavior, which researchers identify as crucial to model validity, while factors such as behavior moderators remain to be addressed [8].

A simulation driver can run simulations for different amounts of time. For instance, we might use the AC student to simulate the next 10 documents a given student would be expected to read. More often, we predict the documents that might be read to complete an entire curriculum and determine if there is enough time during the semester for this activity. Because there is a random aspect to PW and CP, the simulations are run multiple times to get a better estimation. Using REAP's logging and reporting capabilities, we can determine how far a student progressed through the curriculum. We also run simulations with various document retrieval criteria, corpus sizes, and student curricula to determine whether the criteria are too restrictive for the curriculum to be completed in the desired amount of time.

3.2 Assessment of System and Preparation for Study

Our simulations answer two questions. First, how far can a student progress through a curriculum? Second, how do various factors affect this progression? We will use a study using REAP in an English-as-a-second-language (ESL) class at the English Language Institute (ELI) of the University of Pittsburgh to show how we answer these questions. Each student has a personalized set of vocabulary items (focus words) to learn. The system finds documents that meet certain criteria and contain subsets of the student's un-mastered focus words. After a reading, the students answer questions used to update their student models. Simulations were run using AC as a worst case measure as well as CP simulations, and used several sets of focus words.

Could the students complete their curricula in a set period of time, given the experimenters' criteria? Simulations showed it was not possible to complete the curriculum with the existing document corpus and criteria. So, more targeted web crawls were carried out to augment the corpus and new simulations showed students were able to progress through the curriculum. Although the system didn't have enough documents for the whole curriculum, we fulfilled the study requirements. We were also able to make the criteria stricter by reducing the grade level range, improving document quality. Finally, we ran simulations with various sets of focus words and criteria to determine how these factors affected curriculum progression.

4 Conclusions and Future Work

We describe using simulated students in the REAP project to predict student curriculum progress. Various types of simulated students are supported, most trained on data from human students. These simulations were especially useful in preparing to use REAP for reading studies in an ESL classroom. The researchers often requested criteria modifications, and we ran simulations to test their effects.

In the future, we plan to simulate the process of choosing a document to read in addition to answering questions. We also plan to explore how this work could be applied to other domains and tasks. We believe similar simulations would be useful to determine whether there is sufficient curriculum content for a more traditional intelligent tutor with insufficient data to build a traditional simulated student model. In addition, we plan to update our simulated students as we extend the student modeling capabilities of REAP beyond vocabulary and to incorrect or buggy student knowledge.

This project is supported by Award #R305G030123 Funded by the U.S. Department of Education, Institute of Education Sciences. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Department of Education.

References

1. VanLehn, K., Ohlsson, S., Nason, R.: Applications of simulated students: An exploration. In: Journal of Artificial Intelligence in Education (1994)
2. Perfetti, C.A., Britt, M.A., Georgi, M.: Text-based learning and reasoning: Studies in history. Hillsdale, NJ: Erlbaum (1995)
3. Perfetti, C.A., Hart, L.: The lexical quality hypothesis. In: Vehoeven, L., Elbro, C., Reitsma, P. (Eds.): Precursors of functional literacy.: Amsterdam/Philadelphia (2001)
4. Brown, J., Frishkoff, G., Eskenazi, M.: Automatic question generation for vocabulary assessment. In: Proceedings of HLT/EMNLP 2005. Vancouver, B.C. (2005)
5. Collins-Thompson, K., Callan, J.: A language modeling approach to predicting reading difficulty. In: Proceedings of the HLT/NAACL 2004 Conference. Boston (2004)
6. Brown, J., Eskenazi, M.: Student, text and curriculum modeling for reader-specific document retrieval. In: Proceedings of the IASTED International Conference on Human-Computer Interaction. Phoenix (2005)
7. Brown, J., Eskenazi, M.: Retrieval of Authentic Documents for Reader-Specific Lexical Practice. In: Proceedings of InSTIL/ICALL Symposium 2004. Venice, Italy (2004)
8. Ritter, F., Avraamides, M.N., Councill, I.G.: Validating Changes to a Cognitive Architecture To More Accurately Model the Effects of Two Example Behavior Moderators. In: Proceedings of the Eleventh Computer-Generated Forces and Behavior Representation Conference. Orlando, FL (2002)

Designing a Tutoring Agent for Facilitating Collaborative Learning with Instant Messaging

Sheng-Cheng Hsu[1], Min-Yuh Day[3,4], Shih-Hung Wu[5] , Wing-Kwong Wong[2], and Wen-Lian Hsu[3]

[1] Graduate School of Engineering Science and Technology
[2] Graduate School of Computer Science and Information Engineering,
National Yunlin University of Science and Technology, Yunlin, Taiwan
[3] Institute of Information Science, Academia Sinica, Taiwan
[4] Department of Information Management, National Taiwan University, Taiwan
[5] Department of CSIE, Chaoyang University of Technology, Taichung, Taiwan
wongwk@yuntech.edu.tw

Abstract. In this study, we propose a tutoring agent that uses MSN Messenger, a popular synchronous internet media, as the communication platform. Students can invite the agent to a discussion session on MSN Messenger. The agent understands the students' questions in natural language, and provides answers or hints during the group discussion. Unlike a traditional natural language tutoring agent that converses with one student at a time, our agent needs to work with a group of students in MSN chat.

1 Introduction

With the advancement of internet technology, students with distance education can use synchronous internet media to communicate and collaborate with one another. Common synchronous media include chat room, video conference and instant messaging. Among these media, Instant Messaging (IM) is the most popular, e.g., AOL's Instant Messenger, ICQ, MSN Messenger, and Yahoo! Messenger. Some research [1, 2] indicates that IM is a useful tool to carry out distance collaborative learning. Students discuss and exchange ideas with peers simultaneously. During collaborative discussion among a group, a tutor plays the role of information provider when the students have questions. The tutor answers students' questions directly or provides relevant hints. However, it is still impossible for human tutor to stay online for 24 hours a day. Without the presence of a tutor, students often spend too much time searching for answers in order to make progress in a discussion. In this study, we propose a tutoring agent that uses MSN Messenger as the communication platform. It provides a convenient platform for collaborative discussion, recording collaborative process, and sending electronic files. Up to now, this agent is used in an assembly language programming course. Some research has shown the effectiveness of collaborative programming in computer science courses [3]. Therefore, in this study an assembly language programming course is the topic of distance collaborative learning on MSN Message.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 689 – 691, 2006.

2 The Collaborative Programming Process on MSN Message for Assembly Language

In this study, the collaborative learning process for assembly language programming is based on the Learning Together method and problem solving strategies. There are the following seven steps: grouping heterogeneous students, assigning a problem, understanding the problem, making a programming plan, coding a program, testing and debugging the program, and showing the result.

To start collaborative learning, the instructor first divides students with varying ability into groups of four. Then the instructor assigns each group a project, for example, “Checking Password Project: design a program to read and check a 10-bit password from keyboard.” After the group-building activity, the students implement the collaborative programming steps on MSN message. When the students within a group have questions during the discussion, they can invite the tutor agent into the discussion on the MSN Messenger. The students can use natural language to consult with the agent that can provide answers or hints. At the step of understanding the problem, the students try to clarify the problem definition by discussing the following questions: (1) What must the program do? (2) What outputs are required and in what form? (3) What inputs are available and in what form? (4) What are the pre-condition and post-condition? If they don’t understand the problem description, the students can ask the agent. For example, “Does the 10 bit password contain blank bit?” In the programming plan making step, the students within a group need to break down a task into a number of small subtasks until each subtask is easily doable. If the students cannot design a feasible plan, the agent can provide a general plan to them.

At the step of coding a problem, the students need to code a program in assembly language according to the programming plan. The agent can provide means, format and examples if they have any question about the assembly language instruction, for example, if a student asks the agent, “What is interrupt INT 21H with AH=01?” In the testing and debugging step, the students try to execute the program and find out errors, such as syntax error, run time error, and logic error. Through the file transmission function of MSN Message, each group can combine the program files edited by each group member. Moreover, the agent will provide the explanation of the error message. In the step of showing the result, the students compile the result as a report, which includes programming definition, project plan, requirement specification, acceptance test plan, and user's manual.

3 System Architecture

The collaborative tutoring agent has four components: user interface, comprehension module, tutoring module, and student module. In this study, MSN Messenger is used as the user interface, which includes contact list, collaborative discussion, and management interface. To use the tutoring agent, the student has to add the agent contact ID: ita_yun@hotmail.com to the MSN Messenger contact list. Then the students can invite the tutoring agent into the chat in discussion. The chat of MSN Message is used as the collaborative discussion interface for students, and the tutoring agent provides help during the collaborative learning. The management interface is implemented based on a class library, dotMSN. It provides an interface for managing the tutoring agent such as login information, contact list, conversion records, etc.

The comprehension module is used to understand students' questions in natural language and is implemented with the INFOMAP knowledge engineering tool provided by the Intelligent Agent System Lab, Institute of Information Science, Academia Sinica [4]. INFOMAP has been successfully applied to areas such as question answering and intelligent tutoring [5]. A powerful feature of INFOMAP is its capability to represent and match complicated template structures, such as hierarchical matching, regular expression, semantic template matching, and frame (non-linear relations) matching used to extract important concepts from a natural language text. The tutoring module stores the problems' answers involving the lexicon knowledge retrieved from the textbook, the plan knowledge for designing a program, and the hint knowledge for guiding the students to finish this team-project. The student module is used to record the students' portfolio in the collaborative leaning process. The students' portfolio records the discussion process and the questions students ask the agent. On MSN Message, it's easy to store the portfolio in text files.

4 Conclusion

In this study, a tutoring agent for facilitating collaborative learning with instant messaging is proposed. This agent uses MSN Messenger as the communication platform. Students can invite the agent to a discussion on MSN Messenger. The agent understands the students' questions in natural language, and provides answers or hints during the discussion. So far it has been applied to assembly language programming courses. When students get involved in the collaborative programming process, the agent provides help, including understanding problem definition, designing a programming plan, and understanding error messages during debugging.

Acknowledgements

This research is supported by the National Science Council in Taiwan., W-K Wong and S-C Hsu are supported by NSC94-2524-S-224-001 and NSC 94-2520-S-224-001, S-H Wu is supported by NSC94-2218-E-324-003, while M-Y Day and W-L Hsu are supported by NSC94-2752-E001- 001-PAE.

References

1. Nardi, B. A., Whittaker, S., Bradner, E.: Interaction and outeraction: instant messaging in action. The ACM Conference on Computer Supported Cooperative Work (2000) 79-88.
2. Texley S., DeGennaro D. Should We Ban Instant Messaging in School? Learning & Leading with Technology, Vol. 32. (2005) 6-8
3. McDowell, C., Wener, L., Bullock, H., Fernald, J.: The effects of pair- programming on performance in an introductory programming course. The 33rd SIGCSE Technical Symposium on Computer Science Education, 38-42. (2002)
4. Hsu, W. L., Wu, S. H., Chen, Y. S.: Event identification based on the information map – INFOMAP. The 2001 IEEE Systems, Man, and Cybernetics Conference, Tucson, Arizona, USA, (2001) 1661-1672
5. Wong, W. K., Hsu, S. C., Wu, S. H., Lee, C. W., Hsu, W. L.: LIM-G: Learner-initiating instruction model based on cognitive knowledge for geometry word problem comprehension, Computers & Education, Elsevier (In press)

An Approach of Learning by Demonstrating and Tutoring a Virtual Character

Shan-Chun Chi and Chih-Yueh Chou

Department of Computer Science and Information Engineering,
Yuan Ze University, Chung-Li, Taiwan, ROC
s946052@mail.yzu.edu.tw

Abstract. This study purposes an approach of computer-supported learning by teaching. The learning activities of learning by teaching a virtual character are usually involving demonstrating and tutoring. Here we combine demonstrating and tutoring into a teaching virtual apprentice activity. First, student demonstrates solving problems to the apprentice. Then student will monitor and tutor apprentice when the apprentice solves problems. We will try to explore effects of different teaching activities that are demonstrating, tutoring, and demonstrating combine with tutoring.

1 Introduction

The idea of learning by teaching has been proposed by many researchers; for example, in 1531 Valentine Trotzendorf argued that the best way to learn was to teach (Briggs, 1998). Whiteman (1988) also said "To teach is to learn twice". In addition, many studies have shown that learning by teaching is a helpful learning activity (Biswas & Leelawong, 2005). Furthermore, the study showed that the tutee's deep questions will benefit tutor (Roscoe & Chi, 2004). In general, peer tutoring is a common way to realize the idea of learning by teaching, that is, a student plays the role of a tutor to tutor another student which plays the role of a tutee. However, in peer tutoring, the responses of the tutee are various and may not be beneficial to the tutor. Instead of a real student tutee, a tutee which is played by a computer simulated virtual character can be designed to respond to be beneficial to the tutor.

There are several computer supported learning by teaching systems which use virtual characters to play the role of a tutee. For examples: RTS system (Chan & Chou, 1997; Wong *et al.* 2003), DENISE (Nichols, 1994), PALs (Scott & Reif, 1999), LECOBA (Ramírez Uresti, 1999; Ramírez Uresti & du Boulay, 2004), STEPS (Ur & Vanlehn, 1995) and Betty's Brain (Biswas & Leelawong, 2005). In these systems teaching activities are various and probably can be divided into two classifications: demonstrating and tutoring (Table 1). Some systems enable students to teach a virtual tutee by demonstrating some examples or knowledge. In other systems, students teach a virtual tutee by tutoring, that is, the students monitor, correct and guide the virtual tutee's problem solving.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 692 – 694, 2006.

Table 1. Two classifications of virtual character enhanced learning by teaching systems

Demonstrating	Tutoring
Betty's Brain, DENISE	LECOBA, PALs, RTS, STEPS

2 Using Apprenticeship Approach to Combine Demonstration and Tutoring

Existed virtual character enhanced learning by teaching systems enable students either to learn by demonstrating or to learn by tutoring. Demonstration forces students to practice or present the knowledge. Tutoring makes students observe, analyze, and guide the tutee's problem solving. Although RTS and PALs systems engage students in practicing knowledge and tutoring by playing the role of a tutee and a tutor in turns, students practice and tutor in different problems. We try to let students learn by both demonstrating and tutoring in the same learning task or problem. Thus we adopt apprenticeship, which involves demonstration and tutoring, as teaching activity (Collins, Brown & Newman, 1989). First, master demonstrates or presents knowledge to apprentices (Collins *et al.* term this activity as modeling), that is, master executes process repeatedly and apprentices observe. Then apprentices will try to execute the same process and master will tutor apprentices, such as giving some helps or advices when apprentices meet some problems (Collins *et al.* term this activity as coaching). So, we use apprenticeship to naturally combine demonstration and tutoring to engage students in learning by preparing, learning by doing, and learning by tutoring. At beginning, students need to study the material. While they were told that they need to teach someone, they could learn better by preparing to teach (Bargh & Schul, 1980). Then students will demonstrate to apprentice thus students do the learning task at last once. If students found that they can not demonstrate correctly, they need to study the material again. After demonstrating, apprentices will practice themselves and students tutor them to solve problems or to present knowledge.

3 Experiment Design

In order to explore effects of different teaching activities involving demonstrating and tutoring, we will proceed an experiment after implementing a system supporting learning by demonstrating and tutoring a virtual tutee. The procedure of experiment is designed as follows. First, all the students will be asked to do a pre-test. Then they will be informed that they will teach someone after they studied the material. This is to make sure that every student will achieve the "learning by preparing" activity. All participants will be divided into four groups based on different teaching activities (Table 2). First group is control group and students continue to study the material. The students of second group participate in learning by demonstrating to a virtual character. The students of third group engage in learning by tutoring a virtual character. The students of fourth group employ demonstration and tutoring to teach a virtual character. After the teaching process, students will be asked to do a post-test. Then we will analyze the grades of pre-test and post-test to find out if there is obvious difference on learning performance while adopting different teaching activities.

Table 2. Four groups of experiment

Teaching activities			
No teaching activity (control group)	Demonstrating	Tutoring	Demonstrating + Tutoring

In this experiment, we suppose that students in the forth group will have better post-test grades. However, we need to explore the real effects of different teaching activities. Besides of comparing performance, we will investigate the behaviors of students during the teaching activities; for examples, whether students can demonstrate the solutions correctly during demonstration and whether students find the apprentice's errors during tutoring. Furthermore, students will be asked to fill in a questionnaire to investigate their feelings and cognition during these teaching activities. The experiment will be held in May 2006 and preliminary results will be reported on the conference.

References

Bargh, J., & Schul, Y. (1980). On the cognitive benefits of teaching. *Journal of Educational Psychology*, 72 (5), 593-604.

Briggs, D. (1998) A Class of Their Own – When Children Teach Children, Bergin & Garvey.

Biswas, G., & Leelawong, K. (2005). Learning by teaching: a new agent paradigm for educational software. *Applied Artificial Intelligence*, 19 (3), 363-392.

Chan, T.W. & Chou, C.Y. (1997). Exploring the design of computer supports for reciprocal tutoring. *International Journal of Artificial Intelligence in Education*, 8, 1-29.

Collins, J.S. Brown, & S.E. Newman, (1989). "Cognitive apprenticeship: teaching the craft of reading, writing, and instruction," Essays in honor of Robert Glaser, Hillsdale, NJ: Lawrence Erlabum Associates Publishers.

Nichols, D. (1994). Issues in designing learning by teaching systems, *In Proceedings of the East-West International Conference on Computer Technologies in Education (EW-ED'94)*, 176-181.

Ramírez Uresti, J.A. (1999). LECOBA: A LEarning COmpanion system for binary Boolean Algebra. In Johnson, L. (Ed.), Proceedings of Workshop 1: *Animated and Personified Pedagogical Agents AI-ED'99 conference*, 56-61. Le Mans, France.

Ramirez Uresti, J. and du Boulay, B. (2004). Expertise, Motivation and Teaching in Learning Companion Systems. *International Journal of Artificial Intelligence in Education,* 14, 193-231.

Roscoe, R.D. & Chi, M.T.H. (2004). The influence of the tutee in learning by peer tutoring. *To appear in the 26th Annual Meeting of the Cognitive Science Society*, Chicago, IL.

Scott, L. A. & Reif, F. (1999). Teaching Scientific Thinking Skills: Students and Computers Coaching Each Other. *The 9th International Conference on Artificial Intelligence in Education (AI-ED 99)*, Le Mans, France, 285-293

Ur, S. & VanLehn, K. (1995). STEPS: A Simulated, Tutorable Physics Student. *Journal of Artificial Intelligence in Education*, 6 (4), 405-435.

Whiteman, N.A. (1988) Peer teaching: to teach is to learn twice. Washington, DC: *Asscition for the Study of Higher Education*

Wong, W. K., Chan, T. W., Chou, C.Y., Heh, J. S., & Tung, S. H. (2003). Reciprocal tutoring using cognitive tools, *Journal* of *Computer Assisted Learning*. 19 (4), 418-428.

e-Learning System to Provide Optimum Questions Based on Item Response Theory

Takahiro Murase[1] and Yukuo Isomoto[2]

[1] Nursery Department, Chukyo Junior College,
Toki-cho 2216, Mizunami-shi, Gifu-ken, 509-6101, Japan
murase@j-chukyo.ac.jp
[2] Division of Economics and Information, Gifu Shotoku Gakuen University,
Nakauzura 1-38, Gifu-shi, Gifu-ken, 500-8288, Japan
yisomoto@gifu.shotoku.ac.jp

Abstract. With the goal of providing instruction with an optimum method to achieve the target given in the learning process, PID control theory was applied and an assessment was made of the proportion(P), integral(I), and differential(D) of the learner's level of understanding. Using these parameters, a web server system was constructed as a teaching model adopting learning rules extracted from optimum problems. Using the item response theory, problem characteristics were analyzed by obtaining the discrimination power and difficulty level of each problem, and each learner's ability was also estimated. Characteristic functions of one set of tests were identified by combining the problem characteristics obtained. These functions were incorporated in the Matlab simulation software, and the relation between the difficulty level of problem determined by manipulation using PID control and the learner's ability was obtained. With this simulation, a method was constructed to optimally arrange learning materials based on the hierarchical learning theory.

1 Introduction

With the application of automatic control theory and other aspects of systems engineering in the field of educational technology, we formulated mathematical model for high level educational strategies, in order to provide problems or explanations suited to the understanding of individual learner and implemented this model on web server using computer technology. Test characteristic functions are obtained through item response theory for the discrimination power and difficulty level that are characteristics of a problem, and through classification and combination the probability of a correct answer according to the learner's ability can be obtained. In addition, using the "theory of learning" that examined the learning behaviors of learners, these functions are applied to modeling of learning support systems. Together with the use of multimedia technology, these are beneficial as a learning support. We treated the arithmetical computation and a certification examination for information technology engineer as a case for the evaluation.

2 Hierarchical Learning Theory and Item Response Theory

Gagne's hierarchical learning theory is fundamental to methods of hierarchical structuring of learning tasks. According to this theory, the primary significance of the hi-

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 695 – 697, 2006.

erarchy is to identify prerequisites that should be completed to facilitate learning at each level. In item response theory, the probability of a correct answer to any given question is expressed as a function of the characteristic value to be measured. These functions express the features of questions, and are called item characteristics functions. Thus, the item characteristics function is a probability density function determined from the probability of the correct answer $Pj(x)$ for item j against learner's ability x. If the item characteristic function is determined, then the probability of correct and incorrect answers can be identified, and used to decide the characteristics of the test as a whole.

This system introduces a 3-parameter logic model incorporating a pseudo-chance level for the possibility of a correct answer if the answer is as supposed for multiple-choice type problems, and a 2-parameter logic model for non-selection problems. BILOG-MG Ver. 3 was used in the estimation of each parameter. An item information function is obtained with this item characteristic function, to express which level of learner can distinguish correct and incorrect answers with which level of acumen. Similarly, a test information function of problem groups formed with problems having these characteristics can be obtained. Considering advice by computer matched to these trends, the learner can be provided with a strategy to overcome learning impasses.

3 Analysis of PID Parameter Characteristics

A characteristic of feedback systems is that response speed and response accuracy are generally antagonistic: if emphasis is placed on response speed, the response series vacillates and response accuracy declines, increasing the degree of instability. Thus, if problems that are easy given the learner's ability are continued, the time until the goals are achieved lengthens. Conversely, if the problems are too difficult, rapid results drop off and the level of difficulty of problems vacillates.

To analyze the characteristics of such systems, we developed a dynamical system simulation model using learner's transfer function expressed with a differential equation. The time needed to reach the target level was minimized by the system evaluation. Next, let us consider a model of learner characteristics. For problem difficulty, we obtain the test information function using the item response theory mentioned above, and put it into MATLAB. With reference to the item response theory and item score sheet, in the learning model of the learner we used a model expressed with a differential equation in which the ability level of various learners is expressed as time (number of times) on the horizontal axis, and the level of learning on the vertical axis.

Fig.1 shows the models thus far explained, designed with the simulation software called Matlab/Simulink. The learner's character model is seen above, while the PID control model for the learner control is shown below. The learner model outputs the score using the understanding level of learner at the point in time when he addresses the difficulty level.

The upper part in Fig.2 presents in graphic form the understanding level of the learner following the exercise N repeat. At that time, the understanding level of the learner is evaluated by the difficulty level of the exercise following the N repeat. The Fig.2 on the left indicates when only the Kp parameter is moved, and with difficulty level of exercise for the 10th, 15th and 30th exercise, respectively. The larger Kp parameter can be made to achieve a higher level more quickly but it oscillates at over 10.

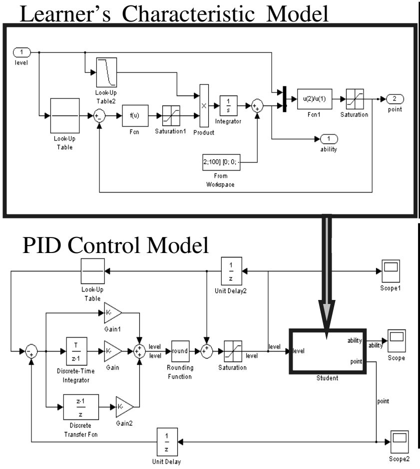


Fig. 1. PID Control Model

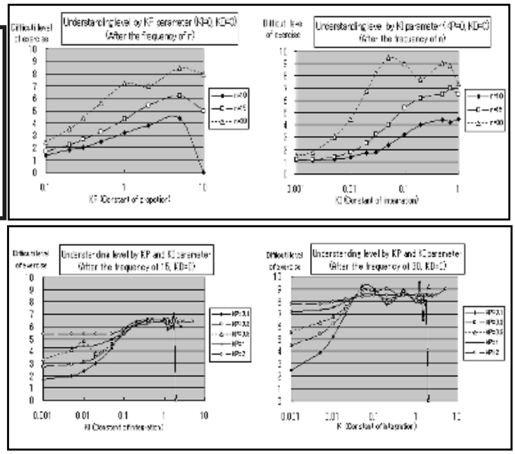


Fig. 2. PID Parameter Characteristic by Simulation

This is because the problems given are too difficult and inappropriate in relation to the learner's level of understanding. The right-hand figure also indicates when only the Ki parameter is moved, but oscillation occurs with more than Ki=1. Likewise, the lower part on Fig. 2 shows the understanding level of the learner after 15 and 30 exercises with both Kp and Ki parameters moving. Since the two parameters of Kp and Ki are moved, each parameter can be used over a wide range, enabling a higher rate of response. Also, the range of the non-oscillating Ki is expanded, and stability is obviously higher as well.

4 Conclusion

As examples of this method, we developed web server system for the arithmetic computation and a certification examination for information technology engineer, but systems can also be developed for other fields using the same method. An environment in which PID parameters could be analyzed and evaluated was established, and the characteristics of each parameter were analyzed. In the future, their appropriateness will need to be verified.

References

1. Anderson, J.R. (1985). Cognitive psychology and its implications. New York: Freedman.
2. Brown, J.S. & VanLehn, K. (1980). Repair Theory: A generative theory of bugs in procedural skills. Cognitive Science, 4, 379-426.

BEEweb: A Multi-domain Platform for Reciprocal Peer-Driven Tutoring Systems

Ari Bader-Natal and Jordan B. Pollack

DEMO Lab, Brandeis University
Computer Science Department MS018
Waltham MA 02454. USA
{ari, pollack}@cs.brandeis.edu

Abstract. Tutoring systems that engage each student as both a tutee and a tutor can be powerfully enhanced by motivating each tutor to try to appropriately challenge their tutee. The BEEweb platform is presented as a foundation upon which to build such systems, based upon the Reciprocal Tutoring protocol and the Teachers Dilemma. Three systems that have recently been built on the BEEweb platform are introduced.

1 Introduction

Educational research on peer tutoring has shown beneficial effects on the achievement and attitude of the tutors and tutees [1]. These peer-driven methodologies have more recently been introduced into the arena of Intelligent Tutoring Systems [2]. Our SpellBEE system incorporated a reciprocal tutoring approach, in which each participating student is engaged both as tutor and tutee [3]. Over 12,000 people have used SpellBEE during its first two years online at `http://SpellBEE.org`. This work has informed the development of a platform upon which tutoring systems could be built to leverage the Reciprocal Tutoring protocol and the Teachers Dilemma. This platform, called BEEweb, was designed to enable the rapid development of highly-scalable new tutoring systems that require minimal domain expertise to prepare. Here, we describe the core protocols of the BEEweb platform and three recently-released tutoring systems in different task domains built on this platform.

2 Reciprocal Tutoring and the Teachers Dilemma

All web-based tutoring sessions using BEEweb systems have a common structure: Initially, each student is presented a list of pseudonyms of other currently-available students and must indicate whom they are willing to be matched with. Mutual interest between two students initiates a match between them consisting of a fixed number of rounds of interaction. The structure of these rounds defines the Reciprocal Tutoring (RT) protocol. Each round is a four-step process that the pair of students synchronously progress through. From the student's perspective, one round consists of the following steps:

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 698–700, 2006.

1. Student assumes the **Tutor** role by preparing a challenge for their Tutee.
2. Student assumes the **Tutee** role by preparing a response to their Tutor.
3. Student is given feedback about the response that they prepared in Step 2.
4. Student is given feedback about how their Tutee responded to the challenge that they prepared in Step 1.

While the RT protocol[1] specifies the structure and progression of the interactions among a pair of students, it makes no attempt to motivate or influence the student's actions when preparing challenges and responses. The Teachers Dilemma refines this RT protocol (TD-RT) with an extrinsic motivational mechanism biasing the tutor towards selecting *appropriate challenges* [3]. This is accomplished by adding the following constraints:

a The difficulty of any challenge in the task domain can be estimated.
b The accuracy of a response to a challenge in the domain can be assessed.
c The feedback provided in Step 3 and Step 4 is supplemented with a role-specific reward. Acting as Tutee, the student is rewarded for response accuracy; acting as Tutor, the student is rewarded for selecting challenges that reveal the tutee's strengths and weaknesses (see [3] for more details.)

3 BEEweb Tutoring Systems

Each BEEweb tutoring system applies the TD-RT protocol to a different task domain by uniquely specifying the domain's *challenge* and *response* structures, the user interface *toolkits* for interacting with these structures, and the challenge *difficulty estimators*.[2] Each new BEEweb system is introduced accordingly.

PatternBEE (`http://PatternBEE.org`) focuses on a spatial-reasoning task, loosely based on Tangram puzzles, in which a challenge is an outline of a geometric shape, and a response is an arrangement of available pieces attempting to fill one such outline. The challenge toolkit consists of a space into which the tutor drags, rotates, flips the pieces. The response toolkit is similar, but also presents an outline of the target goal shape. PatternBEE estimates the difficulty of a challenge based on the number of pieces required and perimeter of the outline.

MoneyBEE (`http://MoneyBEE.org`) focuses on a coin-based elementary algebra task, in which a challenge characterizes a set of coins by its combined value and number of coins, and a response is a guess at how many of each type of coin was being described. The challenge toolkit consists of stacks of coins, from which the tutor selects some number of quarters, dimes, nickels, and pennies. The response toolkit is similar, but also states the challenge (in terms of number of coins and combined value.) Challenge difficulty estimates are based on the number of steps required for a heuristic search algorithm to reach a solution.

[1] The protocol can be further elaborated as follows: In Steps 1 and 2, the challenge and the response are each either selected from a list of options or constructed from a suitable toolkit. In Steps 3 and 4, the feedback generally includes the challenge posed, the response submitted, and the correct response to the challenge.

[2] Our own student programmers do this now, and we expect to release a public API.

GeograBEE (`http://GeograBEE.org`) focuses on a geographical knowledge domain, in which a challenge contains a question about one of the states in the U.S. Three categories of questions are currently deployed: (a) identify the capital city in the specified state, (b) locate the specified state on a map of the U.S., or (c) identify a state by an illustration of its boundaries. The challenge toolkit is divided into two steps: first choosing a state on the map about which to pose a question, and then selecting one of the three types of questions to ask about that state. The response toolkit for the identification-based questions (categories a and c) states the question in multiple-choice form, from which the tutee must make a selection. The response toolkit for the location-based questions (category b) state the question and display a map, upon which the tutee clicks to respond. Challenge difficulty estimation in GeograBEE currently takes into account the specific state and question category selected.

All of these BEEweb systems have been deployed to publicly-accessible websites, each of which has been instrumented to collect action and interaction data. Now that these websites are being used by students, both in and out of the classroom, we are beginning to accumulate this data for analysis.

4 Challenges and Future Work

The initial research aim of the BEEweb was to convincingly determine whether the TD-RT protocol can serve as a principled basis for peer learning while requiring minimal curriculum domain expertise and content costs. Many new research challenges and opportunities presented themselves once the platform was built. For example, we are working on replacing the hand-built difficulty estimators with adaptive ones based on statistical analysis of student match data, using techniques similar to Conejo et al. [4]. We are also experimenting with ways of using collected data to further enhance the tutoring experience (by making portions accessible to the student and their parents, teachers, and peers.) Finally, we are developing programming tools for others to develop and contribute their own reciprocal tutoring systems to the BEEweb.

References

1. Cohen, P.A., Kulik, J.A., Kulik, C.L.C.: Educational outcomes of tutoring: A meta-analysis of findings. American Educational Research Journal **19**(2) (1982) 237–248
2. Chan, T.W., Chou, C.Y.: Exploring the design of computer supports for reciprocal tutoring. International Journal of Artificial Intelligence in Education **8** (1997) 1–29
3. Bader-Natal, A., Pollack, J.B.: Motivating appropriate challenges in a reciprocal tutoring system. In Looi, C.K., McCalla, G., Bredeweg, B., Breuker, J., eds.: Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology. Volume 125 of Frontiers in Artificial Intelligence and Applications. IOS Press (2005) 49–56
4. Conejo, R., Guzmán, E., Millán, E., Trella, M., Pérez-De-La-Cruz, J.L., Ríos, A.: Siette: A web-based tool for adaptive testing. International Journal of Artificial Intelligence in Education **14** (2004) 29–61

A Multimedia Tool for English Learning Using Text Retrieval Techniques on DVD Subtitles

Jia-Hong Lee[1], Jia-Shin Cheng[1], and Mei-Yi Wu[2]

[1] Department of Information Management, Kun Shan University
[2] Department of Information Management, Chang Jung University
Tainan, Taiwan, R.O.C
jhlee@mail.ksu.edu.tw

Abstract. In this paper, a multimedia tool named *Supplementary Multimedia Material Provider* (SMMP) is proposed. To extract relevant sentences from subtitles of audiovisual, learners input keywords which enable that the sentence containing the target vocabulary repeatedly be exposed. The SMMP employs text retrieval techniques using vocabulary concept similarity measurement and followed by displaying the found sentences in various ways like text, audio and video clips. Since not every single movie covers the proper content for every viewer for the intention of learning. The SMMP also provides the lexical level and category statistical analysis for viewers. These analyses are very useful as the criteria for language learners in selecting appropriate movies as their learning materials.

1 Introduction

In recent years, DVD movies are very popular in colleges as the audiovisual for learning foreign language. Many researchers have shown that video materials can promote the motivation of learning foreign languages. Students really enjoy watching movies to get exposure to natural language in a non-threatening setting. In addition, movies and videos provide common ground to students of any international background. For these reasons, it is attractive for the learners to use the audiovisual for learning language; just imagine that you can have both general teaching resources as well as fully-developed lessons on various films so that you can make yourself some popcorn and settle down in front of your computer for some fun with movies and learning the language. Lin[1] looked at the incorporation of DVD films into a two-semester general English class for non-majors at a Taiwan university. Their study addresses that: Was it feasible to use DVD films as the major course material in a university level listening and speaking course for English majors? The answer was yes from the teacher's point of view after one semester. But the biggest drawback is the time it takes to develop the materials. In addition, once materials are developed, they can be used again in subsequent classes. As with any materials, sometime they do not work well at first and need improving or modification to use with different levels.

We try to develop a multimedia tool to help teachers and learners for fast obtain materials from different DVD films to achieve their needs.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 701 – 703, 2006.

2 The Proposed Method

The tool, called Supplementary Multimedia Material Provider (SMMP), is designed to self-access language learning. Teachers or learners can select DVD movies for English learning by themselves. The SMMP provides the lexical level and category statistical analysis for the selected movie captions.

To describe the method of the SMMP, it will be helpful to imagine a particular setting. We assume a learner who has bought or rent an interesting DVD film and want to learn English from the contents of the DVD repeatedly. During the video playing process, if some target vocabularies appeared on the English caption, he needs to stop playing and find the meanings of the vocabularies from a dictionary. In addition, when he wants to repeat currently passing sentence, he will suffers to moving the control scroll bar on the DVD player to extract position. These drawbacks will waste the learner's time and reduce the fun of language learning.

To improve the drawbacks for language learning caused by using general DVD player software, a user friendly DVD language learning interface in The SMMP is designed. The architecture of the SMMP is shown is Figure 1. The learning interface is displayed in Figure 2.

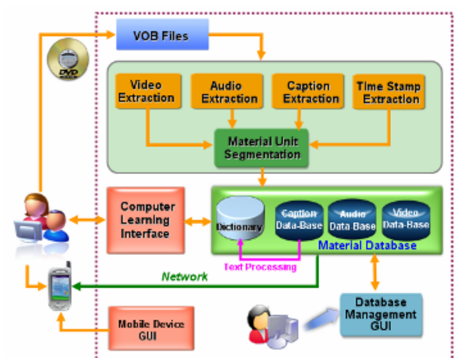


Fig. 1.

Fig. 2.

Figure 1: The architecture of the proposed DVD language learning framework
Figure 2: The learning interface of SMMP

Three main functions are provided by the SMMP.

(1) Multimedia material extraction from DVD films and material database construction with sentence by sentence.

The file sizes of DVD films are very large. Some video compression software is required to convert the original DVD VOB file format into mpeg4 or avi compressed format. We used a commercial OCR package to extract the captions and another one to convert the audio file into mp3 format. The captions and audio are segmented into clips according to the timestamps of captions and these clips are put into database for searching. The database can accumulate a huge of material data extracted from different DVD films.

(2) Vocabulary search and concept based similar keyword search
Research on reading and on vocabulary acquisition suggest the value of providing supplementary readings which offer learners repeated exposure to target vocabulary in context[2]. By using text retrieval technique, two approaches are provided to find clips from database. One matches keywords to find the sentences in the database. The other indexes a concept categorized dictionary[3] and list the similar keywords. The dictionary categorizes 30,000 English words into 130 groups of related meanings with definitions. Similar vocabulary searching function is indispensable for English writing, translation and vocabulary learning.

(3) Lexical level and category statistical analysis for captions
There are about 1200 sentences extracted from the captions of a DVD movie. The lexical level in the sentences can be analyzed using text retrieval techniques [4]. After removing the stopwords in the extracted movie caption, we compute the frequency of keywords which are appeared on the referenced vocabulary table of Taiwan General English Proficiency Test (GEPT)[5]. In addition, we also draw a radar map to show the vocabulary categories according to the keyword characteristic distribution according to the categorized dictionary .

3 Conclusions

We have propose a multimedia tool named *Supplementary Multimedia Material Provider* (SMMP). The SMMP employs text retrieval techniques using vocabulary concept similarity measurement and followed by displaying the found sentences in various ways like text, audio and video clips. Since not every single movie covers the proper content for every viewer for the intention of learning. The SMMP also provides the lexical level and category statistical analysis for viewers. These analyses are very useful as the criteria for language learners in selecting appropriate movies as their learning materials.

References

1. H.C. Liou, J.E. Katchen, and H. Wang, Teaching a Listening and Speaking Course with DVD Films: Can It Be Done?, Lingua Tsing Hua,, pp. 221-236 Taipei: Crane, 2003.
2. D. Wible, C.H. Kuo, F.Y. Chien and N.L. Taso, Automating Repeated Exposure to Target Vocabulary for Second Language Learners, IEEE International Conference on Advanced Learning Technologies, pp. 127 – 128, Aug. 2001
3. Ricardo Baeza-Yates and Berthier Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley Inc., 1999
4. Longman Lexicon of Contemporary English (English-Chinese), Longman Press
5. http://www.gept.org.tw/, The Language Training & Testing Center in Taiwan.

A Profiling System and Mobile Middleware Framework for U-Learning

Seong Baeg Kim, Kyoung Mi Yang, and Cheol Min Kim

Department of Computer Education, Cheju National University,
66 Jejudaehakno, Jeju-si, Jeju-do, 690-756, Korea
{sbkim, kmyang, cmkim}@cheju.ac.kr

Abstract. With recently increased emphasis on learner-centered education, a u-learning system, which can provide learners with a learning at any time in any place, has been becoming the focus of attention. Also, the advance of technology enables learners to have easier access to internet through PDA, thus to acquire information needed to learn on the spot at the moment without time or space constraints. Based on the current trend of the technology, we focus on how learners obtain and access information suitable for them. We shall propose a mobile middleware framework and profiling system, which take into account circumstances of a learner with a mobile device, specifications on a mobile device, and the learner's level or preferences for individualized u-learning. The middleware system is devised to make use of the information collected from RFID reader for outlining the learning environment of the learner.

1 Introduction

The trend in education has been changing from teacher-oriented learning activities to learner-centered ones which learners search, process, and make proper use of the information or knowledge they need for themselves. Accordingly, the ways of learning has been switching from passive learning to active one, from collective learning to individual one.

In the course of this paper, we shall suggest an RFID middleware framework and profiling system regarding mobile devices for u-learning learners. Our system provides a u-learning server with the data collected from the PDA, RFID reader, and a learner. The information serves as a basis for customized learning corresponding with each learner's learning style, level, and goal. In other words, a server can create a learning environment suitable for a specific learner and reorganize properly it to meet his/her demands and environmental changes.

2 Profiling System and Middleware Framework

We describe the profiling system and RFID middleware framework to provide an elaborate individualized u-learning using an RFID reader attached to a PDA device. The profiling system is made up of two components; server, which offers agents connected to database, and client, which provides a mobile learning user interface.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 704 – 706, 2006.

PDA is used as a mobile device, which satisfies such requirements as RFID information, and a user's interface environment considering a mobile device's level.

The mobile learner profile gathered in this way will be sent to a server through a communication module. Some of the data profiled are saved in the learner's database at server side, and some of them are sometimes changed in the learning process. With the database of learner's profile, a profile agent manages learner profiles which should be kept through many learning sessions. It receives such information as the learning level, preferences, goals, hour and location of each learner from the mobile device and constructs a customized u-learning environment for the learner [1,2]. Learning materials are saved and managed in a learning database, and metadata about learning content such as relations between learning materials and knowledge structure are kept in a learning management database. A learning content agent plays a role of a search engine for the two databases. A customized learning agent extracts a diagnostic questionnaire or optimal learning content.

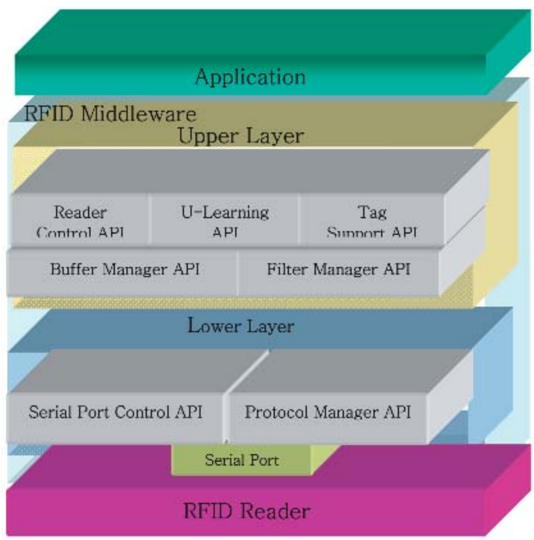


Fig. 1. The RFID system framework

Figure 1 shows the overall system framework based on RFID reader. Specially, we propose the RFID middleware framework for developing u-learning application. The RFID middleware consists of two layers : upper layer and lower layer. The upper layer provides the APIs to be used in application program such as reader control API, u-learning API, and so on. The lower layer has the functionality to communicate with RFID reader.

The scenario for learning in our system is at the following. A learner (assume that he/she is L) wants to learn using a PDA at some place. As the first step, Learner L can make a diagnostic of his/her learning level and submits the diagnostic data to the server, which returns the diagnostic results. Also, the profiling system collects and encodes the learner profile from PDA, RFID reader, and the learner L. Then, it sends the collected profile to the u-learning server. Now, L can be provided a learning service suitable for his/her PDA and learning level.

In this paper, we also propose the approach to apply the RFID information to learning including the basic profile information. For example, we can assume that the learner L goes to the museum for study and the articles in the museum have RFID tags themselves. L reads the RFID tag of the article of interest using the tag reader attached to the PDA. After reading it, on the spot, L can get additional learning information about the article such as motion learning or hypermedia-based learning contents provided by his/her instructor on the Internet. Under this situation to learn on the spot, it is very important to provide a customized learning service to L using the profiling information. As another example, situation learning is necessary for L, who is interested in learning English as a foreign language. If each object in any place has a RFID tag, L who wants to know its expressions, can read it using a RFID reader attached to PDA. Then, the information of the RFID tag will be sent to a server database to get the English expressions corresponding to the tag.

3 Conclusion

We proposed a profiling system and RFID middleware framework designed for customized u-learning. That is to come up with a way of providing a best-suited learning environment for each mobile learner, who uses a mobile device. The profiling system manipulates a key information such as learner profile, learning level, specifications on mobile devices, and so on. In particular, the learner's profile is drawn through a diagnostic evaluation with the aid of RFID reader. Also, we suggested the RFID middleware framework, which has the common API set including an additional APIs related to a u-learning application.

We're developing a profiling system and implementing RFID APIs to be used on Windows CE in PDA.

Acknowledgements

This study was partly supported by RIS (Regional Innovation System) Program and Jeju Cultural Technology Education Center NURI (New University for Regional Innovation) Program, Cheju National University, Korea.

References

1. Chan, P.,"Constructing Web User Profiles: A Non-invasive Learning Approach", In Proceedings of WEBKDD'99 Workshop on Web Usage Analysis and User Profiling, pp39-55, Springer-Verlag, 1999.
2. Ricardo Carreira, Jaime M. Crato,Daniel Gonçalves, and Joaquim A Jorge, "Evaluating Adaptive User Profiles for News Classification", In Proceedings of IUI'04, 2004.

Responding to Free-Form Student Questions in ERM-Tutor

Nancy Milik, Melinda Marshall, and Antonija Mitrovic

Intelligent Computer Tutoring Group
Department of Computer Science and Software Engineering
University of Canterbury, Christchurch, New Zealand
{nmi14, tanja}@cosc.canterbury.ac.nz

Abstract. We present ERM-Tutor, a constraint-based tutor that teaches logical database design (i.e. mapping conceptual to logical database schemas). Students practice this procedural task in ERM-Tutor by solving each step and receiving feedback on their solutions. We also present a new feature added to the system, which enables students to ask free-form questions. A preliminary evaluation carried out on ERM-Tutor investigated how students use free-form questions, and provided promising results. We plan to perform a bigger study in early 2006.

1 Introduction

Constraint-based tutors have been successful in a variety of domains, such as conceptual database design, database queries, data normalization, UML and language learning [2,5,6]. Building on successful work, we have developed ERM-Tutor, in which students practice the algorithm for mapping conceptual database schemas (i.e. ER diagrams) into relational schemas. The ER-to-relational algorithm [4] consists of seven steps, which map the ER components in the following order: 1) regular entities, 2) weak entities, 3) 1:1 binary relationships, 4) 1:N binary relationships, 5) M:N relationships, 6) multivalued attributes, and 7) *n*-ary relationships. Although the algorithm is well-defined and short, students typically find it difficult to learn and apply consistently.

ERM-Tutor is a web-based system, the main components of which are the pedagogical module, problem solver, student modeler, session manager and user interface. The tutor also contains a set of problems and 121 constraints representing the domain knowledge. The problem-solving process is broken into seven tasks, corresponding to steps in the mapping algorithm, each task presented to the student on a separate page. The student has to complete the current task in order to move on to the next one. The student can request feedback at any time. The short-term student model consists of a list of satisfied and a list of violated constraints. This model is used by the pedagogical module to present feedback to the student. ERM-Tutor also maintains a long-term student model, which is used for problem selection.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 707 – 709, 2006.

2 Question Asking Module

Intelligent Tutoring Systems provide feedback on students' actions, but students do not always understand the feedback they receive. Therefore, it would be beneficial for students to be able to ask free-form questions at any time. ALPS [1,3] allows the student to ask any question, to which the system replies with a pre-recorded video clip. The results show that the rate of unprompted questions is lower than in the case of one-on-one human tutoring. Furthermore, half of the questions are not related to problem-solving, but are rather social interactions. Most of the remaining questions are performance-oriented, and not deep questions that would facilitate learning.

In this light, we added a question-asking module to ERM-Tutor. We defined 98 distinct questions, based on our experiences in teaching the mapping algorithm and our experience with other constraint-based tutors. These questions can be categorized into interface usage ("What does the button *Check Step* do?"), definitions of terms ("What is a foreign key?"), diagram notations ("How is an attribute represented in the ER-diagram?"), mapping regulations ("How is a relationship mapped?"), and deeper questions ("Why are the steps arranged in this order?"). The question database additionally includes a number of repeated questions that are phrased differently, resulting in a total of 182 questions. In contrast to ALPS, the answers to questions are textual.

The TFIDF (Term Frequency Inverse Document Frequency) vector weighting scheme [7] was chosen as the information retrieval mechanism, as is the case in ALPS. In our system, the questions are read from the database and separated into words. The weight of each question and word is calculated, and words are indexed in a hash table. When the student asks a question, the same calculations are applied to the query string: it is also broken-up into words and their weights are calculated. Each question is then allocated a query weight. Finally, the answer corresponding to the question with the highest query weight is returned to the student. To evaluate the subjective relevance of answers, students are encouraged to submit their ratings of answers; however, the system does not enforce it to avoid mode errors and distractions from the problem solving task.

3 Preliminary Evaluation

We performed a preliminary study of ERM-Tutor with students enrolled in an introductory database course at the University of Canterbury in 2005, in order to investigate the usage of free-form questions. 29 students logged into ERM-Tutor at least once, but five students used it for less than two minutes and so their logs were excluded from analyses. The average interaction time was under one hour (mean=54min, sd=63min), ranging from several minutes to 4.5 hours over several weeks. The number of sessions ranged from one to four (mean=1.67, sd=0.96). On average, students attempted 4.6 problems and completed 25% of them.

Only eight students asked questions, with a total of 24 questions submitted. The number of questions per student ranged from one to five. The questions can be categorized into task-focused (50%), definition-focused (8%) and phatic questions (42%). Task-focused questions ask directly for help solving the problem (e.g. "How could I solve this table?"). For instance, three students copied the feedback messages, added a question mark at the end or a "How to" at the start, and submitted them as the questions. Definition-focused questions ask for definition of terms. There were only two such questions submitted: "What is foreign key?" and "What is multivalue?" Phatic questions establish a sense of social mood (e.g. "What is your name?", "How are you?" and "How do you answer questions?"). Excluding phatic questions, 14 questions were relevant for students' actions. Five of these questions were answered correctly, and for two of these, the students specified highest relevance. The answer could not be found for one question. The remaining questions received answers which were related to the query, but were not useful to students. This happened when the students did not formulate questions well, but instead copied a part of the feedback message, adding a question mark at the end (e.g. "Make sure the relationship is 1:1?"). We intend to enhance our question database with these questions.

4 Conclusions

The paper presented ERM-Tutor, a new constraint-based tutor that teaches the procedural task of mapping ER diagrams into relational schemas. We enhanced ERM-Tutor with a question-asking module, which allows the student to ask a free-form question, which the system processes and returns the answer with the highest relevance weight, using the TFIDF weighting scheme. Our preliminary study showed some evidence that students welcome the idea of asking free-form questions and confirmed the need for eliciting deeper questions. We are currently investigating various techniques to encourage students to use the module, such as prompting students to ask more questions and even suggesting a question to be asked based on their student model. We plan to conduct a full evaluation study of ERM-Tutor in March 2006.

References

1. Anthony, L., Corbett, A., Wagner, A., Stevens, S., Koedinger, K. Student Question-Asking Patters in an Intelligent Algebra Tutor. In: J. Lester, R. M. Vicar, Paraguacu, F. (eds) Proc. 7th Int. Conf. Intelligent Tutoring Systems, pp. 455-467, 2004.
2. Baghaei, N., Mitrovic, A., Irwin, W. A Constraint-Based Tutor for Learning Object-Oriented Analysis and Design using UML. In: C.K. Looi, D. Jonassen, M. Ikeda (eds), Int. Conf. Computers in Education ICCE 2005, pp. 11-18, 2005.
3. Corbett, A., Wagner, A., Chao, C., Lesgold, S., Stevens, S., Ulrich, H. Student Questions in a Classroom Evaluation of the ALPS Learning Environment. C-K. Looi, G. McCalla, B. Bredeweg, Breuker, J. (eds) Proc. 12th Int. Conf. Artificial Intelligence in Education, pp. 780-782, 2005.
4. Elmasri, R, Navathe, S. Fundamentals of Database Systems. Addison Wesley, 2004.
5. Mitrovic, A., Mayo, M., Suraweera, P., Martin, B. Constraint-based tutors: a success story. In: L. Monostori, J. Vancza and M. Ali (eds), Proc. 14th Int. Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems IEA/AIE-2001, Springer-Verlag Berlin Heidelberg LNAI 2070, pp. 931-940, 2001.
6. Mitrovic, A., Suraweera, P., Martin, B., Weerasinghe, A. (2004) DB-suite: Experiences with Three Intelligent, Web-based Database Tutors. Journal of Interactive Learning Research (JILR), vol. 15, no. 4, pp. 409-432, 2004.
7. Salton, G., Buckley, C., Term Weighting Approaches in Automatic Text Retrieval. Tech. Report #87-881 Computer Science Dept, Cornell University, 1987.

Behaviors Analysis with a Game-Like Learning System for Pre-school Children

Ren-Jr Wang, Ching-Hsieh Lee, and Chung-Jen Chiu

Industrial Technology Research Institute, ICL, Bldg. 51, 195, Sec. 4, Chung Hsing Rd.
Chutung, Hsinchu, Taiwan 310, R.O.C
{RJWang, C.H.Lee, crchiu}@itri.org.tw

Abstract. Teaching by fun is the most ideal teaching strategy in the educational field. In that case we are trying to establish a game-like learning system with Multi-modal interface and mix-initiative dialogue system for pre-school children.

1 Introduction

By providing several communication channels between human and computer, Multi-modal interface can increase recognition rate of user's intention through various natural input modalities [4][5]. In this poster, we are concerned about how to merge Multi-modal interface into digital learning environment, particularly a game-like learning environment for per-school children. In addition, dialogue management (DM) has also been integrated into our system to generate dynamic content according to the interaction between computer and children. Finally, we collect some video clips from pre-school children through our system and try to analyze possible learner's behaviors with NHUE (National Hsinchu University of Education) to enhance the game flow and tutoring strategy in the future.

2 System Architecture

Currently, we propose an extendable multi-modal integration framework. Mouse, keyboard and Speech input will be our default modalities. (Figure 1)

For domain portability and system extensibility, we utilize a unified Semantic Feature Structure [4] for semantic meaning representation and multi-modal integration. Each input modality has its own understanding process and need to follow the unified Semantic Feature Structure to convert user's input into more accurate semantic frame (For example, NLU [2] and Device Understanding).

After retrieving semantic frames from various modalities, integration module aligned these frames by rules. Currently we use time as a main feature to alignment semantic frames. In the future, we will try other features to align those semantic frames, according to different characteristics of learners.

When alignment process completed, system obtains groups of semantic frames, and then integrates into new semantic frame represented learner's operations. For this

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 710 – 712, 2006.

purpose, we adopt a semantic multimodal integration approach [3][4] that use Ontology to represent possible semantic features in semantic frames. During the integration process, we not only use ontology to verify the comparability of feature types, but also derive rules to combine multiple unimodal semantic frames by inferring from ontology. Once system achieves understanding learner's operations, it passes the integrated semantic frame to dialogue management module. Dialogue management module will maintain learner's input log and dynamically generate suitable content back to learner [1].

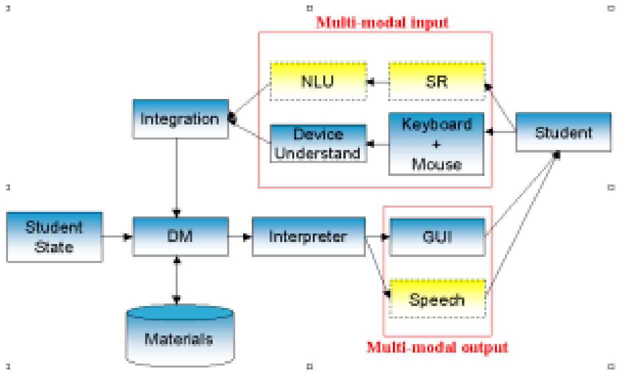


Fig. 1. Extendable multi-modal integration framework

3 Data Collection and Analysis

We have completed a proof-of-concept system that fits within our integration framework. . Children will learn how to recognize animals through first scene by moving animals to the right place in the farm. After moving all the animals, system will change to the next scene that will let children to choose their desired animals and then ask the number counting question according to their choice.

To track the behaviors of students, we studied 50 children that age from 4 to 7 in our research. 33 children are novice computer users but the reset children aren't. To make the data collection more realistic, we let them playing our system by themselves in the music classroom of kindergarten and put the camera away from them.

4 Experiment Result and Future Work

After few weeks analysis and discussion with educational experts, we preliminarily found several interesting learner's behaviors of pre-school children through our system as following. We are now proceeding to model learner's behaviors [6][7].

(1) Distraction /Leaving: When finding children have distracted from computer game with time or leave their seats, we will classify these children in this category.

(2) Discovery learning: Children in this category mostly belong to novice computer users. They cost much time to learn how to handle mouse when playing the game.
(3) Search for answer: Different from discovery learning, children in this category enjoy in the game. In the while, we also find some children have high cognitive ability.
(4) Practiced operation: Children in this category operate the mouse to play game and answer the question smoothly.
(5) Play: Children try to use their way in the game process instead of following the instruction provided by our system.
(6) Impatience: After several times of playing, some children are quite familiar with the game flow and are able to predicate the next step.

In the future, we will integrate learner's model into our system and hope to enhance the game flow and tutoring strategy to make the system more attractive to the learners.

Acknowledgements

This paper is a partial result of Project 5352B23100 conducted by ITRI under sponsorship of the Ministry of Economic Affairs, R.O.C. Especially thanks to Dr. Ya-Hui Chen and Dr. Cheng -Ta Yang for their technical support.

References

[1] H.-M. Wang, Y.-C. Lin and P.-H. Hung, "Error-tolerant and Goal-oriented Approach to Designing a Robust and Scalable Spoken Dialogue System", CCL Technical Journal , pp. 45-55, 2001
[2] Y.-C. Lin, H.-M. Wang and C.-J. Chiu, "Error-Tolerant Spoken Language Understanding with Confidence Measure", CCL Technical Journal , pp. 13-19, 2003
[3] Johnston, M., et al., "Unification-based Multimodal Integration", Proceedings of 35th Annual Meeting of the Ass. for Comp. Ling., 1997
[4] Joyce Chai, S. Pan, M. Zhou, and K. Houck, "Context-based Multimodal Understanding in Conversational Systems", Proceedings of 4th International Conference on Multimodal Interfaces (ICMI 2002)
[5] Sharon Oviatt, Rachel Coulston, and Rebecca Lunsford , "When Do We Interact Multimodally? Cognitive Load and Multimodal Communication Patterns", Proceedings of 6th International Conference on Multimodal Interfaces (ICMI 2004)
[6] Bunt A. and Conati C., "Probabilistic Student Modelling to Improve Exploratory Behaviour". Journal of User Modeling and User-Adapted Interaction, vol 13 (3), p. 269-309, 2003
[7] Burton, R.R. and Brown, J.S., "A tutoring and student modelling paradigm for gaming environments", ACM SIGCSE Bulletin, 8, 236-246.1976

Studying Human Tutors to Facilitate Self-explanation

Amali Weerasinghe and Antonija Mitrovic

Intelligent Computer Tutoring Group
Department of Computer Science and Software Engineering
University of Canterbury, Private Bag 4800, Christchurch, New Zealand
Tel.: +64 3 364 2987 ext. 7771
{acw51, tanja}@cosc.canterbury.ac.nz

Abstract. This paper reports the first phase of a project with the goal of developing a general model of self-explanation support, which could be used in both open- and closed-ended domains. We studied how human tutors provide additional support to students learning with an existing intelligent tutoring system designed to help students learn database modelling. We report on the findings from this study, which will serve as the basis for defining the model. We also discuss directions for future work.

1 Introduction

Studies indicate that students acquire shallow knowledge even in the most effective Intelligent Tutoring Systems (ITS) [1]. Self-explanation was shown to facilitate the acquisition of deep knowledge [2]. Several ITSs were enhanced with self-explanation support in domains such as physics [3], mathematics [1], database design [6] and data normalization [5]. With the exception of database design, all these domains are closed-ended, as problem solving is well structured, and therefore self-explanation expected from learners can be clearly defined. Database design is an open-ended task: the final result can be defined in abstract terms, but there is no algorithm to find it. Although the above ITSs were shown to improve student performance, none of these self-explanation models have been used in both open- and closed-ended domains.

Our long-term goal is to develop a model to facilitate self-explanation which can be used in both open- and closed-ended domains. We have chosen Entity-Relationship (ER) modelling as the open-ended domain, and ER-to-relational mapping as the closed-ended domain. The later task is a well-formed one, and therefore is a deterministic algorithm that students learn in database courses. EER-Tutor [7] and ERM-Tutor [4] are two existing constraint-based tutors. Our goal is to develop a general self-explanation model that can be used to enhance these systems.

In order to develop a model for self-explanation, we need to consider three basic decisions: when to prompt for self-explanation, what to self-explain and how to obtain self-explanation from learners. As the first step, we conducted a study to observe how students interacted with the EER-Tutor, while providing additional help by a human tutor through a chat interface. Section 2 presents this study. The next section discusses the findings of this study and how they can be incorporated in a self-explanation model. Section 4 details the conclusions and the directions of future work.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 713 – 715, 2006.

2 Preliminary Study

The study was conducted in August 2005 at the University of Canterbury, and involved volunteers enrolled in an introductory database course and professional tutors. The professional tutors will be referred to as tutors, while EER-Tutor as the system hereafter. EER-Tutor provides a problem-solving environment and complements classroom instruction. The version of EER-Tutor used in the study was enhanced with a chat interface, so that the tutors could provide one-to-one feedback to students. We wanted to make the bandwidth between the student and the tutor very similar to that between the student and the ITS. As a result, tutors could observe only the students' interactions with the ITS. Participants interacted with the system in one room and the tutors observed their interactions in another room.

The tutors were not given any specific instructions on providing assistance to students. Student participants were not told that a human tutor was involved in the study. Students also could ask for help through the chat interface or the *More Help* button in the interface. All interactions were recorded. Students themselves decided when to end the session. All participants filled out a questionnaire on their perceptions about the system and interventions through the chat interface. The tutors were also interviewed to understand their views on the tutoring experience.

3 Observations and Prototype for the Self-explanation Model

Seven students and four professional tutors participated in the study, with at most two students per tutor. The average duration of sessions was 85 minutes (sd=20). The average number of problems attempted was 11 (sd = 5), and all the participants completed all the problems attempted. The timing of tutor interventions differed significantly. Some tutors intervened in the first problem in which the student needed help, while in other sessions, the tutors intervened mostly in 4th or the 5th problem. In one situation, the tutor waited until the 19th problem to intervene.

The self-explanation model will be developed on the basis of the findings from this study. The model will decide when and what to self-explain, and how to obtain self-explanations. As all tutors provided delayed feedback, which was well-received by the participants, the model will provide delayed feedback. With delayed feedback, specific guidelines to decide on the timing of interventions need to be incorporated into the model. In the study, delayed feedback was provided in the following situations: (i) the student has been inactive for a pre-determined period of time, (ii) the student has made the same mistake repeatedly or (iii) the student seems to be reacting to feedback without much reflection.

In the first scenario, it will be beneficial to prompt the student to ask a question in order to understand the difficulty in completing the solution, to which the system can respond appropriately. This either requires natural language capabilities or obtaining the response through menu options. For instance, we can ask the student which concept he/she is having difficulties with, and provide a menu for the student to select the concept he/she needs assistance with. As noted in (ii), if the student makes the same mistake repeatedly, it is clear that there is a misconception or gap in his/her knowledge. Then, it will be more beneficial to provide a problem-independent explanation initially. Later on, the student may need assistance to understand how to apply the domain concept to the current state of the problem. A student seems to be reacting to feedback without reflection if he/she makes a single change without reflecting on the other changes that need to be performed as a result. In such situations, the student will be prompted to reflect on other related changes.

Our model also needs to decide how to prompt learners to self-explain. As explained earlier, we have seen tutors provide problem-independent explanations when there is evidence that a student has difficulty with a domain concept. Later on, he/she can be prompted to understand how the corresponding domain concept relates to the current problem state. At other times, the student may have difficulty with the current problem. Then the student can be guided using a series of prompts ranging from rephrasing feedback, discussing problem-specific details to providing the answer directly. If the system has natural language processing capabilities, students would be able to correspond with the ITS in a natural manner using partial answers.

4 Conclusions and Future Work

This research focuses on developing a self-explanation model for both open- and closed-ended domains. As the first step, we conducted a study to observe how tutors help students to solve a problem using the EER-Tutor. In addition to the system's feedback, the students were prompted by human tutors through a chat interface. Although different kinds of prompts were used, all of them provided delayed feedback and guided the students towards the solution without giving the answer directly. Both timing and content of interventions were well received by the students. They also felt that the help received through the chat interface was very useful for understating mistakes on their own, providing an opportunity for self-explanation and reflection.

The findings from the study are being used to develop the model of self-explanation, which will be used in the next study with ERM-Tutor to understand its applicability in a closed-ended domain. If necessary, the model will be modified and then implemented in both EER-Tutor and ERM-Tutor. These enhanced systems will later be evaluated in authentic class room environments.

References

1. Aleven, V., Koedinger, K. R., Cross, K.: Tutoring Answer Explanation Fosters Learning with Understanding. In: Lajoie, S.P. and Vivet, M.(eds.): AIED 1999, IOS Press, 199-206.
2. Chi, M. T. H.: Self-explaining Expository Texts: The dual processes of generating inferences and repairing mental models. Advances in Instructional Psychology, (2000) 161-238.
3. Conati, C., VanLehn, K.: Toward Computer-Based Support of Meta-Cognitive Skills: a Computational Framework to Coach Self-Explanation. Artificial Intelligence in Education, 11 (2000) 389-415.
4. Milik, N., Marshall, M., Mitrovic, A.: Teaching Logical Database Design in ERM-Tutor. In Ikeda, M., Ashley, K. (eds.): Proc. ITS 2006.
5. Mitrovic, A: The Effect of Explaining on Learning: a Case Study with a Data Normalization Tutor. In: C-K Looi, G. McCalla, B. Bredeweg, J. Breuker (eds) Proc. AIED 2005, 499-506.
6. Weerasinghe, A., Mitrovic, A.: Supporting Self-Explanation in an Open-ended Domain. In: Gh.Negoita, M., Howlett, R. J., Jain, L.C. (eds.) Proc. of KES 2004 306-313.
7. Zakharov, K., Mitrovic, A., Ohlsson, S.:, Feedback Micro-engineering in EER-Tutor. In: Looi, C-K., McCalla, G., Bredeweg, B., Breuker, J. (eds.) Proc. AIED 2005, (2005) 718-725.

An Ontology-Based Solution for Knowledge Management and eLearning Integration

Amal Zouaq[1], Claude Frasson[1], and Roger Nkambou[2]

[1] University of Montreal, CP 6128, succ Centre-Ville, Montreal, QC, H3C3J7
Tel.: (514) 343-5648
{zouaq, frasson}@iro.umontreal.ca
[2] UQAM, CP 8888, succ Centre-Ville, Montreal, QC, H3C3P8
Tel.: (514) 987-3000 # 8395
nkambou.roger@uqam.ca

Abstract. As knowledge becomes a crucial asset to organization's survival, an efficient knowledge management policy should be set up and should result into an organizational memory. Similarly, an effective eLearning program must be implemented in the organization and exploit the organizational memory to manage competence evolution. In this paper, we introduce a framework to manage organizational knowledge through the creation and management of an ontology-based Organizational Memory (OM). We also explain how an Intelligent Tutoring System (ITS) can benefit from this OM in order to provide a just-in time, just enough learning solution to the organization's members.

1 Introduction

Knowledge is the key asset of the modern organization. Thus, the capture of organizational knowledge becomes a necessity as well as the development of internal competences. In this context, the set-up of just-in time just enough learning connected to the real work processes becomes also urgent. The integration of knowledge management and Intelligent Tutoring Systems [10] could be an answer to this need [7].

2 An Ontology-Based Organizational Memory

Inside an organization, most of the corporate knowledge is created and stored in the form of documents, with little or no metadata about document content and context. This situation creates two kinds of problems: From one side, this lack of metadata and structure leads to very poor capabilities to query corporate knowledge sources within an automatic process and makes documents knowledge invisible when trying to inventory the organization intellectual capital. From the other side, the implementation of eLearning programs does not beneficiate from the knowledge capital inside the organization which results into training materials disconnected from the real work processes and knowledge.

Our proposed architecture, indicated in Fig. 1, relies on the creation and management of an ontology-based Organizational Memory (OM) [1, 2, 3, 9].

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 716 – 718, 2006.

Ontology is used to create and organize content metadata and to provide a shared understanding across the organization. In fact, ontology is used to describe the organization, the domain in which the organization works, its competences, the organizational documents, etc. Each object in the system (Employee, Competence, Knowledge Object, Role, etc.) is an OWL [5] object which can be uniquely identified by a URI. All these objects are described by a number of concepts of the domain ontology and can thus be retrieved by similarity-based techniques using the vector cosine ranking algorithm [4].

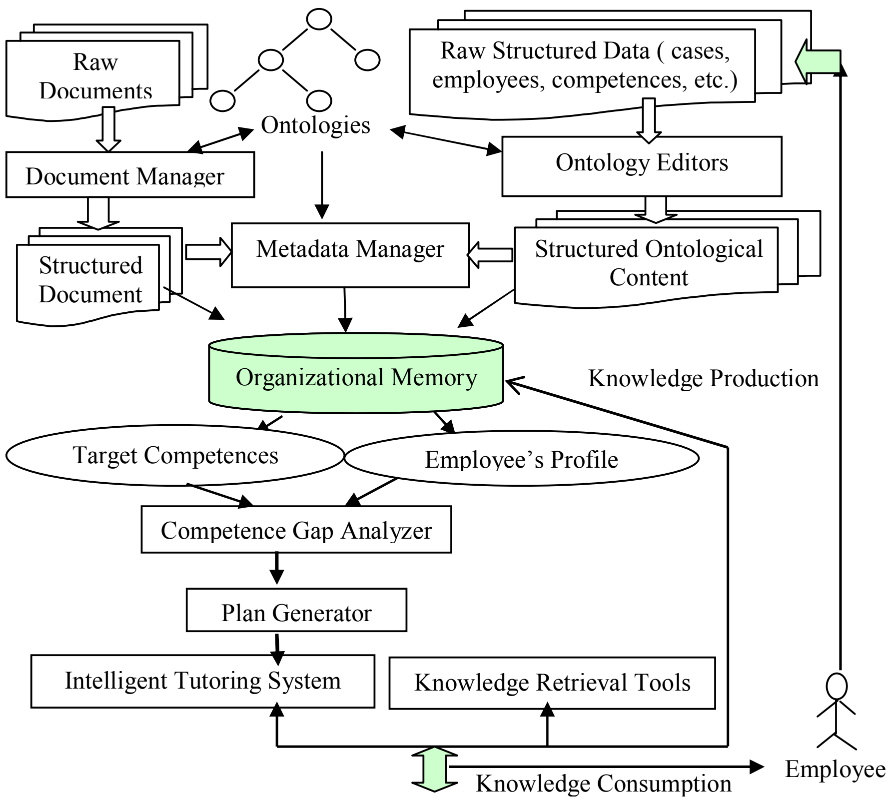


Fig. 1. Conceptual Architecture

Briefly speaking, our architecture takes as input structured data (database, case, employee, competence, etc.) and unstructured data (documents) provided by an employee who wants to share his knowledge (Knowledge Production). The system then identifies manually (with the help of the employee) or semi-automatically (with information retrieval techniques) relevant knowledge parts and concepts and link them to the existing ontologies or creates new concepts in the ontologies. Then SCORM [8] compliant metadata is created through a metadata manager, results into OWL files and refers to the OWL objects (with their URI). A knowledge Object is then added to the Organizational Memory. Because each object is defined according to the ontologies, and especially competences, the system is then able to offer efficient knowledge retrieval tools and to perform competence gap analysis. An eLearning plan is then generated with the corporate knowledge thus integrating training material with organizational knowledge (Knowledge Consumption).

Our platform is developed in java and the OWL ontologies are developed with the Protégé Ontology editor [6].

3 Conclusion

In this paper, we presented a solution to the knowledge acquisition problem in intelligent tutoring systems through the use of the organization available knowledge in training sessions and the set-up of an ontology-based organizational memory (OM). At present, we elaborated and we built tools to create, organize, retrieve and effectively exploit the knowledge residing in the OM to achieve corporate competence development and training. Future work may explore the interest of document summarization coupled with automatic metadata generation. The next step in this research will be to experiment the proposed framework in real corporate settings. We are currently working on domain ontology development inside a real corporate environment and first experiments have shown the advantage of our architecture.

References

1. Abecker, A., Bernardi, A., and Sintek, M. 1999b. Proactive Knowledge Delivery for Enterprise Knowledge Management. In Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering, Learning Software Organizations, Methodology and Applications, Lecture Notes In Computer Science, 1756:103–117.
2. Conklin, E.J. 2001. Designing Organizational Memory: Preserving Intellectual Assets in a Knowledge Economy, white paper, Group Decision Support Systems, Washington, D.C. http://cognexus.org/dom.pdf
3. Gandon, F. 2002. A Multi-Agent Architecture For Distributed Corporate Memories. Third International Symposium, From Agent Theory to Agent Implementation, at the 16th European Meeting on Cybernetics and Systems Research (EMCSR 2002), 623-628.
4. Harman, D. 1992. Ranking Algorithms, in W. Frakes & R. Baeza-Yates. Information Retrieval, Prentice-Hall, USA.
5. OWL : Web Ontology Language. Available at : http://www.w3.org/TR/owl-ref/
6. The Protégé ontology builder. Available at: http://protege.stanford.edu/
7. Schmidt, A. 2005. Bridging the Gap Between E-Learning and Knowledge Management with Context-Aware Corporate Learning (Extended Version) In: Professional Knowledge Management (WM 2005) Post Proceedings, Springer.
8. SCORM (Sharable Content Object Reference Model). Available at : http://www.adlnet.org
9. Van Elst, L., and Abecker, A. 2002. Domain Ontology Agents for Distributed Organizational Memories. In Rose Dieng-Kuntz et al., Knowledge Management and Organizational Memories, Kluwer Academic Publishers.
10. Zouaq, A., Frasson, C., and Rouane, K. 2000. The Explanation Agent. Intelligent Tutoring Systems 2000: 554-563.

Computer-Mediated Collaborative Visual Learning System to Generate Evidence-Based Nursing Care Plans

Woojin Paik[1] and Eunmi Ham[2,*]

[1] Dept. of Computer Science, Konkuk University,
322 Danwol-Dong, Chungju-Si, Chungcheongbuk-Do, 380-701, Korea
wjpaik@kku.ac.kr
[2] Dept of Nursing Science, Konkuk University
322 Danwol-dong, Chungju-si, Chungcheongbuk-do, 380-701, Korea
hem2003@kku.ac.kr

Abstract. For the nurses to provide quality patient care, the nurses need to generate an accurate and reliable nursing care plan. To prepare a nursing care plan, the nurses need to follow a number of steps starting from the data collection stage and ending with the nursing diagnoses, outcome, and intervention selection stage. Many nurses are responsible for many patients. The patients come and go. The nurses in many critical settings work in the three eight-hour shifts. Therefore, it is essential and necessary for many nurses to share the generation and the use of the nursing care plan. Instead of text-based nursing care plan, we are developing a visual system to represent patient state and the corresponding diagnosis through collaborations amongst the nurses. By using this system, the novice nurses learn from what others do.

1 Collaborative Nature of the Nursing Care

For the nurses to provide quality patient care, it is necessary for the nurses to generate an accurate and reliable nursing care plan. The nurses follow a number of steps starting from the data collection and ending with the nursing diagnoses, outcome, and intervention selection. There are 167 nursing diagnoses, which are categorized as organized as 13 domains and 46 classes [1]. Each diagnosis is associated with the definition and one or more defining characteristics. The nurses learn to select and differentiate diagnoses by reviewing the existing cases and also by observing what other nurses do. Each diagnosis is linked to one or more potentially suitable nursing outcomes. Therefore, the nurses select the most appropriate nursing outcomes based on the initially chosen nursing diagnosis. Similarly, each diagnosis and outcome is linked to one or more potentially suitable nursing interventions. The nurses learn to select the correct outcomes and interventions by reviewing the existing cases and also by observing what other nurses do.

Many nurses are responsible for many patients. The patients come and go. The nurses in many critical settings work in the three eight-hour shifts. Therefore, it is essential and necessary for many nurses to share the generation and the use of the nursing care plan. One nurse might not be able to collect all necessary information

* Corresponding author.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 719 – 721, 2006.

about a patient to reach a diagnosis at one time. Thus, other nurses will need to collaborate in the data collection stage. In addition, one nurse might develop doubts, questions, or further data collection needs while trying to select the most appropriate nursing diagnosis. Other nurses can either find then share answers to the questions for others including the nurse to study or complete the nursing diagnosis step with the additional information.

2 Training by Collaboratively Generating Nursing Care Plan

Nursing care plan is an embodiment of how nurses apply a clinical reasoning process to analyze and evaluate gathered facts about patients. The nurses develop a plan of care that prescribes interventions to attain the expected outcomes. The care plan is prepared to provide continuity of care from nurse to nurse, to enhance communication, to assist with determination of agency or unit staffing needs, to document the nursing process, to serve as a teaching tool, and to coordinate provisions of care among disciplines [2].

We are developing a collaborative nursing care plan generation system for the nursing science undergraduate students. Every semester, the Juniors, who major in nursing science at the Department of Nursing Science, Konkuk University in Chungju, Korea, develop detailed care study reports of patients while they work as a student intern at a psychiatric warden for four weeks. The nursing care plan is one section of the case report, which is submitted at the end of the internship period. Traditionally, each case study report including the nursing care plan was prepared by one student. However, we wanted to introduce the group project aspect into the internship assignment by having the students to generate the nursing care plan collaboratively.

To enable the easy collaboration amongst the group members, we developed a web community for the group members to communicate and also to collaboratively generate the nursing care plan. At the center of the web community, we incorporated an interactive visualization system to allow visual inspection of the collected factual information about the patient. The visualization system allows the students to link one or more diagnoses and the corresponding outcomes as well as interventions given a specific patient. This representation is inspired by MindMap, which is an alternative way to organize the patient information especially devised to enhance the educational experiences by the nursing students [3]. One notable feature of the MindMap-based visualization system is the exemplification of the previously selected outcomes, and interventions for other patients given the same diagnosis. For example, if a certain outcomes or the interventions were often chosen for a particular diagnosis then the linkage will automatically appear before the nursing students select any outcomes or the interventions. This allows the nursing students to consider the evidences, which are accumulated during the nursing care plan generation tasks carried out during the previous semesters. This is an attempt to promote the evidence-based nursing care. The collaboration as well as the learning amongst the nursing students is designed to occur when they collectively manipulate visual representation of the nursing care plan.

The Figure 1 shows ‘Health Maintenance, Altered’ and ‘Social Isolation’ as the nursing diagnoses for the patient HS. Each nursing diagnosis is preceded by ‘D’. Nursing Outcome is preceded by ‘O’ and Intervention is preceded by ‘I’.

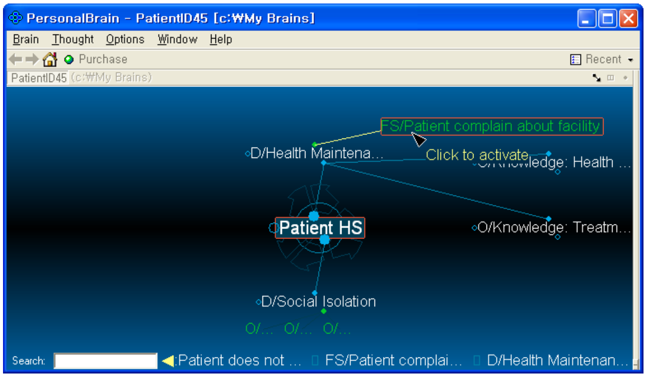


Fig. 1. Mind Map for Patient HS

We are in the early stages of developing a computer-mediated collaborative visual learning system to generate evidence-based nursing care plans. We developed a prototype system, which enables the students to visual representation of the nursing care plans. We are in the process of the evaluating the system. We expect the resulting system to aid both nursing students and practitioners. They can learn from each other by working toward the same goal of generating a quality nursing care plan.

References

1. Ralph, S.S., Craft-Rosenberg, M., Herdman, T.H., and Lavin, M.A., NANDA Nursing Diagnoses: Definitions & Classification 2003-2004, NANDA International, Philadelphia, 2003.
2. Doenges, M., and Moorehead, M.F., Application of Nursing Process and Nursing Diagnosis: An Interactive Text for Diagnostic Reasoning 4th Edition, F.A. Davis Co., Philadelphia, 2003.
3. Mueller, A., Johnston, M. and Bligh, D., Joining mind mapping and care planning to enhance student critical thinking & achieve holistic nursing care. Nursing Diagnosis 13(1):24-27, 2002.

Prevention of Off-Task Gaming Behavior in Intelligent Tutoring Systems

Jason A. Walonoski and Neil T. Heffernan

Worcester Polytechnic Institute, Computer Science Department, 100 Institute Rd, Worcester, MA 01601 USA
{jwalon, nth}@wpi.edu

Abstract. A major issue in Intelligent Tutoring Systems is off-task student behavior, especially performance-based gaming, where students systematically exploit tutor behavior in order to advance through a curriculum quickly and easily, with as little active thought directed at the educational content as possible. This research developed both active interventions to combat gaming and passive interventions to prevent gaming. Our passive graphical intervention has been well received by teachers, and our experimental results suggest that using a combination of intervention types is effective at reducing off-task gaming behavior.

1 Introduction

Intelligent Tutoring Systems (ITS) have been shown to have a positive effect on student learning [1], however these effects may be negated by a lack of student motivation or student misuse, particularly "gaming" of the system [2]. Gaming is the systematic use of tutor feedback and help methods as a means to obtain correct answers with little or no work, in order to advance through a curriculum as fast or as easily as possible.

Within ITS there have been a variety of approaches towards remediation of gaming behavior in students [2], which are mostly active interventions focused on combating student gaming, with few approaches focused on prevention. This research aimed at exploring a more comprehensive approach using active interventions to combat gaming along with a passive method to prevent gaming within the *Assistments* mathematics ITS [3].

2 Prevention of Gaming

We developed three gaming interventions, two traditional active interventions, and one passive deterrent or prevention mechanism. The interventions were deployed and evaluated experimentally. Two active interventions were used to respond separately to the two types of hallmark gaming behavior: rapid-fire guessing-and-checking and hint/help abuse. These interventions were triggered by simple gaming detection algorithms, which marked a student as guessing-and-checking or abusing-hints *prima facie* of the appropriate surface-level characteristics [4]. When triggered a message was displayed to the offending student encouraging them to try harder, ask a teacher for help, or pursue other suitable actions.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 722 – 724, 2006.

The passive intervention sought to prevent gaming by providing visual feedback on student actions and progress. It had no triggering mechanism, and was continuously featured prominently on-screen for easy viewing by the student and teachers. Theoretically, gaming would then be prevented through Panopticon-like paranoia (when a fear of being watched, without knowing whether one is being watched at any given moment, causes self-corrective behavior) [5].

Our passive intervention (not shown) graphically plots all recorded student actions (such as problem attempts, hint requests, bottom-out hints) in a horizontal timeline. Each action has associated summary text that identifies and provides relevant details and results of the action on mouse-over. The horizontal distance between points reflects the amount of time between the actions. The vertical height of actions is based on their outcome (correct actions are higher than incorrect actions). Throughout the design, the ubiquitous traffic-light color conventions of modern society are used, where green is implicitly "good" or "correct," yellow is "caution," and red is therefore "bad" or "incorrect." The graphical plot was designed to (1) allow teachers and students to easily identify gaming behavior via emergent visual patterns, (2) thereby preventing gaming behavior in the students by the students themselves, and (3) providing a launching point for teacher intervention where gaming behavior or student misunderstandings are identified.

Once all three interventions were designed and implemented, we conducted an experiment to test their effectiveness within the *Assistments* system. One group of students (70 students) received both the active and passive interventions (group 1); while a second group (57 students) received no interventions (group 2). Both groups of students used the tutoring system for an average of 3 class periods (approximately 45 minutes each period), each session having their rate of gaming measured by our *prima facie* gaming recognition algorithm [4]. Then we swapped the conditions, so that group 1 no longer received interventions, while the group 2 began to encounter them. The students used the tutoring system for another class period, and the rate of gaming was compared before and after the swapping of conditions.

Before switching conditions, group 1 had an average rate of gaming that was almost half the rate of group 2 (an average of 3.62 occurrences of gaming per session compared to 6.235), suggesting that the interventions were perhaps having some sort of effect. However, in order to show that those differences were not the result of some sort of selection effect in the groups, the conditions were swapped. After the swap, both groups had decreased amounts of gaming. Group 1 reduced gaming on average by 2.8 occurrences per session, while group 2 decreased their gaming by an average of 4.4 occurrences per session. One-side t-tests were performed on both groups, to see if the resulting change was significantly different from zero, and in both cases the answer was yes ($p < 0.0001$, in both tests).

To determine whether there really was a bigger impact with group 2 – turning the intervention mechanisms on versus off – we conducted an analysis of variance (ANOVA) and the resulting p-value of .08 suggests that turning the interventions on (group 2) makes a bigger impact on *prima facie* gaming than turning them off (group 1). One possible interpretation and explanation of these results would be that when interventions are turned on students learn not to game, and once interventions are turned off, they simply continue not to game. Further analysis might reveal whether actual invocation or receiving of the active interventions is correlated with this decrease in gaming, as opposed to simply the *possibility* of receiving them (a student might have never seen the active interventions when they were turned on if they were never gaming). Otherwise, we might be able to conclude that the decrease in gaming was due more to the passive intervention, or perhaps other factors. We leave the identification of the particular effects each factor for future work.

3 Conclusions

The goal of this research was to explore the intervention and prevention of off-task gaming behavior within the *Assistments* system. Three dynamic mechanisms were designed: two active interventions for hint-abuse and guessing-and-checking, and one passive intervention. Our experimental results suggest that the combined application of active and passive interventions successfully reduces off-task gaming behavior more effectively than no intervention mechanisms.

References

1. Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). *Intelligent tutoring goes to school in the big city*. International Journal of Artificial Intelligence in Education, 8, 30-43.
2. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. (2004) *Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System"*. Proceedings of ACM CHI 2004: Computer-Human Interaction, 383-390.
3. Razzaq, L, Feng, M., Nuzzo-Jones, G., Heffernan, N.T. et. al (2005). *The Assistment Project: Blending Assessment and Assisting*. Proceedings of the 12th Annual Conference on Artificial Intelligence in Education 2005, 555-562.
4. Walonoski, J.A., Heffernan, N.T. (2006). *Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems*. Proceedings of the 8th International Conference on Intelligent Tutoring Systems.
5. Bentham, Jeremy. *The Panopticon Writings*. Ed. Miran Bozovic (London: Verso, 1995).

Learning Linear Equation Solving Algorithm and Its Steps in Intelligent Learning Environment

Marina Issakova*

University of Tartu, Institute of Computer Science, J. Liivi 2, 50409 Tartu, Estonia
marina.issakova@ut.ee

Abstract. T-algebra is an interactive learning environment for step-by-step solving algebra problems, including linear equations. A step in T-algebra combines conversion by rules with entering of the result. The rules, which correspond to the steps of school algorithms, give the program information about the student's intentions and enable the program to check the knowledge of the student, to understand the mistakes and give feedback. T-algebra is intended to help learning solution algorithms and their steps with designed rules. This article describes the rules designed for linear equation solving in T-algebra.

1 Introduction

At school most of algebra problems (including linear equations) are solved using some algorithms. To solve linear equations, the student should know step-by-step solution algorithm and know how to perform each algorithm step: choose a transformation rule corresponding to a certain operation in the algorithm, select the operands for this rule, and replace them with the result of the operation.

There are two different kinds of interactive learning environments available, which allow building step-by-step solutions. In the first kind of environments (such as MathXpert [2], AlgeBrain[1]), the student solves a problem working in terms of rules: selects a part of the expression and the rule. In such environments the student learns solution algorithm, but the learning of performing algorithm steps is passive, because the transformation itself is made by the computer. In the second kind of environments (for example, Aplusix [4]), the student can produce step-by-step solution themselves, because a solution step consists simply of entering the next line. Yet the program does not handle the solution algorithms of different types of problems.

T-algebra is an environment enabling to solve algebra problems step-by-step, including solving of linear equations. In the design of the T-algebra environment, we have been guided by the principle that all the necessary decisions and calculations at each step should be made by the student. For that T-algebra combines the two approaches described above: selection by rules is supplemented by entering the result. This gives the student the possibility to learn the algorithms and their steps. And it enables the program to check the knowledge and skills of the student, to monitor, whether the student works according to the algorithm and to diagnose errors.

* The author was supported by ICT graduate school.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 725 – 727, 2006.

2 Designed Rules for Linear Equation Solving in T-algebra

Problem solving in T-algebra takes place step-by-step. To make the program more intelligent, our own rule dialogue was designed. Each solution step consists of three stages: selection of the transformation rule, marking the parts of expression, entering the result of the operation. The student can make mistakes, receive feedback and ask for help at all three stages. The problem solution window is shown on Figure 1.

At the first stage of each solution step, the student has to choose the rule that he is going to apply. The information on which rule is applied enables the program to estimate, whether the student knows the algorithm for solving this problem and check more efficiently, whether the student's actions on the next stages are correct. The textbook algorithms were followed as closely as possible in the design of the rules, which correspond to the steps of school algorithms. We have tried to make the student's approach to solving the problems within the program parallel to the approach the student would take solving on paper. We hope that such work in the program will help the student to develop skills, which will carry over to the work on paper.

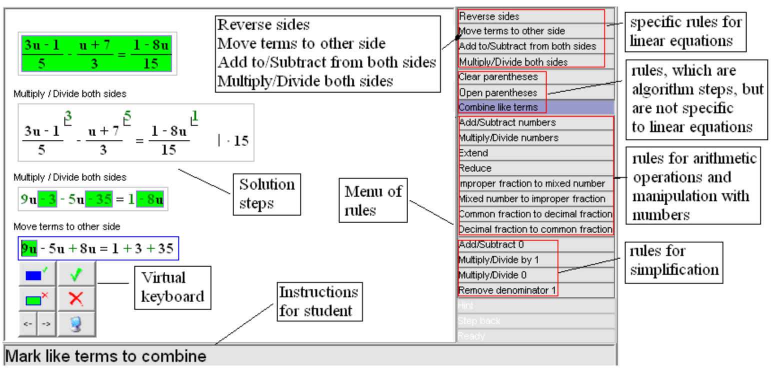


Fig. 1. The problem-solution window of the T-algebra program

The designed set of rules is complete, i.e., all exercises in this field are solvable with these rules. The set of rules consists of the new rules for the algorithm, which is being learned, and of the simplification and computation rules learned before (Fig. 1 shows the set of rules for linear equations). This set of rules is small enough to be displayed in the menu at all times, and it gives the possibility to diagnose whether the student knows which step to perform at the moment. In other environments the set of rules, which allows solving the same equations as in T-algebra, is much larger. For example, MathXpert has 11 specific rules instead of our 4 rules. With so many rules in MathXpert the student cannot see all the rules and cannot select an unsuitable rule, because "…only correct rules are offered as menu choices for you to choose" [2].

Designing the rules, we have taken into account results of research on students' mistakes made on paper [3], and have attempted to leave an opportunity for the student to make the same mistakes in T-algebra. We have also tried to make the rules interface as transparent as possible to be sure that mistakes made by the student are caused by misconceptions not by poor interface design.

Application of the rules *Multiply/Divide both sides* and *Move terms to other side* is displayed on Figure 1. Let us take a closer look at the last rule (Fig. 2). In order to apply this rule, the student has to mark the terms that he wants to move. The program checks, whether the selected parts are appropriate. If the marking was correct, the equation with boxes is written onto the next line. The boxes appear on other side of selected parts. The student can make most common errors [3], which he would do on paper: forget some moved term and not change the sign of a moved term. The program can diagnose these mistakes, can give appropriate error message and show the exact position (box) of the incorrect part. During the input, the student can ask for help (button with computer) and the program will put the right answers to the boxes.

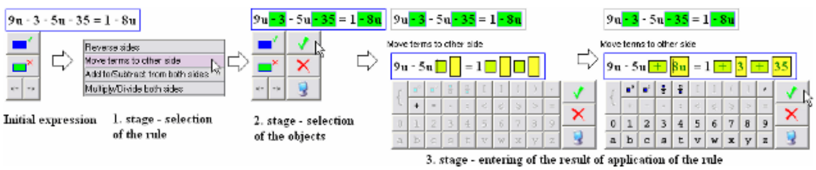


Fig. 2. Applying the rule *Move terms to other side*

3 Conclusion

In existing systems the student either can learn algorithm steps and the program does not handle the solution algorithms (Aplusix) or the student can learn only algorithm and the learning of performing algorithm steps is passive (MathXpert, AlgeBrain). We have succeeded to create such rule dialogue in T-algebra that gives the student the possibility to learn both solution algorithms and their steps, to make the same mistakes as on paper and enables the program to give understandable feedback about mistakes. Designing the set of rules, we followed school solution algorithms. As we have seen the designed set of rules is small enough to be displayed in the menu and this gives the student the possibility to choose. The design of rules and of rule dialogue is most important part in T-algebra that distinguishes it from other environments.

References

1. Alpert, S.R., Singley, M.K., Fairweather, P.G.: Deploying Intelligent Tutors on the Web: An Architecture and an Example. International Journal of Artificial Intelligence in Education 10(2) (1999) 183-197
2. Beeson, M.: MathXpert : un logiciel pour aider les élèves à apprendre les mathématiques par l'action. Sciences et Techniques Educatives 9(1-2) (2002)
3. Issakova, M.: Possible Mistakes during Linear Equation Solving on Paper and In T-algebra Environment. In: Proceedings of the 7th International Conference on Technology in Mathematics Teaching, Vol. 1. Bristol, UK (2005) 250-258
4. Nicaud, J., Bouhineau, D., Chaachoua, H.: Mixing microworld and cas features in building computer systems that help students learn algebra. International Journal of Computers for Mathematical Learning 5(2) (2004) 169-211

The Design of a Tutoring System with Learner-Initiating Instruction Strategy for Digital Logic Problems

Sheng-Cheng Hsu

Graduate School of Engineering Science and Technology,
National Yunlin University of Science and Technology, Yunlin, Taiwan
g9010802@dyuntech.edu.tw

Abstract. In this study, a tutoring system with a learner-initiating instruction strategy is proposed to help learners solve digital logic problems that they input to the system. After the system comprehends the problem inputted by the learner and determines the category of the problem, the system produces a solution plan for the learner. The learner solves the problem by following the plan step by step. After that, the system diagnoses the learner's answer when s/he executes a plan. If the learner's answer is incorrect, the system provides hints for the learner to revise the answer. Empirical evaluation results indicate that the system is effective in tutoring digital logic problem.

1 Introduction

Problem-solving oriented approaches have been widely applied to the instruction of mathematics, science, and engineering. Furthermore, many computer-based tutoring systems based upon problem-solving approaches are developed and applied to various fields successfully [1, 2, 3]. Most of these systems are based on an instructor-initiating instruction strategy and provide pre-designed problems for learners. When learners are asked to solve a problem, the system will instruct the learners what to do. This strategy is appropriate for beginners who develop their competence for problem solving from the very beginning. Nevertheless, such systems are generally not helpful if a learner encounters a problem that does not exist in the pre-designed database. Therefore, it is definitely not enough for an instruction system to be merely equipped with an instructor-initiating instruction strategy. In order to address this question, a tutoring system with a learner-initiating instruction strategy is proposed for helping more advanced learners. In this system, a learner becomes active in posing problems that he is interested in. After comprehending the problem posed by the learner, the system can provide him with relevant instruction materials and hints on problem solving. Currently, the system is applied to a course in digital logic, which is an important course in the department of electrical engineering in a university. In this course, students are taught techniques of digital circuit design including binary, combination and sequential logical circuit design, algorithmic state machine design, and VHDL modeling of digital circuits, etc. [4]. After accepting a problem inputted by a student, the system provides a solution plan and hints to help the student solve the problem.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 728 – 731, 2006.

2 System

The system assists a learner in the problem solving process described by Glass and Holyoak [5]. Three steps followed by the system are explained in the following.

Step1: Comprehend a problem given by the learner: In order to use the system, a learner enters the six components of a problem through the input interface (Fig. 1). If a learner has a problem, “Find the minimum product-of-sums expression for the function, F = BC’D’+BC’D +A’C’D’+BCD’+A’B’CD’, using Karnaugh map.” Firstly, the learner needs to select the problem topic “Boolean function minimization using Karnaugh map.” Then the learner selects the problem goal “Find the minimum POS expression.” Next the learner selects the problem operation “Karnaugh map”. Then the system generates the problem constraints automatically according to the problem operation. Finally, the learner selects the problem object and inputs the content of the object. The problem object is “SOP expression” and the system provides a textbox for learner to input its content: “F = BC’D’+BC’D +A’C’D’+BCD’+A’B’CD’ ” (Fig. 1).

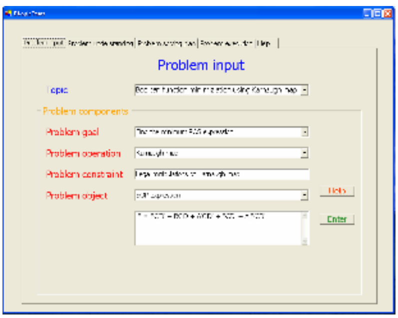


Fig. 1. The problem input interface

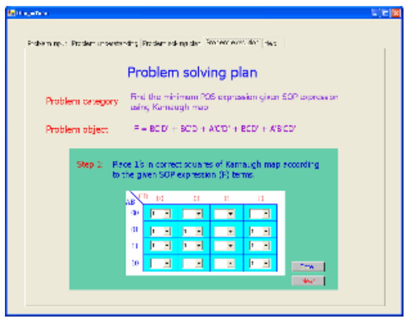


Fig. 2. The plan execution interface

Step2: Generate candidate solution plans for the learner to choose: After accepting the problem components from a learner, the system understands the problem with the help of a knowledge base. If the learner’s input is correct, the system can identify the category of the problem and provides the relevant information (lexicon, definition, theory, and method) to help the learner understand the problem. For example, if the learner does not understand the term “Minimization product of sums”, s/he can select the term, and then the system will provide its definition. After the learner understands the problem, the system retrieves the problem solving plans according to the category of the problem. All these plans can produce correct answers. The learner can choose any plan and the system will provide explanations and examples to assist the learner to comprehend the plan.

Step3: Diagnose the execution of the learner’s chosen plan: After the learner understands the plan, he solves the problem step by step according to the chosen plan (Fig. 2). The system diagnoses the learner’s answer when s/he executes a plan. If the learner’s answer is incorrect, the system provides a hint for him to revise the answer.

3 Experiment

We evaluated the effectiveness of this system in training students to solve digital logic problem. Ninety freshmen in two classes participated in this experiment. One class with 47 students was assigned to the experimental group while the other class with 43 students was assigned to the control group. The following experiment steps were taken: a pre-test, a problem solving experiment, and a post-test. Firstly, the pre-test including thirty questions was used to assess the students' problem solving skills. The results showed no significant difference between the two classes in their pre-test scores (t = 0.108, $p = 0.914 > 0.05$). This implied that the two groups had no difference in their initial problem solving skills. The problem solving experiment lasted for three weeks. The instructor asked the students to solve twenty problems each week. The experimental group solved the problems with the help of the system, whereas the control group solved the problem without the system. After the experiment, a post-test with thirty questions was given and analyzed with an independent sample T-test shown in Table 1. The results showed the scores of the experimental group were significantly higher than those of the control group. This shows that system is useful in training students to solve digital logic problems.

Table 1. The result of independent samples *T*-test

Group	N	Average score	SD	*t*	*p*
Experiment	47	79.7234	9.10495	4.927*	0.00
Control	43	68.1628	12.97237		

*$P < 0.01$

4 Conclusion

In this study, a computer-assisted system with a learner-initiating instruction strategy is proposed to help freshman solve digital logic problems. A learner can input the four components of a digital logic problem and the system comprehends the problem with a knowledge base. Next, the system constructs the problem component content and generates the problem solving plans. The learner chooses a plan and solves the problem step by step. The system can diagnose the learner's answer when s/he executes the plan. In the experiment to evaluate the tutoring effectiveness of the system, 90 freshman participated as the experimental and control groups. By comparing the two groups' performance in the pre-test and the post-test, we conclude that the system is effective in tutoring.

References

1. Chang, J. Chang M., Lin J.-L., Heh, J.-S.: Implements a diagnostic intelligent agent for problem solving in instructional systems. Proceeding of IWALT. (2000)
2. Looi, C. K., Tan, B. T.: A cognitive-apprenticeship-based environment for learning word problem solving. Journal for Research in Mathematical Education. 17, (1998) 339–354

3. Reisslein, J. Atkinson, R. K., Seeling, P., Reisslein, M.: Investigating the presentation and format of instructional prompts in an electrical circuit analysis computer-based learning environment, IEEE transactions on education. 48 (2005) 531-539
4. Roth, C. H.: Fundamentals of logic design. 5th edn. Thomson Brooks/Cole (2003)
5. Glass, A. L., Holyoak, K. J.: Cognition. 2nd edn. Random House, New York (1986)

Reflective Tutoring for Immersive Simulation

H. Chad Lane, Mark Core, Dave Gomboc, Steve Solomon, Michael van Lent, and Milton Rosenberg

Institute for Creative Technologies, University of Southern California
13274 Fiji Way, Marina del Rey, CA, 90292 USA
{lane, core, gomboc, solomon, vanlent, rosenberg}@ict.usc.edu

Abstract. Reflection is critically important for time-constrained training simulations that do not permit extensive tutor-student interactions during an exercise. Here, we describe a reflective tutoring system for a virtual human simulation of negotiation. The tutor helps students review their exercise, elicits where and how they could have done better, and uses explainable artificial intelligence (XAI) to allow students the chance to ask questions about the virtual human's behavior.

1 Introduction

Reflection is widely regarded as a critical meta-cognitive skill for learning. Not surprisingly, researchers have found that human tutors often continue discussing a problem with a student after it has been solved [1]. For problem solving that occurs under hard time constraints (e.g., flight training), reflection is even more important since there is limited time for decision making during practice. Additionally, the tutor must avoid becoming a distraction (e.g., by giving too much advice). In business and the military, it has long been recognized that the *after-action review* (AAR) is when a bulk of the learning occurs [2].

In this paper, we report on an intelligent tutoring system (ITS) that scaffolds reflection activities with the student, such as reviewing salient events from an exercise, discussing ways to improve, and asking questions of entities involved in the simulation. Salient events typically include critical decision points in the simulation, errors of omission and commission, and also instances of positive actions on the part of the student. Question-asking is achieved by the use of explainable artificial intelligence (XAI) [3], a simulation-independent approach to explanation with roots in earlier simulation-dependent explanation research (e.g., [4]). XAI allows entities to describe their reasoning, choices, and state changes through questions derived from their AI models. Using XAI as a standalone tool would be tantamount to unguided discovery [5], and so our ITS monitors its use and gives hints on the best questions to ask. This differentiates our approach from other reflective systems, such as SCoT-DC [6].

2 Planning and Executing a Reflective Tutoring Session

To conduct a reflective tutoring session, it is necessary to both plan ahead and execute the plan. Because the student's answers might reveal new misconceptions,

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 732–734, 2006.

1	T	In the simulation history window, can you select a line where you could have done better?
2	S	**Line 5**
3	T	Why do you think line 5 upset the doctor?
4	S	**He didn't want to move his clinic**
5	T	That's right. You abruptly admitted your goal was to have him move his clinic. What else could you have said?
6	S	**I don't know**
7	T	You needed to build more familiarity. For example, you could have asked how he was doing today. Let's focus on line 10.
8	T	In the XAI window, please ask the doctor some questions and find out how this utterance contributed to failure of the negotiation.
9	S	**Why did the negotiation fail?**
10	D	I no longer trusted the captain.
11	S	**How was your solidarity with the Captain affected here?**
12	D	It decreased.
13	S	**Why did your solidarity with the Captain decrease?**
14	D	I disagreed with the intentions of the Captain expressed in this utterance.
15	T	Good job. In general, you want to avoid revealing sensitive information unless absolutely necessary.

Fig. 1. A reflective dialogue in which the student learns that it was unwise to reveal an undesirable fact before sufficiently building trust. References to "lines" are to the actual exercise (not shown here) and questions asked in the XAI subdialogue were selected from a menu.

it is important that the tutor be able to adapt an AAR plan on the fly. Our system begins its planning process by loading a log file from the target simulation and performing the following steps:

1. **analyze student's exercise:** highlight important events from the exercise that are candidates for discussion.
2. **create agenda:** organize and prioritize the highlighted events.
3. **prepare XAI:** load exercise log, action representations, and natural language generation knowledge (details in [3]).

The first two steps roughly model what human instructors need to do to perform an AAR: judge the student's performance, make decisions about what merits discussion, and finally, decide how they might go about addressing these issues. Currently, steps 1 and 2 require human support, but we are working on automating these tasks as part of an in-game tutor that assesses turn-by-turn choices of the student. The resulting agenda is then passed to a planner and executor that conduct the dialogue – an example appears in figure 1. Prior to this, the student had completed a session with a virtual doctor who is running a clinic in a dangerous location [7]. The student's task is to convince the doctor to move willingly to a safer location through building trust and bargaining.

The reflective tutor's actions are determined by a hierarchical task network planner. Our prototype uses 12 recipes that implement various reflective activities, such as asking the student to identify mistakes (e.g., line 1 of the figure), suggesting ways to improve (line 7), and using XAI to perform "investigations" (lines 9-14). To support XAI, we use a simple model of investigation comprised of a sequence of ideal questions and associated hints that are given when the student fails to ask the right questions. Natural language generation is accomplished via templates and we currently use a keyword-based approach to handling answers to open-ended questions (e.g., line 4).

3 Ongoing and Future Work

We are currently porting our system to a serious game for teaching cultural awareness and negotiation. Although the version of our system presented here assumes no tutor presence during an exercise, our new version coordinates the reflective activities with advice received during the simulation (Katz et. al. refer to this as *distributed* tutoring [1]). We are also exploring more advanced natural language generation and understanding techniques.

Acknowledgments

Special thanks to Bill Swartout and David Traum for their guidance and support. The project described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the U.S. Government, and no official endorsement should be inferred.

References

1. Katz, S., Allbritton, D., Connelly, J.: Going beyond the problem given: How human tutors use post-solution discussions to support transfer. International Journal of Articial Intelligence in Education 13 (2003) 79–116
2. Morrison, J.E., Meliza, L.L.: Foundations of the after action review process. Technical Report 42, U.S. Army Research Institute (1999)
3. Core, M.G., Lane, H.C., van Lent, M., Gomboc, D., Solomon, S., Rosenberg, M.: Building explainable artificial intelligence systems. In: Proc. of the 18th Conference on Innovative Applications of Artificial Intelligence (IAAI06), Boston, MA (in press)
4. Johnson, W.L.: Agents that learn to explain themselves. In: Proceedings of the Twelfth National Conference on Artificial Intelligence. (1994) 1257–1263
5. Mayer, R.: Should there be a three-strikes rule against pure discovery learning? the case for guided methods of instruction. American Psychologist 59(1) (2004) 14–19
6. Peters, S., Bratt, E.O., Clark, B., Bon-Parry, H., Schultz, K.: Intelligent systems for training damage control assistants. In: Proc. of Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC04), Orlando, FL (2004)
7. Traum, D., Swartout, W., Marsella, S., Gratch, J.: Fight, flight or negotiate: Believable strategies for conversing under crisis. In: Proc. of the 5th International Working Conference on Intelligent Virtual Agents. (2005)

Knowledge Engineering for Intelligent Tutoring Systems: Assessing Semi-automatic Skill Encoding Methods

Kevin Kardian and Neil T. Heffernan

100 Institute Road, Computer Science Department, Worcester Polytechnic Institute
Worcester, MA, USA
nth@wpi.edu

Abstract. Building a mapping between items and their related knowledge components, while difficult and time consuming, is central to the task of developing affective intelligent tutoring systems. Improving performance on this task by creating a semi-automatic skill encoding system would facilitate the development of such systems. The goal of this project is to explore techniques involved in text classification to the end of improving the time required to correctly tag items with their associated skills.

1 Introduction

One of the more difficult problems in creating intelligent tutoring systems involves knowledge engineering for a given domain. In order to accurately gauge performance, thousands of questions, referred to hereafter as "items," must be tagged with a fine grained mapping of knowledge components (KCs). A knowledge component represents a set of concepts, skills, or strategies needed to solve items within a domain; a complete set of KCs is referred to as a transfer model. In the ASSISTment Project [1], encoding a set of approximately 300 items with their corresponding KCs has been shown to take as long as three sessions of six to eight hours for two people, which translates to up to 48 person-hours. It is evident that reducing the amount of work required in building these mappings would greatly benefit those who develop subject matter for intelligent tutoring systems.

Inspired by Rosé et al [2], the purpose of this paper is to investigate the possibility that machine learning algorithms may be able to assist the author by suggesting what KCs can be paired with a given question based only on the question text. Rosé reported that "even in cases where the predictions cannot be made with an adequate level of reliability, there are advantages to starting with automatic predictions and making corrections, in terms of reliability, validity, and speed of coding." This paper does not attempt to do any empirical analysis to measure coding time, and instead is first applying Rosé's ideas to our dataset. Specifically, this paper explores the accuracy of selecting several of the most likely KCs, instead of only one. The worth associated with imposing a hierarchical model, beginning with a substantially less specific skill set as a basis for a more specific skill classification, is also investigated.

2 Experiments

The data set used for all experiments consisted of mathematics questions released from the Massachusetts Comprehensive Assessments Systems (MCAS) state test

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 735 – 737, 2006.

combined with questions that were written as part of the assembly of a tutoring system. The original data contained 1258 items, where many of the question texts were tagged with more than one skill. Due to the difficulty of knowing how to evaluate our classifier, we decided to focus only on questions tagged with a single skill, which left us with 878 question text items. Items were assigned tags from the "April" transfer model, which contains 78 different KCs, as well as from the more general MCAS5 transfer model, which contains only five KCs. Experiments were run using the Mallet text classification package[3]; we selected to use a NaiveBayes algorithm for the purposes of simplicity. The total data set was divided at random in every trial, and the items were classified using 90% of the data for the training set.

First Experiment. The first experiment conducted was a study of the advantages associated with selecting more than one KC from the April transfer model for each given item and testing to see if any of the selected KCs was correct. The results of this classification are shown below in Table 1.

Table 1. Accuracies for Top N Choices

N	1	2	3	4	5
Accuracy	0.4096	0.5194	0.5663	0.6005	0.6369
N	6	7	8	9	10
Accuracy	0.6738	0.6875	0.7098	0.713	0.7303

The accuracies shown in Table 1 display the chance that the correct classification for a given item is one of the top N choices generated. These data indicate a substantial improvement over the initial accuracy by adding one or two additional KC selections. However, if the top five KCs or more are selected, each additional selection seems to add between 2% and 3% accuracy. The goal is to narrow the selection of choices to as few as possible while still providing an accuracy that is high enough to assist the user in tagging items.

Second Experiment. Our second experiment was an assessment of our hierarchical classification model; this model first selects a broad category from the MCAS5 transfer model and then classifies it into one of the April transfer model KCs. As is described by Rosé et al [3], the broad category is not selected with absolute certainty, but the results for selecting a single skill hierarchically are expected to be an improvement over selecting one of April transfer model skills directly. The results of this experiment are reported below in Table 2.

Table 2. Basic Accuacy vs. Hierarchical Accuracy

N	1	2	3	4	5
Basic Acc.	0.4096	0.5194	0.5663	**0.6005**	**0.6369**
Hier. Acc.	**0.4519**	**0.5207**	**0.5722**	0.5745	0.6137

We see that a hierarchical classification is more effective when few KC choices are selected. However, when the top four or more KCs for a given item are selected, a direct classification into the April transfer model is more accurate than a hierarchical classification. The rate of improvement in the performance of the hierarchical classification decreases sharply, relative to the basic classification. This could indicate that the effectiveness of hierarchical classification can be severely limited by the accuracy of the top tier when many options are selected from the lower tier possibilities.

3 Conclusions

From an HCI perspective, it is reasonable to suggest several choices to the user when dealing with semi-automatic skill encoding. By limiting the choices presented to a user when they are tagging new items, one could expect a significant decrease in the amount of time taken to enter new items. Based on the results of the first experiment, it is apparent that the greatest benefits are achieved through selecting a small number of top choices. If additional decisions are presented to the user, the domain nears the size of the original transfer model, which would be self-defeating.

The second experiment reinforced Rosé's conclusions that a hierarchical model can improve classification accuracy when selecting a single skill. However, as more skills are selected, this model is surpassed by a more basic classification. This evidence suggests that a hierarchical model would serve as an effective part of a semi-automatic skill coder, but would work most effectively if supplemented in some way. For instance, we could use the confidence of the broad classifier to inform selection of the best classification at more specific levels.

Future work can be done in quantifying the value of presenting more than one KC to the user. There is an inherent threshold that has yet to be discovered regarding the potential time-saving benefits of presenting multiple KCs. Additionally, investigations into more powerful text classification algorithms could prove beneficial.

References

1. Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R, Walonoski, J.A., Macasek. M.A., Rasmussen, K.P. (2005). The Assistment Project: Blending Assessment and Assisting. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press.
2. Rose, C, Donmez, P., Gweon, G., Knight, A., Junker, B., Cohen, W., Koedinger, K., & Heffernan, N. T. (2005). Automatic and Semi-Automatic Skill Coding With a View Towards Supporting On-Line Assessment. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press.
3. McCallum, A. & Kachites (2002). "MALLET: A Machine Learning for Language Toolkit." http://mallet.cs.umass.edu.

Towards Teaching Metacognition: Supporting Spontaneous Self-Assessment

Ido Roll, Eunjeong Ryu, Jonathan Sewall, Brett Leber, Bruce M. McLaren, Vincent Aleven, and Kenneth R. Koedinger

Human Computer Interaction Institute,
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh PA 15213
{iroll, eryu}@andrew.cmu.edu,
{sewall, bleber, mclaren, aleven, koedinger}@cs.cmu.edu

Abstract. The Self-Assessment Tutor (SAT) is an add-on component to Cognitive Tutors that supports self-assessment in four steps: *prediction, attempt, reflection*, and *projection*. The SAT encourages students to self-assess their ability spontaneously while problem solving, and to use help resources accordingly. For that reason its episodes precede the students' work with the Cognitive Tutor, which itself remains unchanged. The SAT offers detailed feedback and help function to support the Self-Assessment process. A complementary instruction is given to students before working with the SAT. We hypothesize that working with the SAT will encourage students to self-assess on subsequent problems requiring similar skills, and thus will promote learning. A classroom evaluation of SAT is currently in progress.

1 Introduction

Supporting students' metacognition while working with Intelligent Tutoring Systems contributes to deep, meaningful learning of the relevant domain knowledge [1; 2] and can promote future learning in various domains and learning environments by equipping students with better metacognitive skills [6; 7].

One such skill is self-assessment, i.e., the ability and tendency of students to evaluate correctly their knowledge level. Self-assessment can be used by students to choose their actions and monitor their progress [3], and by the tutoring system to update its assessment of the student [8]. Research shows, however, that students are not good at self-assessing their knowledge [4].

To address that, and as part of an overall metacognitive suite, Gama [3] prompts students to evaluate their knowledge level before each problem, and to reflect on their assessment once they are done. Zapata et al. [8] allow students to self-assess their knowledge by describing their experience with similar concepts.

2 The Self-Assessment Tutor (SAT)

The Self Assessment Tutor (SAT) we describe here was built using the Cognitive Tutors Authoring Tools – an environment for authoring tutors by demonstration [5],

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 738 – 740, 2006.

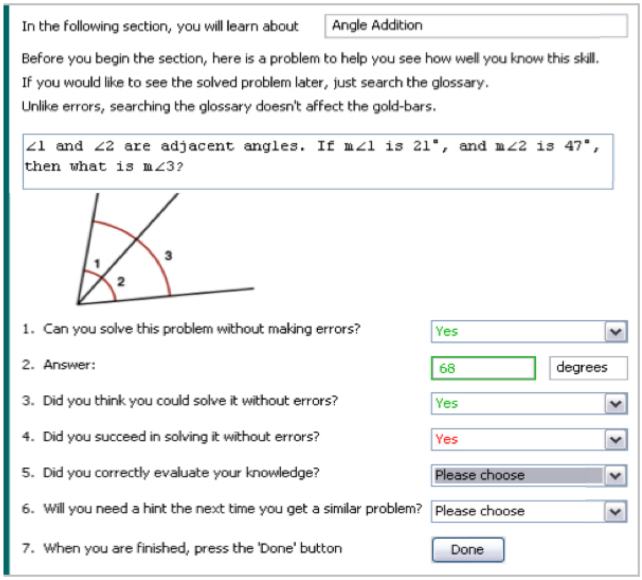


Diagram 1. The Self-Assessment Tutor: prediction (q. 1), attempt (q. 2), reflection (q. 3-5) and projection (q. 6)

and is an add-on component to existing Cognitive Tutors. Each unit of the Geometry Cognitive Tutor (e.g. Angles) is composed of sections (e.g. angles between parallel lines) during which students practice a specific set of skills (e.g. same-side interior angles). SAT does not prompt students to self-assess their ability on every individual problem since it is very time consuming, and might become annoying to the students. In addition, since students are being prompt to self-assess, they are not engaged in that behavior spontaneously. To encourage *spontaneous self-assessment*, SAT adds a self-assessment preparatory activity before each set of problems, and does not interfere with the problem-solving process itself. In practice, between any two sections of the Cognitive Tutor (that remain intact), students engage in a self-assessment episode, in which they assess their ability on the relevant set of skills.

The self-assessment process. Each self-assessment episode includes problems on which students assess their ability – one problem per skill (each episode includes 4-5 such skills). In designing the interface of the SAT we used similar principles to those detailed in Gama [3]. Each problem includes the following steps: (1) *Prediction* – how well do I think I can solve this problem? (2) *Attempt* – what is the answer to the problem? (3) *Reflection* – how well did I do? Did it match my prediction? (4) Projection – what does this imply about my ability to solve problems using similar skills in the future? Will I need help the next time I attempt a problem requiring a similar skill?

Each of these steps (besides attempting the problem) is scaffolded with drop-down menus (see diagram 1). Additional support is made available in on-demand hints (not seen in diagram). Using these steps, we try to relate the current self-assessment experience to relevant future situations.

Feedback on self-assessment. Giving feedback on self-assessment should be based on the students' assessment of themselves, not the system's assessment of the student. On this assumption, SAT traces students' attempts and assessment and gives feedback according to the following principles:

– Where possible, the tutor should base feedback on previous responses (e.g., feedback to question 3, "Did you think you could solve it without errors?" is based on their answer to question 1).
– When the student reports a need for help, it should be given. The system does not assume that the student knows unless the student reports so.
– When several answers are possible, the tutor should allow for all of them.

Self-assessment instruction. Before working with the SAT, each student receives an instruction through a movie, describing the importance of self-assessment and demonstrating the interface of the new tutor.

The SAT is currently being evaluated in a classroom study, and is well received by the students. We hypothesize that it will contribute to learning since students would be more aware of their knowledge level, and would be engaged in self-assessment spontaneously more often.

Acknowledgements. We express thanks to Ido Jamar, Jo Bodnar, Mari Murphy, Sabine Lynn, Kris Hobaugh and Dale Walters for their help carrying out this study.

References

1. Aleven, V., & Koedinger, K.R. (2002). An effective metacognitive strategy: learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science* 26(2), 147-79.
2. Bunt, A., Conati, C., & Muldner, K. (2004) Scaffolding self-explanation to improve learning in exploratory learning environments. *in proceedings of the 7th International Conference on Intelligent Tutoring Systems,* 656-67. Berlin: Springer-Verlag.
3. Gama, C. (2004) Metacognition in Interactive Learning Environments: The Reflection Assistant Model. *in proceedings of the 7th International Conference on Intelligent Tutoring Systems,* 668-77. Berlin: Springer-Verlag.
4. Glenberg, A.M., Wilkinson, A.C., & Epstein, W. (1982). The Illusion of Knowing: Failure in the Self-Assessment of Comprehension. *Memory and Cognition* 10, 597-602.
5. Koedinger, K.R., Aleven, V., Haffernan, N., McLaren, B.M., & Hockenberry, M. (2004). Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. *in proceedings of the 7th International Conference on Intelligent Tutoring Systems.*
6. Roll, I., Aleven, V., McLaren, B.M., Ryu, E., Baker, R.S.J.d., & Koedinger, K.R. (to appear) The Help Tutor: Does Metacognitive Feedback Improves Students' Help-Seeking Actions, Skills and Learning*? to appear in proceedings of The 8th International Conference on Intelligent Tutoring Systems.* Berlin: Springer-Verlag.
7. Roll, I., Baker, R.S., Aleven, V., McLaren, B.M., & Koedinger, K.R. (2005) Modeling Students' Metacognitive Errors in Two Intelligent Tutoring Systems. *in proceedings of User Modeling 2005,* 379-88. Berlin: Springer-Verlag.
8. Zapata-Rivera, J.D., & Greer, J.E. (2004). Interacting with Inspectable Bayesian Student Models. *Int J of Artificial Intelligence in Education* 14(2), 127-63.

Automated Vocabulary Instruction in a Reading Tutor

Cecily Heiner[1,2], Joseph Beck[2], and Jack Mostow[2]

[1] School of Computing, University of Utah, Salt Lake City, UT USA 84112
cecily@cs.utah.edu
Tel.: 801-792-4352

[2] School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213
joseph.beck@cmu.edu, mostow@cs.cmu.edu

Abstract. This paper presents a within-subject, randomized experiment to compare automated interventions for teaching vocabulary to young readers using Project LISTEN's Reading Tutor. The experiment compared three conditions: no explicit instruction, a quick definition, and a quick definition plus a post-story battery of extended instruction based on a published instructional sequence for human teachers. A month long study with elementary school children indicates that the quick instruction, which lasts about seven seconds, has immediate effects on learning gains that did not persist. Extended instruction which lasted about thirty seconds longer than the quick instruction had a persistent effect and produced gains on a posttest one week later.

1 Introduction

Many tutorial domains require learning vocabulary. We present an experimental comparison of three ways to introduce new words. In this paper, we automate a published sequence for human-administered vocabulary instruction and evaluate the effectiveness of the automated sequence compared. As baselines, we use a null treatment with no instruction, and a quick treatment consisting of exposure to the definition. For the quick treatment, we included the definition of the word as part of the story text, immediately after the sentence containing the vocabulary word. For third condition, we included the same instruction as the quick treatment plus extended instruction at the end of the story. The extended instruction comes from experts in vocabulary instruction [1] who suggest the following steps for classroom teaching:

1) *Give the context of the vocabulary word in the story.* The Reading Tutor displayed and read a sentence of the form "In this story, ..." followed by the story sentence containing the vocabulary word.
2) *Repeat the word to create a phonological representation of the word.* The Reading Tutor displayed the word, read the word, and asked the student to repeat the word.
3) *Explain the meaning of the word.* The Reading Tutor displayed and read a sentence with the vocabulary word followed by "means" followed by the definition.
4) *Provide examples of the word in other contexts outside the story.* The Reading Tutor displayed and read one additional hand-crafted sentence containing the vocabulary word.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 741 – 743, 2006.

5) *Allow the children to practice the word in new contexts.* The Reading Tutor displayed and read another hand-crafted sentence containing the vocabulary word and asked "Does this make sense?". The student could click on the words "yes" and "no" to respond, but received no feedback.
6) *Say the word again to reinforce the phonological representation.* The reading Tutor displayed the word, read the word, and asked the student to repeat the word.

We will call these six steps of vocabulary instruction the Beck battery.

Each story contained three selected vocabulary words with each of the three words assigned randomly to a different treatment condition: null, quick, and Beck battery. In the Null condition, the student simply encountered the word in context without instruction. In the Quick condition, students received in-story instruction in the form of a handcrafted definition after the sentence where the vocabulary word first appears. On the word assigned to the Beck condition, students received the same kind of hand-crafted definition as in the quick condition plus the previously described extended instruction based on the Beck battery.

Each student was tested on each word three times: in a pretest, in a posttest, and in a delayed posttest. The pretest contained three questions that required matching a vocabulary word to its definition. In addition to the same matching questions on the pretest, the immediate posttest and delayed posttest contained cloze (fill in the blank) questions. This project is implemented in the context of Project LISTEN's Reading Tutor which uses automatic speech recognition to assist children (mostly ages 6-10) in reading aloud.

2 Evaluation

Students used Project LISTEN's Reading Tutor [2] during July 2005 reading clinics at two elementary schools. We analyzed trials where the student had completed all of the items in the pretest, the immediate posttest, and the delayed posttest for all three words in a story. This left a dataset with fourteen students who had read a total of eighteen stories (four distinct texts) for a total of eighteen trials. In each of the conditions with instruction, the students answered three more matching questions correctly on the Immediate Posttest than they did on the pretest, but only Beck instruction resulted in gains that lasted a week or longer on the matching task (five questions). With the cloze questions, the gains were even more remarkable; students who received Beck instruction answered almost twice as many delayed posttest questions correctly compared to other conditions (ten instead of six).

We used a logistical regression in SPSS to test statistical significance because the data did not have a normal distribution and trials were correlated [3]. The dependent variable was whether or not a posttest item was correct. The other factors were the delay, the treatment, the question type, the performance on pretest, and the interaction between treatment and delay. One output of a logistical regression is a set p-values that represent the probability that the results happened by chance. The question type and delay were both statistically insignificant predictors. Beck instruction was more effective than no instruction ($p=0.058$). Beck instruction was also more effective than quick instruction. Whether the student received Beck instruction was a stronger predictor of posttest performance than whether the pretest question was correct.

Acknowledgements

This work was partially supported by the National Science Foundation, under ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the official policies, either expressed or implied, of the sponsors or of the United States Government.

References

1. Beck, I.L., M.G. McKeown, and L. Kucan, *Bringing Words to Life*. 2002: The Guilford Press.
2. Mostow, J. and G. Aist, *Evaluating tutors that listen: An overview of Project LISTEN*, in *Smart Machines in Education*, K. Forbus and P. Feltovich, Editors. 2001, MIT/AAAI Press: Menlo Park, CA. p. 169-234.
3. Menard, S., *Applied Logistic Regression Analysis.* Quantitative Applications in the Social Sciences, 1995. **106**.

Visualizing Errors and Its Evaluation

Hidenobu Kunichika[1], Tsukasa Hirashima[2], and Akira Takeuchi[1]

[1] Dept. of Artificial Intelligence, Kyushu Institute of Technology
680-4 Kawazu, Iizuka, Fukuoka, 820-8502 Japan
{kunitika, takeuchi}@ai.kyutech.ac.jp
[2] Dept. of Information Engineering, Hiroshima University
1-4-1, Kagamiyama, Higashi-hiroshima, Hiroshima 739-8527 Japan
tsukasa@isl.hiroshima-u.ac.jp

Abstract. The aim of our research is to realize a system for supporting English composition in second language learning which allows learners to compose some sentences and gain appropriate awareness of their errors in the sentences with indirect information from the system. This paper proposes a method when and how the system provides stimuli to the learners, describes its implementation briefly, and reports validity of the method. We adopt a method of visualizing learners' errors as indirect information for the learners.

1 Introduction

Children make mistakes and learn from them. Learning is much more meaningful if the child is allowed to experiment on his own rather than listening to the teacher lecture [4]. In order to learn from mistakes, learners need to first be aware of the mistakes. Ways to be aware are the three types: by themselves, by indirect information and by direct information.

The aim of our research is to realize a system to support English composition in second language learning which allows learners to gain appropriate awareness of mistakes by giving indirect information. Its target learners are beginners in English: we assume that the vocabulary and grammar used by the learners are basic and taught in Japanese junior high school. Learners make various sorts of errors in English composition. One type of errors is discrepancy between thinking and writing (DTW) in which a learner writes is different from what she wants to represent. Current intelligent computer-assisted language learning systems [1] do not support learners in the correction of such DTW errors except by informing them that "the answers are incorrect". The goal of our approach is to emphasize DTW errors by animation visualizing the content of the sentences. We expect that the emphasizing such DTW triggers a conflict awareness enabling the learner to reflect on her activities and revise the sentences in order to reduce the conflict. The reason we select the method "error visualization [2] by animation" to inform learners of errors is that we will image what we want to write first, and then we write them in English actually. We believe as expression of errors is close to how we think, the expression always remain in our memory.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 744 – 746, 2006.

2 Error Visualization and the Learning Support System for English Composition

(1) Error Visualization

A sentence which corresponds to one event is composed of objects, their attributes and the case elements of the event. Therefore, DTWs appear as differences, i.e. lack, excess and replacement, between the components of a sentence a learner writes and the components of what she wants to represent. Here, we describe these types of DTW errors and a method of visualizing them. *Information lack* means cases where a learner does not represent the necessary information in the sentences she has composed. If a learner composed a sentence without a necessary information, our system generates an animation with particular values for the lacked information in order to realize the errors. For example, when a learner should input "a short tree" but instead uses "a tree", our system will show a picture of a tall tree. *Information excess* errors are the cases in which sentences have an excess of attributes for the objects and case elements. When such cases occur, the system also generates animation which represents the content of a sentence as it is. *Information replacement* error is that a learner writes other objects, attributes, case elements or events than what the learner actually wants to represent. For the error, the system also generates an animation as the learner writes.

(2) The Learning with the System

Our system consists of a natural language processing module (NL) [3], a knowledge processing module (KP) including a conceptual dictionary, and an animation generation module (AG). The NL interprets English sentences which are inputted by users, and extracts “case frames” from the sentences. The KP generates an internal representation called “state transition information”. Then, the AG shows animations to the users following to the state transition information.

The system works in two stages: an authoring stage and a learning stage. In the learning stage, the system provides the *original* animation for a learner, which she should represent in English later. When the learner composes sentences, the system extracts “*learner’s* case frames”, identifies DTW errors, and then, generates "*learner's* state transition information" which reflects the errors. After that, the system shows an animation corresponding to the *learner's* state transition information. The animation represents how the sentences are interpreted by the system. The learner is expected to compare the animation with the *original* animation, have conflict awareness if the animations have any discrepancy, and reconsider the sentence composed by the learner in order to reduce the conflict.

3 Conflict Awareness and the Applicability of Our System

We have investigated the following two points to confirm whether our method works well or not.

(1) Can our system grasp the contents of sentences inputted by learners?
(2) Can learners be aware of errors?

We showed each subject original animation which included one event, and asked the subject to express the animation in English. If the composed sentence has any DTW errors, the subject proceeds to the next step. Our system showed the subject's animation generated from the composed sentence. After that, we asked the subject to identify the difference between the original animation and the subject's animation, and to re-compose the sentence again. We used three animations, and the number of subjects was 18. They were graduate and undergraduate students.

As the results of the investigation, the followings are shown.

(1) Can our system grasp the contents of sentences inputted by learners?
The subjects composed 90 sentences in total. In the sentences, 83 sentences (92%) are interpreted correctly and the system generated animations from the interpreted sentences. The rest, that is, 7 sentences, are correct sentences but not interpreted because the sentences have words which are not stored in the conceptual dictionary.

(2) Can learners be aware of errors?
The subjects were aware of 66 errors (92%) out of 72 errors. Since almost errors were been aware of, we can say that our method is useful for allowing learners to gain appropriate awareness of errors. The main reason of failing to be aware of errors is that the animation generated from the subject's sentence is very close to the original animation.

4 Conclusions

This paper described a method of visualizing DTW errors in order to allow learners become aware of their own errors by themselves. As the result of our investigation, we found that our system can grasp the contents of almost sentences inputted by subjects though many sentences were inputted. Furthermore, the subjects were aware of almost DTW errors visualized by the system. Therefore, we can say that our method is useful to help learners be aware of DTW errors by themselves.

At present, although the system emphasizes errors of information lack in animation, in the case of information excess and information replacement, the system generates animation as the sentence is. The remaining issues are to investigate the way of giving original animation for learners who do not understand an appropriate verb and to realize a method of confirming the visibility of errors of information excess and information replacement.

References

[1] Gamper, J. & Knapp, J.: A Review of CALL Systems in Foreign Language Instruction, Proc. of AIED2001, pp.377-388 (2001).
[2] Hirashima, T., Horiguchi, T., Kashihara, A. & Toyoda, J.: Error-Visualization by Error-Based Simulation, Intl. Jl. of AIED, Vol.9, pp.17-31 (1998).
[3] Kunichika, H., Takeuchi, A. & Otsuki, S.: An Authoring System for Hypermedia Language Learning Environments, and its Evaluation, Proc. of ICCE95, pp.73-80 (1995).
[4] Piaget, J.: To Understand Is To Invent. NY: The Viking Press, Inc. (1972).

Applying an Assessment Agent to Evaluate the Learning Effectiveness

K. Robert Lai and Chung Hsien Lan

Department of Computer Science and Engineering, Yuan Ze University, Taiwan
krlai@cs.yzu.edu.tw, chlan@nanya.edu.tw

Abstract. This study presents an assessment agent that models collaborative learning as multi-issue agent negotiation using fuzzy constraints for peer assessment. The proposed method aggregates student marks to avoid the subjective judgments and unfair assessments. Experimental results indicated that students and instructors generally acknowledged the peer assessment as a valuable process for enhancing student critical thinking skills and improving learning performance.

1 Introduction

Peer assessment describes an encounter between equals in professional education, qualifications, and positions in which one's pursuits are examined, discussed, or critiqued by the others [1]. Researchers have investigated the effectiveness of peer assessment systems in various learning scenarios [2][3]. However, some issues associated with reliability hindered the acceptance of the assessment process. Potential biases such as friendship, gender or race cause students to mark good performance down or vice versa, thus discrediting peer assessment's reliability and validity [4]. Furthermore, students often lack ability and experience in peer assessment and encounter difficulties in interpreting assessment criteria. These obstacles often result in subjective and unfair assessments and, thus, students' reflection and learning effectiveness are not enhanced.

Assessment agent [5] is an effective tool that supports the assessment process through which the proposed methodology aggregates students' marks to reduce personal bias. In this system, students define individual fuzzy membership functions based on their evaluation concepts and agents facilitate student-student negotiations during the assessment process. The whole process of peer assessment can be conducted anonymously and managed via Internet. By applying the assessment agent, agents can reach mutually acceptable agreements that overcome the unfair assessment as a result of students' various degree of understanding the assessment criteria.

2 Assessment Agent

Assessment agent is a computational model that relies on multi-issue agent negotiation and fuzzy constraints for peer assessment learning [5]. By using this model, students

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 747 – 749, 2006.

are able to construct personal evaluation criteria using fuzzy membership functions and reach an agreement for peer assessment via agent negotiation. Following the framework of the peer assessment process [6], the workflow of the assessment agent is divided into the following steps (Figure 1).

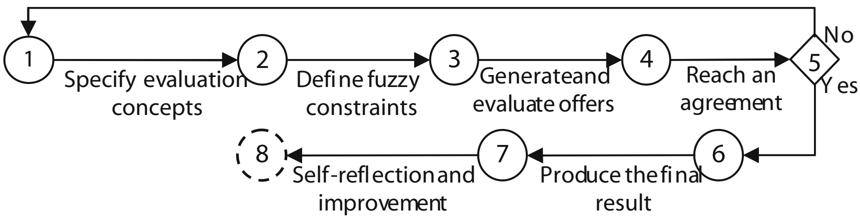


Fig. 1. The workflow of the assessment agent

Multi-issue agent negotiation is supported in the assessment process. In particular, the core methodology of the assessment agent focuses on using fuzzy constraints to represent personal interests and applies negotiation strategies in making concessions or trade-offs between different possible values for negotiation issues. By applying constraints to express negotiation proposals, the model can execute the negotiation with increased efficiency and determine final results for overall assessments. The scores achieved through this process of peer assessment are more reliable than the ones assigned by an individual. In addition to eliminating individual bias, student learning effectiveness is enhanced through interaction with students.

3 Effectiveness of Peer Assessment

An experiment was designed to demonstrate the usefulness of the assessment agent. 92 students were assigned to 26 teams which were randomly divided into two groups. All students were required to submit assignments that analyze the assigned project. One group included 13 teams was the experimental group that participated in peer assessment using the assessment agent, whereas the other group did not take part in any peer assessment activities.

According to Fig. 1, students in the experimental group were allowed to construct their own fuzzy membership functions for the evaluation concepts, *Completeness*, *Correctness* and *Originality*, to assess assignments through three rounds. After defining fuzzy constraints, the assessment agent begins to generate and evaluate ideal offers through negotiate strategies. The team submitting their assignment to be assessed by other teams receives final scores, understands the peer assessments, and can reflect upon the assessment and revise the assignment.

Paired t-test analysis for performance from round 1 to round 3 indicates that the improvement in learning for the experimental group was significant (completeness: *t-value*=5.04, p<0.001; correctness: *t-value*=5.25, p<0.001; originality: *t-value* =4.66, p<0.001). Furthermore, the instructor's assessment is utilized to evaluate the performance of the two groups. By t-test analysis, the difference between the two groups was significant (completeness: *t-value*=2.13, $p<0.05$; correctness: *t-value*=2.43, $p<0.05$; originality: *t-value* =2.28, $p<0.05$). Furthermore, quantitative scores indicate that students in the experimental group improved their performance via peer assessment.

Finally, students provided feedback via a questionnaire. Questionnaire results indicate that students regarded the assessment agent as a satisfactory approach for flexibly assessing peer assignments, receiving fair feedback, and improving their performance. Although some students considered it time and effort consuming, most students believed that the system helped them to reflect on and improve their learning activities.

4 Conclusion

This study has presented an assessment agent for peer assessment learning. By using this web-based system, students are able to reach an agreement for peer assessment via agent negotiation. Analysis of experimental data indicated that students generally have a positive attitude toward this process facilitated by the assessment agent and significantly improved learning performance. Students acknowledged the positive impact of the peer assessment process on understanding the evaluation concepts and improving project quality. In addition, students also agreed that the assessment agent is flexible and benefits the learning process.

References

1. Hunter, W. M., Schmidt, E. R., & Zakocs R.: Evaluation of a peer assessment approach for enhancing the organizational capacity of state injury prevention programs. Journal of Public Health Management and Practice, Vol. 11 (2005) 29-36
2. Lin, S. S. J., Liu, E. Z. F., & Yuan, S. M.: Web-based peer Assessment feedback for students with various thinking styles. Journal of Computer-Assisted Learning (2001) 420-432
3. Davis, R., & Berrow, T.: An evaluation of the use of computer supported peer review for developing higher-level skills. Computers and Education, Vol. 30 (1998) 111-115
4. Li, L., & Steckelberg A. L.: Using peer feedback to enhance student meaningful learning. Association for Educational Communications and Technology (2004) 576-587
5. Lai, K. R., & Lan, C. C.: Development of an assessment agent to promote the learning effectiveness in a computer supported collaborative learning environment. IEEE International Conference on Advanced Learning Technologies (2005) 354-358
6. Sitthiworachart, J., & Joy, M.: Web-based peer assessment in learning computer programming. The 3rd IEEE International Conference on Advanced Learning Technologies (2003) 180-184

TELAT: An Educational Authoring Tool

Seong Baeg Kim, Hye Sun Kim, and Cheol Min Kim

Department of Computer Education, Cheju National University,
66 Jejudaehakno, Jeju-si, Jeju-do, Korea
sbkim@cheju.ac.kr, tonaa07@nate.com, and cmkim@cheju.ac.kr

Abstract. Recently, a lot of new educational games have been exploited rapidly due to the advance of information technology. However, it is unpractical for instructors to apply the existing educational games as they intend. In this paper, we propose an educational authoring tool called *TELAT* (Template-based Entertainment Learning Authoring Tool) for an individualized learning on the web, which can be easily developed by instructors. In general, it's known that game-style learning has the advantage in increasing the learner's interest and immersiveness. However, it's not difficult to find out that if someone has completed certain learning game, he/she tends to lose the interest on the game and the game becomes a useless one. To tackle the problem, we design and develop a specific authoring tool, which enables instructors to develop the various educational games according to the levels of learners and learning contents.

1 Introduction

Many educational softwares have been developed on the web because web-based learning systems are easy to provide self-oriented, customized, and interactive learning. Internet's convenient access and variety of material like multimedia is extending traditional instruction by enabling a profound degree of adaptivity and interactivity [1]. In addition, cooperative learning with certain students in a foreign country is feasible. For example, there have been ways to realize the functionality or environment for collaboration on the information network [2]. It is true that a computer game has not been generally used for educational purposes. However, a computer game, which keeps the characteristics of educational playing or learning in playing, can be an alternation for the existing educational methods.

However, by accomplishing an educational game, learners get bored as they repeat it. To help learners keep studying, instructors have to find another game. However, the time spending for preparing a new game is regarded too long and ineffective. Furthermore the learners get tired and loss precious time by doing it. By considering the fact, in this research, we design and develop a tool for developing an educational game to draw active learner's participation and interest as corresponding with intentions of examiners by transforming all the elements as well as questions shown on the game screen to instances.

2 *TELAT* Architecture

In this part, a template-based educational authoring tool for making an educational game is described. Teachers can perform game style learning in the class to provide

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 750 – 752, 2006.

an interesting lecture. However, it is very difficult to offer more suitable game to learners. Moreover, even though learners have much interest when they play a game at the first time, they're getting bored as they play it repeatedly.

It's very appealing to develop a template based authoring tool for teachers, which is easy to produce educational games in accordance with specific situations of learners without making additional effects for learning multimedia programs. To meet the various requirements of instructors and learners, many elements should be considered such as the time to solve questions, the contents of learning, types of games, hints, points, characters or backgrounds, and so on. By paying profound attentions to the elements mentioned above, it is possible for instructors to construct customized game instances. The one of the significant characteristics of this system is organizing a game template by web basis regarding these instances as variables by teachers.

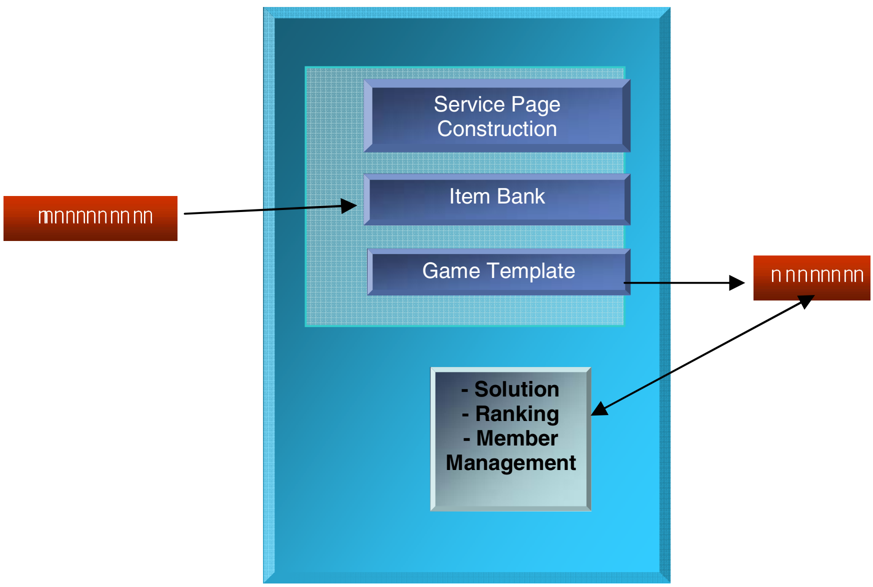


Fig. 1. The Key Components of *TELAT*

Figure 1 shows the main components of *TELAT*. An instructor can develop items, which are stored in an item bank database. Also, an instructor can set the preferences using 'Service Page Construction' interface regarding the settings of game components such as screen, animation, item, and so on. The screen settings include the elements to be displayed on the screen. For example, the game developers can choose the background images of the game through search or preview and also assign the position and the number of button. Besides, they can control various resources including game points to be displayed on the screen. The settings of animation enable to make the dynamic effects combined with character, icon, sound, etc. Also, game developers can choose the styles of the item to be solved by students. Each game is produced by the procedure that integrates an item and a game template into a game instance. After generating a game instance, it is saved with history information through XML data transmission as the form, which is easy to modify. It is possible for students to execute a game instance after finishing the procedure of a simple installation. Each game instance corresponds to one file in a directory.

Up to now, we have developed a few games in a simple form such as crossword puzzle, O/X problem, and guessing word, etc. Also, these games are combined with a learning video, which has the feature that is basically difficult to bring an interaction with learners. By providing a mechanism to insert a game instance into a learning video at a specific position, the interaction with learners would be enhanced. *TELAT* has the capability to incorporate a new game through a minor modification for an educational purpose. Thus, we're making *TELAT* extend to provide templates for games such as tetris. Furthermore, we're finding out a possibility to make an intelligent game template using ontology and semantic web technology. For example, it would be plausible to make a crossword puzzle instance using the ontology database, which represents effectively relations between words.

3 Conclusion

We are extending *TELAT* to incorporate a popular game, which can be used in learning and developed in template form. The key advantages of *TELAT* are at the following. First, learners can study in interesting and funny circumstances through game-style e-learning. Also, leading to the sense of rivalry and the sense of cooperation, the productivity on learning will be enlarged. Second, *TELAT* is able to generate a game instance based on template. Therefore, the instructors, who do not have the basic knowledge about making games, can easily provide various games to learners without time-consuming efforts.

Acknowledgements

This study was partly supported by DCRC (Digital Contents Cooperative Research Center) and Jeju Cultural Technology Education Center NURI (New University for Regional Innovation) Program, Cheju National University, Korea.

References

1. Trude Heift and Devlan Nicholson, Theoretical and Practical Considerations for Web-Based Intelligent Language Tutoring Systems, In Proceedings of the 5th International Conference, ITS'2000 (2000)
2. Tomoko Kojiri and Toyohide Watanabe, Collaborative Learning Environment which Enforces Students to Induce Ideas, In Proceedings of the 5th International Conference, ITS'2000 (2000)

A Modular Architecture for Intelligent Web Resource Based Tutoring Systems*

Sergio Gutiérrez, Abelardo Pardo, and Carlos Delgado Kloos

Department of Telematic Engineering
Carlos III University of Madrid, Spain
{sergut, abel, cdk}@it.uc3m.es
www.it.uc3m.es

Abstract. In this document a platform for the development of ITSs is presented. The objective of this architecture is to provide a tutoring platform with a modular structure suitable to accommodate different sequencing paradigms through a common functional interface. The platform has been tested with positive results.

1 Introduction

The World Wide Web (WWW) has become the biggest source of educational material, and web browsers have become a standard tool for information access and search.

In this paper, an architecture called SIT, for the deployment of web-based tutoring systems is presented. It is oriented towards the reuse of learning units (digital learning content such as web exercises, commented photographs, etc) accessible through the web. Using a specific adaptive sequencing strategy and a set of URLs pointing to readily available learning resources, the presented architecture simplifies the design of an adaptive learning module.

Two ways of adaptation are considered: through the content (dynamic content, i.e. parametric exercises) and through the sequencing of learning units. The architecture presented in this paper assumes a clear separation between content and sequencing and is focused on providing generic support for sequence adaptation.

Content is assumed to be produced by external sources. Its form may range from simple static documents to arbitrarily complex and adaptive units. The presented architecture assumes that this content is divided into self-explanatory units and available in a web server. The platform proposes a framework in which multiple sequencing strategies may be easily deployed and make use of available content resources. Sequencing strategies may range from trivial linear traversal to sophisticated adaptive strategies. The paper presents, as an example, a graph based paradigm for sequence adaptation.

The system assumes also a clear distinction between the pedagogic aspects of e-tutoring (learning content, adaptive sequencing) and the technical ones (connections, client-server model). All information related to content sequencing is contained in a data structure within the tool. Such data structure only takes external URLs as references to learning resources. The tool decides the right sequence of such URLs based on the given strategy and the observed user behavior.

* Work partially funded by Programa Nacional de Tecnologías de la Información y de las Comunicaciones, project TSI2005-08225-C07-02 and Ministry of Education grant BES-2003-0898.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 753–755, 2006.

The objective of this architecture is to provide a tutoring tool with a simple web interface for students, modular and easily extensible sequencing capability using a potentially large set of previously designed external learning units.

Other proposals for the developing of intelligent tutors from the recombination of existing material have been published before. One of the most interesting ones is [1], that reuses the best parts of heterogeneous materials on the web, and is similar to the work presented here but with a greater emphasis on the ontologies developed for the integration of different material rather than on adaptation of the sequencing of that material. Another interesting paper is [2], where tutors are made combining different task-based expert systems. The approach is simpler than ours at first, but has a smaller level of granularity.

2 General Architecture

The general architecture of the platform is depicted in Figure 1.

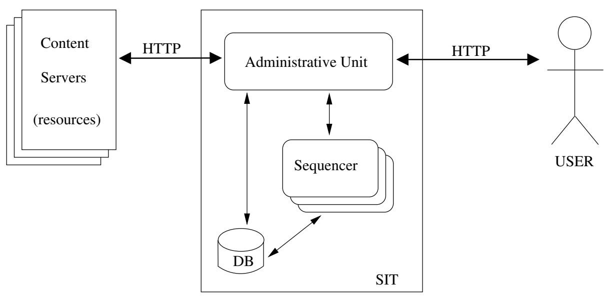


Fig. 1. General architecture

The architecture is based on the reuse of learning resources already available through a web interface with either static or dynamic content. These resources are organized within the tool as modules. A module is a collection of learning units (i.e. resources) with one or several learning goals. Module structures are fully described in XML files. The platform allows for multiple sequencers to coexist. The syntax of a module description depends on the sequencer used to process it. Different sequencers may share an entire module description or parts of it.

The resources (identified solely by a URL) are shown to the student as a regular web page including two additional buttons with labels “Advance” and “Logout”. The former instructs the tool to decide which resource is to be shown next (see below), while the latter stores the current state and terminates the session. Students may continue the activities at any time.

The content server contains the resources to be delivered by the system to the student. It can be a set of remote machines anywhere in the Internet, or a private one in the same LAN as the sequencing server. Any resource that can be accessed (i.e. downloaded) through an URL is suitable to be used. It is important to note that in the case of dynamic web pages (PHP, JSP, etc) only the resulting HTML code for the HTTP request can be downloaded, not the producing code.

The Administrative Unit is the central part of the system. It communicates the user, the Sequencer module and the content servers. This unit takes care of all the details of the HTTP communication such as sessions, requests, and so on. It is also responsible of forwarding the learning units (i.e. resources) specified by the Sequencer to the user, and collecting all the data to feed the Sequencer.

The database contains three main tables. The first one stores data about the users (i.e. login, password, profile). The second one stores data about the sessions already open in the system (who is running which learning module, etc). The last table stores historical information about the platform (i.e. users, modules, events, timestamps, data sent or received) for data mining.

The Sequencer module is responsible for all decisions related to the sequencing of learning units, so it must take care of things such as learning goals for the students or expected difficulty level for next activities. It must also keep track of all required historical data such as learning units already delivered to the student or elapsed time between units. This information is stored in additional tables part of the relational database. It must be noted that the platform's historical table (see Section 2) cannot be used, because it is designed for another purpose and stores very low-level data.

The student model is included in the Sequencer. In this architecture, the same sequencing strategy with two different student modeling approaches implies two different Sequencers. There is no clear separation yet between these two domains, and there is no defined interface for the interchange of information between the user model and the rest of the system. This interface is assumed to be part of the sequencer implementation.

The communication between the Administrative Unit and the Sequencer is done through a simple three-function interface: init, close, and processRequest. The last one gives the sequencer all the information needed (user, module, results from last activity, etc) by it to decide which learning unit is to be delivered next.

A proof of concept experiment and results are fully described in [3]. The platform has significantly evolved to the architecture described in this document.

Two more experiments are taking place now using this architecture. Results will be published soon.

References

1. Hrin, D., Sala, M., Pompidour, P.: Evaluating and revising courses from web resources educational. In: Intelligent Tutoring Systems, Springer-Verlag (2002)
2. El-Sheikh, E., Sticklen, J.: Generating intelligent tutoring systems from reusable components and knowledge-based systems. In: Intelligent Tutoring Systems, Springer-Verlag (2002)
3. Gutiérrez, S., Pardo, A., Delgado Kloos, C.: An adapting tutoring system based on hierarchical graphs. In: Adaptive Hypermedia. Volume 3137 of Lecture Notes in Computer Science., Springer (2004)

The Design of an Intelligent Software Architecture Based on a Generic eLearning Environment

Ana M. Gonzalez de Miguel

Dpto. Sistemas Informaticos y Programacion
Facultad de Informatica, Universidad Complutense de Madrid
Profesor Jose Garcia Santesmases s/n, Ciudad Universitaria, 28040 Madrid, Spain
ana.gonzalez@fdi.ucm.es

Abstract. This paper presents the design of an intelligent software architecture called *Learning Virtual Object Tool* (LVOT) to facilitate the definition, registration and delivery of objects in a *Generic eLearning Environment* (GeLEnv). The research work brings our initial explorations towards the definition and use of simple and more complex learning objects in cooperative environments. We use a top-down analysis process [4] to describe GeLEnv models. Our architecture serves to statically and dynamically generate learning objects using a special purpose registry and, two core functional engines. Basic, composite and cooperative LVO structures can be defined and registered using the first engine. The second engine intelligently processes these structures to provide the learning resources to the final users of our LVOT platform.

Keywords: Intelligent Software Architecture, eLearning Environments, Cooperative Models, Knowledge Structures, Delivery Mechanisms.

1 Introduction

This paper presents the design of our LVOT architecture as an intelligent, open, modular and standard-based solution which allows universities, research centres, companies and many other eLearning entities to define, create, register and deliver LVOs in a flexible and efficient way. Further, the work brings a major contribution to knowledge in the modelling of *large-scale eLearning environments*.

In the last few years, there has been a wide interest in the development of eLearning standards [5]. The work focuses on the definition of technical specifications to support distributed learning and interoperability. From research carried out so far, it seems clear that, in order to produce intelligent and effective architectures, the underlying software platforms will need to adapt to the demands of truly complex cooperative environments. Some *IMS-based architectures* (like [2] and [6]) have been proposed to address the design of collaborative models. However, the design of large-scale cooperative learning environments does not seem to be resolved using any of these system.

We have analyzed a cooperative model that we called GeLEnv using a *top-down analysis method* inspired, mainly, by our research work around the development of *new intelligent systems* [4]. The LVOT platform provides full computational support

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 756 – 759, 2006.

to our GeLEnv models and LVO structures. This tool helps to create *virtual ecosystems* of *intelligent learning entities*. The underlying control mechanism of this engine highlights most of the *intelligence* being processed by our platform

2 Research Objectives and Motivations

Our main research objectives can be summarised as follows: a) exploring the latest *Artificial Intelligence* (AI) insights and *eLearning Developments*, b) presenting our initial ideas for solving "complexity" in an intelligent and effective way, c) analyzing real, large-scale eLearning environments and, d) planning our future research work to produce and test a number of LVOT platforms.

In general, AI has provided *Software Design* with a number of techniques to cope with the development of intelligent platforms, and we believe that since [1] established the key fundamentals of the behaviour viewpoint in AI, *Software Engineering* will emerge as a productive application of his pioneering work. In this paper, we present the design of a *hybrid intelligent solution*. Basically, we analyze our GeLEnvs in terms of knowledge structures while we exploit some behaviour-oriented AI benefits to automatically retrieve the eLearning information, generate new learning objects and, deliver these by performing intelligent actions. Further, this research work has been motivated by some of our *solution architecture designs* in technology-oriented projects like [3] and [4].

3 The Analysis of the GeLEnv

The key concept explored here is "level of abstraction". The GeLEnv analysis method incorporates three levels of abstraction. The first level of abstraction is a qualitative network where the nodes correspond to eLearning *entities* and the edges represent their *cooperations*. It is a dynamical model of eLearning entities. In the second level of abstraction, we analyze these elements in more detail introducing *sub-entities* and *interactions*. This consists of dividing the high level entities into sub-entities and, mapping the high level links into a number of interaction activities. Finally, the third level of abstraction integrates the core designing elements of the LVOT platform into the GeLEnv model. Thus, as we move down to lower abstractions, we convert the top knowledge-based mechanisms into eLearning structures and activities. We use this analysis method to adopt abstract views of eLearning entities and their cooperations. Then we establish lower abstractions through *refinement stages*. This basic analysis method helps to define how to cooperate in complex eLearning situations.

4 The LVOT Architecture

The LVOT architecture provides an efficient way to create, integrate and manage eLearning resources. It gives the computational support that is required to store and process LVOs.

We have identified three types of LVOs: *basic*, *composite* and *cooperative*. A basic LVO is a simple object that encapsulates some information of one learning resource. Composite LVOs can result from either the combination of basic LVOs or, the integration between several learning resources. Finally, we define cooperative LVOs as the composition of basic and/or composite LVOs. All these objects can be designed using two types of LVO structures: *Technical Descriptors* and *Delivery Flows*. The first structure organizes the LVO metadata. The second structure represents an execution template.

Fig. 1 shows an overall view of our LVOT platform and its users. The deployers look for some eLearning information, create new LVOs and store these within the *LVOT Registry*. The managers supervise and publish these objects for external collaborations purposes. The final users (e.g. researchers, students, lectures, etc.) access the LVOT platform to make use of the learning resources.

As shown in Fig. 1, the LVOT platform includes two core engines: the *LVOT Registration Engine* and the *LVOT Delivery Engine*. The deployers and managers use the first engine to create and manage LVOs. The second core engine incorporates a set of functional modules to intelligently deliver the registered LVOs to final users. The overall performance of this engine results from collecting and processing LVO structures. This engine incorporates a controller module to provide intelligent support to the GeLEnv model. This module controls the execution of all the LVO structures.

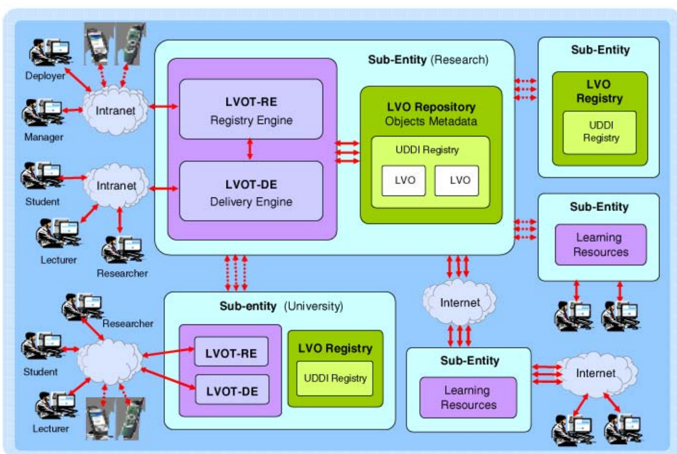


Fig. 1. LVOT Architecture & Users

5 Conclusions

The paper has outlined the top-down analysis of our GeLEnv model as a contribution to knowledge in the modelling of large-scale cooperative eLearning environments. The LVOT platform has been presented as part of our ongoing investigations in the development of intelligent software platforms.

6 Future Work

We are planning to continue proving the validity and efficiency of our LVOT platform in real working environments. We will implement several *LVOT prototypes* and *case studies* for testing and evaluating its performance in cooperative research-oriented environments. In this future work, we will be adding more intelligence into our platform, solving truly complex, cooperative and adaptive models.

References

1. Brooks, R. A.: Intelligence Without Reason, In: Proceedings of IJCAI-91, Sidney, Australia (1991).
2. CORDRA: Content Object Repository Discovery and Registration/Resolution Architecture (2005).
3. Gonzalez de Miguel, A. M.: MES: A Mobile e-Services Platform for Telefonica Moviles España, Hewlett Packard Consulting, Spain (2003).
4. Gonzalez de Miguel, A. M.: Explorations into the Behaviour-Oriented Nature of Intelligence: Fuzzy Behavioural Maps, PhD Thesis, SHU, UK (2003).
5. IMS: IMS Content Global Packaging Information Model Version 1.1.4, http://www.imsglobal.org (2004).
6. OKI: The OKI Project, http://www.okiproject.org (2005).

The Remote Control Approach – An Architecture for Tutoring in Pre-existing Collaborative Applications

Andreas Harrer, Nils Malzahn, and Benedikt Roth

Collide Research Group, University Duisburg-Essen, 47048 Duisburg
{harrer, malzahn.roth}@collide.info

Abstract. In this paper we present an architecture for the integration of tutoring and process scaffolds into existing collaborative applications. The architecture allows to combine existing research results concerning collaborative processes and their formalization, and existing and tested collaborative learning environments. The architecture allows controlling the learning environments either by a human or a pedagogic agent. Both types of tutors are using the same set of primitives – either via an intuitive user interface or a slim Java interface.

1 Structuring and Scaffolding Collaboration

Collaboration has become an important factor in learning activities, especially in disciplines that require substantial phases of working in teams, such as computer science, communication sciences etc. This can be seen in the emergence of the research field Computer-Supported Collaborative Learning (CSCL) in the last decade. Yet, just reducing the computer-based support to providing the suitable technological means to communicate is most often insufficient to promote the collaborative learning activity: Studies, like [1] showed, that collaboration does not happen effectively in every situation just by initiating the collaborative situation.

Scaffolds [2] or collaboration scripts are means to structure the learning activity and support the learners in organizing their activities or acquiring the skills to collaborate effectively. Thus their use in computer-based learning support environments (LSE) is a major topic of recent research in the CSCL community ([3];[4]).

Interestingly a parallel discussion occurs also in a field of computer-supported learning that has evolved independently of CSCL, the discipline of Intelligent Tutoring Systems (ITS): The support of the learners by the system to promote them in the learning process is often called tutoring or interventions.

It is obvious that the expertise and experiences of these two fields should be combined in collaborative computer-supported learning activities. One of the grand challenges for the shared interest between the communities will be the representation and implementation of scaffolds respectively tutoring for collaborative scenarios.. This article will present our approach of combining aspects from CSCL, pedagogical design, and ITS in an integrated architecture for supporting collaborative learning activities.

Up to now complex learning support environments and explicit scaffolding/tutoring models are largely unrelated and co-exist, but do not co-operate. On the one hand LSEs, such as WISE, CoLab, or Belvedere, either have a specific ("hardwired") process model embedded or do not have an explicit learning process model at

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 760 – 762, 2006.

all. On the other hand environments that use explicit process models for supporting the learning process typically fall short in either re-usability or expressiveness of the process model. Most systems using a formal model define their own proprietary model for the learning process which is not understandable and thus re-usable by other applications. Systems that have explicit mechanisms for structuring activities usually tend to have a very narrow focus, such as sequencing the presentation of learning material in web-based hypertext systems or intervening on the first deviation from an "ideal" learning path. There are very few approaches that explicitly try to scaffold collaboration with adaptive approaches: coming from the ITS area, like the "Collaborative Tutor" approach and the GridCOLE approach from the CSCL field.

IMS Learning Design can be considered as a formal approach with explicit representation of both the models and the operational semantics. We assume that the formal character of IMS LD can also be utilized to scaffold and apply tutoring support for pre-existing LSEs. The availability of learning design engines (LDE), such as CopperCore , could provide explicit process support without having to implement a process model from scratch for each individual environment, if we can meet the challenge of integrating pre-existing LSEs and LDEs in a flexible, interoperable architecture. The idea is to combine the flexibility of learning scripts, which can be adapted to different learning groups and tasks, with the often task-oriented and domain specific ITS systems. Given this it will be possible to use one learning flow for more than one learning environment at the same time. That means the script (agent, tutor) can be used for arbitrary (within certain limits) collaborative learning environments, enabling students using different learning environments to collaborate with each other.

2 A Flexible Architecture for Tutoring in Collaborative Settings

We propose an approach that aims at a clear separation of the learning design engine together with the specification and implementation of the learning flow (as LD documents) and the collaborative learning environments. In this proposal we assume that the learners interact exclusively with the LSE without having to know anything about being "scripted" or "scaffolded" by the LDE respectively the LD document. In our approach the LDE is used as a process regulation facility that interacts with the LSE using a common generic vocabulary of communication primitives. This has the advantage, that the LD document can be used with a variety of different LSEs without any changes to the document.

For the concrete realization of our approach we defined an architecture that brings together LSEs and LDEs without having to make substantial changes in either of the two components: the schematic overview of the architecture can be found in figure 1 and the components introduced have the following function:

- Engine Extension (CopperCore Extension): this component extends the event propagation mechanism of the learning design engine so that events are sent to the LSE to remotely control the learning process via the Remote Control Component.
- Remote Control Component: this component maps events coming from the LDE to one or more communication primitives, that build the vocabulary for remotely controlling learning support environments

- LSE Remote API (Translator): this interface accepts communication primitives that have been defined for a variety of different LSEs and maps these primitives to the specific functionality available in the concrete LSE.

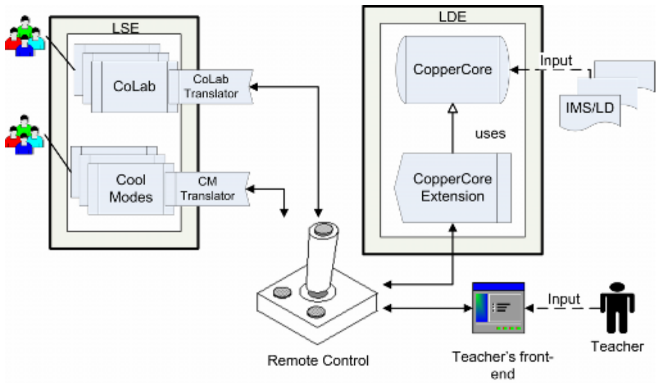


Fig. 1. Remote control architecture for interaction between LDE and LSE

An interesting feature of this architecture is, that besides our main purpose - the realization of collaboration scaffolds in pre-existing learning support environments - the remote control can be used by a variety of different actors such as intelligent tutors or human teachers.

Since the already present systems shall not be rewritten we decided to use a loosely coupled approach that allows to be adjusted for different learning support environments and different scaffolding agents to be applied. The proposed architecture has been implemented using the Cool Modes LSE and Copper Core Engine (LDE).

References

[1] Weinberger, A.:Scripts for Computer-Supported Collaborative Learning Effects of social and epistemic cooperation scripts on collaborative knowledge construction. PhD Thesis, Ludwig-Maximillans-Universität München.. (2003).

[2] Quintana, C., Krajcik, Joseph, Soloway, Elliot: A Case Study to Distill Structural Scaffolding Guidelines for Scaffolded Software CHI (2002).

[3] Weinberger, A., Fischer, F. Häkkinen, P., Dillenbourg,, P., Harrer, A.: Computer-supported scripting of interaction in collaborative learning environments. Workshop at the Conference on Computer Supported Collaborative Learning Conference (CSCL-05). Taipei, Taiwan (2005).

[4] Harrer, A., Hernandez Leo D. Dimitriadis, Y.: Languages for modeling of collaborative learning processes: Formalization, Practical uses and tools. Workshop at CSCL-05. Taipei, Taiwan (2005).

[5] Koper, R., Tattersall, C. (ed): Learning Design: A Handbook on Modelling and Delivering Networked Education and Training Springer Verlag (2005).

Scrutable Learner Modelling and Learner Reflection in Student Self-assessment

Judy Kay and Lichao Li

School of Information Technologies, Madsen F09, The University of Sydney,
NSW 2006, Australia
{judy, lli1}@it.usyd.edu.au
http://www.it.usyd.edu.au/~judy
http://www.it.usyd.edu.au/~lli1

Abstract. Learner reflection is critical to effective, deep, transferable learning, especially in cognitively demanding areas, such as learning programming. This paper presents KrAssess, a programming education system, which aims to facilitate student self-assessment and promote learner reflection through scrutable learner models. KrAssess supports learning by helping students learn to "see" solutions according to the criteria valued by their teacher. It also supports learning to write, review and improve solutions to design tasks.

1 Introduction

User modelling or learner modelling is at the core of intelligent tutoring systems (ITS) as it allows an ITS to provide individualized instruction and a personalised learning experience. It was not generally realized that such models can also contribute to learning directly; until, in the 1990's, the concept of open learner models emerged [1, 2], noting that a learner model is a valuable learning resource on its own. Learners can benefit from viewing, questioning and manipulating their individual learner models. A common and fundamental belief supporting open learner models is that it supports an important meta-cognitive [3] activity, namely learner reflection [4-6].

2 System Description

Our system, KrAssess, is built to help students learn to program in C. Fig. 1 shows the main steps in the system's student self-assessment [7] process:

- Students study a task description (top left of the figure);
- They note the concepts the task teaches (top right of the figure);
- They study a set of examples in the following stages (Step 1 in the figure) :
  - They read a set of examples;
  - They evaluate it according to criteria provided by the teacher.
- They provide their own solutions to the task (Step 2 in the figure) and;
- They evaluate their own solutions with the same criteria they used to evaluate the examples (Step 3 in the figure).

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 763 – 765, 2006.

The examples students study are not necessarily ‘perfect’ solutions. Rather they provide opportunities to explore interesting and important ideas associated with the learning goals. We usually create these examples, which demonstrate common misconceptions, after grading final exam questions. We also strive to provide multiple examples, so we can illustrate different ways to do one task.

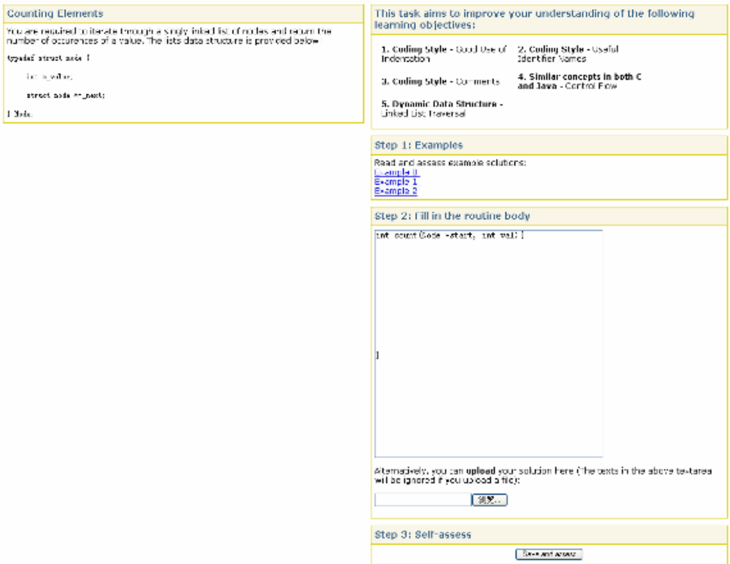


Fig. 1. A KrAssess Task

KrAssess users need to assess example solutions they read and their own solutions with criteria provided by the teacher. Each criterion corresponds to one of the learning objectives of the task. Consequently, each criterion asks the student to rate one aspect of the set of domain knowledge, thereby promoting reflection-in-action [5, 6]. Moreover, when assessing example solutions, since these examples have been pre-assessed by teachers using the same criteria, students’ assessments can then be compared with the teacher’s assessment, and this contributes to a model of the learner’s knowledge.

KrAssess provides a user profile to promote reflection-on-action [5, 6] as it shows students a visualization of their learning progress by externalizing their learner models with SIV [8], which allows a learner model to be explored in certain constructive ways. From this information, students are encouraged to think about what they have learnt and how they have learnt it, i.e. reflecting on their experience. Furthermore, students are able to compare the system’s beliefs about their programming knowledge with their own beliefs. In this way, reflection is further encouraged; especially if the system’s beliefs and the student’s own beliefs differ.

3 Evaluation and Conclusion

KrAssess was used by 260 students over one semester in our C programming subject. We conducted detailed analysis of the work done by 46 of them, 20 at the top of the class and 26 at the borderline of a pass. The main findings were:

- The self-assessments of more able students' answers were often similar to how a teacher would assess them; while the less able students tended to over-rate their solutions. This was in line with earlier studies in [9];
- Students were influenced by the supplied examples. However, it shows that the approach of reading examples before writing a solution does get students to incorporate elements from our code. This is beneficial to learning so long as the tasks are designed to ensure that straight copying is not attractive and;
- Stronger students could identify incorrect elements in an example solution, while weaker students could not.

In conclusion, KrAssess is a student self-assessment system with scrutable learner models to support and promote learner reflection. Two types of learner reflection, namely reflection-in-action and reflection-on-action are supported by allowing the student to self-assess solutions and by providing intelligent and informative learning progress feedback through open learner modelling. The system was used by students over one semester. We found KrAssess can provide more able students help that they value and appreciate and the system measures give a good indication of student's ability to "see" code from the perspective the teacher intended for the learning activity.

References

1. Crawford, K., Kay, J.: Shaping learning approaches with intelligent learning systems. Intl Conf for Technology in Education, Vol. 3, France (1992) 1472-1476
2. Kay, J.: Invited keynote address: Learner know thyself: Student models to give learner control and responsibility. In: Halim, Z.T., Ottomann, Z.R. (eds.): ICCE'97 International Conference on Computers in Education (1997) 17-24
3. Flavell, J.H.: First discussant's comments: What is memory development the development of? Human Development **14** (1971) 272-278
4. Boud, D., Keogh, R., Walker, D.: Promoting reflection in learning: A model. In: Boud, D., Keogh, R., Walker, D. (eds.): Reflection: Turning Experience into Learning. Kogan Page, London (1985) 18-40
5. Schön, D.A.: The reflective practitioner. Basic Books, New York (1983)
6. Schön, D.A.: Educating the reflective practitioner. Jossey-Bass Publishers, San Francisco (1987)
7. Boud, D.: Implementing student self-assessment. Campbelltown: Higher Education Research and Development Society of Australasia. HERDSA Green Guide **5** (1986)
8. Kay, J., Lum, A.: Exploiting Readily Available Web Data for Scrutable Student Models. 12th International Conference on Artificial Intelligence in Education, Amsterdam, Netherlands (2005) 338-345
9. Boud, D.: The role of self-assessment in student grading. Assessment and Evaluation in Higher Education **14** (1989) 20-30

Using Learning Objects Features to Promote Reusability of Pedagogical Agents

Eduardo Rodrigues Gomes[1], Ricardo Azambuja Silveira[2], and Rosa Maria Vicari[1]

[1] Universidade Federal do Rio Grande do Sul, Av. Bento Gonçalves, 9500 - Campus do Vale - Bloco IV, Porto Alegre - RS –Brazil.
{ergomes, rosa}@inf.ufrgs.br
[2] Universidade Federal de Santa Catarina, Campus Universitário, s/nº - Caixa Postal 476 Florianópolis - SC –Brazil.
silveira@inf.ufsc.br

Abstract. The Learning Object idea is based on the premise that the reuse of learning material is very important for designing learning environments. The reusability of learning objects results from the product of three main features: modularity, discoverability and interoperability. In this paper we discuss how these features can be useful when added to pedagogical agents. This approach considers learning objects built according to agent architectures: the Intelligent Learning Objects approach.

1 Introduction

The Learning Object (LO) approach is based on the premise that the reusability of learning material is very important to designing learning environments [2]. The reusability of learning objects is given as a result of three features: interoperability, discoverability and modularity. In this paper we discuss how these features can be useful when added to pedagogical agents. To achieve this goal, we propose the convergence between learning objects and agents technologies: the Intelligent Learning Objects (ILO).

An ILO is an agent that provides learning experiences to students in the same way as LO's used to do. According to this approach a learning environment is composed of an **agent society** and a **communication framework**. The agent society encompasses three kinds of agents: *Intelligent Learning Objects – ILO* agents (responsible for generating learning experiences to students); *Learning Management Systems - LMS* agents (responsible for dealing with administrative and pedagogical tasks); and *ILO's retrieve - ILOR* agent (responsible for storing data about ILO's). The communication framework is based on the FIPA reference model [6], composed of an ontology and a set of dialogues that give the features to enable the agents to share information.

2 Pedagogical Agents as Intelligent Learning Objects

An ILO must be reusable, interoperable, discoverable and modular. As the technological basis of an ILO is composed of agents and LOs technologies, we need to treat these features in these two levels. This section presents how achieve this.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 766 – 768, 2006.

Discoverability: The discoverability of learning objects is the capability of being discovered based on their educational content. The Learning Objects Metadata standards allow the description of the educational content of LOs, and the Learning Object Repositories provide storage features and make that information available. To achieve discoverability, the ILO approach adopts the IEEE *1484.12.1 Standard for Learning Object Metadata* (LOM) [4] and defines a set of dialogues to be used to consult ILOs' metadata information.

Interoperability: This concept means that a set of LO's can communicate each other to share pedagogical information and work together to solve the student's learning difficulties. To achieve interoperability, the ILO approach adopts two IEEE standards for learning objects: the LOM and the *IEEE 1484.11.1 Standard for Learning Technology – Data Model for Content Object Communication* (DMCOC) [4]. The LOM is used to describe the metadata information of the ILOs and the DMCOC is used for the communication of pedagogical information among the ILOs. It also adopts the concepts defined by FIPA [6], which defines standards to enable interoperability for MAS.

Modularity: The content of ILO's must be comprehensive enough to be unitary and coherent, but small enough to be reused in different courses. To achieve modularity, the ILO approach claims that modularity can only be reached by a good pedagogical project. It also adopts the idea that agents as coarse-grained computational systems, each making use of significant computational resources that maximizes some global quality measure [5].

Reusability: The Learning Management Systems (LMS) are systems used to deliver courses using LOs. To complete the scenario, the ILO approach defines the LMS Agents to work as an LMS.

3 Test Bed

We developed a framework composed of a set of Java classes designed to built ILOs as easy as possible and applied this framework to the agent-based learning environment described in [1]. Such system is composed of an ILO playing the role of a special calculator and an Animated Pedagogical Agent (APA) playing the role of an animated tutor to help primary school students to learn some fundamental mathematical properties of multiplication.

The Animated Pedagogical Agent works like an LMS and the Calculator Pedagogical Agent is a typical ILO. The communication between the Animated Pedagogic Agent and the Calculator Pedagogic Agent is performed according to the DMCOC and the ILO communication framework's dialogues.

4 Conclusions

This case study showed that using the ILO approach potentially improves the reusability of pedagogical agents. With learning object features, the agents can be reused in different courses concerning their learning contents. However, in order to share information, agents also need to interoperate in the communication level. Based on the FIPA concepts and IEEE standards, the communication structure proposed in the ILO approach contributes to solve this issue.

While the development of educational content outside of the Intelligent Tutoring Systems and Intelligent Learning Environments approach is converging to the use of standards towards the reusability, we are still developing *ad-hoc* pedagogical-agents-based learning environments. This is the issue that this paper addressed. We should start to think about reusability when developing pedagogical agents. We need to go towards to the use and development of technologies that enable our agents to be reusable and interoperable. The convergence between learning objects and agents' technologies seems to be promising.

References

1. Lucas, J. P.; Widges, B.; Silveira, R. A. (2005). Inserting Animated Pedagogical Agents Inside Distributed Learning Environments by Means of FIPA Specifications. In: Fourth International Joint Conference on Autonomous Agents & Multiagents Systems (AAMAS) – Proceedings of Agent-Based Systems for Human learning Workshop (ABSHL) 2005, Utrecht.
2. Friesen, Norm: What are Educational Objects? Interactive Learning Environments, 3 (9). (2001).
3. Silveira, R. A., Gomes, E. R, Vicari, R. M. Inteligent Learning Objects: An Agent-Based Approach of Learning Objects. In Weert, Tom Van, Tatnall, Arthur (Eds.) Information and Communication Technologies and Real-Life Learning.Boston Springer (2005), 1103 - 110.
4. IEEE Learning Technology Standards Committee (LTSC): Specifications (2004). retrieved July 7, 2005 from http://ltsc.ieee.org.
5. Wooldridge, M.; Jennings, N. R.; Kinny, D. A methodology for agent-oriented analysis and design. In: Proceedings of International Conference on Autonomous Agent AAMAS (1999), v.3. 1999.
6. FIPA: The foundation for Intelligent Physical Agents: Specifications (2002). retrieved July 7, 2005 from http://www.fipa.org.

Error Diagnosis in Problem Solving Environment Using Action-Object-Input Scheme

Dmitri Lepp*

University of Tartu, Institute of Computer Science, J. Liivi 2, 50409 Tartu, Estonia
dmitri@ut.ee

Abstract. In T-algebra problem solving environment students solve expression manipulation problems step-by-step using transformation rules and following the three stage input dialogue. The errors can be made either when selecting the rule or the objects of the rule or while inputting the resulting expression. The program is able to diagnose different errors and help to correct them. This paper gives an overview of the error diagnosis principle that is used in the T-algebra.

1 Introduction

Expression manipulation problems are very important in school mathematics. When using traditional instruction technology, the teacher is not able to give immediate feedback or draw attention to errors of every student. The use of intelligent problem solving environments can produce maximum effect in this field. The essential properties of such environment should include: the possibility to solve problems using exactly the same steps as on paper; possibility of making errors (the same errors as on paper); ability of the program to diagnose errors and help to correct them. Existing environments don't have all these properties at once. In the Aplusix [4] the student enters the new expression and the program checks for the equivalence of the expressions. Mathpert [2] and AlgeBrain [1] use transformation rules with no input of result.

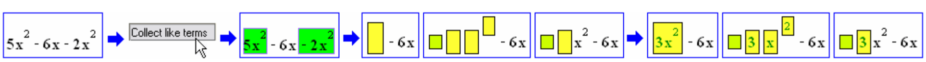


Fig. 1. Three stages of application of rule collect like terms

T-algebra enables step-by-step problem solving in four fields of mathematics. T-algebra uses so-called action-object-input dialogue scheme [3] where each step consists of three stages (Fig. 1): selecting a transformation rule, marking the parts of expression, entering the result of the application of the rule. Three different input modes are implemented for the input stage (Fig. 1). In free input mode the whole result is entered into one box. In structured mode the structure for the result is given. In partial input, boxes are given only for input of rule-specific components.

* The author was supported by ICT graduate school.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 769 – 771, 2006.

2 Error Diagnosis in T-algebra

At the beginning of each solution step, the student has to select the transformation rule he is going to apply. If the selected rule is not applicable to the current expression, no error message is displayed until the student confirms his selection of objects – he is given a chance to realize that no suitable objects can be found and correct the choice of the rule. The only check performed immediately upon selecting the rule is whether the current expression is already in the form of the answer to the problem.

After selection of the transformation rule in T-algebra the student has to select the objects to which the rule is applied. In some programs (Mathpert, AlgeBrain) the student selects a subexpression and the program applies the rule to the appropriate parts of this subexpression. The approach of T-algebra is different – it requires from the student precise selection of operands. At this stage many errors arise. When selecting separate objects the program immediately checks whether the selected parts are syntactically correct expressions (one of the brackets is not selected, etc.).

After confirmation of objects T-algebra is able to diagnose whether the student knows and considers the priorities of the operations (Fig. 2). T-algebra also checks whether the objects are of the correct structure and their number is correct (for example, the rule *Collect like terms* requires selection of at least two similar monomials).

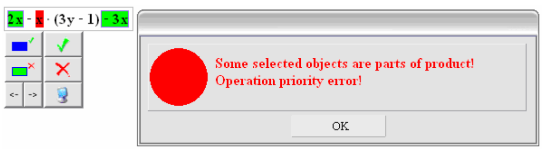


Fig. 2. Error in selection of the object for the rule *Collect like terms*

When applying the rules, T-algebra copies unchanged parts of the expression and protects them from modification. It only lets the user to enter the exact result of applying the rule (Fig. 1). When the user confirms his input, the program has full information on the transformation rule, operands as well as the result of the application of the rule offered by the student. In most cases, the issue whether the entered expression is equivalent to the previous one is not the only aspect that can be clarified. T-algebra is able to apply the rule itself and compare the student's result with the correct one.

Some common checks are performed when applying all the rules. First program checks whether the entered parts and the whole resulting expression is syntactically or mathematically correct. Another common check for all the rules is whether all necessary input boxes are filled in. Further checks depend on the input mode and the rule.

In free input mode, a check of the structure of the result is performed in most cases (for example, when collecting like terms, the result should be a single monomial). This is because we want the student to apply exactly the same rule that he chose and not to simplify something else. The other issue that is checked is the priority of operators – whether the student adds brackets if needed. After the structure of the result is checked, the content is checked in exactly the same way as in structured input mode.

In structured input and partial input modes the resulting expression already has the correct structure, because the student is prevented from entering unsuitable parts into corresponding boxes. In these input modes the program checks the essential parts of the resulting expression separately in order to find the exact error and diagnose its cause. When the result is a single monomial, the operation sign, coefficient, variables and their powers are checked separately. When the result is a polynomial then the set of the monomials in the student's result is compared to the set of monomials in the correct result. If a difference is found, the exact error is diagnosed if possible (Fig. 3).

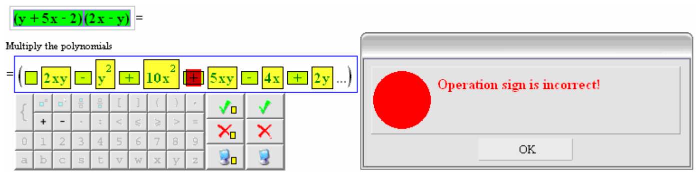


Fig. 3. Error in input of the result – wrong operation sign entered

3 Conclusions

T-algebra gets full information on the student's intentions and the result of applying the rule. Using this information, T-algebra is able to provide better error diagnosis – in addition to simple expression equivalence it is able to recognize many typical errors in all the transformation rules, recognize operation priority errors, incorrect rule object selections, etc. T-algebra is able to show the student the exact position of the error and help to correct it. Error checks are performed after all three stages of single step: after selection of the rule, after selection of objects and after entering the resulting expression. All this could improve teaching of different mathematical topics (available in T-algebra), because the students get immediate feedback, see the errors made, correct and memorize them.

References

1. Alpert, S. R., Singley, M. K., Fairweather, P. G.: Deploying Intelligent Tutors on the Web: An Architecture and an Example. International Journal of Artificial Intelligence in Education 10(2) (1999) 183-197
2. Beeson, M.: Design Principles of Mathpert: Software to support education in algebra and calculus. In: Kajler, N. (ed.): Computer-Human Interaction in Symbolic Computation. Springer-Verlag, Berlin Heidelberg New York (1998) 89-115
3. Lepp, D.: Extended Solution Step Dialogue In Problem Solving Environment T-algebra. In: Proceedings of the 7th International Conference on Technology in Mathematics Teaching, Vol. 1. Bristol, UK (2005) 267-274
4. Nicaud, J., Bouhineau, D., Chaachoua, H.: Mixing microworld and cas features in building computer systems that help students learn algebra. International Journal of Computers for Mathematical Learning 5(2) (2004) 169-211

Learning How Students Learn with Bayes Nets

Chao-Lin Liu

National Chancier University, Taipei 11605, Taiwan
chaolin@nccu.edu.tw

Abstract. This extended abstract summarizes an exploration of how computational techniques may help educational experts identify fine-grained student models. In particular, we look for methods that help us learn how students learn composite concepts. We employ Bayesian networks for the representation of student models, and cast the problem as an instance of learning the hidden substructures of Bayesian networks. The problem is challenging because we do not have direct access to students' competence in concepts, though we can observe students' responses to test items that have only indirect and probabilistic relationships with the competence levels. We apply mutual information and backpropagation neural networks for this learning problem, and experimental results indicate that computational techniques can be helpful in guessing the hidden knowledge structures under some circumstances.

Summary

Behavior models of activity participants are crucial to the success of computer systems that interact with human users. When using Bayesian networks (BNs) as the language for model construction, Mislevy et al. asked where we could obtain the numbers for the conditional probability tables (CPTs) [1]. We could ponder where we could obtain the structures of the BNs in the first place. For educational practitioners, an obvious and practical answer to this inquisitiveness may be that we should consult experts of the targeted domains to provide the knowledge structures, such as the prerequisite relationships between concepts, for building student and instructor models. Indeed this is an effective and the de facto approach to building computer-assisted educational software in general. Can computers be more helpful than finding the detailed numbers in the CPTs for student modeling? More specifically, can computers assist in any way for finding the structures of student models? Given a *composite concept*, say $dABC$, that requires knowledge about three *basic concepts*, say cA, cB, and cC, how can we tell how students learn $dABC$ from cA, cB, and cC? Do students combine cA and cB into an intermediate product, dAB, and then combine dAB and cC into $dABC$? Or, do students integrate the basic concepts directly to learn $dABC$?

In this exploration, we assume that students learn the composite concept from ingredient constructs that do not include overlapping basic concepts. For instance, we subjectively exclude the possibility of learning $dABC$ from two intermediate composite concepts dAB and dBC, because they both include cB.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 772–774, 2006.

This assumption simplifies the search space. However, the size of the search space still grows explosively with the number of basic concepts included in the target composite concept, and is related to the Stirling number of the second kind.

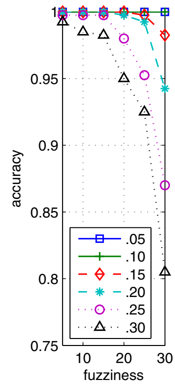
We assume that educational experts provide a set of possible ways that students may, implicitly or explicitly, employ to learn the composite concept, and our job is to help experts identify which of these learning patterns is the most likely answer. Hence, the process of learning how students learn begins with the acquisition of a set of candidate answers. We use the set of candidate learning patterns to build BNs for simulating possible student behaviors, and employ the simulated data to train backpropagation neural networks (BPNs). The learned BPNs can then be used to classify the unobservable learning pattern, based on students' item responses, into one of the candidate answers.

Following the steps of many researchers who explored methodologies for building computer-assisted tutoring systems, we employ simulated students in this study. Simulated students were generated from Liu's simulation system that considers the probabilistic relationships between students' responses to test items and students' competence levels in concepts [2]. The degree of uncertain relationship between these two factors was controled by a parameter called *fuzziness*. We set *fuzziness* to a larger value when we simulated a more uncertain relationship between responses to items and competence in concepts. The other parameter, named *groupInfluence*, affected the uncertain relationship between the students' actual behaviors and students' stereotypical behaviors. We set *groupInfluence* to a larger value to make students more likely to deviate from their typical behaviors. In short, it became harder to guess the real mental states of a student when either *fuzziness* or *groupInfluence* were set to larger values in the simulation.

Students' responses to test items and students' competence levels were represented with different, though directly connected, nodes in the BNs that were used to generate simulated students. States of nodes that represented competence levels in concepts were not observable, and only states of nodes that represented correctness of item responses were accessible. Hence, our job was to guess the substructure of the unobservable nodes based on the data that had only indirect and probabilistic relationships with the true answers. Due to this reason, known algorithms for learning structures of Bayesian networks, such as the PC algorithm implemented in Hugin, were not directly viable for this learning problem.

We employed estimated mutual information (EMI) for comparing the candidate solutions. If students learn $dABC$ from dAB and cC rather than from cA and dBC, the EMI between the nodes for both dAB and cC and the node for $dABC$ may be larger than the EMI between the nodes for both cA and dBC and the node for $dABC$. (In this case, $EMI(dAB, cC|dABC)$ is expected to be larger than $EMI(cA, dBC|dABC)$.) Namely, we used the EMI to represent the merits of a competing substructure. We had to estimate the mutual information between two sets of nodes, since we did not have direct access to the states of the nodes that represented concepts. We estimated the state for the node that represented a concept with the percentage of correct responses to test items designed for the concept, and used the estimated states of nodes to calculate the EMIs. In addition to the EMIs for all competing substructures, we introduced ratios between the EMIs for training the BPNs. Experience indicated that ratios between the EMIs, e.g., the ratios between the EMIs and the largest EMI, were useful for improving the prediction quality of the trained BPNs.

We tested the proposed procedure for guessing how students learn *dABC*. There were four possible answers. We randomly sampled 500 network instances that had different underlying joint probability distributions for each of these four answers, and simulated item responses of 10000 students that were generated from these 2000(=4×500) networks. Each simulated students responded to three items for seven concepts, i.e., *cA*, *cB*, *cC*, *dAB*, *dAC*, *dBC*, *dABC*, and the responses must be either correct or incorrect. We calculated the EMIs and their ratios for each network instance for training BPNs, so we trained the BPNs with 2000 training instances. We then applied the trained BPNs to predict the learning patterns of 400 groups of students—100 groups generated for each of the four answers. We repeated the above procedure for 36 combinations of *fuzziness* and *groupInfluence*, each ranging between 0.05 to 0.30. The figure on this page shows the results. The horizontal axis shows the decimal part of *fuzziness*, the legend shows the values of *groupInfluence*, and the vertical axis shows the percentage of correct identification of hidden structures in 400 test cases. The results suggest that it is possible to identify the hidden structure better than 80 percent of the time, if *fuzziness* and *groupInfluence* are not large and if educational experts' guess list does include the correct structure.



Do we really need student models of better quality? Experimental results reported by Carmona et al. suggested that student models of higher quality could help us improve the effectiveness of computerized adaptive tests [3]. Hence, we hope results outlined in this extended abstract can be useful. We have expanded our experiments to cases where we learned how students learn composite concepts that included four basic concepts [4]. The accuracy remained above 75% in unfavorable conditions. We thank reviewers for their invaluable comments on the original manuscript. This work was partially supported by the research contract 94-2213-E-004-008 of National Science Council of Taiwan.

References

1. Mislevy, R.J., Almond, R.G., Yan, D., Steinberg, L.S.: Bayes nets in educational assessment: Where do the numbers come from? 15th UAI (1999) 437–446
2. Liu, C.L.: Using mutual information for adaptive item comparison and student assessment. J. of Educational Technology & Society **8**(4) (2005) 100–119
3. Carmona, C., Millán, E., Pérez-de-la-Cruz, J.L., Trella, M., Conejo, R.: Introducing prerequisite relations in a multi-layered Bayesian student model. Lecture Notes in Computer Science 3538 (2005) 347–356
4. Liu, C.L., Wang, Y.T.: An experience in learning about learning composite concepts. 6th IEEE ICALT (2006) to appear

AlgoLC: A Learning Companion System for Teaching and Learning Algorithms

Patrícia Gerent Petry[1] and Marta Costa Rosatelli[2]

[1] Departamento de Informática e Estatística, Universidade Federal Santa Catarina INE-CTC, Cx.P. 476, Campus Universitário, Florianópolis-SC, 88040-900, Brazil
patricia@mbox1.ufsc.br

[2] Programa de Pós-Graduação em Informática, Universidade Católica de Santos R. Dr. Carvalho de Mendonça, 144, Santos-SP, 11070-906, Brazil
rosatelli@unisantos.br

Abstract. Learning algorithms involves translating sequences of simple steps that have complex results into instructions that can be followed by a computer. To accomplish this, the students have to develop skills of logical reasoning and algorithmic problem solving. This paper presents AlgoLC, a Learning Companion System that supports teaching and learning algorithms. Teaching is facilitated as the system provides the teacher with elements to better identify the students' doubts and errors. Learning is facilitated by the learning companion as it provides support to the student identify and correct his or her own mistakes.[1]

1 Introduction

Several times learning algorithms in Computer Science creates a barrier for many students entering an undergraduate course. Learning algorithms involves translating sequences of simple steps that have complex results, into instructions that can be followed by a computer. To accomplish this, the students have to develop skills of logical reasoning and algorithmic problem solving.

This paper presents AlgoLC, a Learning Companion System that supports teaching and learning algorithms. Learning Companion Systems are Intelligent Tutoring Systems that include, besides the traditional modules of the architecture, a Learning Companion (LC) [1]. LCs are virtual peers that support the students during the learning process.

AlgoLC makes use of Constrain-Based Modelling (CBM) to support its reasoning and the LC interventions - the feedback messages that are presented to the students. CBM is based on a theory of learning from the identification of errors [3]. Identifying errors is quite important for learning as usually the students are not able to detect their own mistakes. CBM represents the domain knowledge as a set of constraints on correct solutions. The constraints divide the solution possibilities into correct and incorrect [2].

[1] This research is partially supported by FAPESP grant no. 03/08279-2.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 775–777, 2006.

2 AlgoLC Description

2.1 Architecture

AlgoLC domain model is represented with CBM. It contains constraints on corrects solutions to problems regarding the algorithms structures.

The student model includes general information about the student and his or her correct and incorrect knowledge (i.e., which constraints were violated or not). The problems presented to the student are chosen by the system based on the student model, which is an overlay [4] of the domain model. The information stored about a student that interacts with the system includes: the number of LC interventions; the number of non-violated constraints; the number of constraints violations; and the constraints that were violated.

The tutor model includes the pedagogic strategies. The tutor role is to coordinate the learning activities: to present the problems to be solved, to control the prerequisites between the problems, and to control the interaction between student and LC. In addition, the tutor model chooses the feedback messages presented by the LC.

When the student develops a solution to a problem, the system verifies which constraints are relevant to the current solution and, among the relevant ones, which ones are satisfied. If the solution violates any constraint, the LC intervenes presenting the feedback message that is associated to the violated constraint. The LC is pro-active and collaborative. It initiates an interaction with the student whenever it is necessary, without a request from him or her.

2.2 Constraints

Currently, the domain model consists of a set of 6 problems and 35 constraints. The constraints were modelled based on the identification of usual and frequent problems of teaching and learning algorithms. The problems are organised in 4 groups, according to the concept that is approached by the problem and its level of difficulty. The 4 groups are: variables declaration, sequence structures, decision structures, and loop structures.

2.3 Prototype

AlgoLC prototype allows (1) visualizing the problems solved; (2) verifying the algorithm entered by the student and generating a report; (3) generating a student report (violated constraints and the number of times each one was violated); and (4) generating a general report that shows the number of times that each constraint was violated by the student so far (for all the problems solved).

3 Evaluation

AlgoLC prototype system was tested in an empirical evaluation that was carried out with 15 Computer Science undergraduate students of a 1st year course on

Algorithms. The evaluation aimed to check the appropriateness of the modelling approach in identifying students' errors and providing feedback.

The 6 problems presented during the evaluation were about variables declaration and sequence structures in an increasing difficulty level. After entering the solution to each problem (i.e., an algorithm) the students were asked to generate and send their report to the experimenter. After solving the 6 problems the students were asked to request the report that contains the statistics about the constraints violated. From the set of 35 constraints modelled in the system, 24 were related to the problems presented in the evaluation session (i.e., constraints 5 to 15 were not used). Six constraints - 1, 2, 26, 27, 28 and 29 - were not violated. Only one student violated the same constraint - 17 - twice. This constraint is related to assigning a real value to a variable of the integer data type.

Among the 15 students only 2 solved the 6 problems without violating any constraint. Constraint 23 was violated 10 times. This constraint is related to assigning values to a variable of the string data type. Although it can not be assured that the feedback messages presented by the LC after the contraints violation influenced the students' final scores, the 15 students indeed managed to solve the two groups of problems that were proposed during the evaluation.

4 Conclusion and Further Work

This paper presented AlgoLC, a Learning Companion System that supports teaching and learning algorithms in Computer Science undergraduate courses. Currently, a set of 35 constraints for 4 groups of problems are modelled and implemented in a prototype. A first empirical evaluation showed the system potential in supporting teachers and students in identifying errors. Further work includes having a pre- and post-test design to show that students had learned from the system feedback. After that, the aim is to add new types of problems and to model the respective constraints in the system.

References

1. Chan, T.W., Baskin, A.B.: Learning Companion Systems. In: C. Frasson, G. Gauthier (eds.): Intelligent Tutoring Systems: At the Crossroads of Artificial Intelligence and Education. Ablex Publishing Corporation, New Jersey (1990) Cap.1
2. Mitrovic, A., Ohlsson, S.: Evaluation of a Constraint-Based Tutor for a Database Language. Intl. Journal of Artificial Intelligence in Education **10** (1999) 238–256
3. Ohlsson, S.: Constraint-based Student Modeling. In: J. Greer, G. McCalla (eds.): Student Modelling: The Key to Individualized Knowledge-Based Instruction. Springer-Verlag, Berlin (1994) 167-189
4. Holt, P., Dubs, S., Jones, M., Greer, J.: The State of Student Modelling. In: J. Greer, G. McCalla (eds.): Student Modelling: The Key to Individualized Knowledge-Based Instruction. Springer-Verlag, Berlin (1994) 3-35

An Intelligent Tutoring System Based on a Multi-modal Environment for the 300-Certification Program of English Conversation

Youngseok Lee[1], Jungwon Cho[2], and Byung-Uk Choi[3]

[1] Department of Electrical and Computer Engineering, Hanyang University,
17 Haengdang-dong, Sungdong-gu, Seoul, 133-791, Korea
Tel.: +82-2-2220-0363; Fax: +82-2-2281-9912
yslee38@mlab.hanyang.ac.kr
[2] Department of Computer Education, Cheju National University,
66 Jejudaehakno, Jeju-si, Jeju-do, 690-756, Korea
Tel.: +82-64-754-3297; Fax: +82-64-725-4904
jwcho@cheju.ac.kr
[3] Division of Information and Communications, Hanyang University,
17 Haengdang-dong, Sungdong-gu, Seoul, 133-791, Korea
Tel.: +82-2-2220-0363; Fax: +82-2-2281-9912
buchoi@mlab.hanyang.ac.kr

Abstract. Today's mobile world is composed of heterogeneous networks and various devices with different characteristics. Progressive learners want to access and utilize services and information content using all available devices. While use of computers to teach English in a conventional educational environment promotes motivation and effective learning in students, the method generates problems such as provision of learning materials without consideration of teaching methods and evaluation without consideration of individual differences in students. To solve these problems and produce a superior system, we propose an Intelligent Tutoring System (ITS) for learning English that uses multi-modal technology. By overcoming limitations of the mobile environment and using appropriate mobile contents and content negotiation and adaptation strategies, the proposed system provides an effective method of learning based on ITS to support a teacher's role.

1 Introduction

English learning that involves multimedia content can increase learner interest and assist in developing the ability to communicate [1]. However, few studies have focused on learning or educational methods involving mobile devices [2, 3]. Related research has investigated Web-based ITS applications that are useful in various learning contexts. Providing content suitable for specific student levels requires consideration of various factors such as the importance of particular subjects and degree of difficulty; it is necessary to first estimate each student's level and then provide appropriate learning materials [4].

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 778 – 780, 2006.

To solve these problems while retaining the advantages of Web-based learning, this paper introduces an Intelligent Tutoring System (ITS) for learning English. Its design and implementation were based on an intelligent tutoring system to provide content suitable for specific student levels in a multi-modal education platform that supports various communication environments and devices [5]. We described subject contents and created a learner model that determined the provision of material appropriate to specific student levels, as assessed by an inference engine based on Item Response Theory (IRT) [6]. Results indicate that the system minimizes latency time for learning using technology that transmits learning material in real time.

2 System Design

This paper proposes a system consisting of an expression section that presents learning materials to the student. The interface module consists of input interpretation and output creation modules. The input interpretation module recognizes a learner's input entered using a cellular phone keypad and then analyzes the agreement between input data and correct answers. The output creation module presents sound contents using a cellular phone's speaker and image contents. The teacher module establishes an instruction strategy by estimating a learner's level and using the learner model database. The strategy of instruction uses learner levels and preferences to select appropriate items and methods of instruction. The expert module provides knowledge about other modules; the knowledge base of the expert module in elementary school consists of an English sound domain, and each grade has its own textbook and teacher's manual.

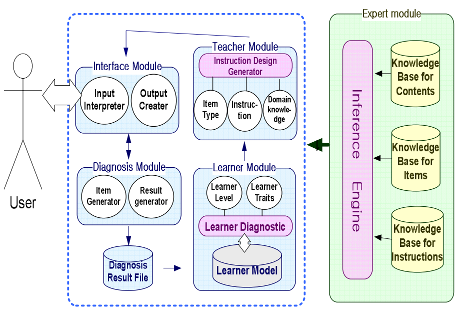


Fig. 1. Core mechanism of the proposed ITS

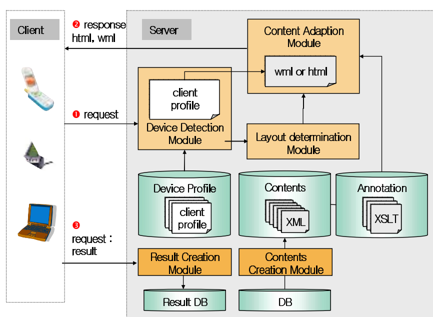


Fig. 2. Multi-modal architecture

Figure 1 shows the core mechanism for the proposed ITS. The proposed system creates learning models to display the student's current state of knowledge. The ITS core was designed using a logical learning formulation, which is called item Type Diagnostic Value (TDV), to measure a learner's traits and knowledge and to act based on rules of inference by the item analysis.

$$TDV = \frac{\sum_{i=1}^{n} QW_i}{\sum_{i=1}^{n} QA_i} \tag{1}$$

i : Total Number oh Sheets, QA : Total Number of Items,
QW : Number of examinees answering incorrectly

When TDV is presented with a problem expressed as a numerical formula based on ratios, it calculates ratios by comparing the total number of item type index to the number of items with incorrect responses. This type of analysis can supply feedback on learning progress and uncover weak points in each student's comprehension.

The communications module supports the Internet, wireless LAN for PDA, and CDMA for cellular phones. Content negotiation and adaptation strategies were implemented by the user-agent matching and XSLT within multi-modal architecture, shown in Figure 2. The network used in this architecture is based on three types of infrastructure. Three kinds of devices can be used to access content: a cellular phone using CDMA, a personal data assistant (e.g. iPAQ 4150) running under Windows CE and connected through the wireless network, and a desktop computer using the wired connection.

3 Conclusion

Existing Web-based education systems focus on providing study materials without considering knowledge levels or abilities of individual students. This paper proposed an intelligent tutoring system to provide materials appropriate for specific student levels via a mobile education platform that supports various communication environments and devices. This study designed and implemented a system that can conduct education appropriate to a learner's level using a mobile environment and that can produce feedback about the learning. Results indicate it is possible to create a multimodal and device-independent mobile service that uses less effort to develop compared to approaches based on traditional systems of tutoring.

Acknowledgment. This work was supported by the HY-SDR Research Center at Hanyang University, Seoul, Korea, under the ITRC Program of MIC, Korea.

References

1. Davidovic A., Warren J., and Trichina E. : Learning Benefits of Structural Example-Based Adaptive Tutoring Systems, IEEE Trans. on Education 46 (2003) 241-251
2. Badjonski M., Ivanovic M., and Budimac Z.: Intelligent Tutoring System as Multiagent System, Proceeding of IEEE Trans. on ICIPS 1 (1997) 28–31
3. Okamoto T.: The Current Situation and Future Directions of Intelligent, IEICE Trans. on Information & System E77-D (1994) 143–161
4. Kaplan R., and Rock D.: New Directions for Intelligent Tutoring. AI Expert, 1(2) (1995) 30–40
5. Leung C.-H., et al.: Mobile Learning: A New Paradigm in Electronic Learning, Proceeding of the 3rd IEEE International Conference on Advanced Learning Technologies (2003) 76–80
6. Crocker L., and Algina J.: Introduction to Classical and Modern Test Theory, Holt, Rinehart and Winston, Inc., Austin, TX (1996)

Adaptive Peer Review Based on Student Profiles

Raquel M. Crespo García, Abelardo Pardo, and Carlos Delgado Kloos

Departamento de Ingeniería Telemática
Universidad Carlos III de Madrid, Spain
{rcrespo, abel, cdk}@it.uc3m.es
http://www.it.uc3m.es/

Abstract. Intelligent tutoring systems cover a wide range of educational processes. However, in the context of peer review methodology, there is no previous work about adaptation of the process according to the student's profile. In this paper, a methodology for adaptive peer review is introduced. Experimental application of adaptive peer review through two courses allows to confirm pedagogical benefits with actual students' results.

1 Introduction

Peer Review amounts to evaluating the work of a colleague and providing feedback about it. This methodology has been widely applied in multiple contexts, ranging from childhood education to academic research. The benefits as well as a detailed topology of peer review in education have been studied in several publications (see [1] for an excellent survey). And a number of supporting programs for peer review in educational environments are reported, such as CPR [2], PG [3], NetPeas [4], OPAS [5] or OASIS [6].

Not all students learn in the same way. Different factors influence not only on how students learn by themselves, but also how do they collaborate with their peers and learn from them. In order to improve the effectiveness of the experience, student roles and adequateness have been frequently analyzed in collaborative learning (see [9] for example), but not so frequently in peer review, although both contexts are indeed similar. Indeed, *"how peer assessors and assessees should best be matched … is discussed surprisingly little in the literature"* [1] [7] [8]. And, to our knowledge, no system supports adapting the peer review process according to the learner's profile.

2 Adaptive Peer Review

Like any other educational process, peer review is suitable to be adapted to student characteristics and needs. In this paper a methodology supporting the dynamic adaptation of the process according to student's profile is presented. A supporting system for adaptive peer review has also been developed, including functionality for, besides peer review management facilities, process adaptation, including students' tracking, profile building and definition and deployment of adaptation criteria.

Defining an adaptive peer review methodology requires analyzing the process and detecting the potential actuation points. Author-reviewer is the only interaction

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 781 – 783, 2006.

exclusive of peer review processes, among all the different types of interactions involving the students in an educational process (student-teacher, student-content and student-student, according to Moore's model). Thus, adaptive peer review mainly concerns the adaptation of author-reviewer interaction, although content adaptation as well as collaborative learning tutoring could also apply.

Adaptive peer review copes with:

1. **Student's profile:** *How to collect the necessary data and select the appropriate variables to build student's profiles.* According to learning theories, particularly social constructivism, Vygotsky's zone of proximal development (ZPD) and scaffolded learning, the presented approach focuses on student's knowledge level as the key variable to influence the process. This selection is endorsed by previous experimental results which confirm the correlation between the level of a submission and how much its reviewer learns from it, and the correlation between the level of the feedback and how much the author learns from it [8]. Experimental data led to discard the predictive approach for building the student's profile (i.e. student's model inducted from past data), as no significant correlation existed between previous marks (neither examination nor submissions marks) and the current ones. On the contrary, correlation between self-evaluation and teachers' scores (ρ=0.71) endorses using self-evaluation-based profiles.
2. **Matching criteria:** *How to match authors and reviewers in order to ensure the educational benefits pursued.* In a first approach, matching authors and reviewers with complementary profiles was proposed, avoiding always pairing together less proficient students [8]. This criterion was based on the hypothesis that non-proficient students would most benefit either from reviewing high level solutions or from proficient students' feedback; while proficient students would not be confused by poor solutions or feedback (avoiding possible negative effects). However, this criterion showed some undesirable consequences in real classroom. Inspired by scaffolded learning and Vygotsky's ZPD theories, matching criteria were redefined in order to avoid too wide gaps which could difficult student comprehension but maintain differences of level which guarantees learning improvements.
3. **Matching system (algorithms):** *How to effectively build the authors-reviewers matching according to their profiles and the matching criteria specified.* Process adaptation focuses on selecting the reviewer(s) for each submission according to the learners' profiles and the specified matching criteria. Fuzzy classification is applied to rank the actual pairs according to the criteria, and genetic algorithms are applied for exploring efficiently the potential combinations of pairs [10]. Experimental evaluation with data from real students proves that the system effectively matches students according to the specified criteria.

3 Results and Concluding Remarks

Both Adaptive Peer Review methodology and supporting system have been applied to two CS2 courses with 52 and 75 participants respectively during the last two years. Students were posed a non-compulsory programming project to be developed in pairs. Peer assessment was done anonymously and individually, and both global marks as well as formative feedback were required from 3 reviewers assigned for each submission. Peer reviews were sent to original authors who could revise and improve their project according to the received feedback. This process was repeated three times through each course.

Promising experimental results have been obtained, confirming the influence of student's profile in the learning outcomes of the peer review process and, thus, the interest of the adaptive peer review methodology. Experimental results have also helped to refine the methodology, particularly the building of the student's model and the definition of the matching criteria.

Acknowledgments. Work partially funded by Programa MEC-CICYT, project MOSAIC-LEARNING TSI2005-08225-C07-02.

References

1. K. Topping, Peer Assessment Between Students in Colleges and Universities, Review of Educational Research, Vol. 68, No. 3, Fall 1998
2. Calibrated Peer Review at http://cpr.molsci.ucla.edu
3. E. F. Gehringer, Electronic peer review and peer grading in computer-science courses, Proc. of the Technical Symposium on Computer Science Education, 2001, pp. 139-143.
4. S.S.J. Lin, E.Z.F. Liu & S.M. Yuan Web peer review: The learner as both adapter and reviewer. IEEE Transactions on Education, Vol. 44, No. 3 2001, pp. 246-251
5. S. Trahasch, From peer assessment towards collaborative learning, Proc. Frontiers in Education Conf., Oct. 20-23, 2004, Savannah, GA.
6. A. Ward, J. Sitthiworachart & M. Joy, Aspects of web-based peer assessment systems for teaching and learning computer programming, IASTED Int. Conf. on Web-based Ed.,. 2004
7. E.F. Gehringer, Assignment and quality control of peer reviewers. Proc. of the American Society for Engineering Education Annual Conf., 2001
8. R.M. Crespo, A. Pardo & C. Delgado-Kloos, An adaptive strategy for peer review, Proc. Frontiers in Education Conf., Oct. 2004.
9. A. Inaba, R. Mizoguchi, Learners' roles and predictable educational benefits in collaborative learning. An ontological approach to support design and analysis of CSCL. Lecture Notes in Computer Science, Vol. 3220, 2004 pp. 285-294
10. R.M. Crespo, A. Pardo, J.P. Somolinos & C. Delgado-Kloos, An algorithm for peer review matching using students profiles based on fuzzy classification and genetic algorithms, Lecture Notes in Computer Science, Vol. 3533, Jun. 2005, pp. 685-69

Correction Marks and Comments on Web Pages

Bo Hu

Universität der Bundeswehr München, D-85577 Neubiberg, Germany
bo.hu@unibw.de

Abstract. In this paper we present our recent work to develop a tool that enables making correction marks and comments to web pages. The tool is written in Javascript/DHTML and supported by major browsers. The correction marks and comments are added directly to a web page in such a way that the layout of the original page is not affected on the screen. Along with our XML centric web content management system this tool provides a browser-based teaching and learning environment for writing with guidance. This technique can also contribute to enhance personalization, because every user may use this technique to make his or her own annotations to any web page of our web site.

1 Introduction

Writing with guidance is a widely used learning and teaching technique. One of the key steps of this process is that the teacher marks the student's written works after handing in, including making his or her correction marks and comments.

In a paper based environment each of these correction marks and comments is made as a combination of textual mark and marginal mark [1,2,3,4]. The textual marks are used to indicate the exact place to which a correction instruction or a comment refers. The marginal marks are necessary because there is often not enough place for detailed instructions within the printing area.

For a web page, in contrast to a page of paper, only some commenting tools are available using special designed margins [5] or simply placing the comments at the end of the commented page. We as well implemented a sticky note [6] commenting tool using layering HTML elements [7]. Because of the lack of textual mark all of these tools can not be used for a proof correction instruction.

In our recent work we developed a Javascript/DHTML user interface which makes it possible to insert textual marks to existing web pages. In section 2 we give an overview of the key requirements and the technical realization briefly before we come to conclusion in section 3.

2 System Overview

The "Tool for Correction Marks and Comments" provides different possibilities to add, read or remove public or private correction marks and comments.

Since different users may add their correction marks and comments to a single web page we introduce a class "CMC" as a collection of all correction marks and comments

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 784 – 786, 2006.

which a user makes to a web page during a session. Depending on the privilege of the user the correction marks and comments are made public or kept private, i.e., only to be displayed to the author(s) of the commented page and the user himself. These private correction marks and comments are not only a possibility to write down personal notes onto a web page but can also be used as a discreet way to inform the web master about typos and other problems of a page.

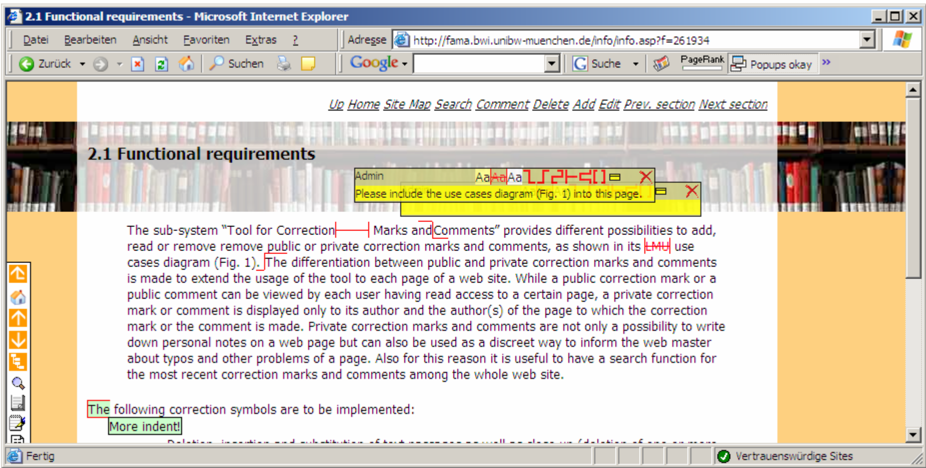


Fig. 1. User interface of the "Tool for Correction Marks and Comments". Note that both sticky notes next to the title can easily be moved to elsewhere. Five correction marks are added to the page: 1) Deletion of multiple spaces; 2) Insertion of a space; 3) Substitution; 4) Start new paragraph; 5) Indent. When the mouse cursor is moved over a textual mark the corresponding correction instruction ("More indent!") is displayed using layer technique.

Each object of the class "CMC" is embodied by a "sticky note". This sticky note has multiple functions: 1) It contains a multi-lined input field in which a user can enter his or her comment to the whole web page. 2) It shows the name of the user. 3) Most importantly it provides a toolbar which can be used to enter correction marks and comments to the text of the web page (Fig.1).

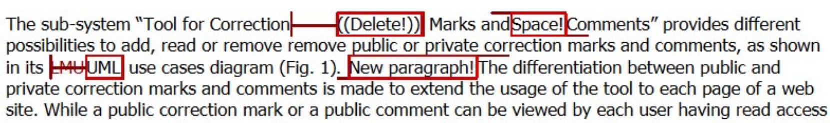

The sub-system "Tool for Correction ((Delete!)) Marks and Space! Comments" provides different possibilities to add, read or remove remove public or private correction marks and comments, as shown in its ~~LMU~~ UML use cases diagram (Fig. 1). New paragraph! The differentiation between public and private correction marks and comments is made to extend the usage of the tool to each page of a web site. While a public correction mark or a public comment can be viewed by each user having read access

Fig. 2. Correction marks in printout

To start a correction and comment session one first selects the text passage to which he or she wishes to make a correction mark or a comment. Then he or she clicks on one of the symbols of the toolbar to activate the corresponding function to place the textual mark into the text passage. Different correction symbols like deletion, insertion and substitution of text, insertion of a space, start new paragraph, run on (no new paragraph) have been implemented. Correction instructions and generic comments can be added. Note that each of the correction marks can simply be deleted.

Since there is often no place for marginal marks on a web page a correction instruction is only displayed when the mouse cursor is moved over the textual mark (Fig.1). This is achieved by Javascript/DHTML technique. When printing out the page a second set of style definition is used so that all correction instructions are printed together with the textual marks (Fig. 2).

3 Conclusion

Using this “Tool for Correction Marks and Comments” which is supported both by IE and Mozilla making correction marks and comments to web pages is much easier and more interactive. Typos and other problems on a web page can be marked and communicated within seconds. From our point of view it is a step towards “Web 2.0” as “an emerging network-centric platform to support distributed, collaborative and cumulative creation by its users” [8]. Comparing with existing systems like Acrobat, for instance, one does not need anymore to download the web page to a local file, to edit, to save, to upload and then to link the file.

However, one limitation of this tool is that it cannot be used to make correction marks or comments to graphics included on a web page. Since graphics and graphic input devices are more and more widely used it is a shall be a challenge to add graphical interactivities to the web.

References

1. British Standards Institution: BS 5261C:1976, Marks for copy preparation and proof correction. ISBN 0 580 09057 4 (1976)
2. U. Hermann et. al: Korrekturzeichen, nach DIN 16511, Die neue deutsche Rechtss-chreibung, Bertelsmann Lexikon Verlag, Gütersloh, P.15-17, ISBN 3-577-10625-5 (1996)
3. Conrad Taylor: Proof Correction Marks, http://www.ideography.co.uk/proof/marks.html (1999)
4. Miles Rind: Key to Correction Marks http://people.brandeis.edu/~rind/bentley/faults.html#symbols (2004)
5. W. Sesink et al: eMargo - the Digital Margin. http://www.emargo.de/www/english/ (2005)
6. David R. Karger et al: Sticky notes for the semantic web, Proceedings of the 8th international conference on Intelligent user interfaces, January 12-15, Miami, USA (2003)
7. Bo Hu, Kai Koster: Commenting Tools For An XML-Centric Collaborative Authoring And Learning Environment, IADIS International Conference WWW/Internet 2005, Proceedings Vol. II, P. 222-225, ISBN 972-8924-02-X (2005)
8. John Hagel: What is Web 2.0? http://edgeperspectives.typepad.com/edge_perspectives/2005/09/what_is_web_20.html (2005)

VIBRANT: A Brainstorming Agent for Computer Supported Creative Problem Solving

Hao-Chuan Wang[1,3], Tsai-Yen Li[2], Carolyn P. Rosé[3], Chun-Chieh Huang[2], and Chun-Yen Chang[1]

[1] National Taiwan Normal University, Taipei, Taiwan
[2] National Chengchi University, Taipei, Taiwan
[3] Carnegie Mellon University, Pittsburgh, PA, USA
{haochuan, cprose}@cs.cmu.edu, {li, g9415}@cs.nccu.edu.tw, changcy@ntnu.edu.tw

Abstract. This paper describes key issues underlying the design of a tutoring system that brainstorms with students in order to support qualitative problem solving. Cognitively oriented and socially oriented support are enabled by two technologies, namely heuristic-based feedback generation and community-data-driven social recommendation. Formal representations and corresponding automated reasoning procedures for these technologies are introduced.

1 Introduction

Research in the area of Intelligent Tutoring Systems (ITS) has achieved impressive results in improving student learning especially in well defined problem solving domains, such as basic algebra (Koedinger et al., 1997) and quantitative physics (Vanlehn et al., 2005). In this paper we begin to consider how to expand the frontiers of this success into areas less touched by ITS research, specifically, the task of more open ended qualitative scientific problem solving, also known as creative problem solving (CPS) that has been attracting more attention in education. Recent work begins to approach this area, such as tutorial dialogue systems (Kumar et al., 2006). A tutoring system that brainstorms with students, called VIBRANT (Virtual Brainstorming), is proposed as an instructional tool for science education.

Consider the following sample CPS question: "What are the possible factors that might cause a debris-flow hazard to happen?", and subsequently, "How could we prevent it from happening?" In our previous work, students were required to answer CPS questions independently without accessing external resources or peer support. Human graders were recruited to score student answers quantitatively using a rubric devised by domain experts (Chang & Weng 2002). However, this operationalization of CPS has previously been used primarily in evaluating particular teaching methods, and the focus was not specifically on how to scaffold students' idea generation.

This work proceeds to focus on system behavior for supporting idea generation, which can benefit from collaboration (Kraut, 2003). Towards maximizing the effectiveness of brainstorming, VIBRANT offers (1) cognitively oriented support providing brainstorming feedback and (2) socially oriented support for discussion group formation. Brainstorming feedback is intended as an interface for students to receive a

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 787 – 789, 2006.

contextualized view of relevant learning resources; while the social support seeks to recommend suitable peers to students for fostering collaborative brainstorming.

2 Overview of VIBRANT

Based on our prior work (Wang et al., 2005), knowledge of solving a CPS task is modeled as a bipartite graph-based formal user profile (fUP), in which the connections between a student's ideation and explanation are explicitly represented.

Intelligent Support. A formal ontology is constructed serving as the core device for organizing experts' CPS ideas and learning resources. The ontology consists of an `is-a` hierarchy organizing experts' ideas into several levels of abstraction. Prescriptive feedback messages are attached to specific idea nodes at lower levels and categorical nodes at higher levels. Finite state machines (FSMs) are designed to retrieve learning resources such as feedback. In FSMs, the finite set of *states*, Q, represents the range of the system's functional behavior, including actions such as `check_coverage` or `move_upward_in_hierarchy`, while the finite *alphabet*, Σ, represents the set of events that trigger transitions from one state to another, such as `all_sub_nodes_covered` which in this case triggers a transition to a state called `get_new_cateogry`. Transition functions $\delta : Q \times \Sigma \rightarrow Q$ represent instructional decisions that trigger appropriate behavior when events are observed.

The *feedback* prepared by the system consists of two parts, a *comment* and a *tutorial*. *Comments* are evaluative texts responding directly to the most recent idea submitted by the student to the system, while the *tutorial* is the instruction that directs the student to the next logical focus node, which may either be an idea node or a categorical node, selected by the system adaptively based on its model of the student. The use of the `is-a` hierarchy is considered beneficial for the FSM-based feedback generation. *First*, the hierarchy of topics provides a basic for supporting a more organized and coherent brainstorming process. The system may select a next focus for tutorial to maximize the students' local coverage of categorical nodes that have been partially addressed by the students' idea entries. *Second*, *comments* can be fetched strategically at a more generalized level in the ontological hierarchy when a particular idea entry is semantically ambiguous and thus results in low similarity scores as computed using vector-based information retrieval (IR) methods. The strategy may help remedy the insufficiency of IR-based methods for computing semantic similarity and to improve the relevance of system-prepared comments against students' ideas.

Social Support. Given a collection of historic fUPs created for previous students, we may re-model the system of fUPs as a tripartite graph with hyperedges. The condition of student p having an idea q and explaining it by reason r can be represented as a hyperedge e_{pqr}, which results in a tripartite graph H=(V, E) where V=$S \cup A \cup B$ and E=$\{e_{pqr}\}$. A variety of analyses can then be computed over the tripartite graph for social structure discovery, such as using local heuristics to extract particular (hyper) edges as cues for social recommendation or applying Social Network Analysis (SNA) methods to incorporate macroscopic and structural information.

3 Evaluation and Current Directions

We conducted an evaluation of VIBRANT feedback generation using a corpus containing 61 CPS idea entries from 10 Taiwanese high school students. The with-hierarchy method generates feedback using the aforementioned approach, while the without-hierarchy method does not make use of a category-based brainstorming plan, so that the selected brainstorming focus motivating the tutorial is based purely on information from the student's most recent contribution. In this case, the comment offered is the comment attached to the most similar node in the domain model.

Two independent judges were recruited to rate the relevance of each feedback message prepared by each of the two versions for every idea entry in the corpus quantitatively. Thus, each coder assigned a relevance score to 122 feedback messages. The result reveals no significant difference on average between the two methods. Furthermore, there was no significant correlation between the relevance scores assigned by the two coders, which casts doubt on the reliability of the evaluation metric. Nevertheless, the trend was in favor of the with-hierarchy approach. Furthermore, the standard deviation of scores across student entries was lower for both coders in the with-hierarchy approach, which indicates that the with-hierarchy approach may be more consistent in its quality. This makes sense given that in some cases local information is sufficient for generating meaningful feedback, while other times context is helpful. In our current work we are refining our operationalization of the notion of "relevance". Follow-up evaluations are planned in the near future.

Moving forward, our plan is to employ VIBRANT as a test-bed for conducting intervention studies and further testing social psychological hypotheses in educational contexts, including verifying and measuring the benefits of group brainstorming.

References

1. Chang, C-Y., Weng, Y-H. (2002). An Exploratory Study on Students' Problem-Solving Ability in Earth Sciences. I*nternational Journal of Science Education*, 24(5), pp. 441-451.
2. Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8(1), 30-43.
3. Kraut, R. E. (2003). Applying Social Psychological Theory to the Problems of Group Work. In J. Carroll (Ed.), *HCI Models, Theories, and Frameworks* (pp. 325-356). NY: Morgan-Kaufmann Publishers.
4. Kumar, R., Rose, C., Aleven, V., Iglesias, A., & Robinson, A. (2006) Evaluating the Effectiveness of Tutorial Dialogue Instruction in an Exploratory Learning Context. *Proceedings of International Conference on Intelligent Tutoring Systems (ITS 2006).*
5. Wang, H.-C., Li, T-Y., & Chang, C-Y. (2005). A user modeling framework for exploring creative problem-solving ability. *Proceedings of 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, 941-943.
6. Vanlehn, K., Lynch. C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., & Wintersgill, M. (2005). The Andes physics tutoring system: five years of evaluation. *Proceedings of 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, 678-685.

Iterative and Participative Building of the Learning Unit According to the IMS-Consortium Specifications

Jamal Eddine El khamlichi, Françoise Guegot, and Jean-Pierre Pecuchet

FRE CNRS 2645 PSI – INSA de Rouen
BP 08, Place Emile Blondel
76131Mont Saint Aignan Cedex France
Phone: +33 02 35 52 84 73
{jamal.elkhamlichi, francoise.guegot, Jean-Pierre.Pecuchet}@insa-rouen.fr

Abstract. This paper presents an iterative and participative approach of designing, automatically producing and using a reuse learning unit. Our approach proposes five design processes: analysis, design, implementation, evaluation, and validation and publication. They are followed by the teacher to build the learning unit. Our approach is also systemic. It proposes a set of tools which supports the designing and the building of learning unit for the e-learning LMS. The designing tool proposes set of formalisms which represent the graphical objects allowing the graphical representation of the educational scenario of the learning unit. The designing and authoring tools produce the learning unit according to the IMS-consortium specifications. The learning unit is a package of physical files. It placed after their building under a sharable repository to be reused by other organisms (universities, companies, etc.). We use the CopperCore player to play the educational scenario of the learning unit.

1 Introduction

In this last decade, two approaches have appeared to treat the problem of contents reuse. The first approach is interested in the reuse of the LOs (Learning objects). The aim is to create repositories of LOs shared on Web. ARIADNE, COLIS, Edusource, DLESE and MERLOT are examples of projects which treat the reuse of LOs. The second approach is interested by the reuse of the educational scenario. Educational languages as IMS (Instructional Management Systems), EML (Educational Modeling Language), MISA (Méthode d'Ingénierie des Systèmes d'Apprentissage) have appeared. They propose models which treat design and reuse of LUs and their educational scenarios.

Our aim is to help the teachers, to design, to product and to publish the LUs and their educational scenarios which can be used by several LMSs. For this purpose, we propose an iterative and participative approach to build a LU according to the IMS-consortium specifications.

This paper is organized in two sections. The first section presents the package of the LU and its educational scenario. The second section presents the processes of design used to design, to produce and to play the LU.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 790 – 793, 2006.

2 Learning Unit

In concrete terms, the LU is a package of physical files (XSD, XML, html, etc.). The package is composed of two major components [1; 2]: (1) the manifest file and (2) the resources files. The package is a zip concise physical file.

The manifest file is an XML (eXtensible Markup Language) document called “imsmanifest.xml”. The <manifest> is the root of this file. It is composed of three direct children: <metadata>, <organization> and <resource>. The <metadata> child describes the manifest and other elements as a whole. The <organization> child is the major element. It describes how the activities are organized to be delivered to the learner. This organization is called educational scenario. Without detailing the elements which describe the educational scenario, we remind that it takes place as a theater play. The actors play the roles to realize the activities by using the computer environment. The different acts of actors form the play. The method of education scenario takes place according to one or more than one play. The educational scenario can be personalized by the conditions and properties applied to the method, acts actors and roles. It can be also become dynamic by using the notification mechanism. The last child is a collection of references to resources.

The resources files are the local and external files making up contents of a LU. These are an electronic representation of media, such as text, sound, images, animations, graphs or any piece of data that can be rendered by a web and presented to the learners. Each of these medias can have a multiple forms of representation. For example, a sound can have the following format WAV, MP3, MPC. A physical file can be created by the teacher of LU or reused from a repository.

3 Design of the Educational Scenario

The design of the educational scenario is consists of five processes:

- Analysis, this process consists in specifying of use context of educational scenario. Thus it is in particularly a question of:
  - Identification of public characteristics: their knowledge, their skills, their master of the computer tools, etc.
  - Identification of material constraints by knowing the users computer equipment (Internet connection speed, computers power, etc.).
  - Precision of resources available: computers, software, humans, etc.
  - Enumeration of the educational objectives (general and\or specific).
  - Description basic of the first ideas in terms of the educational strategies and the Progress of activities.
- The design process consists of modeling the activities of the various roles as well as the input and the products of every activity. We use the MOT tool for designing of the educational scenario. This tool developed in LICEF [3] (Laboratoire d’Informatique Cognitive et Environnements de Formation) Canadian research center. It is a simple and powerful tool. It can be used for graphical representation of knowledge and their relation in several domains (education, architecture, electronic, informatics, etc.) [4].

- In our work, we use this tool, precisely the last version (version 1.5.0) for graphical designing of educational scenario. And for production of content package of LU in a compressed format (zipped format).
- Implementation processes are elaboration of the necessary materials for the realization of the educational scenario and conversion of the educational scenario to a content package. We want to indicate here that the MOT tool does not allow personalizing or managing the dynamics of the educational scenario. For this reason, we use the RELOAD (Reusable ELearning Object Authoring & Delivery) [5] Learning Design Editor. This tool is available as open source on the Web. In fact, the teacher gets back the package generated by the MOT tool and opens it in the Reload tool. Then, he can personalize and make dynamic the educational scenario. The level B corresponds to the educational scenario personalization level. The level C allows making dynamic educational scenario. While the level A corresponds at the general description of the educational scenario already realized by MOT tool.

  When the teacher ends the production of package of LU, he can validate and publish it by using the CopperCore LDE (Learning Design Engine) [6]. This last is released by OUNL (Open Universiteit of NederLand). It is world first open source IMS LDE that supports all three levels of IMS Learning Design (A, B and C). It verifies if a package is in accordance with the IMS-consortium specifications. Also, it publishes the package to be used by the CopperCore player. This last possesses a player which can be used in the test environment as part of a reference implementation of LMS [7]. The actors can play the educational scenarios by using CopperCore player.
- Evaluation process consists of measuring the educational, didactic and material efficiency of the educational scenario. Evaluation has two levels. The most important is to gauge the success of the participant in obtaining and retaining the demonstrated skills and understandings. The second is to determine how successful the educational scenario was in facilitating effective participant learning.
- Validation and publication consists of sharing the package on a repository accessible on a distance server. This process allows the reuse of LU by a multiple LMSs.

These processes are considered as iterative and participative processes:

4 Conclusion

The main objective of our research work is to help the teacher to design, produce and publish the reuse LU and their educational scenario. We proposed in this paper an approach allowing accomplishment of this objective. Its principal characteristics are the iterative of processes design and participation of the actors (learners, tutors, etc.) in the design processes. It is also systemic because it proposes the tools allow the accomplishment of different design processes. The educational scenarios are designed in a graphical way for reasons that we showed previously. Also, they are produced according to IMS-consortium specifications to answer the problem of reuse.

This method allows elaborating the repositories of the LU. The educational scenarios of the LU can be played by the players as CopperCore. This approach allows reducing the costs of creation of the LU. It also allows using effective educational scenarios because they are evaluated before being published on Web.

References

1. Elkhamlichi, J. E., Guegot, F., Pecuchet, J. P. (2005). Design and generation of Collective Educational Activities. In Proceedings of the 5th IEEE International Conference on
2. IMS-LD, http. IMS Learning Design Information Model. Available at: http://www.imsglobal.org/
3. Paquette, G. (1996) La modélisation par objets typés : une méthode de représentation pour les systèmes d’apprentissage et d’aide à la tâche, Sciences et techniques éducatives, 3, 9--42.
4. LICEF, http. Web page of LICEF laboratory. Available at: http://www.licef.teluq.uquebec.ca/eng/index.htm
5. Colin, D. M., Beauvoir, P., Sharples, P., (2005) Learning Design based on Graphical Knowledge-Modelling. In Proceeding of the UNFOLD/Prolearn joint workshop Valkenburg. Edited by R. Koper, C. Tattersall, and D. Burgos. Heerlen, Sept. pp. 317--323.
6. CopperCore, http. Official Web page of CopperCore environment. Available at: http://coppercore.org/
7. McAndrew, P., Nadolski, R., Little, A. (2005) Developing an approach for Learning Design Players. In Proceeding of the UNFOLD/Prolearn joint workshop Valkenburg. Edited by R. Koper, C. Tattersall, and D. Burgos. Heerlen, Sept. pp. 351--358.

Ontology Mapping for Learning Objects Repositories Interoperability

Amel Bouzeghoub and Abdeltif Elbyed

GET/INT, 9, rue Charles Fourier,
91011 Evry, France
{Amel.Bouzeghoub, Abdeltif.Elbyed}@int-evry.fr

Abstract. In order to deal with the need of sharing learning objects within and across learning object repositories most of the recent work argue for the use of ontologies as a means for providing a shared understanding of common domains. But with the proliferation of many different ontologies even for the same domain, it become necessary to provide mapping process to perform interoperability. The main key issue must be addressed is to define educational systems interoperability mechanisms to create a virtual learning space. Although many efforts in ontology mapping have already been carried out, few of them use resources properties to generate semantic relations between local concepts. Our approach uses inference rules to combine several matchers in order to discover mapping dynamically and to improve the results qualities.

Keywords: Learning Resources, Ontology Mapping, Interoperability and Multi-Agent Systems.

1 Introduction

Ontologies offer a great potential in higher education providing in particular the sharing and reusing of information across educational systems and enabling intelligent and personalized learner support. The increased functionalities that ontologies imply will bring new opportunities to e-learning. Learners will be able to interact with distant educational systems easily and in a personalized way. An overview of ontologies for education field and an initial report on the development of an ontology-driven web portal O4E are presented in [2].

We propose in this paper an algorithm which is applied on a Web Based Educational System (WBES) named SIMBAD [1]. To facilitate resources exchange with other WBES, it becomes necessary to find solutions allowing the cooperation between various repositories of learning resources. The user may seek resources out of his/her private reference ontology. The problem is that the comprehension of a new classification (a new ontology) is expensive and does not constitute a justified investment. It is thus necessary to propose mechanisms which allow the user to access to resources of other repositories in a transparent way using his/her favourite WBES.

The particularity of this approach is that (i) it focuses on dynamic ontology mapping using a multi-agent system, (ii) it uses the resources properties to enrich the local ontology by generating semantic relations between local concepts (iii) it is based

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 794 – 797, 2006.

on inference rules to compare the ontologies concepts. These inference rules may be general ones or more specific rules added by an expert. This flexibility allowed the algorithm to be applied to other domains.

In this paper we introduce a dynamic mapping approach for bridging gaps between learning object repositories based on ontologies. Dynamic ontology mapping means that during a user interaction (query), the mapping system receives a sequence of concepts and returns the most specific mapping for each concept.

2 Algorithm Principal

The objective of our approach is to map ontologies dynamically, and only when needed. The system tries to find semantic relations between the concepts of the user query and the concepts in the target ontologies.

The algorithm is based on a multi-agent system and it combines different similarity measures to find mapping candidates between two ontologies. We distinguish three main categories of similarity: linguistic similarity, structural similarity and rule-based similarity. Using these different similarities may increase the precision of the results.

In this section we describe the global architecture and the agents' behaviours of the mapping process.

2.1 Architecture

The ontology interoperability needs to define mapping between ontologies. In our architecture (figure 1) the mapping process is split up into five levels: (i) resources level contains the various resources repositories (ii) ontology level describes the resources classification (iii) interface level where we can find, the users and a set of the ontology agents (OA) which generate new ontologies enriched with semantic relations, (iv) simulation level where we calculate the similarity between candidate concepts and (v) domain expert level allows to a expert to interact with the simulation level.

2.2 Mapping Process and Agents Behaviors

The algorithm begins by generating information from the ontology. The ontology agent OA uses the instances (resources) comparisons for deducing semantic relations between concepts of the same ontology to deduce a new ontology O+. The OA agent which intercepts a user query generates all possible relations between the query concepts and sends both concepts and these relations to all other OA agents.

The simulation level contains four agents: SCA (Similarity Computation Agent), GHA (Generation Hypotheses Agent), FHA (Filtration Hypotheses Agent) and CHA (Choice Hypotheses Agent). We describe in the following the general behaviour of each simulation agent:

The SCA agent determines similarity values of candidate mappings via different matchers. The first iteration consists in providing a basic similarity between concepts. In this iteration we use linguistic tools [6, 7] to compare concepts' names. In the ith iteration we use the similarity produced in $(i-1)^{th}$ iteration and we apply the inference rules. These inference rules are either rules inferred from structural similarity or rules proposed by the domain expert.

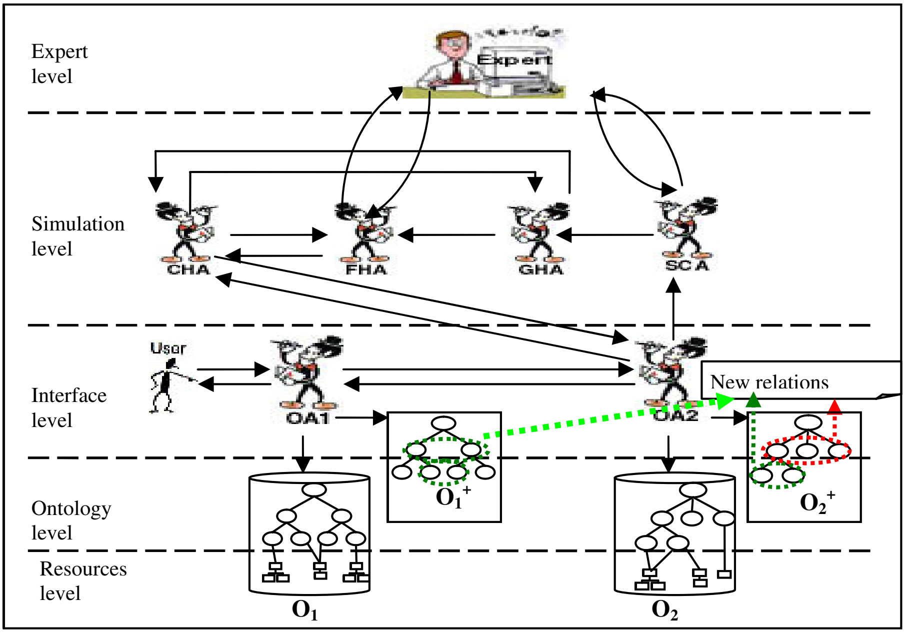


Fig. 1. Architecture approach for mapping process

The GHA agent receives all similarities sent by SCA and it generates hypotheses using inference rules. These hypotheses consist of new correspondences between concepts. The generation of a hypothesis at i^{th} iteration is based on either the mapping set or the similarities generated previously. Indeed, depending on the similarity value, we generate mapping hypotheses between the couple of concepts which have a similarity value enough important.

The FHA agent filters all hypotheses generated by GHA. The hypotheses which do not verify certain constraints (e.g. structural constraints) are removed. The subset of filtered hypotheses is sent to CHA agent.

The CHA agent selects the hypotheses which have the best similarity using both existing mapping and user's feedback.

The final mapping is sent to ontology agents. After several user interactions, each OA acquires more knowledge about other OA(s) and defines a set of most relevant OA(s) (i.e. the OA(s) that answer to its needs). This set is called "agent's accountancies".

3 Conclusion

Various works [3, 4, 5] have been developed for supporting the mapping of ontologies. Most of them are based on syntactic and semantic matching heuristics given by an expert to generate static mapping. None uses inference rules which can be generated for different application domains. In our mapping approach, we try to use as much as possible available information contained in the ontology to determine dynamically and if necessary the relationship between concepts. This information consists of identifier name of concepts, ontology structure, manual/automatic rules and resources properties which generate new semantic relations between concepts of the same ontology.

In future work, we plan to add other match and techniques in order to resolve more complex mapping problems. The implementation of the prototype is in its first phase. It will be followed by a series of the experiments to validate and justify the quality of the approach in sharing learning resources.

References

1. Bouzeghoub A., Defude B., Ammour S., Duitama J.F., Lecocq C.: A RDF Description Model for Manipulating Learning Objects, Proc. IEEE International Conference on Advanced Learning Technologies, Joensuu, Finland, August 2004
2. Dicheva1, D., Sosnovsky, S., Gavrilova, T., Brusilovsky, P.: Ontological Web Portal for Educational Ontologies. International Workshop on Applications of Semantic Web technologies for E-Learning (SW-EL'05) july 2005
3. Euzenat J., Le Bach, T., Barrasa J., Bouquet P., De Bo J., Dieng-Kuntz R., Ehrig M., Hauswirth M., Jarrar M., Lara R., Maynard D., Napoli A., Stamou G., Stuckenschmidt H., Shvaiko P., Tessaris S., Van Acker S., Zaihrayeu I.: State of the art on ontology alignment. Deliverable 2.2.3, IST Knowledge Web NoE, 21p., june 2004
4. Ehrig, M., Sure Y.: Ontology Mapping - An Integrated Approach April 21,2004, ESWS 2004: 76-91
5. Kalfoglou Y., Schorlemmer M.,Ontology mapping: the state of the art, The Knowledge Engineering Review 18(1):1--31, January 2003
6. Silva, N., Rocha, J.A.: Ontology mapping for interoperability in semantic web. In Proceedings of the IADIS International Conference WWW/Internet 2003 (ICWI'2003), Algarve, Portugal, 2003
7. http://www.wordnet.com

Using Agents to Create Learning Opportunities in a Collaborative Learning Environment

Yongwu Miao[1], Ulrich Hoppe[1], Niels Pinkwart[2], Oliver Schilbach[1], Sabine Zill[1], and Tobias Schloesser[1]

[1] Institute of Computer Science and Interactive Systems
University of Duisburg-Essen, Germany
(miao, hoppe)@collide.info
(oliver.schilbach, sabine.zill, tobias.schloesser)@uni-duisburg.de
[2] Human Computer Interaction Institute
Carnegie Mellon University, USA
nielsp@cs.cmu.edu

Abstract. In order to foster situated learning in a virtual community of practice, we developed a multi-user, real-time, 3D car-driving simulation environment. In such a situation-based learning environment, the availability of enough appropriate learning situations is crucial for success. However, we experienced that often a collaborative usage of the system does not result in a large number of these critical situations. This paper introduces the idea of situation creators, intelligent agents who intentionally create specific situations for learners, into our 3D real-time simulation environment. These created situations challenge a learner much more and force him to react in order to master the driving knowledge.

1 Introduction

In recent years, more and more educational 3D simulation systems have been developed. In a typical training course with such a simulator, an individual trainee interacts with the simulation system through a series of pre-defined scenarios that are usually arranged with respect to growing complexity or level of challenge for the trainee. In contrast to these largely "pre-defined" simulation environments that are limited through the number of built-in scenarios, we adopted an alternative design approach for an educational car-driving simulator [1, 2]. Our simulation system allows multiple (also geographically distributed) learners to virtually drive in a shared driving place in a way close to driving in the real world. Furthermore, the simulation system also enables learners to communicate with peers whose cars are close to their own in the virtual driving place. This way, they can discuss the joint problems they face (e.g., who has the right of way in a certain situation). Therefore, each learner is both a member of community of practice and a component of the learning context. When the learner needs help, an intelligent *coach agent* will detect the current driving situations and analyze the learner's difficulties and needs, and then can provide situated guidance. Rather than going through a series of pre-defined driving scenarios (as in other environments), a learner will experience potentially rich and

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 798 – 800, 2006.

unpredictable driving situations in a way analogous to driving in the real world while learning in our simulation environment.

However, our approach has an obvious disadvantage: if there are not enough drivers in the shared virtual driving place, the number of challenging situations that a learner can experience will decrease correspondingly. This paper proposes an idea to address this problem. We introduce a new type of pedagogical agent, the *situation creator*, into our collaborative simulation environment. Such an agent intentionally creates specific situations in the shared virtual driving place. These situations provide learners with learning opportunities, and thus indirectly affect the learning processes.

2 Situation Creator

Our situation creator comprises three components: a learner model, a pedagogical model, and an expert model. The learner model helps to reason about appropriate situations to be generated for learners. The pedagogical model contains knowledge about driving situations and their prerequisite relationships, but does not contain knowledge about how to teach the learner to handle a specific situation. Finally, the expert model contains specific knowledge about how to create situations, but no explicit representation of domain knowledge to be taught. Normally, a situation creator does not even directly interact with learners (therefore, our architecture does not have the "user interface model" component of the classical ITS model), but just indirectly attempts to create the goal situations and thereby "induces" learning opportunities.

Figure 1 depicts the general architecture of a situation creator and its interactions with its environment. The internal structure of the environment depends on the application domain, which here it is represented as a black box. The situation creator works as a repeated process that starts with capturing the current state of the environment (perception) and ends with acting in the environment. Each cycle contains two phases separated by a dashed line in the diagram. The task in the first phase is seeking or maintaining a goal, and the second phase attempts to reach the goal through making and executing an action plan. As the figure 1 illustrates, the perception module monitors the state of the environment. It updates the information in the memory and the student models. The information about the current state of the environment is needed to evaluate if a goal situation has been reached. If not, the agent will check whether the target situation can be created in the current state or not. If it is impossible to achieve this goal, the agent has to give it up. After giving up a goal or achieving one, the situation creator agent tries to seek a new goal. It refers to information in the learner model and pedagogical model and decides which situation will be aimed at in the new state. For each new goal, the agent builds an action plan to achieve it. If the goal remains unchanged, the agent evaluates whether the current action plan needs revision based on the current system state. The specific knowledge about how to create a situation (in the expert model) is used to make and adjust the action plan. Finally, one or more actions will be performed. These, together with student actions, will certainly have an effect on the environment and cause a new process cycle.

According to this design, we have implemented a situation creator which can create three types of situations, A pilot study showed that the situation creator significantly increased the number of situations that a learner can expect to encounter while using the system. With the help of these created situations, a learner is much more challenged and forced to react in order to master the driving knowledge.

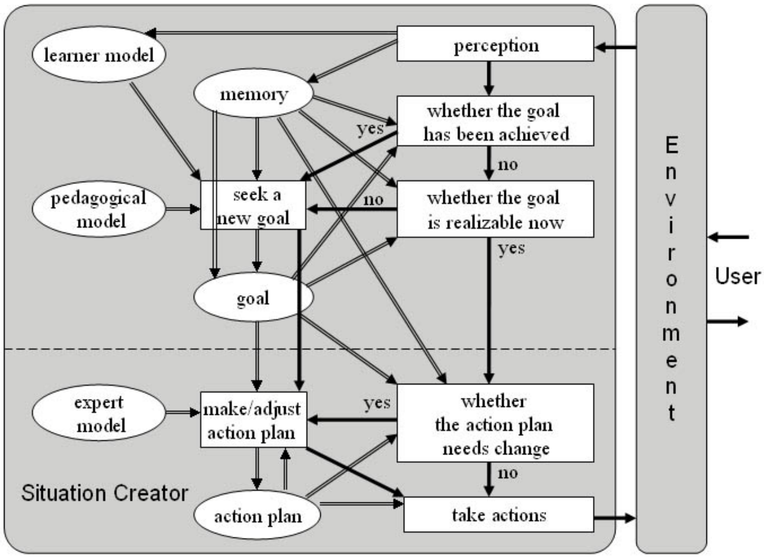


Fig. 1. The abstract system architecture

3 Conclusions

This paper introduced situation creator agents in our 3D collaborative simulation-based learning environment. In order to foster learning in our collaborative simulation environment, these agents create appropriate situations and provide learning opportunities for learners. This paper presented the generic architecture of a situation creator. Based on this design, we have implemented a situation creator that can create simple types of situations. We will extend it to generate more types of situations in the future.

References

1. Miao, Y. (2004). Supporting Situated Learning for Virtual Communities of Practice: Representation and Management of Situated Knowledge. Proc. of the 4th IEEE International Conference on Advanced Learning Technologies, pp. 490-494, Joensuu, Finland.
2. Miao, Y., Pinkwart, N., & Hoppe, U. (2006). Conducting Situated Learning in a Collaborative Virtual Environment. In V. Uskov (Ed.): Proceedings of the 5th International Conference on Web Based Education, pp. 7-12. Anaheim, CA: ACTA Press.

Context-Aware Annotation Patterns for Teachers

Faiçal Azouaou and Cyrille Desmoulins

CLIPS- IMAG, University Joseph Fourier, Grenoble
BP 53 38041 Grenoble cedex 9 France
{faical.azouaou, Cyrille.desmoulins}@imag.fr

1 Introduction

The semantic annotation tool MemoNote [1] dedicated to teacher enables teacher to annotate digital documents with his own comments and point of view as on paper. The annotations created in the tool have an explicit semantics for both teacher and machine. To assist the teacher to rapidly and fluently create these semantic annotations, the tool provides him with annotation patterns facility. However, the same teacher during his/her activity can work in different contexts, where he/she annotates differently, using a specific group of patterns in each specific context. Consequently MemoNote has to be context-aware in order to adapt to each teacher's context situation. This article studies and models the annotation context, in order to design context-aware annotation patterns dedicated to teachers.

2 What Is Teacher's Annotation Context?

From the definition provided by Dey [5], we propose our definition of annotation's context as: a*ny information that can be used to characterize the situation of entities that are considered relevant to the annotation activity in MemoNote.* Schilit and Adams [7] describe the context of a user by his/her environment: *computing environment*, *user environment and physical environment*. Chen & Kotz [4] adds the time to this list.

To identify the teacher's active context, we use generic context's studies [5] and research results about teacher's activity [6]. Finally, we obtain the following elements for the teachers' annotation context

- Teaching elements: Name, Position, Place, Teaching situation
- Computing elements: Device, Software (annotation tool)
- Time elements: Date, Hour, Period
- The teaching situation is described by the teacher's activity (design, doing…), the learning domain, the learning degree and the learner's activity.

The context capture depends on sources of context information: machine software (date/hour…), external device sensors, users themselves. In the case of the teacher annotation activity, the Learning Management Content System (LMCS) the better source for context's extraction if annotated document are managed with such system. If not, the annotation tool itself and the teacher himself/herself can also be asked.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 801 – 803, 2006.

3 Context Modeling

To choose between the possible representation models for context, we follow the assertion of Buccholz in [3]: "*A representation of the context information should be applicable throughout the whole process of gathering, transferring, storing, and interpreting of context information*". The most relevant models to respect this requirement are ontological models. Based on the context elements specified in section 1, we developed ontology of the context of a teacher's annotation (see fig. 1).

Annotation context
- Teaching elements
  - name
  - position
  - place
  - teaching situation
- Computing context
  - device
  - software
- Time

Fig. 1. Teacher's annotation context ontology

- Place
  - School place
    - Library
    - Labs room
    - Meeting room
  - Home
  - Public transportation
- Material
  - Personal assistant
  - Personal computer
  - Tablet pc
- Software
  - MemoNotePC
  - Web tool
- Teaching situation
  - teacher activity
  - learning domain
  - learning degree
  - learner activity
- Teaching activity
  - Assessment
  - Design
  - Doing
  - Preparation
- Learning domain
  - Chemistry
    - Organic chemistry
    - Mineral chemistry
  - Mathematics
    - Algebra
    - Geometry
- Learning degree
  - Bachelor
  - Master
- Learner activity
  - Course
  - Exercise
  - Labs
  - Simulation

Fig. 2. Ontology sub-concepts

4 Context-Aware Pattern

Based on the annotation context model provided in the previous section, we are now able to define context-aware patterns. The MemoNote current pattern formalism [2] is based on Alexander's approach but where the context is only descriptive. The context's annotation's model can be directly integrated into MemoNote pattern model by replacing this descriptive context attribute by the annotation's context specified in section 1, leading to context-aware patterns..

The current implantation of MemoNote's context-aware component provides three functionalities:

Pattern selection: depending on the current context, MemoNote activates only the patterns of which the context attributes suits with the current context.

Context-aware annotating: depending on the pattern used by the teacher to annotate, MemoNote deduces the annotation's semantics and records it with the context.

Pattern's management: when creating a new pattern, MemoNote assists the teacher by predefining pattern's context values using the current's context.

5 Conclusion

Apart from computational and time elements, what makes annotation context specific to teacher is its teaching situation facet, divided into teaching phase, domain, degree and activity. This context is modeled with an annotation context ontology that is directly integrated into the pattern's formalism leading to context-aware patterns.

Improving MemoNote context-awareness concerns first a deeper model of what means context's changes and how to define its granularity. The context use in MemoNote could also be extended to other functionalities of the tool, mainly what we call "remembrance", for example to launch and adapt the annotation retrieval, or to change annotation displaying according to the current device.

References

1. Azouaou, F., Desmoulins, C, "Teachers' document annotating: Models for a digital memory tool". *Int. J. Cont. Engineering Education and Lifelong Learning (IJCEELL)*. Vol. 16(1/2): p. 18-34. 2006.
2. Azouaou, F., Desmoulins, C., Mille, D. "Formalisms for an annotation-based training memory: Connecting implicit and explicit semantics". *10th Artificial and Education conference*. Sydney, Australia. pp. 374-376. 2003.
3. Buchholz, S., T. Hamann, and G. Hübsch. "Comprehensive structured context profiles (cscp): Design and experiences". *IEEE Workshop on Context Modeling and Reasoning (CoMoRea'04)*. Orlando, FL, USA. 2004.
4. Chen, G. and D. Kotz, "A survey of context-aware mobile computing research". Technical Report: TR2000-381 Dartmouth College. Hanover, NH, USA. 2000.
5. Dey, A.K., D. Salber, and D.A. G, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications". *Human-Computer Interaction (HCI) Journal*. Vol. 16(2-4): p. 97-166. 2001.
6. Grandbastien, M., "Ontologies pour l'utilisation de ressources de formation et d'annotations sémantiques en ligne, études et propositions à partir de cas d'utilisation." OURAL report, French government grant. Paris, France. 2005.
7. Schilit, B., N. Adams, and R. Want. Context-aware computing applications *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*. Santa Cruz, California: IEEE Computer Society Press. pp. 85-90. 1994.

On the Definition and Management of Cultural Groups of e-Learners

Ryad Razaki, Emmanuel Blanchard, and Claude Frasson

HERON Laboratory, Computer Science Department, University of Montréal,
CP 6128 succ. Centre Ville, Montréal, QC, Canada
{razakiry, blanchae, frasson}@iro.umontreal.ca

Abstract. One objective of our ongoing research is to be able to culturally adapt e-Learning. This paper is focused on describing a methodology to represent cultural groups of learners and adapt the learning session depending on the membership of learners to one or more cultural group.

1 Introduction

One of the main consequences of globalization in the domain of information is that individuals all over the globe have access to the same global media. However, it is perceived differently depending on the local culture. The important interconnection of the local and global is to be considered cautiously insofar that it generates cultural hybridization from which emerge new cultures [1]. E-learning with the use of Intelligent Tutoring Systems is no exception for it has become a global methodology with the use of Internet. This leads to two major conclusions. First, an ITS, in order to be trustworthy, must provide a response to this local sensibility. Second, since most cultural groups are the fruit of the hybridization of a certain number of cultures, an ITS must stay aware of those cross-cultural relationships. In this paper, we extend a previous work concerning a rules-based methodology to culturally adapt the response of eLearning systems and ITS [2]. We explain how we create the basic cultural groups that are needed in our methodology, how we translate cultural information into usable data for learning decisions. Finally we discuss issues associated with the deployment of a culturally aware system.

2 Reminder on a Process for Cultural Adaptation

Many elements related to the ITS field are obviously culturally-dependant such as emotional management, choice of pedagogical strategies, meanings given to concepts and symbols, reward allocation, test anxiety or ways of motivating people.

Inspired by the concept of Cultural Intelligence [3], we believe that a Culturally AWAre System must have the ability for cultural *understanding* (i.e. culturally interpreting a learner's behavior/feeling/result) and *adaptation* (i.e. displaying different interfaces and/or starting different learning strategies depending on learners' culture).

In our previously proposed methodology for cultural adaptation [2], *Cultural rules* are deduced from the cross-cultural literature. For instance, we can use the *Hofstede's*

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 804 – 807, 2006.

system of values [4] or *Schwartz's Value Inventory* [5] which represent national cultures with a set of dimensions and associated scores. The system uses these data as Cultural Facts that initialize a rule-based system. This system is aimed at determining certainty weights of pedagogically-related attributes (for example the interest for collaboration). To summarize, a cultural group is described as a vector of weighted attributes that we call Rules Weights Vector (RWV). Each learner has a similar RWV that is initialized depending on his cultural profile. The membership of a learner to each cultural group (called Membership Score) is also determined using the normalized distance between the learner's RWV and the RWV of a given cultural group. During the learning process and depending on learners' successes/failures, the weights of learners' RWV evolve and they in turn affect groups' RWV and also all the membership scores. Finally, all pedagogical resources and strategies are dynamically rated in order to represent the interest to use them with learners of a given cultural group. When a learner needs to learn some concept, pedagogical resources and strategies will be selected depending on the membership score of this learner to different cultural groups.

3 Creation and Management of Cultural Rules

The purpose of our system is to select an appropriate tutoring strategy, resource or behavior of a pedagogical agent according to a learner's cultural profile. That goal is reached through a process subdivided in three levels of decision as shown in figure 1.

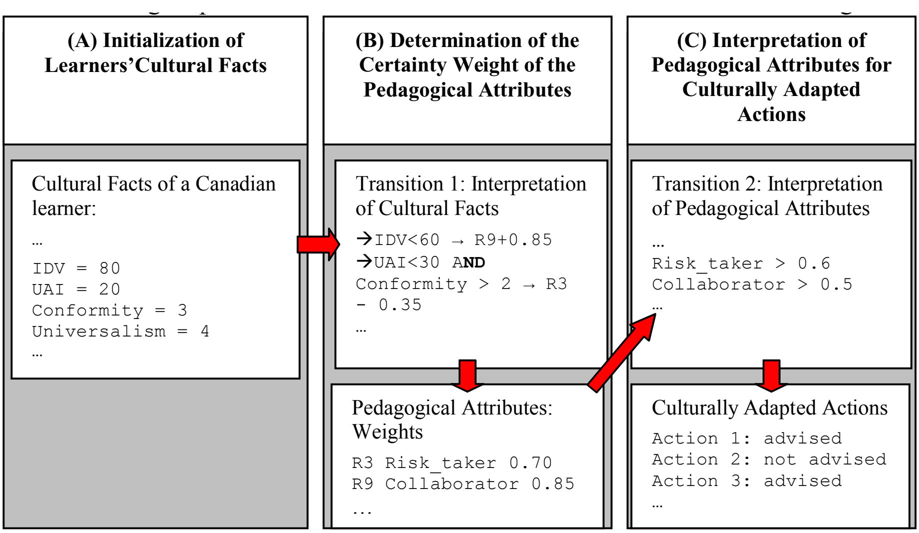


Fig. 1. The decision process for a Canadian Learner (IDV, UAI, Conformity and Universalism are cultural dimensions taken from [4] and [5])

First, the system reads the information submitted by the learner through a culturally focused questionnaire. *Cultural Facts* **(A)** are then deduced according to the cultural groups the learner has been determined to belong to. For instance the cultural facts can be values associated with Hofstede's national culture dimensions like IDV (Individualism) or UAI (Uncertainty avoidance).

Then, *Cultural Facts* are transformed into information usable for pedagogical decisions **(B)**. Those are properties describing the learner, coupled with weights that indicate the degree of his relation to the property. For instance in figure 1, the "collaborator" attribute illustrates that the probability the learner should like to work with other learners is 85%. The weights are increased or decreased using *Cultural Facts* as inputs for predicates in a rules engine.

Finally, the system interprets pedagogical attributes to determine which pedagogical actions are culturally suitable **(C)**. In fact the system selects *Culturally Adapted Actions* according to the weights of pedagogical attributes. For instance, depending of the weight of the learner's "collaborator" attribute, the system selects either a collaborative task or a self-oriented scenario to present the learning content.

To facilitate the process of description of cultural groups, we implemented a tool to create new cultural groups from scratch by attributing scores to cultural facts. In this tool, we introduced a mechanism of *group inheritance*, which means that some groups are specialized groups of a broader one, or a combination of many groups.

4 Discussion on the Deployment of a Culturally Aware System with Large Scalability

In the face of a huge number of learners spread over the world, there are a few issues that must be considered in order to optimize the system's performances.

First issue: a dynamic update or RWV requires all learner profiles, and not only those currently logged into the system.

Second issue: since the cultural adaptation process is performed in real time, it would be more efficient to decentralize as much as possible the adaptation operations to lighten data traffic over the network and in order to respond quickly to changes in the profiles of learners as well as group.

The GRID topology appears to be an interesting way of meeting both these needs but further researches in order to integrate it in our system need to be done.

5 Conclusion

In this paper, we presented an approach to make ITS culturally aware. We described a rule-based decision process based on cultural facts obtained from the cross-cultural research field to allow cultural adaptation in ITS. Culture has a great importance on the way people behave and understand their environments and we believe that dealing with culture is a very promising research avenue.

References

1. Appadurai, Arjun. 1996. Modernity at Large. Cultural Dimensions of Globalization. Minneapolis: University of Minnesota Press.
2. Blanchard, E., Razaki, R., Frasson, C. (2005). Cross-Cultural Adaptation of eLearning Contents: a Methodology. International Conference on E-Learning, Vancouver, Canada.
3. Earley, C., Mosakowski, E. (2004). Cultural Intelligence. Harvard Business Review, October 2004, pp. 139-146.
4. Hofstede, G. (2001). Culture's Consequences: Comparing Values, Behaviors, Institutions, and Organizations across Nations, 2nd edition, London: Sage.
5. Schwartz, S. H. (1992). Universals in the content and structure of values: Theoretical advances and empirical tests in 20 countries. In M.P. Zanna (Ed), Advances in experimental psychology, vol 25. New York: Academic Press. pp. 1–65.

Applying the Agent Metaphor to Learning Content Management Systems and Learning Object Repositories

Christopher Brooks, Scott Bateman, Gord McCalla, and Jim Greer

Advanced Research in Intelligent Educational Systems (ARIES)
Department of Computer Science
University of Saskatchewan
Saskatoon, Saskatchewan S7N 5C9 Canada
cab938@mail.usask.ca, scott.bateman@usask.ca,
mccalla@cs.usask.ca, greer@cs.usask.ca

Abstract. This paper presents a systems approach to facilitate effective learning object acquisition through the use of communications, modeling, and the agent metaphor. By changing the emphasis that is usually placed on metadata creation and interpretation (which can be problematic) we instead focus on the pragmatics of end use data facilitated through agent negotiation.

1 Introduction

The phenomenal rise over the last decade in the use of e-learning systems, in both formal and informal educational settings[1], has begun to bring traditional learning groups and e-learning researchers closer together. Correspondingly, the amount of learning content available freely (or at minimal cost) has increased tremendously.

Despite this increase in the use of e-learning systems, the acquisition, evaluation, and then repackaging of instructional material remain a significant hurdle in the development of online courses. To remedy this, groups of traditional learning researchers have investigated and put into place several different flavours of learning object repositories – typically centralized data stores of content that include descriptive, standardised metadata. While holding out some promise, this approach has recently been criticized as often resulting in incomplete or incorrect metadata [1]. Further, this content usually has to be marked up manually by humans from standardized metadata sets that tend to be very large and contain vague or even fairly useless terms. Moreover, it is assumed that the consumer of the metadata will be a human being (say an instructional designer or a teacher putting together a course) so the metadata tends to be in human readable form. Finally, the approach ignores the importance of the context of end-use of the learning object, in particular the critical role that individual differences in learners plays in the success or failure of the interaction between learners and the learning object. Since the metadata sets are void of descriptions based on actual observed interaction of the learning object with real learners, and there is no role for flexibility or adaptivity during the actual learning process.

[1] See for instance, http://www.ivc.illinois.edu/pubs/enrollment/Fall_04.html

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 808 – 810, 2006.

We argue that fundamentally what is needed is an integration of techniques from "technology-based research", aimed at providing e-learning systems with more flexibility and adaptivity to individual learners in actual end use. This would limit the dependence on standard vocabularies of metadata terms, and reducing the human effort required to make learning object repositories useful. The key to our approach is the use of computational agents to represent the e-learning delivery tool (a pedagogical agent), as well as the learning object repository (a repository agent). The agents essentially act as brokers of content, negotiating contextually sensitive interactions between the various parties in a learning situation. We believe that once agents are introduced, there will be a number of benefits including an increase in the amount of personalization available, the reduction of time spent developing new courses, and an enrichment of the metadata that is associated with learning objects.

2 An Agent-Oriented Approach to Learning Object Workflows

A traditional workflow for learning objects sees an instructor creating a learning design (either formally or informally), acquiring learning objects that fit this design (usually from learning object repositories), and packaging these objects for delivery to students (usually aiming such packages at a course management system where each student will see the same content). How successfully an instructor can do this depends on a number of factors, including their firsthand knowledge of the kinds of students who might be taking the course, the amount of time an instructor has to search through content, and the diversity of the students taking the class.

There are a number of issues with this kind of learning object workflow. First, the need to access multiple repositories increases the amount of time an instructor must spend to build a list of candidate learning objects for a particular purpose. Once a set of candidate objects has been built, it still takes a fair amount of time for an instructional designer to evaluate the suitability of those resources for a given objective. Even if these resources are annotated with metadata (which often they are not), the designer needs to absorb this information, consider the value of the metadata given the context of the publisher of the metadata, and finally make a decision as to which object fits the circumstances best. Our anecdotal observations suggest that instructors spend a minimal amount of time looking at metadata records, and then being to review the learning content directly.

We believe the use of an agent-oriented architecture would allow for a reduction in the amount of time it takes for an instructor to put together a course, an increase in the amount of personalization that can be accomplished, and an enrichment of metadata within learning object repositories. Consider the case of a student trying to complete a class in Artificial Intelligence. In both the traditional and our approach an instructor would create an instructional plan of concepts and competencies they would expect a student to learn. This plan, unlike the traditional approach, would need to be codified into a machine understandable format and passed into the instructional design agent within the learning environment. The instructional design agent would then be responsible for understanding the plan and interacting with learning object repositories to obtain the appropriate content.

In order to provide personalized content, the designer agent would need to have an understanding of the learner for which it is trying to adapt, achieved through an interaction with the learner model the system maintains (either explicitly, though questionnaires, or implicitly through observations of interaction). Once the instructional design agent has understood the learner, an interaction with agents representing repositories can begin to find material of relevance to the learner. This interaction, unlike the traditional approach, is not just a query, but a negotiation between the repositories and the content management system, handled by the agents. In this negotiation both agents provide any information it sees as relevant and can omit that which is believes is unimportant or unreasonable.

At each step in the negotiation process, any two agents reason over the data they have and, apply business rules to try and achieve their goals. For the pedagogical agent the primary goal is to find material that will help this particular student to learn enough to complete the objectives stated in the task plan. Secondary to this, however may be the desire to reduce the amount of time it takes the learner to learn, the amount of financial cost of a learning object, or a reduction in the physical size of a learning object to increase performance. A repository agent, on the other hand, likely has a number of different goals depending on the institution it represents. Corporate repositories may have financial compensation and customer loyalty as significant goals, as well as desire to evaluate the effectiveness of material which is meant to be co-published in a traditional fashion, while institutional or community repositories may seek to provide low cost in-house developed material.

3 Conclusion

The key to the agent approach is to incorporate reasoning and negotiation into the computational methods used to support learning objects. The focus changes from finding universal standardized ontologies to describe content, to understanding the workflows underlying interaction strategies between agents to actually carry out various pedagogical and communication tasks. Re-use of learning objects is achieved by reasoning in context, from taking into account how learning materials are actually used, and from making individual differences among learners a key aspect.

Reference

1. N. Friesen, Final Report on the "International LOM Survey", tech. report Document 36C087, Canadian Advisory Commiittee for ISO/IEC JTC1/SC36, 2004.

Building an Affective Learning Companion

Rosalind W. Picard

MIT Media Laboratory
20 Ames St.; Cambridge, MA 02139
picard@media.mit.edu
http://affect.media.mit.edu

Abstract. About a half century ago, the computer became a model, metaphor and modelling tool privileging the cognitive over the affective, and engendering theories in which thinking and learning are viewed as information processing and affect is ignored or marginalised. In the last decade there has been an acceleration in efforts to redress this imbalance, developing technologies that can begin to measure and manage the role of affect, enabling new theories and interventions in which affect and cognition are appropriately integrated with one another. This invited keynote presents a vision for developing an automated learning companion that jointly supports a learner's affective and cognitive needs. In particular, I will describe our efforts at MIT to invent several of the affective technologies to enable such a learning companion. The talk will show examples of the state of the art with respect to affect sensing and recognition and with respect to developing strategies for responding intelligently to learner affect.

Keywords: Affect recognition, affective tutor, empathetic agents, frustration, learner emotion.

Acknowledgments. I would like to thank past and present members of the MIT Affective Computing group for their many contributions designing and developing new kinds of technologies. Examples in this presentation result from collaborations with Ashish Kapoor, Win Burleson, Rana el Kaliouby, Carson Reynolds, Marc Strauss, Selene Mota, Hyungil Ahn, Shaundra Daily, Yuan Qi, John Rebula, Barry Kort, and Rob Reilly. We are indebted to Ken Perlin and John Lippincott for their development of the agent character shown in this talk. I would also like to gratefully acknowledge collaboration with Art Graesser's group at Memphis. This work has been supported in part by the National Science Foundation under ITR 0325428, by the Media Lab Things That Think consortium, and by funds from NASA provided through Ahmed Noor. Any opinions, findings, conclusions, or recommendations expressed in this presentation are those of the presenter and do not necessarily reflect the views of the National Science Foundation or the official policies, either expressed or implied, of the sponsors or of the United States Government.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, p. 811, 2006.

Integrating Learning Processes Across Boundaries of Media, Time and Group Scale

H. Ulrich Hoppe

Institute for Computer Science and Interactive Systems
University of Duisburg-Essen, 47048 Duisburg, Germany
hoppe@collide.info

Integration. For a long time, computer support for learning has been justified by its potential for the acceleration and multiplication of learning activities, i.e. learning should be faster and reach more learners. Recently, we have seen *integration* as a theme and purpose of educational media usage of its own right. We will distinguish and elaborate on two perspectives: (1) the integration of media to support a smooth and seamless information flow in both virtual and face-to-face classroom scenarios, and (2) the use of ICT to bridge between different conditions of learning, such as individual, small group or large community activities as well as synchronous and asynchronous settings.

Learning objects and learning scenarios. The integration of media and of group scales relies essentially on mechanisms for handling emerging learning objects in terms of production, exchange, re-use and transformation. In the spirit of constructivist, learner centred pedagogical approaches and in contrast to standardised activities around pre-fabricated learning objects or materials, we assume that "emerging learning objects" be created by learners and learning groups in partly unanticipated ways. This assumption gives rise to specific new challenges for the indexing and retrieval of such learning objects (or products). In the absence of indexing through experts, learning object descriptions have to be derived from the learning situation with minimal input from the learners themselves. This constitutes a new challenge for intelligent support techniques, namely for the dynamic recognition and modelling of learning contexts on a semantic level.

Social context and awareness. Contextualised indexing allows for an asynchronous exchange of learning objects within larger anonymous learning communities based on semantic similarities. In this sense, objects of common interest may trigger social processes in learning communities and may complement other techniques for modelling and supporting social relations such as "social network analysis".

Reference

Hoppe, H.U.; Pinkwart, N.; Oelinger, M.; Zeini, S.; Verdejo, F.; Barros, B.; Mayorga, J.I. (2005). Building Bridges within Learning Communities through Ontologies and "Thematic Objects". In Proceedings of the *International Conference on Computer Supported Collaborative Learning* (CSCL2005), Taipei (Taiwan), June 2005.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, p. 812 , 2006.

Improving Adaptivity in Learning Through Cognitive Modeling

Kinshuk

Director, Advanced Learning Technologies Research Centre
Massey University, Palmerston North, New Zealand
kinshuk@ieee.org

Abstract. The increasing demand of distant education and the growing degree of diversity of the learner group have created the widespread practice of e-learning which takes place in virtual learning environments (VLEs). By exploring those VLEs, learners perceive, analyse, assimilate, and interact with the pedagogical presentation and then "construct" their understanding or develop certain skills of the domain.

In order to provide support for learners during the learning process, the VLEs have to demonstrate a certain degree of adaptivity/intelligence in knowing what the learners actually need, and provide means to meet their needs in a way that best suit the learners' cognitive abilities. Cognitive theories are therefore closely examined in this presentation to provide the theoretical basis on which the adaptive techniques can be developed and evaluated.

Although Adaptive learning systems attempt to reduce the cognitive load by tailoring the domain content to suit the needs of individual learners, it is not easy for the educators to determine the effective adaptation techniques. This talk will describe the formalization of cognitive traits to provide the educators an effective and practical way to employ adaptive techniques in their learning systems. Various learner attributes, such as working memory capacity, inductive reasoning skill, domain experience and the perception of domain complexity, need to be monitored and measured to determine the best suitable course of action. This talk will describe the development of cognitive modelling techniques to reliably monitor and measure such attributes.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, p. 813, 2006.

Embodiment, Embeddedness, and Experience: Foundations of Game Play for Identity Construction

Yam San Chee

National Institute of Education, Nanyang Technological University,
1 Nanyang Walk, Singapore 637616
yschee@nie.edu.sg

There is considerable interest today in the use of computer games for student learning. Researchers, as well as educators, recognize that games can engage students in sustained and focused mental activity for extended periods of time. Indeed, game playing often occurs at the expense of more traditional forms of learning and schoolwork. Should we bemoan this fact, or should we seize the opportunity to harness gaming technology for teaching and learning? Does learning by game playing necessarily contradict what education is all about? For those persuaded about the value of learning by game playing, how can the design and use of computer games be introduced into classroom learning that is carried out in the broader context of school-based practices?

In this keynote address, I explore the dimensions of embodiment, embeddedness, and experience in learning by game playing. I argue that these are productive and powerful elements that can help students establish a sense of *being*, develop agency and self-directedness in their learning experience, and, ultimately, construct a personal identity. I shall also examine the construct of *identity* in education and address its importance in the light of New Literacies. The foregoing ideas will be presented in the context of ongoing research into learning by game playing at the Learning Sciences Lab of the National Institute of Education, Singapore. The broader goal of this research endeavor is to investigate and design ways in which game playing might be introduced and used in classroom teaching and learning such that the innovation is pedagogically sound and sustainable.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, p. 814, 2006.

Learner Support: Convergence of Technology and Divergence of Thinking

Jim Greer

ARIES Laboratory Department of Computer Science University of Saskatchewan
greer@cs.usask.ca

In this era of the Wikipedia and the Podcast, it seems that our approaches to supporting both formal and informal lifelong learning are once again being revolutionized. Yet disappointingly e-learning remains dominated by rigid learning management systems that deliver content in much the same way as CBT systems did thirty years ago. Attempts to bring intelligent technologies into mainstream e-learning have been slowly and steadily infiltrating the thinking of the education establishment - but unfortunately the "slowly" has outweighed the "steadily".

At the thirty-year mark in my career as an educator, and after having personally engaged in formal teaching at all levels from pre-school to grad-school, I find that learner support and active learning are the two priorities of greatest concern to me - and perhaps not coincidentally these are two areas where I think intelligent systems have a significant role to play in facilitating learning.

Making learning an active process and engaging learners in that process depends on more than good pedagogy and interesting content. It involves making the content relevant to the context of individual learners and helping learners feel a sense of excitement and connectedness in a learning environment. Exceptional teachers stimulate learner interest. They show learners the relevance of the content to their lives and life goals and extend an invitation to engage that few learners can refuse. Good teaching is often about motivating - motivating the content by showing its relevance and motivating the learners to put in the necessary effort to learn. In the world of e-learning, how do we achieve these kinds of motivation? Those few learners who come to an e-learning setting with sufficient motivation to read independently every page of text in a "Black/CT" course are in that tiny minority of people who will learn despite the environment. We must do better.

Making a learning environment more learner-friendly, more inviting, and more supportive involves:

- putting effort into explicitly preparing learners to embark upon a learning session,
- adapting and customizing the learning content to be suitable for each specific individual learner,
- wrapping around the content a learning community of others who can collaborate, compete or otherwise impart enthusiasm, and
- providing learning support just in time and just in case.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 815–816, 2006.

Our research in the ARIES Laboratory has been focused on supporting cohorts of learners with the iHelp suite of learner support tools. We are working on ways to more easily create communities in which an instructional team consisting of many kinds of helpers including peer helpers and even artificial helpers can support learning needs of students. We are implementing systems that create and sustain a learning community. We are working on ways to use semantic web technologies to prepare packages of learning content for individual learners based on their needs and goals. We are developing approaches for closely monitoring activity during learning in order to try to anticipate learners' needs while respecting learner privacy.

Research by many people in the AIEd/ITS community (and elsewhere) on the topics of learner motivation and using collaboration and community to increase motivation has not yet made sufficient impact on the design of mainstream e-learning technologies or standards. Moving the good ideas from adaptive interactive education systems into the e-learning mainstream is a way to make a difference to millions of learners and perhaps this should be our community's next imperative.

Affect in One-to-One Tutoring

Helen Pain and Kaska Porayska-Pomsta

ICCS/HRCR, University of Edinburgh, 2 Buccleuch Place,
Edinburgh, Scotland, EH8 9LW, UK
{helen, kaska}@inf.ed.ac.uk

Abstract. It is well known that human tutors take into account both the student's knowledge and understanding of what is being taught, in addition to considering the emotional and motivational state of the student. However, there are many gaps in our understanding of the relationship between cognition and affect in tutoring. We have some insight into how human tutors infer student's cognitive and affective states, and current research has attempted to apply this knowledge to the inference of such states by computer tutors. There is ongoing research on how human tutors use their knowledge of student's states in their decisions and actions, and how we might use such research to inform the design of computer tutors.

In this talk we will consider what is currently known about how human tutors infer emotion and motivation in students, whether and how they act of these inferences, and how this relates to the student's cognitive state. We will describe methods that may be used to infer affect, and how these might be adapted for use in Human-Computer educational interactions, providing illustrations from our current work in which we explore how human tutors diagnose and manipulate affect in situational contexts, what affective states may be relevant to tutoring, and how we can model them formally. In particular we will raise questions about how fine grained the diagnosis of affective states needs to be, and whether in fact we need to diagnose (or know how to act on) mid-range affective states - or whether we might consider acting only in response to extreme emotional and motivational state values.

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, p. 817 , 2006.

Author Index