Determining the Significance of Item Order In Randomized Problem Sets

Zachary A. Pardos[1] and Neil T. Heffernan

{zpardos, nth}@wpi.edu

Worcester Polytechnic Institute

Abstract. Researchers who make tutoring systems would like to know which sequences of educational content lead to the most effective learning by their students. The majority of data collected in many ITS systems consist of answers to a group of questions of a given skill often presented in a random sequence. Following work that identifies which items produce the most learning we propose a Bayesian method using similar permutation analysis techniques to determine if item learning is context sensitive and if so which orderings of questions produce the most learning. We confine our analysis to random sequences with three questions. The method identifies question ordering rules such as, question A should go before B, which are statistically reliably beneficial to learning. Real tutor data from five random sequence problem sets were analyzed. Statistically reliable orderings of questions were found in two of the five real data problem sets. A simulation consisting of 140 experiments was run to validate the method's accuracy and test its reliability. The method succeeded in finding 43% of the underlying item order effects with a 6% false positive rate using a p value threshold of $<= 0.05$. Using this method, ITS researchers can gain valuable knowledge about their problem sets and feasibly let the ITS automatically identify item order effects and optimize student learning by restricting assigned sequences to those prescribed as most beneficial to learning.

1 Introduction

Corbett and Anderson style knowledge tracing [3] has been successfully used in many tutoring system to predict a student's knowledge of a knowledge component after seeing a set of questions that used that knowledge component. We present a method that allows us to detect if the learning value of an item might be dependent on the particular context the question appears in. We will model learning rates of items based on what item comes immediately after it. This will allow us to identify rules such as; item A should come before B, if such a rule exists. Question A could also be an un-acknowledged prerequisite for answering question B. After finding such relationships between questions, a reduced set of sequences can be recommended. The reliability of our results is tested with a simulation study in which simulated student responses are generated and the method is tasked with learning the underlying parameters of the simulation.

We presented a method [5] that used similar analysis techniques to this one, where an item effect model was used to determine which items produced the most learning. That method had the benefit of being able to inform Intelligent Tutoring System (ITS) researchers of which questions, and their associated tutoring, are or are not producing learning. While we think that method has much to offer, it raised the question of whether the learning value of an item might be dependent on the particular context it appears in. The method in this paper is focused on learning based on item sequence.
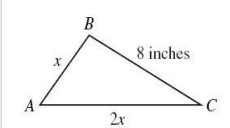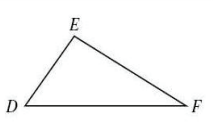

1.1 The Tutoring System and Dataset

Our dataset consisted of student responses from The ASSISTment System, a web based math tutoring system for 7th-12th grade students that provides preparation for the state standardized test by using released math items from previous tests as questions on the system. Figure 1 shows an example of a math item on the system and tutorial help that is given if the student answers the question wrong or asks for help. The tutorial helps the student learn the required knowledge by breaking the problem into sub questions called scaffolding or giving the student hints on how to solve the question.

The data we analyzed was from the 2006-2007 school year. Subject matter experts made problem sets called GLOPS (groups of learning opportunities). The idea behind the GLOPS was to make a problem set where the items in the problem set related to each other. They were not necessary strictly related to each other through a formal skill tagging convention but were selected based on their similarity of concept according to the expert. We chose the five three item GLOPS that existed in the system each with between 295 and 674 students who had completed the problem set. Items do not overlap across GLOP problem sets. Our analysis can scale to problem sets of six items but we

Figure 1. An ASSISTment item

wanted to start off with a smaller size set for simplicity in testing and explaining the analysis method. The items in the five problem sets were presented to students in a randomized order. Randomization was not done for the sake of this research in particular but rather because the assumption of the subject matter expert was that these items did not have an obvious progression requiring that only a particular sequence of the items be presented to students. In other words, context sensitivity was not assumed. We only analyzed responses to the original questions which meant that a distinction was not made between the learning occurring due to answering the original question and learning occurring due to the help content. The learning from answering the original question and scaffolding will be conflated as a single value for the item.

1.2 Knowledge Tracing

The Corbett and Anderson method of "knowledge tracing" [3] has been useful to many intelligent tutoring systems. In knowledge tracing there is a set of questions that are assumed to be answerable by the application of a particular knowledge component which could be a skill, fact, procedure or concept. Knowledge tracing attempts to infer the probability that a student knows a knowledge component based on a series of answers. Presumably, if a student had a response sequence of 0,0,1,0,0,0,1,1,1,1,1,1 where 0 is an incorrect first response to a question and 1 is a correct response, it is likely she guessed the third question but then learned the knowledge to get the last 6 questions correct. The Expectation Maximization algorithm is used in our research to learn parameters from data such as the probability of guess.
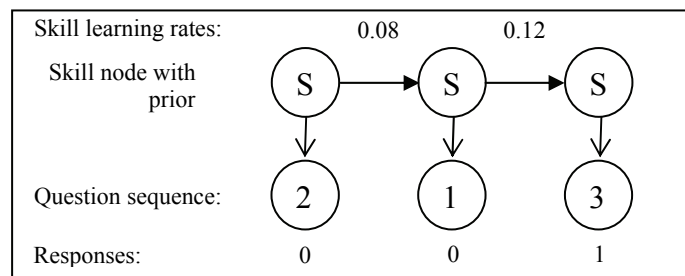

Figure 2. Bayesian network model for question sequence [2 1 3]

Figure 2 depicts a typical knowledge tracing three question static Bayesian network. The top three nodes represent a single skill and the inferred value of the node represents the probability the student knows the skill at each opportunity. The bottom three nodes represent three questions on the tutor. Student performance on a question is a function of their skill knowledge and the guess and slip of the question. Guess is the probability of answering correctly if the skill is not known. Slip is the probability of answering incorrectly if the skill is known. Learning rates are the probability that a skill will go from "not known" to "known" after encountering the question. The probability of the skill going from "known" to "not known" (forgetting) is fixed at zero. Knowledge tracing assumes that the learning on a piece of knowledge is independent of the question presented to students, that is that all questions should lead to the same amount of learning. The basic design of a question sequence in our model is similar to a dynamic Bayesian network or Hidden Markov Model used in knowledge tracing but with the important distinction that the probability of learning is able to differ between opportunities. This ability allows us to model different learning rates per question which is essential to our analysis. The other important distinction of our model is the ability to model permutations of sequences with parameter sharing, discussed in the next section.
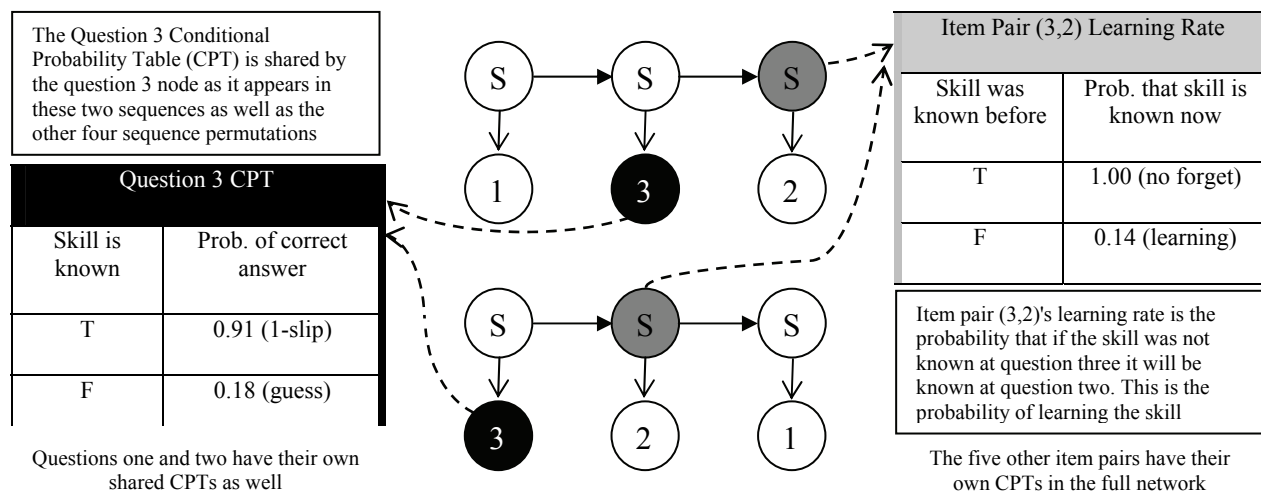
2 Analysis Methodology

In order to represent all the data in our randomized problem sets of three items we must model all six possible item sequence permutations. If six completely separate networks were created then the data would be split into six which would degrade the accuracy of parameter learning. This would also learn a separate guess and slip for each question in

each sequence despite the questions being the same in each sequence. In order to leverage the parameter learning power of all the data and define an individual question's guess and slip values we will use parameter sharing[2] to link the parameters across the different sequence networks. This means that question one as it appears in all six sequences will share the same guess and slip conditional probability table (CPT). The same will be true for the other two questions. This will give us three guess and slip parameters total and the values will be trained to reflect the questions' non sequence specific guess and slip values. In our item order effect model we also link the learning rates of item sequences.

2.1 The Item Order Effect Model

In the model we call the item order effect model we look at what effect item order has on learning. We set a learning rate for each pair of items and then test if one pair is reliably better for learning than another. For instance, should question A come before question B or vice versa? Since there are three items in our problem sets there will be six ordered pairs which are (3,2) (2,3) (3,1) (1,3) (2,1) and (1,2). This model allows us to train the learning rates of all six ordered pairs simultaneously along with guess and slip for the questions by using shared parameters to link all occurrences of pairs to the same learning rate conditional probability table. For example, the ordered pair (3,2) appears in two sequence permutations; sequence (3,2,1) and sequence (1,3,2) as shown in Figure 3.

The Question 3 Conditional Probability Table (CPT) is shared by the question 3 node as it appears in these two sequences as well as the other four sequence permutations

Question 3 CPT	
Skill is known	Prob. of correct answer
T	0.91 (1-slip)
F	0.18 (guess)

Questions one and two have their own shared CPTs as well

Item Pair (3,2) Learning Rate	
Skill was known before	Prob. that skill is known now
T	1.00 (no forget)
F	0.14 (learning)

Item pair (3,2)'s learning rate is the probability that if the skill was not known at question three it will be known at question two. This is the probability of learning the skill

The five other item pairs have their own CPTs in the full network

Figure 3. A two sequence portion of the Item Order Effect Model (six sequences exist in total)

2.2 Reliability Estimates Using the Binomial Test

In order to derive the reliability of the learning rates fit from data we employed the binomial test[3] by randomly splitting the response data into 10 by student. We fit the model parameters using data from each of the 10 bins separately and counted the number

---

[2] Parameter sharing was accomplished in the Bayesian network model using equivalence classes from Kevin Murphy's Bayes Net Toolbox, available at: http://bnt.sourceforge.net/

[3] The binomial test was run with the MATLAB command: binopdf(successes, trials, 1/outcomes)

of bins in which the learning rate of one item pair was greater than its reverse, (3,2) > (2,3) for instance. We call a comparison of learning rates such as (3,2) > (2,3) a rule. The null hypothesis is that each rule is equally likely to occur. A rule is considered statistically reliable if the probability that the result came from the null hypothesis is <= 0.05. For example, if we are testing if ordered pair (3,2) has a higher learning rate than (2,3) then there are two possible outcomes and the null hypothesis is that each outcome has a 50% chance of occurring. Thus, the binomial test will tell us that if the rule holds true eight or more times out of ten then it is <= 0.05 probable that the result came from the null hypothesis. This is the same idea as flipping a coin 10 times to determine the probability it is fair. The less likely the null hypothesis, the more confidence we can have in the result. If the learning rate of (3,2) is greater than (2,3) with p <= 0.05 then we can say it is statistically reliable that question three and its tutoring followed by question two better help students learn the skill than question two and its tutoring followed by question three. Based on this conclusion it would be recommended to give sequences where question three comes before two. The successful detection of a single rule will eliminate half of the sequences since three comes before two in half of the sequence permutations. Strictly speaking the model is only reporting the learning rate when two comes directly after three however in eliminating half the sequences we make the pedagogical assumption that question three and its tutoring will help answer question two even if it comes one item prior such as in the sequence (3, 1, 2). Without this assumption only the two sequences with (2,3) can be eliminated and not sequence (2,1,3).

2.3   Item Order Effect Model Results

We ran the analysis method on our problem sets and found reliable rules in two out of the five problem sets. The results below show the item pair learning rate parameters for the two problem sets in which reliable rules were found. The 10 bin split was used to evaluate the reliability of the rules while all student data for the respective problem sets were used to train the parameters shown below.

Table 1. Item order effect model results

Problem Set	Users	Learning probabilities of Item Pairs						Reliable Rules
		(3,2)	*(2,1)*	*(3,1)*	*(1,2)*	*(2,3)*	*(1,3)*	
24	403	0.1620	0.0948	0.0793	0.0850	0.0754	0.0896	(3,2) > (2,3)
36	419	0.1507	0.1679	0.0685	0.1179	0.1274	0.1371	(1,3) > (3,1)

As shown in Table 1, there was one reliable rule found in each of the problem sets. In problem set 24 we found that item pair (3,2) showed a higher learning rate than (2,3) in eight out of the 10 splits giving a binomial p of 0.0439. Item pair (1,3) showed a higher learning rate than (3,1) also in eight out of the 10 splits in problem set 36. Other statistically reliable relationships can be tested on the results of the method. For instance, in problem set 36 we found that (2,1) > (3,1) in 10 out of the 10 bins. This could mean that sequence (3,1,2) should not be given to students because question three comes before question one and question two does not. Removing sequence (3,1,2) is also supported by rule (1,3) > (3,1). In addition to the learning rate parameters, the model simultaneously trains a guess and slip value for each question. Those values are shown below in Table 2.

Table 2. Trained question guess and slip values

Question #	Problem Set 24		Problem Set 36	
	Guess	*Slip*	*Guess*	*Slip*
1	0.17	0.18	0.33	0.13
2	0.31	0.08	0.31	0.10
3	0.23	0.17	0.20	0.08

3 Simulation

In order to determine the validity of the item order effect method we chose to run a simulation study exploring the boundaries of the method's accuracy and reliability. The goal of the simulation was to generate student responses under various conditions that may be seen in the real world and test if the method would accurately infer the underlying parameter values from the simulated student data. This simulation model assumes that learning rates have distinct values and that item order effects of some magnitude always exist and should be detectable given enough data.

3.1 Model design

The model used to generate student responses is a six node static Bayesian network as depicted in Figure 2 from section 1.2. While the probability of knowing the skill will monotonically increase after each opportunity, the generated responses (0s and 1s) will not necessarily do the same since those values are generated probabilistically based on skill knowledge and guess and slip. Simulated student responses were generated one student at a time by sampling from the six node network.

3.2 Student parameters

Only two parameters were used to define a simulated student, a prior and question sequence. The prior represents the probability the student knew the skill relating to the questions before encountering the questions. The prior for a given student was randomly generated from a distribution that was fit to a previous year's ASSISTment data [6]. The mean prior for that year across all skills was 0.31 and the standard deviation was 0.20. In order to draw probabilistic parameter values that fit within 0 and 1, an equivalent beta distribution was used. The beta distribution fit an α of 1.05 and β of 2.43. The question sequence for a given student was generated from a uniform distribution of sequence permutations.

3.3 Tutor Parameters

The 12 parameters of the tutor simulation network consist of six learning rate parameters, three guess parameters and three slip parameters. The number of users simulated was: 200, 500, 1000, 2000, 4000, 10000, and 20000. The simulation was run 20 times for each of the seven simulated user sizes totaling 140 generated data sets, referred to later as experiments. In order to faithfully simulate the conditions of a real tutor, values for the 12 parameters were randomly generated using the means and standard deviations across 106

skills from a previous analysis [6] of ASSISTment data. Table 3 shows the distributions that the parameter values were randomly drawn from and then assigned to questions and learning rates at the start of each run.

Table 3. The distributions used to generate parameter values in the simulation

Parameter type	Mean	Std	Beta dist α	Beta dist β
Learning rate	0.086	0.063	0.0652	0.6738
Guess	0.144	0.383	0.0170	0.5909
Slip	0.090	0.031	0.0170	0.6499

Running the simulation and generating new parameter values 20 times gives us a good sampling of the underlying distribution for each of the seven user sizes. This method of generating parameters will end up accounting for more variance than the real world since standard deviations were calculated for values across problem sets as opposed to within. Also, guess and slip have a correlation in the real world but will be allowed to independently vary in the simulation which means sometimes getting a high slip but low guess, which is rarely observed in actual ASSISTment data. It also means the potential for generating very improbable combinations of item pair learning rates.

3.4   Simulation Procedure

The simulation consisted of three steps: instantiation of the Bayesian network, setting CPTs to values of the simulation parameters and student parameters and finally sampling the Bayesian network to generate the students' responses.

To generate student responses the six node network was first instantiated in MATLAB using routines from the Bayes Net Toolbox package. Student priors and question sequences were randomly generated for each simulation run and the 12 parameters described in section 3.3 were assigned to the three questions and item pair learning rates. The question CPTs and learning rates were positioned with regard to the student's particular question sequence. The Bayesian network was then sampled a single time to generate the student's responses to each of the three questions; a zero indicating an incorrect answer and a one indicating a correct answer. These three responses in addition to the student's question sequence were written to a file. A total of 140 data files were created at the conclusion of the simulation runs, all of which were to be analyzed by the item order effect detection method. The seeded simulation parameters were stored in a log file for each experiment to later be checked against the method's findings. An example of an experiment's output file for 500 users is shown in Table 4 below.

Table 4. Example output from data file with N=500

Simulated User	Sequence identifier	1st Q	2nd Q	3rd Q
1	5	0	1	1
⋮	⋮	⋮	⋮	⋮
500	3	1	0	1

Each data file from the simulation was split into 10 equal parts and each run separately through the analysis method just as was done in analysis of real tutor data. This analysis step would give a result such as the example in Table 5 below.

Table 5. Example output from item order effect analysis

	(3,2)	(2,1)	(3,1)	(1,2)	(2,3)	(1,3)
Split 1	0.0732	0.0267	0.0837	0.0701	0.0379	0.642
⋮	⋮	⋮	⋮	⋮	⋮	⋮
Split 10	0.0849	0.0512	0.0550	0.0710	0.0768	0.0824

In order to produce a p value and determine statistical reliability to the $p < 0.05$ level the binomial test is used. The method counts how many times (3,2) was greater than (2,3) for instance. If the count is greater than eight then the method considers this an identified rule. Even though there are six item pairs there is a maximum of three rules since if (3,2) > (2,1) is a reliable rule then (3,2) < (2,3) is not. In some cases finding two rules is enough to identify a single sequence as being best. Three rules always guarantee the identification of a single sequence. The method logs the number of rules found and how many users (total) were involved in the experiment. The method now looks "under the hood" at the parameters set by the simulation for the item pair learning rates and determines how many of the found rules were false. For instance, if the underlying simulated learning rate for (3,2) was 0.08 and the simulated learning rate for (2,3) was 0.15 then the rule (3,2) > (2,3) would be a false positive (0.08 < 0.15). This is done for all 140 data files. The total number of rules is three per experiment thus there are 420 rules to be found in the 140 data files.

3.5 Simulation Results

The average percent of found rules per simulated user size is plotted in Figure 2 below. The percentage of false positives is also plotted in the same figure and represents the error.
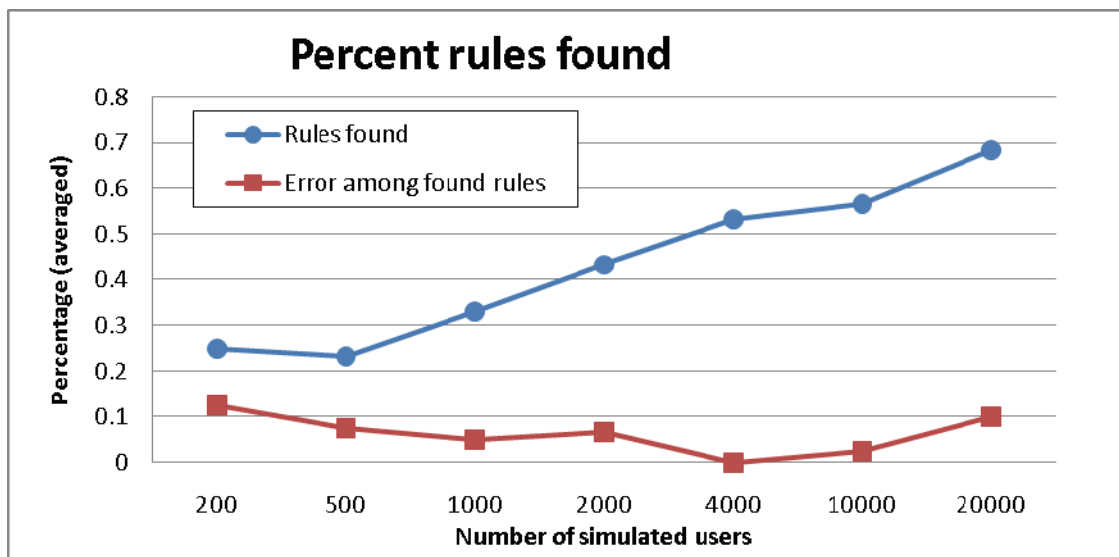

Figure 4. Results of simulation study

Figure 4 shows that more users allows for more rules about item order to be detected. It also shows that the false positive rate remains fairly constant, averaging around the 6% mark. From 200 users to 1,000 users the average percentage rules found was around 30% which would correspond to about 1 rule per problem set (0.30 * 3). This percentage rises steadily in a linear fashion from 500 users up to the max number of users tested of 20,000 where it achieves a 69% discovery rate which corresponds to about two rules per problem set on average. The error starts at 13% with 200 users and then remains below 10% for the rest of the user sizes. The overall average percent of rules found across users sizes is 43.3%. The overall average false positive rate is 6.3% which is in line with the binomial p value threshold of 0.05 that was used and validates the accuracy of the method's results and dependability of the reported binomial p value.

Limitations and Future Work

One of the limitations of this permutation analysis method is that it does not scale gracefully. The number of network nodes that need to be constructed is exponential in the number of items. For a three item model there are six nodes per sequence and six sequences. For a seven item model there are fourteen nodes per sequence and 5,040 sequences (70,560 nodes). One potential optimization would be to only construct sequences for which there is data, which will be at most the number of students.

The split 10 procedure has the effect of decreasing the amount of data the method has to operate on for each run. A more efficient sampling method may be beneficial, however, our trials using resampling with replacement for the simulation instead of splitting resulted in a high average false positive rate (>15%). A more sensitive test that takes into account the size of the difference between learned parameter values would improve reliability estimates. The binomial accuracy may also be improved by using a Bonferroni correction as suggested by a reviewer. This correction is used when multiple hypotheses are tested on a set of data (i.e. the reliability of item ordering rules). The correction suggests using a lower p value cut-off.

There is a good deal of work in the area of trying to build better models of what students are learning. One approach [1] uses a matrix of skill to item mappings which can be optimized [2] for best fit and used to help learn optimal practice schedules [7] while another approach attempts to find item to item knowledge relationships [4] such as prerequisite item structures using item tree analysis. We think that the item order effect method introduced here and its accompanying paper [5] have parallels with these works and could be used as a part of a general procedure to try to learn better fitting models.

Contribution

This method has been shown by simulation study to provide reliable results suggesting item orderings that are most advantageous to learning. Many educational technology companies [8] (i.e. Carnegie Learning Inc. or ETS) have hundreds of questions that are tagged with knowledge components. We think that this method, and ones built off of it, will facilitate better tutoring systems. In [5] we used a variant of this method to figure out what items are causing the most learning. In this paper, we presented a method that

allows scientists to see if the items in a randomly ordered problem set produce the same learning regardless of context or if there is an implicit ordering of questions that is best for learning. Those best orderings might have a variety of reasons for existing. Applying this method to investigate those reasons could inform content authors and scientists on best practices in much the same way as randomized controlled experiments do but by utilizing the far more economical means of investigation which is data mining.

Acknowledgements

References

[1] Barnes, T. (2005). Q-matrix Method: Mining Student Response Data for Knowledge. *Proceedings of the AAAI-05 Workshop on Educational Data Mining*, Pittsburgh, 2005. (AAAI Technical Report #WS-05-02).

[2] Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning factors analysis - A general method for cognitive model evaluation and improvement. *In Proc. the 8th International Conference on Intelligent Tutoring Systems*. pp. 164-175

[3] Corbett, A. T., Anderson, J. R. & O'Brien, A. T. (1995) Student modeling in the ACT programming tutor. In P. Nichols, S. Chipman, & R. Brennan (Eds.), *Cognitively diagnostic assessment* (pp. 19-41). Hillsdale, NJ: Erlbaum.

[4] Desmarais, M. C., Meshkinfam, P. & Gagnon, M. (2006). Learned student models with item to item knowledge structures. *User Modeling and User-adapted Interaction*, 16(5), 403–434.

[5] Pardos, Z. A., Heffernan, N. T. In Press (2009) Detecting the Learning Value of Items In a Randomized Problem Set. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education. Brighton, UK. IOS Press.*

[6] Pardos, Z. A., Heffernan, N. T., Ruiz, C. & Beck, J. (2008) Effective Skill Assessment Using Expectation Maximization in a Multi Network Temporal Bayesian Network. *The Young Researchers Track at the 20th International Conference on Intelligent Tutoring Systems*. Montreal, Canada. pp. 31-40

[7] Pavlik, P. I., Jr., Presson, N., & Koedinger, K. R. (2007). Optimizing knowledge component learning using a dynamic structural model of practice. *In proceedings of the 8th International Conference on Cognitive Modeling*. Ann Arbor, Michigan, USA.

[8] Stevens, R. H., & Thadani, V. (2006) A Bayesian Network Approach for Modeling the Influence of Contextual Variables on Scientific Problem Solving. In M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): *ITS 2006*, LNCS 4053, Springer-Verlag. pp.71-84.