Student Modeling in Intelligent Tutoring Systems

Yue Gong
Worcester Polytechnic Institute
November 2014

Committee Members:

Dr. Joseph E. Beck, Associate Professor, Worcester Polytechnic Institute. Advisor.
Dr. Neil T. Heffernan, Professor, Worcester Polytechnic Institute. Co-advisor.
Dr. Carolina Ruiz, Associate Professor, Worcester Polytechnic Institute.
Dr. Cristina Conati, Associate Professor, University of British Columbia.

Contents

Abstract

After decades of development, Intelligent Tutoring Systems (ITSs) have become a common learning environment for learners of various domains and academic levels. ITSs are computer systems designed to provide instruction and immediate feedback without requiring the intervention of human instructors. All ITSs share the same goal: to provide tutorial services that support learning. Since learning is a very complex process, it is not surprising that a range of technologies and methodologies from different fields is employed.

Student modeling is a pivotal technique used in ITSs. The model observes student behaviors in the tutor and creates a quantitative representation of student properties of interest necessary to customize instruction, to respond effectively, to engage students' interest and to promote learning. In this dissertation work, I focus on the following aspects of student modeling.

Part I: Student Knowledge: Parameter Interpretation. Student modeling is widely used to obtain scientific insights about how people learn. Student models typically produce semantically meaningful parameter estimates, such as how quickly students learn a skill on average. Therefore, parameter estimates being interpretable and plausible is fundamental. My work includes automatically generating data-suggested Dirichlet priors for the Bayesian Knowledge Tracing model, in order to obtain more plausible parameter estimates. I also proposed, implemented, and evaluated an approach to generate multiple Dirichlet priors to improve parameter plausibility, accommodating the assumption that there are subsets of skills which students learn similarly.

Part II: Student Performance: Student Performance Prediction. Accurately predicting student performance is one of the common evaluations for student modeling. The task, however, is very challenging, particularly in predicting a student's response on an individual problem in the tutor. I analyzed the components of two common student models to determine which aspects provide predictive power in classifying student performance. I found that modeling the student's overall knowledge led to improved predictive accuracy. I also presented an approach which, rather than assuming students are drawn from a single distribution, modeled multiple distributions of student performances to improve the model's accuracy.

Part III: Wheel-spinning: Student Future Failure in Mastery Learning. One drawback of the mastery learning framework is its possibility to leave a student stuck attempting to learn a skill he is unable to master. We refer to this phenomenon of students being given practice with no improvement as wheel-spinning. I analyzed student wheel-spinning across different tutoring systems and estimated the scope of the problem. To investigate the negative consequences of wheel-spinning, I investigated the relationships between wheel-spinning and two other constructs of interest about students: efficiency of learning and "gaming the system". In addition, I designed a generic model of wheel-spinning, which uses features easily obtained by most ITSs. The model can be well generalized to unknown students with high accuracy classifying mastery and wheel-spinning problems. When used as a detector, the model can detect wheel-spinning in its early stage with satisfactory precision and recall.

Chapter 1. Introduction

1. Intelligent Tutoring Systems

After its birth in the late 1970s, Intelligent Tutoring Systems (ITS) are growing more sophisticated with increasingly large influence in education. ITS are computer systems designed to provide direct customized instruction and immediate feedback to students, but without requiring the intervention of human beings. The goal of the systems is to provide tutorial services that support learning (Nkambou, Bourdeau et al. 2010).

The original intention of designing and developing such systems was due to the vision that Artificial Intelligence could produce a promising solution to the limitations educational professionals were facing: how to effectively teach and help students learn in a large scale. The main concern was that the effectiveness of teaching improves with small student to teacher ratios. In 1984, Bloom conducted experiments comparing student learning under conditions, a 30 students per teacher class vs. one-to-one tutoring, and found that individual tutoring is much more effective as group teaching (Bloom 1984). Therefore, on one hand, there were increasingly pragmatic needs, such as how to achieve high student learning without requiring an impractical number of teachers, and how to support student learning outside school without constraints of time and location. On the other hand, AI researchers were keenly seeking a meaningful venue for their enthusiasms to spread the power of AI in many traditional fields at the time when AI was blossoming. Computer scientists, cognitive scientists, educational professionals viewed the newborn Intelligent Tutoring Systems (ITS) as a means to fulfill their various goals. An ITS uses AI techniques and supports quality learning for individuals with no or little human assistance. As a result, ITS research is a multi-disciplinary effort and requires seamless collaborations of a variety of disciplines, such as education, cognitive science, learning science, and computer science.

A computer system, to be called an ITS, in particular needs to be able to provide immediate feedback and individualized assistance. A study has shown that, from a 'most-wanted' list of specific features, students primarily desire an ITS that provides individualized teaching and learning (Harrigan, Kravcík et al. 2009). Many research studies have also confirmed that because of immediate feedback, ITSs resulted in substantial successes in improving student learning in different domains, such as mathematics (Razzaq, Feng et al. 2007), physics (VanLehn, Lynch et al. 2005), and reading (Mostow and Aist 2001).

This ability of providing immediate feedback and individualized assistance is achieved by different parts of the system collaborating together. There are four major components: domain modeling, tutor modeling, student modeling and the user interface. Domain modeling is a technique to encode domain knowledge, such as concepts, rules and procedures, facilitating their use in computer systems. This part of the system is often called expert knowledge, and systems focusing more on domain modeling are called expert systems. An ITS uses this part as a knowledge base to evaluate student performance with the expert knowledge in the context. Student modeling is a technique used to understand students, including their knowledge level, and their behaviors and their emotions; it provides a computer-interpretable representations to the system. The tutoring model consumes the knowledge from the domain model and the student model. It directs the system to provide human-like tutoring with applications of several different pedagogical strategies. For example, given the estimation of student misconception, the output of the student model, comparing it with the domain knowledge, the output of the domain model, the tutor model may need to decide whether tutorial actions are necessary to conduct. If an intervention decision has been made, the tutor model also needs to make a wise decision as to when and how to intervene. Finally, all these three models collaborate as services at backend and the user interface works as a presentational tier to blend the services together to interact and communicate with the user.

2. Student Modeling

Student modeling, despite existing as one of four major components in the classic architecture of ITSs, already forms a large base of research of its own. I think that there are several reasons.

First, the student model is the core component in an ITS. From an architectural point of view, student modeling is essential. Traditionally, the four-component architecture was adopted in engineering an ITS. However, with decades of developments of ITSs, there are other architectures that have been adopted, meeting some specific design objectives and system tradeoffs. In 1990, Nwana presented his work, in which he reviewed other architectures and pointed out that the choice of the architectural design of an ITS actually reflect the tutoring philosophies varying in their emphases in different components of the learning process: domain, student or tutor (Nwana 1990). As a consequence, some components were enhanced taking more responsibilities, while some components were even eliminated from the classic architecture. In many cases, two or more components were mixed and functioned together. Student modeling has always found its solid ground in many types of architectures. For example, model tracing is considered a major achievement in domain modeling by many researchers, as the model encodes learning as a series of rule-based cognitive steps and represent required domain knowledge accordingly. Model tracing can also serve as a student model, as it assumes the student actions can be identified and explicitly coded through topics, steps, or rules (Anderson and Reiser 1985). The other aspect is that, student modeling is the fundamental part in an ITS which is actually in charge of decision making. The tutor model relies on the knowledge provided by student modeling so as to adjust its intelligence to perform immediate feedback and individualized tutoring.

Second, student modeling solves a wider range of research questions, and more importantly, many are not limited to solely involve ITSs. This perhaps is the most important reason why after decades of development of ITSs, student modeling gradually attracts attentions and becomes a new emerging research topic. As early as the 1980s, Self (1988) identified six major roles for the student model:

1) Corrective: to help eradicate bugs in the student's knowledge;
2) Elaborative: to help correct 'incomplete' student knowledge;
3) Strategic: to help initiate significant changes in the tutorial strategy other than the tactical decisions of 1 and 2 above;
4) Diagnostic: to help diagnose bugs in the student's knowledge;
5) Predictive: to help determine the student's likely response to tutorial actions;
6) Evaluative: to help assess the student or the ITS.

Nowadays, we commonly see student modeling's roles in 3), 5) and 6). Moreover, student modeling is a good means that can extend its abilities out of the tutor. In 5) and 6), there are many applications of building student models using the student in-tutor data to predict his out-of-tutor performance, as well as evaluate, analyze and understand student learning in from a quantitative point of view.

Third, student modeling is an interesting, yet very challenging, problem, where researchers with various backgrounds can find their own spots to fit in. Ideally, the student model should contain as much knowledge as possible about the student's cognitive and affective states and their evolution as the learning process advances (Nkambou, Bourdeau et al. 2010). A student model must be dynamic, as it needs to provide the current knowledge about the student while he is using the ITS. Therefore, compared to domain modeling, which is relatively static and could be designed and engineered ahead of time, more challenges exist in building a good student model. To evaluate whether a student model is good or not, based on the purpose of using the model, two aspects are often considered: predictive accuracy and parameter plausibility. Predictive accuracy is used when a student model is mainly used to classify, predict or detect some student behaviors, where ground truth is easily observed. By comparing the ground truth with the predicted value, some metrics of accuracy are calculated to evaluate the goodness of the model. There are a great number of prior works that have been done to improve student model's predictive accuracy. A recent competition in the Knowledge Discovery and Data Mining Cup 2010, which focused on predictive accuracy of student models, also reflects the great challenge of building an accurate student model. On the other hand, parameter plausibility is used when a student model is mainly used to assess students. For example, when a tutoring system is used in a mastery learning setting (where learners keep solving problems until they have mastered the skill), the student knowledge is used to determine his mastery. Student knowledge is typically obtained by using a student model to get its estimation. The estimation is represented by one of the model parameters. This specific requirement needs

the model parameter reflects the true level of the student knowledge. Whether the parameter is plausible is very important for the system to make its tutoring decisions. Therefore, parts of efforts on improving student models also focus on building a more plausible model.

3. ASSISTments: an Intelligent Tutoring System

Most of this dissertation work is conducted based on ASSISTments, a web-based math tutoring system. It was first created in 2004 as a joint research conducted by Worcester Polytechnic Institute and Carnegie Mellon University. Its name, ASSISTments, came from the idea of combining *assist*ing the student with automated assess*ment* of the student's proficiency at a fine-grained level (Feng, Heffernan, & Koedinger, 2009). Thousands of middle- and high-school students are using ASSISTments for their daily learning, homework, and preparing the MCAS (Massachusetts Comprehensive Assessment System) tests. In 2010-2011, there were over 20,000 students and 500 teachers using the system as part of their regular math classes in and out Massachusetts.

The ASSISTments system is a typical step-based tutoring system (VanLehn 2006). The student practices a problem in a linear manner and once the student begins a problem, the tutor is responsible to give feedback and/or help. Inside the system, there is a key concept called an "Assistment," which bundles together a question for the student to solve and the question's associated tutorial actions that can be used to help the student. Figure 1 shows an Assistment, which consists of an original question, also called a main question, and a tutoring session. In this example, the main question shows at the top asking the length of side DF in triangle DEF. The tutoring session is for assisting student learning when the student fails to answer the original question correctly.

Depending on the tutorial strategy associated with the Assistment, assistance is provided by different forms, including:

1. A sequence of scaffolding questions

When a student gave a wrong response on the original question, ASSISTments presents scaffolding questions so as to break the original question down into steps. In Figure 1, two of the total three scaffolding questions were shown as the second and the third questions. The student must answer each scaffolding question correctly in order to proceed to the next scaffolding question.

2. A sequence of hints

Hints are messages that provide insights and suggestions for solving a specific question. Typically, there are 2 to 5 hints associated with each scaffold and main question. In Figure 1, although the first scaffolding question offers 3 hints for helping students solve the question, the student succeeded without the system's help; whereas in the second scaffolding question, the student requested hints. After viewing a hint, the student is allowed to make another one or more attempts to answer the question. If he or she still has difficulty in solving the question, he or she could ask for more hints until finally a bottom-out hint is presented which provides the student the answer. Bottom-out hints are necessary to avoid the problem of a student becoming stuck and unsure how to proceed within the tutor.

As a computer-based tutoring system, ASSISTments collects more information than traditional practice methods such as paper and pencil. Beyond basic information such as the correctness of student response and the problem presented, the system logs every student action such as requesting help or submitting a response, so the system is able to know more about the students than in traditional homework. For the analyses presented in this paper, only the student's first attempt at the original question is used to score correctness of student response. Thus if the student generates an incorrect response in his or her first attempt of a question, that question would be marked wrong even though the student will probably eventually solve it. Usually, students perform multiple actions when solving a question. The system logs all student actions which include: to give a response, to request a hint and to answer a scaffolding question. Equally important is that the system also time-stamped those actions, so that not only what the student did is known, but when he or she did it and how long it took is recorded.

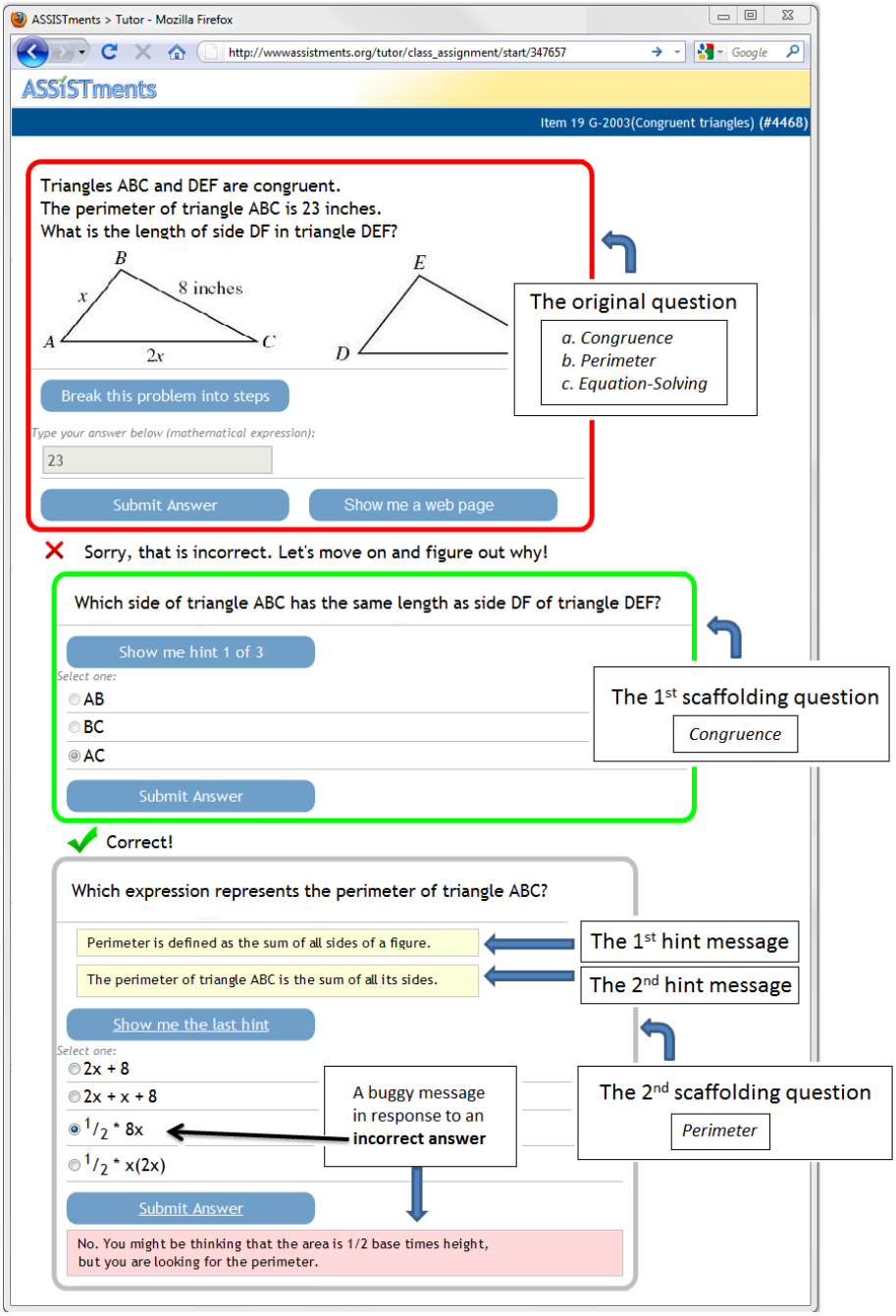


Figure 1. An Assistment showing a student being tutored. The third scaffolding question is about equation solving and the fourth one on substitution is not shown.

4. Issues Addressed in the Dissertation Work

This dissertation work focuses on constructing and improving student models. The work consists of the following three aspects.

Part 1. Improve a student model to produce more believable parameter estimates in order to better analyze and understand student learning. One objective of educational research is to understand students, especially their learning. Student modeling techniques can be used to fulfill this task. In addition to predicting performance, researchers are often interested in using the parameters learned from student

models to answer research questions. Therefore, being able to produce believable parameters is important. I proposed an approach of using Dirichlet priors to improve a student model, so that more plausible parameter estimates can be learned (Rai, Gong et al. 2009; Gong, Beck et al. 2010).

Part 2. Improve student modeling techniques to generate accurate predictions of student behavior. Intelligent tutoring systems use student models to understand student proficiencies and to represent student progress in his learning process. Also, the system uses student models to predict student behaviors, such as a student response to a question. High predictive accuracy of a student model is sought, as higher accuracy of the model means higher accuracy of the system in terms of understanding student learning process. Towards this problem, I have analyzed a student model's components to determine which parts provides power in predicting student performance (Gong and Beck 2011). I have also proposed approaches to improve a student model to produce more accurate predictions of student performance. I proposed to look beyond the transfer model and make use of students' overall proficiency in the domain to better predict student performance (Gong and Beck 2011). I also have presented an approach to model multiple distributions of student performances and used multiple classification models to predict student performance. The models outperformed the student model which had worked the best on our data (Gong, Beck et al. 2012).

Part 3. Model student *wheel-spinning*, which is students stuck in the mastery learning process, for understanding the impact of the phenomenon, discovering interesting patterns in student behaviors and accurately detecting wheel-spinning at an early stage. Some intelligent tutoring systems use the mastery learning framework, in which the system provides students practice opportunities with the individualized amount to ensure efficient mastery. Wheel-spinning depicts the phenomenon that the student fails to master a skill even when granted a large number of practice opportunities. I have analyzed the wheel-spinning problem across two different tutoring systems and demonstrated that it is a broad problem that hurts a significant number of students with considerably serious negative impacts in the two different student populations (Beck and Gong 2013). I also explored the relationship between the wheel-spinning problem and other non-productive student behaviors (Gong and Beck (in preparation)). I constructed a general model for detecting wheel-spinning, which is free from ASSISTments specific features. I evaluated the model using the data from ASSISTments and the Cognitive Tutor Algebra and showed the model showed superior predictive accuracy for both tutor system data (Gong and Beck (in preparation)).

Chapter 2. Student Modeling Background

Student modeling is an important technique used in intelligent tutoring systems (ITSs). Student models observe the student's behaviors in the system, and create quantitative representations of his properties of interest, which inform other modules of the system. The key use of a student model in an ITS is to support making instructional decisions. A good student model that matches student behaviors to student properties of interest can often provide insightful information to both the system and the researchers.

Two essential factors are involved in the definition of student modeling: student behaviors and properties of interest. Student behaviors can be viewed as the input of a student model, which include a variety of observations, such as student answers and student actions. Properties of interest represent what about the student is being modeled. Depending on the requirements, the range of things being modeled could be fairly broad: student knowledge, student performance, student emotion and other constructs of interest. Student models create quantitative representations, which are consumable to other modules within a computer system, and most of which are also interpretable to humans outside a computer system.

There are two categories of methods for building student models: cognitive science methods and machine learning methods. Different techniques work better or worse for different academic domains. Moreover, two categories of techniques are sometimes used conjunctively to achieve a superior result. My dissertation work lies in the category of machine learning methods; therefore the majority of this chapter is used to discuss the related work in the machine learning methods following a brief description of the related work in the cognitive science methods.

1. Cognitive Science based Student Models

Cognitive science-based student models were introduced, developed and thrived in the 1980s and 1990s. The big assumption of adapting cognitive science in student modeling is that how humans learn can be modeled as a computational process (Nkambou, Bourdeau et al. 2010). Therefore, traditional ways to construct student models require a significant amount of time and human labor. The common techniques include structured interviews and think-aloud protocols. Despite high construction costs, these student models are inevitably subjective. Previous studies have shown that human engineering of these models often ignores distinctions in content and learning that have important instructional implications (Koedinger and Nathan 2004; Koedinger and Mclaughlin 2010) .

Two common techniques are model-tracing (MT) and the constraint-based modeling (CBM). The development of model tracing is grounded in cognitive psychology based on the ACT-R (Adaptive Control of Thought – Rational) cognitive theory. The belief is that human learning processes can be modeled by some form of structures describing how a task is procedurally accomplished. The technique is closely related to domain modeling and expert systems. In the model tracing framework, student actions are atomized as encoded topics, steps and rules forming to the path through the problem space (Anderson and Reiser 1985). The student model uses these rules or steps to represent student knowledge. By tracing student execution of these rules, the model reasons about student knowledge, infers whether they followed the path and diagnoses the reason why an error or divergence occurs.

The assumption made by the constraint-based modeling (CBM) contrasts that of model tracing. CBM in particular disagrees with MT in the computer system's ability to model the human learning process. MT believes in the computer techniques' full ability to accurately modeling learning step by step or rule by rule. CBM thinks that only errors can be recognized and captured by a computer system. Therefore, only constraints matter, as it is believed that the actual thinking could follow various paths yet lead to the same destination. The cognitive theory underlying CBM is that knowledge, and learning accordingly, is refined into two categories, procedural knowledge and declarative knowledge (Self 1988; Ohlsson 1994). MT models learning in a rigid manner, where only the explicit procedural learning is considered. However, it is possible that the student has already obtained enough declarative knowledge which allows him to apply it in problem-solving. The actual problem-solving, helping him acquire procedural knowledge, does not necessarily follow the procedures in which MT assumes how learning

should occur. One aspect of CBM that is similar to MT is that, it is also considered domain modeling. Instead of a full stack of domain knowledge, only basic domain knowledge, such as rules, and pedagogical states are represented by constraints which should not be violated during problem-solving. When constrains are violated by the student, an error is triggered and the CBM uses pattern matching to search the domain model to respond to the student's incorrect actions.

2. Machine Learning based Student Models

With the development of new generations of ITSs, one of the most remarkable characteristics differing from the old generation is a large number of interactions between the system and students, such as the student responding to a question, the student requesting assistance, or the system providing feedback. Therefore, large amounts of data are generated during interactions across thousands of students. This new development calls for more researchers with computer science background joining the field. In particular, machine learning techniques provide new means for performing student modeling. There are two extraordinary advantages of the machine learning-based student models over the traditional cognitive science based student models.

First, the construction of the new type of models does not base itself on the assumption that human learning can be modeled regardless of extent. Machine learning based methods are agnostic with regard to this assumption (Nkambou, Bourdeau et al. 2010). They are simply using any reasonable machine learning techniques to understand student properties of interest.

Second, the range of objects being modeled is enriched by the use of the machine learning based student models. On the contrary, cognitive theories underlying the models are essential. It requires a tremendous amount of work in cognitive science, philosophy and learning science on an object to provide a sound foundation to the construction of a student model of that object. This in some sense puts obstacles to the development of student modeling. The student models based on cognitive science therefore focus on student knowledge. For the sake of machine learning techniques, it becomes possible that a variety of constructs could be well modeled, such as student performance, student affect, student robust learning, etc.

With regard to the two commonly used roles for a student model: (1) Predictive: to help determine the student's likely response to tutorial actions; (2) Evaluative: to help assess the student or the ITS (Self 1988), Machine learning-based student models can fulfill the roles naturally.

For the role of predictive, a classic task that machine learning techniques are designed to target is predictive tasks: to predict the value of a particular attribute based on the values of other attributes. The attribute to be predicted is commonly known as the target or dependent variable, while the attributes used for making the prediction are known as the explanatory or independent variables (Tan, Steinbach et al. 2005). Student modeling provides an ideal platform, as in ITSs, many variables, typically behaviors of students, are of interest for prediction, such as the most popular, the correctness of a student's response on a question.

For the role of evaluative, as a sub-discipline of data mining, machine learning based student models can be used to discovering useful information in large data repositories. People use data mining to convert raw data into novel and useful information, and then, in the "closing the loop" phase, the results from data mining are integrated into decision making (Tan, Steinbach et al. 2005). In ITSs, machine-learning based student models could typically be used to estimate some student constructs or the impact of certain student behaviors, pedagogical strategies, and tutorial interventions.

2.1 Student Knowledge

Student knowledge is the most intriguing construct to educational workers and researchers. A variety of mechanisms were invented for the purpose of estimating student knowledge. A traditional method with the longest history is to test students through some forms of "ask-and-answer" based manner, such as through quizzes, exams and questionnaire. The idea is from that the best estimation of student knowledge

is obtainable by observing student performance. The key fact is that student performance is observable, whereas student knowledge is latent.

With regard to this characteristic of student knowledge, machine learning based student modeling opens a new door for student knowledge estimation. Specifically, this unique characteristic leads student knowledge estimation naturally to Bayesian networks. Applying this technique, Bayesian Knowledge Tracing was introduced in (Corbett and Anderson 1995). The model takes the form of the Hidden Markov Model, where student knowledge is a hidden variable and student performance is an observed variable. The model assumes a causal relationship between student knowledge and student performance; i.e. the correctness of a question is (probabilistically) determined by student knowledge. There are four parameters estimated by the model: 1) prior knowledge, which is the probability that a particular skill was known by the student before interacting with the tutoring systems; 2) learning rate, which is the probability that student's knowledge transits from unlearned to learned state after each learning opportunity; 3) guess, which is the probability that a student can answer correctly even if he/she does not know the skill required in the problem; 4) slip, which is the probability that a student responds to a question incorrectly even if he/she knows the required skills.

The classic BKT has been used broadly and successfully across a range of academic domains and student populations, including elementary reading (Beck and Chang 2007), middle-school mathematics ((Koedinger 2002), (Gong, Beck et al. 2010)), middle school science (Sao Pedro, Baker et al. 2013) and college-level genetics (Corbett, Kauffman et al. 2010). However, largely due to the simple model structure and the underlying assumptions the BKT model has, it seems to leave promising opportunities to improve. There is no lack of continuous efforts being placed towards enhancing the BKT model. As a consequence, a number of KT variants emerged.

One issue with BKT is that it is skill oriented. The assumption is learning differs across skills, but students do not different as individuals. The outcome is that for a skill, all students share the same BKT parameters, including prior knowledge, learning rate, guess rate and slip rate. Questioning this assumption, researchers have thought about individualization. The initial effort was done in the original work where BKT was introduced (Corbett and Anderson 1995). Apparently, the researchers acknowledged that solely skill-oriented knowledge estimation seems an incomplete assumption. Their solution is to estimate an individualized weight for each student and then adjust the model's generated parameters accordingly. However, the big drawback of this approach is the optimization can only be conducted off-line, meaning only after all data is obtained a weight can be estimated, so makes the approach a no run-time solution. More recently, the prior-per student individualization BKT variant was presented by (Pardos and Heffernan 2010). This model is able to provide run-time estimation by augmenting the original BKT with an additional observed variable, representing student factor. With this modification, BKT parameters are individualized for each student.

Another issue with BKT is that its estimated parameters are constant for a skill. This assumption applies that students learn/guess/slip at constant rates. They remain the same regardless of external factors, such as the time spent in learning, the problems practiced, or the mood the student is in, etc. The original intention of such design is made so as to reduce the number of parameters with the focus on refining a cognitive model rather than on evaluating students' knowledge growth (Draney, Pirolli et al. 1995), which seems opposing the goal of student modeling. A variant of BKT attempting to solve this issue is the contextual guess and slip BKT model. The model focuses on relaxing the assumption that guess and slip probabilities are fixed (Baker, Corbett et al. 2008). Another attempt tackles the problem of the constant learning rate. A moment-by-moment learning BKT model was proposed to detect how much learning occurs in each problem step (Baker, Goldstein et al. 2010), which contradicts the original assumption that learning rate is constant through all problems of a skill.

The third drawback with BKT is its lack of the ability to handle multiple skill problems. A classic BKT model is designed per skill. If a problem requires multiple skills to solve, it raises difficulty deciding to which skill this particular observation should belong. Due to the shortage of proper solutions, the strategy of splitting one observation of multi-skill problem into multiple observations of single skill problems is adopted. This solution is based on the assumption that all subskills are independent(Pardos,

Beck et al. 2008). An alternative, LR-DBN, is a solution to the problem without assuming subskill independency. LR-DBN uses logistic regression over each step's subskills to model transition probabilities for the overall knowledge required by the step(Xu and Mostow 2011).

Since Dynamic Bayesian Network, the form of BKT, is an open network, it eases implementing and investigating any plausible ideas which could be beneficial to better model student knowledge. A line of extensions of BKT is to incorporate help requesting behaviors. (Beck, Chang et al. 2008) and (Sao Pedro, Baker et al. 2013) added an additional observed variable in the BKT topology, representing whether the student has requested a help message or seen scaffolding in a problem. More extensions were investigated by leveraging a variety of student behaviors. For example, (Yudelson, Medvedeva et al. 2008) and (Wang and Heffernan 2012) take time into account, respectively focusing on time intervals the user spent on problems and the time intervals the student spent before taking his first response to the attempted problem. (Gong, Beck et al. 2010) uses the augmented BKT to analyze the impact of non-serious learning behaviors to student knowledge and learning. Other generic information can also help improving BKT. For example, (Pardos and Heffernan 2011) investigated the idea of incorporating item difficulty into the BKT model.

The openness of the BKT framework also enables the integration of BKT and other models. (Xu and Mostow 2013) uses Item Response Theory to refine BKT. Instead estimating prior knowledge by BKT, they approximate students' initial knowledge as their one-dimensional overall proficiency and combine it with the estimated difficulty and discrimination of each skill. A more comprehensive work was presented by (Khajah, R. et al. 2014), where the authors created a hybrid model, LFKT, which combines the latent-factor model and BKT. The model personalizes the guess and slip probabilities based on student ability and problem difficulty estimated by the latent-factors model. Dynamic Cognitive Tracing is a unified model based on BKT simultaneously addressing two problems, student modeling and cognitive modeling, which factorizes problems into the latent set of skills required to solve the problems (Gonzalez-Brenes and Mostow 2012).

2.2 Student Performance

In a broad sense, student performance denotes how students perform in a variety of contexts. However, in student modeling, in most cases, it refers in particular to the correctness of student response to the very next practice opportunity. There has always been great enthusiasm for modeling student performance. This focus is largely due to the belief that knowing how well the student will perform in the future helps to inform the system and allows the system to adapt better to suit the student's individual leaning needs.

A common technique used for modeling student performance is Bayesian Knowledge Tracing. Despite the original intention of modeling student knowledge, all BKT variants and extensions could also be used to model student performance. Based on student knowledge estimated by the model, the predicted performance can be obtained by calculation combining with the slip and guess rates. The related work of BKT has been elaborated in Section 2.1 .

2.2.1 Learning Curve Based Student Performance Modeling

Another major line of student models are based on fitting learning curves. Learning curves are a powerful tool used for evaluation of learning systems and measurement of students learning. Learning curves plot the performance of students with respect to some measure of their proficiency over time, such as response time, error rate, success rate, or mastery rate, etc, indicating how much students learn as a result of practicing (Anderson, Bellezza et al. 1993). Several main functions have been used to depict a learning curve, such as exponential growth, exponential rise or fall to a limit, and power law. With respect to the superior between two major functions, exponential law and power law, there always have been different voices (Newell and Rosenbloom 1981; Heathcote, Brown et al. 2000; Ritter and Schooler 2002; Leibowitz, Baum et al. 2010). The common used form of learning curve follows the so-called "power law

of practice", which states that the logarithm of the reaction time for a particular task decreases linearly with the logarithm of the number of practice trials taken (Newell and Rosenbloom 1981). To model student performance, instead of response time as the measure of the learning curve, the probability of a correct answer is employed.

Additive Factors Models (AFM) is a generalized linear mixed model applying a logistic regression to fit a learning curve to the student performance data (Boeck 2008). The central idea of AFM was originally proposed by (Draney, Pirolli et al. 1995), and introduced into the ITS field by (Cen, Koedinger et al. 2006), where the authors renamed the model as the Learning Factors Analysis model (LFA). The model has a logit value representing the accumulated learning for a student using single or multiple skills. The model captures the ability of the student and the easiness of the required skills. It also considers the benefit from prior practice by estimating the amount of learning on the skills for each practice opportunity.

Learning Decomposition (LD) is a variant of learning curves, which estimates the relative worth of different types of learning opportunities. The approach is a generalization of learning curve analysis, and uses non-linear regression to determine how to weight different types of practice opportunities relative to each other (Beck 2006). Unlike AFM, LD selects the form of exponential curves over power curves. The model has a free parameter to represent how well students perform on their first trial performing the skill, and a set of free parameters representing how quickly students learn the skill by performing a particular type of practice, such as reading the same story repeatedly or reading a new story.

The Performance Factors Analysis model (PFA) was presented by Pavlik et al. (Pavlik, Cen et al. 2009), which is on the reconfiguration of LFA. PFA drops the student ability parameter of LFA, which allows PFA to generalize across different subjects. Depending on the reconfiguration on the difficulty parameter, there are two implementations of PFA: one captures problem difficulty, and the other captures skill difficulty. Aside from the learning rate parameter, same as in LFA, the model also estimates an additional parameter for each skill reflecting the effect of prior unsuccessful practices.

The Instructional Factors Analysis model (IFA) was presented by Chi et al (Chi, Koedinger et al. 2011), which tailors PFA for their specific needs. The model, other than tracking the effect of prior successful practices, i.e. learning rate, and the effect of prior unsuccessful practices, estimates an additional variable's effect, what they called "tells", a form of instruction without yielding a correct or incorrect answer.

2.2.2 Recommender System Based Student Performance Modeling

Before 2010, most work addressing the problem of modeling and predicting student performance applies traditional machine learning techniques, such as classification and regression. The competition in the Knowledge Discovery and Data Mining Cup 2010 brought broad attentions from outside of the ITS fields. Participants are asked to learn a model from students' past behavior and then predict their future performance. A new flow since then is researchers start thinking about using recommender system techniques for modeling student performance. One of the competition winners pointed out that the basic problem of predicting missing ratings of users in recommender systems looks very similar to the problem of predicting missing performance of students in learning systems (Toscher and Jahrer 2010), so predicting student performance can be considered as rating prediction since the student, task, and performance would become user, item, and rating in recommender systems, respectively. Toscher et al adopted methods from collaborative filtering, such as KNN and matrix factorization, and also blended an ensemble of predictions using a neural network. Thai-Nghe et al. proposed tensor factorization models to take into account the sequential effect (for modeling how student knowledge changes over time). Thus, the authors have modeled the student performance as a 3-dimensional recommender system problem on (student, task, time) (Thai-Nghe, Horvath et al. 2011).

2.2.3 Tabling Based Student Performance Modeling

The idea behind this type of student models is to check the percentage of students with the same pattern of behaviors who have correctly answered for the next question. For the training data, a table is constructed for each skill which works as a look up table mapping student behaviors to the percentage of students who yield a correct answer. Each row (or column) in the table represents a category of student behavior pattern, such as ask for fewer than 5 hints. Due to its low dimension, the table can typically deal with a small number of types of student behaviors. The Assistance Model (AM) applies the tabling technique, which consists of a table of probabilities of a student answering a question correctly based on the number of attempts and the percentage of available hints used on the previous problem of the same skill(Wang and Heffernan 2011). Extended the AM model, Hawkins et al. proposed the Assitance Progress Model" (APM). The authors pointed out that AM only takes into account the number of attempts and percentage of hints required on the previous question without considering the progress the student is making over time in terms of attempts and hints used (Hawkins, Heffernan et al. 2013). APM broadens the scope of examination to previous two problems to predict performance on the next question.

2.3 Student Robust Learning

In recent years, a voice starts to be heard which debates whether the research thread of modeling student performance, i.e. correctness of next problem, has probably progressed beyond a useful point (Beck and Xiong 2013), and student modeling research has paid limited attention to modeling the robustness of student learning (Baker, Gowda et al. 2011).

Rather than focusing on student short-term performance, giving more attentions to the long-term learning seems more meaningful. The argument is that the ultimate goal of tutoring systems is not to improve future performance within the system itself but to improve unassisted performance outside the system (Baker, Gowda et al. 2011). Therefore, intelligent tutoring systems should promote robust learning (Koedinger, Corbett et al. 2012). There are three components of robust learning: preparation for future learning, transfer to novel contexts and retention over time.

There are studies demonstrating the effectiveness of some interactive learning environments on preparing students for future learning (Tan and Biswas 2006; Chin, Dohmen et al. 2010). Research of modeling preparation for student future learning (termed "PFL") has not started until recently. Baker and his colleagues pioneered a line of research of PFL. In their first work (Baker, Gowda et al. 2011), the researchers designed a model to predict a student's later performance on a paper post-test of preparation of future learning. They constructed the model in a form of linear regressions, informed the model with features of student learning and behaviors within a computer tutoring system, and evaluated their model against BKT. They showed that the model predicts PFL better than BKT and accommodates limited amounts of student data. Extending this work, Hershkovitz et al. proposed an alternate method of predicting PFL. They used quantitative aspects of the moment-by-moment learning graph created in their prior work, which represents individual students' learning over time and is developed using a knowledge-estimation model which infers the degrees of learning that occurs at specific moments rather than the student's knowledge state at those moments (Hershkovitz, Baker et al. 2013). They showed better predictive performance of the new model at student-level cross-validation.

Knowledge transfer includes two aspects: the transfer from in-tutor performance to out-of-tutor performance and the transfer from one skill to new skills. In the past, a small number of studies have been conducted to design student models to investigate the same-skill transfer in and out of tutor (Corbett and Anderson 1995; Baker, Corbett et al. 2010; Corbett, Kauffman et al. 2010). On the aspect of the cross-skill transfer, student modeling approaches are still in their early age. Some early efforts were placed towards demonstrating a question of yes or no: whether student knowledge of one skill will transfer to another skill (Martin and Vanlehn 1995; Desmarais, Meshkinfam et al. 2006; Zhang, Mostow et al. 2007) . For example, Zhang et al. used the method of learning decomposition to study students' mental representations of English words with a particular interest in whether practice on a word transfers to

similar words with the same root (e.g. “cat” and “cats”). Until recently, studies have started to directly predict knowledge transfer across skills quantitatively. Baker et al. presented an automatic model to predict student knowledge transfer. They applied feature engineering and built a linear regression model to predict student scores in a post-test of knowledge transfer. The post-test involved related but different skills than the skills studied in the tutoring system. The model was evaluated against BKT at student level cross-validation (Baker, Gowda et al. 2011). Sao Pedro et al. studied transfer in the domain of science. They based their studies on an existing educational data mining model, BKT, and built an augmented model of transfer to track students’ performance across topics, which was viewed as evidence of skill transfer of data collection inquiry skills inter-science topics (Sao Pedro, Gobert et al. 2012). As an extension of their prior work, Sao Pedro et al. constructed machine-learned models which were trained using labels generated through the method of “text replay tagging”. The researchers compared two approaches, BKT and an averaging approach that assumes static inquiry skill level to predict student performance on a transfer task requiring data collection skills (Sao Pedro, Baker et al. 2013) .

Retention is the third component in the robust learning framework. It is often referred to as delayed-performance reflecting knowledge retained over time. Pavlik and Anderson used a modeling approach including an extended ACT-R model and the Atkinson Mardov model to predict how long knowledge will be retained after learning foreign language vocabulary. Accordingly they developed a quantitative algorithm to dynamically adjust the balance between increasing and decreasing temporal spacing to maximize long-term gain per unit of practice time (Pavlik and Anderson 2008). Wang and Beck investigated predicting student delayed-performance after 5 to 10 days to determine whether and when the student will retain the studied material. While applying feature engineering, they found some of the traditionally-believed useful features for predicting short-tem performance have little predictive power for predicting retention. They then built a student model in the form of logistic regression on the basis of the performance factors analysis model to predict the correctness of student response after a delayed period (Wang and Beck 2012). A follow-up work of Wang and Beck (2012) was done by Li et al. The researchers inherited the main framework from the prior work, but with a new goal of exploring features that are specially targeted to measure retention. They found that that mastery speed is the best predictor and the effects of performance on initial mastery persists across a lengthy interval.

2.4 Student Affect

A key factor in student learning is their emotions. Emotions can be used in the learning content to increase learners’ attention and improve his memory capacity (Nkambou, Bourdeau et al. 2010). Many works in Computer Science, Neurosciences, Education and Psychology have shown the significant impact of emotions in learning activity. Different emotions and moods are often compiled in the more general constructs of positive versus negative affect (Tellegen, Watson et al. 1999). Positive affect comprises emotions such as enjoyment, pride and satisfaction, and negative affect comprises anxiety, frustration and sadness (Tellegen, Watson et al. 1999). In general, positive affect has a profound facilitative effect on cognitive functioning and learning (Isen, Daubman et al. 1987; Hidi 1990; Ashby, Isen et al. 1999; Fiedler 2001). Relations between learning and negative affect have been found inconsistent. For example, negative relations have been found between learning and general negative affect, such as anxiety, anger, shame, boredom and hopelessness (Hembree 1988; Boekaerts 1993; Linnenbrink, Ryan et al. 1999; Pekrun, Goetz et al. 2002; Pekrun, Goetz et al. 2004). However, some emotions traditionally viewed as negative can prompt more analytical, detailed and rigid ways of processing information (Isen, Daubman et al. 1987; Hidi 1990; Ashby, Isen et al. 1999; Fiedler 2001). For example, confusion is associated with learning under certain conditions (Liu, Pataranutaporn et al. 2013; D'Mello, Lehman et al. 2014)

The biggest challenge of understanding and modeling human emotions is its difficulty to obtain observations. Traditional methods used in conducting studies include self-report (Arroyo, Cooper et al. 2009), retrospective emote-aloud protocols (D'Mello, Craig et al. 2008), field observations (Baker, Moore et al. 2011) and video observations (D'Mello, Taylor et al. 2007). The drawbacks are evident, that some

methods disrupt natural affective flow and some are expensive to conduct, so none is suitable for the needs of computer systems.

2.4.1 Sensor-based Modeling

Automatically detecting emotions using a variety of sensors seems a good solution to address these issues. The use of sensors, such as facial expression sensors, postures analysis seats, and eye gaze detection equipment, etc., can largely enrich data types collected from students, attract researchers with various knowledge backgrounds to contribute their expertise to student modeling, and presumably improve understanding of learning.

Kapoor et al. proposed a multi-sensor affect recognition approach, which uses multimodal sensory information from facial expressions and postural shifts of the learner and information about the learner's activity on the computer to classify affective states of children trying to solve a puzzle on a computer (Kapoor and Picard 2005). Litman and Forbes-Riley extracted acositic-prosodic features from the student speech and combined them with student and task dependent features to predict student emotion states (Litman and Forbes-Riley 2006). In the study of (Arroyo, Cooper et al. 2009), a sensor framework including various sensors, such as video camera, posture chair, skin conductance bracelet were integrated into intelligent tutors which were used by students in classroom experiments. The researchers constructed a linear regression model including students' self reported data, learning related variables and tutor-collected sensor data to predict emotions. D' Mello et al. developed and evaluated a multimodal affect detector that combines conversational cues, gross body language and facial features. The detector uses feature-level fusion to combine the sensory channels and linear discriminant analyses to discriminate between naturally occurring experiences of boredom, engagement/flow, confusion, frustration, delight and neutral (D'Mello and Graesser 2010). Muldner et al. used a combination of tutor and sensor data to predict student delight moments during learning activities (Muldner, Burleson et al. 2010). Grafsgaard et al. presented an automated facial recognition approach to analyzing student facial movements during tutoring. They also built predictive models to examine the relationship between intensity and frenquency of facial movements and tutoring session outcome, which highlights relationships between facial expression and aspects of engagement, frustration and learning (Grafsgaard, Wiggins et al. 2013).

2.4.2 Sensor-free Modeling

One limitation of the sensor-based approaches is their applications are strictly dependent on the use of sensors. Other than the potential issues such as sensor malfunction and connection failures, a practical challenge is that the cost of equipping the sensors might raise economic obstacles to tutor users. Therefore, sensor-free emotion modeling seems more intriguing. A robust sensor-free emotion model can not only be more broadly applicable, also be easily validated empirically across different tutor environments and tutor populations.

Arroyo et al. presented an approach to modeling student affect. They took students' survey answers and log file data as students' observed behaviors and constructed a Bayesian Network, integrating behavioral, cognitive and motivational variables, to infer the students' cognitive and affective state (Arroyo and Woolf 2005). D'Mello et al. explored the reliability of detecting a learner's affect from conventional features extracted from integrations between students and AutoTutor. The obtained the ground truth based on ratings given by the learner, a peer and two trained judges. They applied multiple regression analysis and confirm that dialogue features could predict the affective states of boredom, confusion, flow and frustration (D'Mello, Craig et al. 2008). Conati et al. presented a probabilistic model of user affect. The model applies a Dynamic Bayesian Network to take the causes and effects of emotional reactions into account for handling uncertainty involved in recognizing a variety of user emotions (Conati and Maclaren 2009). Lee et al. presented a sensor-free detector of confusion to study novice Java programmers' experiences of confusion and their achievements. They found that confusion

which is resolved in associated with statistically significantly better midterm performance than never being confused at all (Lee, Rodrigo et al. 2011). Baker and his colleagues in (Baker, Gowda et al. 2012) elaborated a few sensor-free affect models that can detect student engaged concentration, confusion, frustration and boredom solely from students' interactions within a tutor system. They obtained the ground truth using field observations of affect and attempted to fit the detectors using eight common classification algorithms, such as J48 decision trees, step regression, etc. As another work in the line of researching confusion, Liu et al. used sensor-free affection detection to explore the relationship between affect occurring over varying durations and learning outcomes among students using a tutoring system. The researchers in particular distinguished two main negative affects, frustration and confusion, and provided correlation analyses between student test scores and sequences of two affective states independently and in combination(Liu, Pataranutaporn et al. 2013). A richer range of affects were studied in (Wixon, Arroyo et al. 2014), where the researchers developed and analyzed affect detectors for confidence, excitement, frustration and interest. They relied on self-report "ground truth" measurements of affect within a tutor and model them as continuous variable that are later discretized into positive, neutral and negative classifications. Moreover, the authors discussed the opportunities and limitations of scaling up the approaches by cross-validation with regard to potentially distinct sample groups. A detailed review as to knowledge elicitation methods for affect modeling in education was provided in (Porayska-Pomsta, Mavrikis et al. 2013), where the researchers provided a synthesis of the current knowledge elicitation methods that are used to aid the study of learners' affect and to inform the design of intelligent technologies for learning. In particular, they discussed Advantages and disadvantages of the specific methods are discussed along with their respective potential for enhancing research in this area, and issues related to the interpretation of data that emerges as the result of their use.

3. Commonly-Used Student Models

3.1 Knowledge Tracing

The knowledge tracing model (Corbett and Anderson 1995), shown in Figure 2 is a graphical model composed of two binary nodes: student knowledge and student performance. Student knowledge is the hidden variable, which, as a convention, is shown by an oval. Student performance is the observed variable, shown by a rectangle. The arrow between student knowledge and student performance reflects a causal relationship, indicating this model assumes that student performance, i.e. the correctness of a question, is (probabilistically) determined by student knowledge.

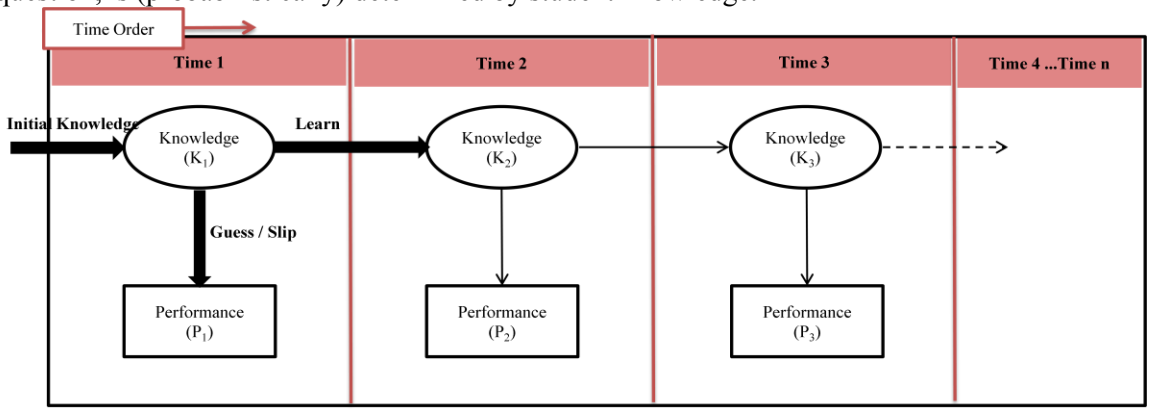


Figure 2. The knowledge tracing model

Furthermore, the knowledge tracing model, as we can see in Figure 2, consists of a chain of such units, which makes the model a dynamic Bayesian network. The model has n time slices, where n is determined by how many practices the student actually did for the particular skill. If the student practiced

10 questions for a skill, say "Addition and Subtraction", the model of that student on "Addition and Subtraction" should contain a chain of 10 units in a time order. This sequence of units is ordered by time; thus Time 2 is the problem on this skill occurring after the problem at Time 1. There is another important causal relationship represented by the arrow pointing from student knowledge at time t-1 to student knowledge at time t. It reflects the idea that how much a student knows about a skill at a certain time point is affected by how much he or she knew in the previous time point.

The knowledge tracing model can be trained by fitting the data of student performances on a skill to the model. In this work, a student performance is the correctness of student response in an Assistment. The model takes student performances and uses them to estimate the student's level of knowledge.

The knowledge tracing model was designed as skill-oriented, i.e. each skill has four associated parameter estimates learnt. Among them, there are two learning parameters. The first is initial knowledge (K0), the likelihood the student knows the skill when he or she first uses the system to practice the skill. The second learning parameter is the learning rate (L), the probability a student will acquire a skill as a result of an opportunity to practice it. In addition to the two learning parameters, there are two performance parameters: guess and slip (G and S). Student performance is assumed to be a noisy reflection of student knowledge, mediated by these two performance parameters. The guess parameter represents the fact that the student may sometimes generate a correct response in spite of not knowing the correct skill. For example, some ASSISTments items have multiple choice questions, so even a student with no understanding of the question could generate a correct response. The slip parameter acknowledges that even students who understand a skill can make an occasional careless mistake.

3.2 Performance Factors Analysis

Performance Factors Analysis (PFA) is a student modeling approach proposed by (Pavlik, Cen et al. 2009). It takes the form of logistic regression with student performance as the dependent variable. We chose PFA as our framework as, relative to Bayesian networks, logistic regression is more flexible to incorporate more (or different) predictors.

It is particularly important to note that there are two student models, both of which were named as Performance Factors Analysis. Both models were designed based on the reconfigurations of Learning Factors Analysis(Cen, Koedinger et al. 2006) by dropping student variable and considering a student's prior correct and incorrect performances. The two models vary in their independent variables. The model presented in (Pavlik, Cen et al. 2009) estimates item difficulty (i.e. one parameter per question); the other (Pavlik, Cen et al. 2009) estimates skill difficulty (i.e. one parameter per skill. Note that in the original paper (Pavlik, Cen et al. 2009), the authors used the term "knowledge components (KC)" while we use the term "skills"). In this work, I refer to the first model as the PFA-item model; the other is represented as the PFA-skill model.

$$m(i, j \in required\_skills, q \in questions, s, f) = \beta_q + \sum_{j \in required\_skills} (\gamma_j s_{i,j} + \rho_j f_{i,j}) \quad \textbf{(1) PFA-item}$$

$$m(i, j \in required\_skills, s, f) = \sum_{j \in required\_skills} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}) \quad \textbf{(2) PFA-skill}$$

The *m*s in Equation 1 and 2 are logits (i.e., are transformed by $e^x/(1+e^x)$ to generate a probability). They represent the likelihood of student *i* generating a correct response to an item. In the equations, $s_{i,j}$ and $f_{i,j}$ are two observed variables, representing the numbers of the prior **s**uccessful and **f**ailed practices done by student *i* on skill *j* . The corresponding two coefficients (γ_j and ρ_j) are estimated to reflect the effects of a prior correct response and a prior incorrect response of skill *j*. Rather than considering all of the skills in the domain, the PFA model focuses on just those skills required to solve the problem.

The PFA-item model estimates a parameter (β_q) for each question representing its difficulty. In the PFA-skill model, as seen in Equation 2, the β parameter has a subscript of *j*, indicating that it captures the

difficulty of a skill. Also, it is moved to the inside of the summation part to incorporate multiple skills, i.e., in PFA-skill an item's difficulty is the sum of its skills' difficulties.

Chapter 3. Parameter Interpretation

1. Introduction

In educational research, one fundamental goal is assessing students and estimating constructs, such as their knowledge levels, behaviors, goals and mental states, etc. Unfortunately, most of those attributes are difficult to measure directly. The traditional method used by researcher in Education is to design an experiment, which address a particular interest, find appropriate subjects for the experiment, conduct the experiment, and finally analyze the data collected from the experiment. Apparently, such method is very expense in terms of human labor and even in economic.

An intelligent tutoring system provides a platform where students can learn, while the system can understand the learning. Taking the advantage of computer-based learning environments, students using such systems can contribute many more data than an experiment. The system is able to log every interaction with the student. With great amounts of data, data mining can be nicely used for exploring and analyzing the data, and further many scientific questions could be answered.

As the most important means, building a model to describe data is commonly used. The model fits the data, and the model parameters capture patterns that summarize the underplaying relationships in data. In many works, the estimated model parameters are interpreted to convey interesting information. Arroyo, et al. applied a Bayesian Network on log-data, and through interpreting parameter estimates inferred a student's hidden attitude toward learning, amount learnt and perception of the system (Arroyo and Woolf 2005). Beal, et al. used Hidden Markov Models to fit log data, and extracted learning patterns of students by summarizing the models' parameter estimates (Beal, Mitra et al. 2007). Baker, et al. analyzed student log data using a linear regression model, and used the learnt coefficients to answer questions with regard to a phenomenon: gaming the system (Baker, Corbett et al. 2008).

It appears that in order to answer research questions and establish scientific insights, interpreting model parameters is a major way. The problem arises here, as we would like to have a method to evaluate the model in terms of whether the parameter estimates from the model make sense. Typically a common evaluation to a model focuses on how well the model fits the training data and how well the model can generalize to unknown test data. This measure captures the model's fitting goodness and predictive accuracy, while how good its parameter estimates are is not covered. Therefore, in order to accomplish the task of finding good parameter estimates which can convey useful information, we need a means to evaluate the goodness of the parameters. The property is refereed to as parameter plausibility. Moreover, we need efforts to build models which can provide parameters with high plausibility.

The knowledge tracing model is the most commonly used student model. For a single skill, it produces four parameters, which capture four different properties in student learning: prior knowledge, guess, slip and learning rate. As a commonsense, learning rate should be in a reasonable range, as topics students are learning should not be extremely easy, so that students can learn it immediately. However, the fact is without any controls, KT could produce a parameter estimate whose value is far beyond what people believe, for instance 0.9 of learning rate. This situation also applies to other parameters as well. For instance, the guess parameter could be estimated as a value over 0.5, which means that the student can guess correctly more than a half of problems he attempts. A few works have been done concerning parameter plausibility of the KT model.

Beck, et al. first pointed out as a Hidden Markov Model, the KT model has the problem of identifiabiliity: observed student performance corresponds to an infinite family of possible model parameter estimates, all of which make identical predictions about student performance. These parameter estimates make different claims, some of which are clearly incorrect about the students' hidden learning properties. That is to say, the KT model is prone to converging to erroneous degenerate states. The authors present using Dirichlet priors, which is a natural mechanism in graphical models to provide a means of encoding prior probabilities of the parameters, to bias the model search process. They examined learning curves constructed by the model's parameter estimates to evaluate those parameter estimates' plausibility, and showed that using Dirichlet priors resulted in more sensible models.

In a follow-up study, the author detailed the reasons which cause the problem of Identifiability (Beck 2007). Moreover, instead of manually setting Dirichlet priors based on the researcher's understanding to the domain, the work presented a method to automatically generate Dirichlet priors based on the statistics of data. The study is lack of a controlled field study, which leads to a difficulty to evaluate parameter plausibility. The results suggested that the method might result in more plausible parameter estimates.

Aside from using Dirichlet priors, there is another approach used to address the problem of implausible learnt parameters. Researchers can impose a maximum value that the learnt parameters could reach, such as a maximum guess of 0.30 that was used in the parameter fitting procedure (Corbett and Anderson 1995).

To seek a better understanding of the behavior and accuracy of the EM algorithm in fitting the KT model, Pardos, et al. used synthesized data that comes from a known set of parameter values, and by observing the results from model fitting procedures they explored the knowledge tracing parameter convergence space. Knowing the ground truth of the parameters, they examined the estimated parameters resulted from different runs of model fitting process with different starting priors.

2. Automatically Generating Dirichlet Priors to Improve Parameter Plausibility

2.1 Background

Depending on model fitting approaches, KT is possible to generate multiple parameter estimates which fit training data equally well. Therefore, How to estimate the model parameters is an important issue to KT. There are a variety of model fitting approaches. The Expectation Maximization (EM) algorithm is majorly used. It finds parameters that maximize the data likelihood (i.e. the probability of observing the student performance data). Compared to other model fitting approaches for KT, using EM to learn the parameters has been shown to be able to achieve the highest predictive accuracy (Gong, Beck et al. 2010). However, it suffers two major problems that are inherent in the KT model's search space: local maxima and multiple global maxima ((Rai, Gong et al. 2009), (Beck and Chang 2007)).

Local maxima is common in many error surfaces. The issue is that the algorithm has to start with some initial value of each parameter, and its final parameter estimates are sensitive to those initial values. The EM algorithm is such an algorithm. To use EM to fit the knowledge tracing model, for all its four parameters, manually setting initial values are necessary. The estimates of parameters after the training procedures could be impacted by the initial choices of seeding values.

Multiple global maxima is another issue. This issue is also known as identifiability. In particular, the problem of identifiability regards to that for the same model, given the same data, there are multiple (differing) sets of parameter values that fit the data equally well (so called multiple global maxima). Based on statistical methods, there is no way to differentiate which set of parameters is preferable to the others.

Consequently, we have to be more careful to select the parameters' initial values when using EM to fit the model, as we want to neither be stuck with some local maxima, nor get unbelievable parameters which are meaningless for making scientific claims, even if those parameters make accurate predictions.

In order to solve the problems, (Beck and Chang 2007) proposed that, rather than using a single fixed value to initialize the conditional probability table when training a knowledge tracing model, it is possible to use Dirichlet priors to start the algorithm.

Dirichlet prior is an approach used to initialize conditional probability tables when training a Dynamic Bayesian network. Using Dirichlet priors in KT assumes that across all skills, their corresponding parameters' values are drawn from a Dirichlet distribution, which is specified by a pair of numbers (α, β).

Figure 3 shows an example (the dashed line) of the Dirichlet distribution for (9, 6). If this sample distribution were of K_0, it would suggest that few skills have particularly high or low knowledge, and we expect students to have a moderate probability of mastering most skills. Conceptually, one can think of

the conditional probability table of the graphical model being as seeded with 9 instances of the student knowing the skill initially and 6 instances of him not. If there is substantial training data, the parameter estimation procedure is willing to move away from an estimate of 0.6. If there are few observations, the priors dominate the process. The distribution has a mean of $\alpha/(\alpha+\beta)$. Note that if both α and β increase, as in the solid curve, whose Dirichlet parameters are 27 and 18, in Figure 3 , the mean of the distribution is unchanged (since both numerator and denominator are multiplied by 3) but the variance is reduced. Thus, Dirichlets enable researchers to not only specify the most likely value for a parameter, as using fixed priors can do, but also the confidence in the estimate.

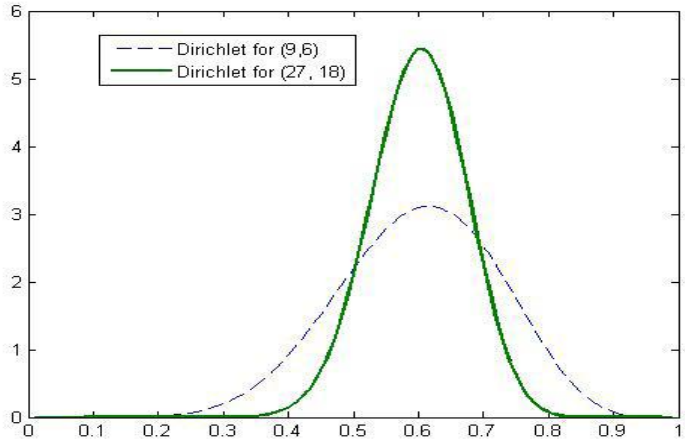


Figure 3 Sample Dirichlet Distributions demonstrating decreasing variance

There are two benefits from using Dirichlet priors to initialize EM for training a KT model. First, it allows the injection of knowledge engineering. Through setting the Dirichlet parameters, researchers can specify their confidence about what value, or what range of values, of a parameter should be. If they have strong prior knowledge about a parameter, they could use larger Dirichlet priors, indicating that they are very sure about what the parameter should be. For example, when being taught a new skill, students should have no prior knowledge about it. Taking this fact into account, researchers could set the Dirichlet priors of K_0 to (1,10), reflecting their belief that students are not likely to understand the new skill. If they set the Dirichlet priors to (10, 100), contrast to (1, 10), the values indicate the confidence of the researchers is very high about the students having almost no possibility to know the skill.

The other benefit is that using Dirichlet priors helps reduce extremely bad estimates of parameters for, especially, skills with few observations. When data used to train a model of a skill are sparse, there is little constraint provided by the data. Thus, parameter estimates can be extreme values due to over-fitting. Since the assumption is that parameters across all skills are sampled from a Dirichlet distribution, it is reasonable to assume that the parameter of the skill with sparse data should be similar to the parameters of other, better-estimated, skills. Dirichlet priors provide additional observations of the skill, which bias the estimates towards the mean of the distribution. As a result, models with few data are more influenced by the priors towards the mean and those estimates are expected to become more reasonable.

2.2 The Algorithm

If researchers have strong knowledge about the domain, using their prior knowledge is a reasonable way to set Dirichlet priors (Beck and Chang 2007). However, one complaint is that such an approach is not necessarily replicable as for different domains and different subjects; different experts may give different answers. Although work in (Beck 2007) has proposed a method to automatically generate Dirichlet priors. There are two limitations. First, the study was conducted on the basis of a fairly small data set, and thus more exploration to the method is necessary. Second, the proposed way to calculate Dirichlet priors treats

all data equally weighted, i.e. a data instance corresponding to few observations is viewed as equally important to a data instance obtained from a large number of observations.

In this dissertation work, we extend the previous work and present an automatic method used to generate Dirichlet priors. The method considers the weights of the observations and the study was conducted on a large size of data.

The detailed procedure of the algorithm is shown as follows.

The algorithm of using the automatic generated Dirichlet priors to train KT

```
1: Let D[] denote the training data, D[i] denote the i^th skill's data
    in D, K0[] denote the parameters of prior knowledge, G[] denote
    the parameters of guess, S[] denote the parameters of slip, L[]
    denote the parameters of learning.
2: def train_KT_by_Dirichlet()
3:   {K0[], G[], S[], L[]} = train_KT(KT, D[], fixed_priors[])
4:   for each of {K0[], G[], S[], L[]}
5:     param[] = K0[]                    // take K0 for an example
6:     Dirichlet_priors[] = auto_gen_Dirichlet(param[])
7:   end for
8:   {K0[], G[], S[], L[]} = train_KT(KT, D[],Dirichlet_priors[])
9: end

10: def train_KT(model, data[], priors[])
11:   for i=1 to data.length do             //i.e., for each skill
12:     {K0[i], G[i], S[i], L[i]} = EM (model, data[i], prior[]);
13:   end for
14:   return {K0[], G[], S[], L[]}
15: end

16: def auto_gen_Dirichlet(param[])
17:   μ = mean(param[])
18:   σ^2 = var(param[])
19:   for i=1 to D.length do
20:     weight[i] =  sqrt(D[i].length)
21:     sum_weight += weight[i]
22:   end for
23:   for i=1 to D.length do
24:     μ' += (weight[i] * param[i])/sum_weight
25:     σ^2' += (weight[i] * (param[i] −μ)2)/sum_weight
26:   end for
27:   α =(μ'^2 / σ^2')*(1−μ')−μ'
28:   β = α *((1/ μ')−1)
29:   return {α, β}
30: end
```

First, the algorithm trains a KT model for each skill in the data, shown in line 3. Each KT model is fit by the data of a skill and learnt by an EM algorithm, where EM is initialized by fixed priors for each KT parameter, K_0, G, S, L. The fixed priors are obtained by rough estimates of the domain. As a result, for each skill, a set of KT parameters are estimated. For example, if there are *n* skills, there would be *n* sets of KT parameters, i.e. *n* values of each of the K_0, G, S, L parameters.

Next, for each KT parameter, the algorithm generates its Dirichlet priors by calling the method "auto_gen_Dirichelt". This method takes the *n* values of that KT parameter, and based on those values calculates Dirichlet priors. In detail, the method calculates the mean and variance of the *n* values. At this point, Dirichlet priors could be induced based on the mean and variance, following the standard transformation formulas. However, simply using such mean and variance gives all skills equal weight. This can be problematic, since as we mentioned earlier, skills with few cases are susceptible to error: going to extreme values such as getting 0 as student's learning parameter. Therefore, we weight each estimate by the square root of the number of cases used to generate the estimate, since $\sqrt{N}$ is how the standard error decreases. Thus, the method, instead, calculates Dirichlet parameters by using the weighted mean and weighted variance. In this way, each KT parameter has its Dirichlet priors.

At last, the algorithm trains a KT model for each skill again, shown in line 8, using the generated Dirichlet priors to initalize EM to estimate model parameters.

As an extra attempt, the algorithm could be iterated by looping back from line 8 to line 3. The logic behind the iteration is that instead of using fixed priors to initialize the EM algorithm, we could also start EM with the Dirichlet priors obtained from the last loop. It is interesting to see whether using automatically-generated Dirichlet priors could be able to improve parameter plausibility. It is also interesting to see that how using iteratively-generated Dirichlet priors impact parameter plausibility. Corresponding to using iteratively-generated Dirichlet priors, fixed priors could also be obtained by iteration. For example, in the second loop, for the parameter K_0, rather than using the rough estimate as its prior to start EM, the algorithm can use the mean of K_0 values estimated from the first loop as the prior. In this way, the prior is able to reflects the characteristic of the data .

2.3 Results

For this study, we used data from ASSISTment. The data are from 199 twelve- through fourteen- year old 8th grade students in urban school districts of the Northeast United States. These data consisted of 66,311 log records of ASSISTment during January 2009 to February 2009. Performance records of each student were logged across time slices for 106 skills (e.g. area of polygons, Venn diagram, division, etc). We split our data into training set and test set with the proportion of 2:1.

For each skill, we trained a few KT models using different types of priors-setting methods, including fixed priors, Dirichlet priors, iterative fixed priors and iterative Dirichlet priors. We compared parameter plausibility resulted from those methods. Quantifying parameter plausibility is difficult since there are no well-established means of evaluation. In this study, we explored two metrics for this analysis.

The first metric is the number of practice opportunities required to master each skill in the domain. We assume that skills in the curriculum are designed to neither be so easy to be mastered in very few opportunities nor too hard as to take a large number of opportunities. We define mastery as the same way as was done for the mastery learning criterion in the LISP tutor(Corbett 2001): students have mastered a skill if their estimated knowledge is greater than 0.95. Based on students' prior knowledge and learning parameters, we calculated the number of practice opportunities required until the predicted value of P(know) exceeds 0.95, indicating students have mastered the skill. In particular, if students master a skill using fewer than 3 practice opportunities, we refer to this situation as "extremely-fast-learnt". If students do not master a skill until over 50 practice opportunities, we refer to this situation as "extremely-slowly-learnt". For each priors-setting method, we inspected how many skills with the unreliable extreme cases it resulted in. The comparisons are shown in Table 1.

Fixed priors resulted in more extreme cases, 29 extremely-slowly-learnt skills and 2 extremely-fast-learnt skills, than Dirichlet priors, shown in the first row of the table. This result implies that Dirichlet prior model estimates more plausible parameters. With more iteration, the extreme cases remain constant with fixed prior whereas the number slightly decreases with Dirichlet priors. The skills that are found implausible by Dirichlet are a subset of those found by fixed priors. Hence, Dirichlet is fixing the implausibility of fixed priors and is not introducing new problems of its own.

Table 1 Comparison of extreme number of practice until mastery

	# of extremely-slowly-learnt skills		# of extremely-fast-learnt skills	
	Fixed priors	Dirichlet priors	Fixed priors	Dirichlet priors
1 iteration	29	17	2	0
2 iterations	29	16	2	0
3 iterations	29	15	2	0

The second metric used to evaluate parameter plausibility is student prior knowledge assessed by a pretest. The traditionally-assessed student prior knowledge works as an external measure. By comparing to this standard, we could be able to evaluate the KT parameter, K_0, as K_0 also estimates student prior knowledge and so should have large correlation with the external measurement.

To obtain K_0 at the student level, we trained a KT model for each student, rather than each skill. A KT model of a student was fit by the responses to questions he solved across skills. The model then estimated a set of parameters (prior knowledge, guess, slip and learning) for the *student*, which represents his aggregate performance across all skills. The parameter, prior knowledge, particularly captures the student's overall prior knowledge on all skills in the domain.

The students in our study had taken a 33-item algebra pre-test before using ASSISTments. The pretest questions covered the skills which would be practiced later when students were using ASSISTments. We used the percent of correct as the pretest score. We calculated the correlation between the students' prior knowledge estimated by the models and their pretest scores. In Table 2, we can see that the Dirichlet prior model produces slightly stronger, but not reliably so, correlations than the fixed prior. Neither method improves with more runs of iteration.

Table 2 Comparison of correlation between prior knowledge and pretest

	Fixed priors	Dirichlet priors
1 iteration	0.76	0.80
2 iterations	0.73	0.81
3 iterations	0.73	0.81

3. Automatically Generating Multiple Dirichlet Priors to Improve Parameter Plausibility

3.1 Background

Modeling all skills using the same set of Dirichlet priors assumes that for all skills, their KT parameters are drawn from a single set of Dirichlet distributions. For example, across all skills, their K_0 values are drawn from a Dirichlet distribution of K_0; their guess values are drawn from a Dirichlet distribution of guess. So are slip and learning. That is to say, skills are assumed to have distributional similarities with each other, in all of their KT parameters, prior knowledge, guess, slip and learning. Regardless of what skill it is, due to using a single set of Dirichlet priors, its KT parameters are respectively biased towards the means of the distributions of prior knowledge, guess, slip and learning. The bias is particularly stronger for those abnormal outlier skills with insufficient observations. Specifically, with sparse data, the model of a skill is trained with few constraints from the evidence; thus although it achieves the highest predictive accuracy it could get, still generates implausible parameter estimates. As a result, the skill appears an outlier. Since for such skills, it is preferred to have parameter estimates which are more similar to the other, better-estimated, skills, Dirichlet priors provide bias to them. As shown in Figure 4, Skill A and Skill B are at the tail of the distribution. By using Dirichlets, those outliers are biased towards the mean of the distribution. The hypothesis is that it is probably good that they are moved towards the center.

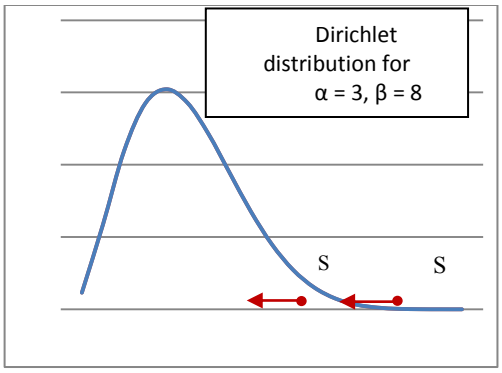


Figure 4 Dirichlet distribution with two outliers

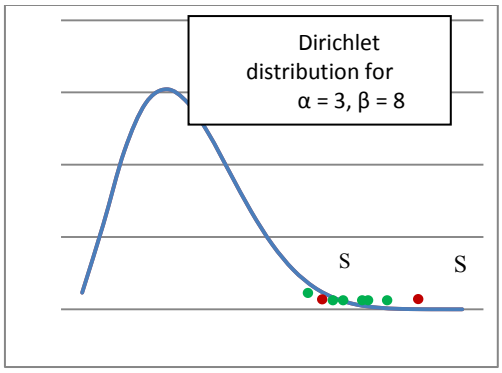


Figure 5 Dirichlet distribution with more "outliers"

Dirichlet has been shown to work well on positively biasing outliers ((Beck and Chang 2007), (Rai, Gong et al. 2009)). However, a second-thought question is: are the outliers really outliers?

The assumption of using a single set of Dirichlet distributions is that KT parameters of all skills in the domain are from that single set of distributions. As a result, based on this assumption, those skills which are located further away from the means are considered outliers. However, it is also reasonable to assume that KT parameters of skills are sampled from multiple Dirichlet distributions. Take the above example, in Figure 6, which shows the same distribution as Figure 4, if there are many skills with similar parameter estimates to Skills A and B, perhaps they are not really outliers. A plausible hypothesis is that they are sampled from a separate Dirichlet distribution, so that they behave differently from the other skills in the domain. To such case, moving those skills towards the mean may be inappropriate as they are better modeled separately corresponding to the additional distribution.

3.2 The Approach

3.2.1 Identify KT Parameters from Multiple Dirichlet Distributions

We used clustering for identifying skills sampled from multiple Dirichlet distributions. For skills sampled from a Dirichlet distribution, a unique set of Dirichlet priors should be used. Therefore, skills are classified into clusters, each of which is considered a region in the 4-dimentional knowledge tracing parameter space contains skills with homogeneity. For example, possibly a cluster of skills are well described as "not previously known (low K_0), but easy to learn (high learning)", or "hard to learn, but students have partial incoming knowledge".

We used the k-means cluster analysis to classify the skills, as intuitively skills with similarity would be spatially located close to each other in the parameter space. The four KT parameters, learnt from KT models with fixed priors, are used as the attribute set of a skill.

We did not have prior knowledge about how many Dirichlet distributions generate the skills, so we did not specify a certain k for the k-means method. Nor we used any self-adaptive k-means clustering methods to automatically determine the number of clusters. Self-adaptive clustering methods always have their own metrics to evaluate the goodness of the current clustering results, such as the algorithm converges without changes bigger than a pre-set threshold between iterations. Our goal, however, is to see how many clusters could result in better parameter plausibility, so we had no *a priori* reason to believe that an automated clustering approach would also optimize our metrics. Therefore, we attempted several values of k, until the number of clusters that works best on parameter plausibility is found.

3.2.2 Train KT with Multiple Dirichlet Distributions

After identifying clusters of the skills, for each cluster, skills of the cluster use the same set of Dirichlet priors to initialize their KT models. We used the same algorithm, shown in the previous section, to automatically generate the set of Dirichlet priors for each cluster of skills. The detailed procedure of the algorithm is shown as follows. Some methods used in the algorithm were defined in the previous section.

First, the algorithm trains a KT model for each skill in the data, shown in line 3. Each KT model is fit by the data of a skill and learnt by an EM algorithm, where EM is initialized by fixed priors for each KT parameter, K_0, G, S, L. The fixed priors are obtained by rough estimates of the domain. As a result, for each skill, a set of KT parameters are estimated. For example, if there are *n* skills, there would be *n* sets of KT parameters, i.e. *n* values of K_0, G, S, and L, respectively.

Next, taking those KT parameters as the attribute sets of skills, the algorithm applies the k-means method to cluster those skills. We attempted several successive k values, shown in line 4, from 1 to *n*. When k=1, the algorithm is equivalent to the algorithm proposed before in Section X, shown in line 5-line 6. In other words, a single Dirichlet distribution is assumed. Otherwise, skills are classified into *k* clusters, shown in line 7- line 9, indicating *k* Dirichlet distributions.

Next, for each of the *k* clusters, the algorithm automatically generates Dirichlet priors only using KT parameters of the skills of that cluster, shown in line 10-line 16. Using the generated Dirichlet priors, new KT models are trained for skills of that cluster, shown in line 17. Therefore, for skills in different Dirichlet distributions, separate Dirichlet priors are applied. We determine the maximum value of k, *n*, by observing the changes of parameter plausibility between iterations. When the algorithm becomes convergence, the algorithm halts by our intervention.

The algorithm of using the multiple automatic generated Dirichlet priors to train KT

```
1: Let D[] denote the training data, D[i] denote the i^th skill's data in
    D, K0[] denote the parameters of prior knowledge, G[] denote the
    parameters of guess, S[] denote the parameters of slip, L[] denote
    the parameters of learning and n denote the number of clusters.
2: def train_KT_by_multiple_Dirichlets()
3:   {K0[], G[], S[], L[]} = train_KT(KT, D[], fixed_priors[])
4:   for k=1 to n do
5:     if (k == 1)
6:        cluster[] = k-means({K0[], G[], S[], L[]}, k)
7:     else
8:        cluster[1] = { K0[], G[], S[], L[]}
9:     end if
10:    for j=1 to k do
11:      {K0'[], G'[], S'[], L'[]} = {K0[], G[], S[], L[]} in cluster[j]
12:      data = the corresponding D[]s in cluster[j]
13:      for each param[] in {K0'[], G'[], S'[], L'[]}
14:        param[] = K0'[]                  // take K0' as an example
15:        Dirichlet_priors = auto_gen_Dirichlet(param[])
16:      end for
17:      {K0[], G[], S[], L[]} = train_KT(KT, data, Dirichlet_priors[])
18:    end for
19:  end for
20: end
```

It is important to know that automatically generated Dirichlet priors might be hurt by the outliers with extreme values. Since similar to calculating the arithmetic mean, outliers might distort the parameter estimates. In this study, we trimmed the data for lowering the impact of extreme values on calculating

Dirichlet priors. It's worth emphasizing that trimming was only applied for calculating Dirichlet priors. In other parts of the algorithm, we used original data.

We trimmed data in two ways after obtaining KT parameters from models initialized by fixed priors. First, for each of the KT parameters, 5% largest values and 5% smallest values were trimmed. Note that trimming was done separately for each parameter. For example, the learning rate of Pythngorean Theorem, 0.0001, is in the lowest 5% so is screened out. Meanwhile its prior knowledge of 0.45 could be believed as a normal value, thus is maintained. Second, in each cluster, bottom 10% skills with largest distances from the cluster centroid were also removed.

3.3 Results

For this study, we used data from ASSISTments. The data are from 345 twelve- through fourteen- year old 8th grade students in urban school districts of the Northeast United States. These data consisted of 92,180 log records of ASSISTment during Dec. 2008 to Apr. 2009. Performance records of each student were logged across time slices for 105 skills (e.g. area of polygons, Venn diagram, division, etc).

We used BNT-SM (Chang, Beck et al. 2006) to apply the EM algorithm to estimate the KT model's parameters. We focused on parameter plausibility. The metrics used for measure models are the same as the two used in the previous section: number of skills which require extreme number of practice opportunities until mastery and the correlation between model-estimated K_0 and pretest-assessed student prior knowledge. We compared models initialized with fixed priors, a single set of Dirichlet priors, and multiple sets of Dirichlet priors.

Table 3 shows the comparisons of models, based on the first metric. We found in the two cases, extremely- slowly-learnt skills and extremely-fast-learnt skills, the performances of models are inconsistent. The model with fixed priors generated fewer skills mastered extremely slowly, while the other models with Dirichlet priors produced 5-6 more. It is worth pointing out that the skills found to be slowly mastered by the fixed model is a subset of those found by the other three models. Furthermore, the skills with low mastery rates found by the three Dirichlet models have high overlap. In the other extreme case, models with Dirichlet priors produced no skills with extremely high mastery rate, while the model with fixed prior resulted in slightly more.

Table 3 Comparison of extreme number of practice until mastery

	# of extremely-slowly-learnt	# of extremely-fast-learnt skills
Fixed prior	22	2
Single Dirichlet	28	0
2 Dirichlet Distr.	27	0
3 Dirichlet Distr.	27	0

Figure 6 shows the comparisons of correlations between student pretest scores, an external standard that measures student prior knowledge, and the parameter estimates, K_0, from the models.

Since we classified skills into k clusters for calculating their own Dirichlet priors and skills of each cluster were trained separately using their own Dirichlet priors, it is a fairer comparison if skills of each of the k clusters can be trained separately using their own fixed priors as well. In this way, the same granularity of training is guaranteed, so any difference in parameter plausibility between using fixed priors and Dirichlet priors would be due to the difference of priors. In detail, after line 11 of the pseudo code, for the j^{th} cluster, we calculated the mean of each KT parameter, and used the mean as the fixed prior to re-train KT models for skills of the j^{th} cluster. We compared the results of using fixed priors with the results of using Dirichlet priors.

First, more Dirichlet distributions generally resulted in higher plausibility of the student knowledge parameters. Both lines of the Dirichlet and Dirichlet-trimming models have the up-going trend. The

correlation values above 0.88 are significantly higher than the baseline value 0.83 from the fixed prior model with p-values < 0.05. It suggests classifying students in a fine-grained level provides the models more confidence about the distributions where the data are from, thus taking the extra information specified by the Dirichlet priors, the models produce more plausible parameter estimates. We also found that with more clusters, the correlation values dropped. It suggests that with too stronger bias, parameter estimates are skewed towards the mean too much to learn parameters which can reflect the original data.

Second, the results showed the evidence of the automatic generated Dirichlet priors being hurt by extreme parameter values of outliers. In the case of one cluster, i.e. all data were fitted by models using the same priors, the Dirichlet model using Dirichlet priors produced lower correlation (0.80 vs. 0.83) compared to the fixed prior model. However, the Dirichlet-trimming model catches up the fixed prior model, indicating the necessity of trimming for Dirichlets. However, the advantage from trimming decreases as the number of cluster increases, until eventually the untrimmed Dirichlet has better performance. Thus, the power from trimming is reduced as presumably the higher similarity of the students in a distribution reduced the problem of outliers.

Finally, the results showed that increasing plausibility is not simply a result of having multiple distributions; rather there is an interaction effect between multiple distributions and the use of Dirichlet priors. The figure shows a series of correlation values, corresponding to multiple distributions + fixed priors, the line with spade. We see that fixed prior models performance is independent of the number of distributions (except for possible over-fitting with 6 distributions). Thus, the improvement from multiple Dirichlet distributions is not an artifact of multiple distributions necessarily resulting in better performance.

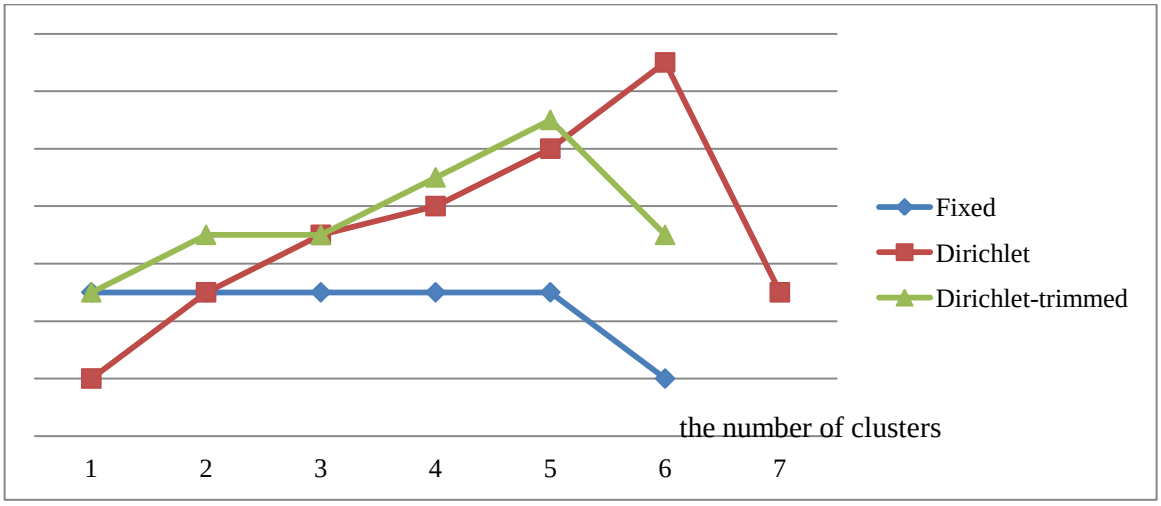


Figure 6 Correlation between prior knowledge and pretest, by number of clusters

Chapter 4. Student Performance Prediction

1. Introduction

Student modeling is a technique used in intelligent tutoring systems to represent student proficiencies and learning. Traditionally, human teachers learn about students through years of experience. They acquire their understanding of students' learning through many ways, such as students' responses, questions and misconceptions, as well as their facial expressions and body language. Similarly, when it comes to a computer tutor, the system needs to be aware of student learning status as well. Student modeling techniques are used to make inferences and predictions as to students. The applied model assesses students in real-time and supply knowledge to other tutor modules, particularly the teaching module, so as to enable the system to respond effectively, engage students' interest and promote learning.

Aside from being used to track students while they interact with the system, student modeling techniques can also be used to obtain scientific insights about student learning. Student models typically produce parameter estimates after being trained on a great amount of data. Most of those parameter estimates are semantically meaningful. They may capture impacts of some student behaviors, or reflect probabilities of certain actions. Therefore, parameter estimates being interpretable and plausible is fundamental, as through interpreting them, researchers could understand students, such as their level of knowledge, interests, preferences, stereotypes, etc.

Towards these two main usages of student modeling, a student model can be evaluated by two measures, predictive accuracy and parameter plausibility. Each of the measures captures the goodness of a student model of one aspect. This dissertation work focuses on improving a student model, in both predictive accuracy and parameter plausibility. The corresponding contents are presented in this and next chapters, respectively.

This chapter focuses on the task of prediction. In particular, the prediction is for one of the most important student behavior: student performance on the next problem. In this chapter, I first introduce two popular student models and present related work on improving predictive accuracy of student models. Next, in Section 3, I analyze the existing models and conduct studies in order to find out which features can inform an accurate model. In Section 4 and 5, I analyze the shortcomings of student models, and towards them I propose two approaches to improve the model's predictive accuracy.

Predicting student behaviors is a very important task for computer tutors. Accurately predicting student performances enables the tutor to be aware of a student's mastery status, so that the tutor can determine the necessity of more practice (Koedinger, Anderson et al. 1997). By accurately assessing student bad behaviors, such as "off-task" or "abusing help", the tutor is better able to intervene at the right time and place so as to decrease student disengagement (Baker, Corbett et al. 2006).

Student modeling plays a key role in prediction and further drives decision-making in computer tutors. The model in use should be able to accurately predict a student's *individual* behaviors at the problem level. In particular, according to the information collected so far, the model should be able to make a prediction on how the student will behave in the very next problem. Aside from predicting seen students' behaviors, the model is also required to correctly respond to new students, who have no historical data to inform the model. This requires the model in use has ability to be generalized across populations.

Unfortunately, predicting individual trials is a difficult task with model-fit statistics generally being fairly low. Taking R^2 as the metric, for predicting student individual correctness, we have found R^2 values ranging from 7.2% to 16.6% ((Gong and Beck 2011), (Gong, Beck et al. 2010)) on data sets from different computer tutors using common student modeling approaches. This lack of model fit is not specific to our data; psychology studies predicting student individual response time, a continuous value and thus easier to see incremental improvements in performance, R^2 values ranged from 5.4% 67.9% (Heathcote, Brown et al. 2000) on 40 sets of data representing learning series. Most existing student models fail to produce satisfyingly high predictive accuracy ((Baker, Pardos et al. 2011), (Gong, Beck et al. 2010)).

The knowledge tracing model (KT), which emerged over a decade ago, has been established as a standard to evaluate new models and being used in real application. Even being such a classic model, KT has been shown, by studies on a variety of data sets sampled from different populations, to have predictive accuracy generally between 0.65 and 0.70 in AUC (Area Under the Curve) of the ROC (Receiver Operating Characteristic) curves ((Baker, Pardos et al. 2011), (Gong, Beck et al. 2010)).. More frustratingly, although there have been a number of efforts dedicated to improving accuracy, none have dramatically improved model fit.

One class of efforts, which attracts a large amount of attention, is tweaking existing models ((Baker, Pardos et al. 2011), (Pardos and Heffernan 2010), (Pardos and Heffernan 2011), (Baker, Corbett et al. 2008), (Xu and Mostow 2011)). In the evaluations of predicting unknown students' step-level performances, these models generally performed similarly to the original KT, and some even underperforming KT. Several papers have reported performance improvements in terms of AUC. The prior per student model, enhancing KT by incorporating individualization, resulted in an improvement of 0.007 (Baker, Pardos et al. 2011). The contextual guess and slip model, fitting KT by contextually-computed *guess* and *slip* parameters, resulted in negative improvement of -0.21 (Baker, Pardos et al. 2011).

Another class of efforts, which is relatively fewer, is to construct new modeling approaches. Performance Factors Analysis (PFA) is an alternative of KT (Pavlik, Cen et al. 2009). However, its predictive performances relative to KT varied. Gong, et al. (Gong, Beck et al. 2010) found that PFA worked substantially better than KT, on a data set from ASSISTments, with 0.071 gains in AUC. Baker et al. (Baker, Pardos et al. 2011)found the model did not perform as well as KT, about 0.033 worse in absolute in AUC, on a data set from Cognitive Tutors. Therefore, it seems that attempts on building new models have not resulted in clear and consistent improvement.

2. Analyze Student Models: Determining Sources of Power in Understanding Student Performance

2.1 Methodology

To improve a student model's predictive accuracy, I start with analyzing existing student models so as to understand what information could possibly inform a student model and enable it to result in accurate prediction of student performance. In particular, I want to determine sources of power in understand student performance. I break a student model down and inspect its individual components (in the rest of the proposal, "predictor" and "feature" are conceptually equivalent to "component") to understand which component is essential to an accurate model of student performance. The PFA model was selected as the framework for this analysis.

Many student model components could be important, in terms of enabling a student model to achieve high accuracy in predicting a student response's correctness for a problem. I choose to examine three: 1) student proficiencies on required skills, 2) problem difficulty and 3) skill difficulties, as those are the most commonly used components across different student modeling techniques. I detail each of them in the following.

1) Student proficiencies on required skills

This feature is widely used in many student modeling techniques ((Cen, Koedinger et al. 2006), (Corbett and Anderson 1995), (Pavlik, Cen et al. 2009), (Pavlik, Cen et al. 2009)). Required skills of a problem are indicated by a transfer model. A transfer model is a cognitive model that contains a group of knowledge components and maps existing questions to one, or more of the knowledge components model (Croteau, Heffernan et al. 2004). For instance, based on our transfer model, the original question of the Assistment shown in Figure 1 was tagged with 3 skills (Congruence, Perimeter, and Equation-Solving). Since the transfer model is responsible for providing which skills are required to solve the problem, we refer to "using student proficiencies on required skills to predict" as "using transfer models to predict".

The transfer model is often treated as the primary component in student modeling, so is the first component we considered.

Our question was simple: how much variance do transfer models account for? Specifically, how much can a model's predictive accuracy benefit from observing a student's prior performances on required skills? To answer this question, we designed a model that solely considers student proficiencies on the transfer model. We trimmed the PFA-item model and dropped its predictor of item difficulty (β_q), from Equation 1, as item difficulty has nothing to do with the transfer model. As a result, the new model has student performances on a series of question as the single predictor, so the only variable predicting the possibility of a student's correct response is his proficiencies on required skills.

2) Item difficulty (question difficulty)

This feature has been less studied in student modeling. Considering that it is used in Item Response Theory (IRT) (Embretson and Reise 2000), a generally effective technique for assessing students ((Desmarais 2011), (Hernando 2011)), we think of it as reasonable to infer that item difficulty is an important predictor of student performance.

Item difficulty hasn't been widely used in student modeling until recently when the PFA-item model was proposed (Pavlik, Cen et al. 2009), as well being integrated into Knowledge Tracing in order to better predict student performance (Pardos and Heffernan 2011). Hence, in student modeling, there were few attempts for exploring the ability of item difficulty to accurately predict student performance.

Similar to how we test the effect of the transfer model in isolation, in order to test the effect of item difficulty we modify the PFA-item model by dropping the part corresponding to student proficiencies (the part inside the Σ in Equation 1). So the model only has the parameter β_q. Since the model has excluded other features, it can be used to discover the pure ability of item difficulty to contribute the model's predictive accuracy.

3) Skill difficulties

This feature is also not commonly used. Only Learning Factors Analysis (Cen, Koedinger et al. 2006) uses skill difficulty in the model. Since the PFA-skill model was reconfigured based on the LFA model, it inherits this feature. To examine skill difficulties, we built a model based on the PFA-skill model (Equation 2) and removed the part corresponding to student proficiencies. Only the skill difficulty parameter (β_j) after the sigma sign is left to capture the effect of the required skills for the question.

2.2 Data Pre-processing

The data used in this study are a portion of the algebra-2005-2006 development data set for the KDD cup competition 2010 from the Cognitive Algebra Tutor. Since the original data set is very large, to form our working data set, we randomly selected 74 students and their performance records, 94,585 steps completed by the students. We don't have access to the transfer model used in this data set. Thus for determining which skills are required in a question, we directly used the skill labels given in the data. There are a number of questions that do not specify which skills are required to solve them. For those questions, we removed them from the data set. Therefore, in the remaining data set, there are 117 algebra skills, including: Addition/Subtraction, Remove constant, Using simple numbers, Using small numbers, etc.

With respect to modeling item difficulty, we were forced to make a compromise when designing the models. Due to a characteristic of the Cognitive Tutor data, it is not sensible to use the question's identity. In the Cognitive Tutor, a question can have multiple steps, each of which typically requires different skills. Therefore, in the Cognitive Tutor, if a question identity occurs multiple times in the student performance records, we cannot simply assume that they concern the same question. For example, a record might be the first step of a question, while another record with the same question identity might be the tenth step of the question. The difficulties of the two steps are probably not the same as they involve different skills and different aspects of the question. For modeling skill difficulty, there is no difficulty, but it presents clear problems for modeling item difficulty. A solution is to build a new question identity combining the

original question identity and the skills required in a step (Pardos and Heffernan 2011). For instance, if the original question id is Q1 and the first step of the question requires "Addition", we can build a new question id, Q1-Addition; while if the tenth step requires "Using small numbers", we have another question id, Q1-UsingSmallNumbers. However, this way results in a very large number of question identities, over 8000 in our data, and it causes a severe computational problem for logistic regression and an inability to fit the model within SPSS, even with increased memory. Therefore, we made a pragmatic decision: for each step, we represented its difficulty using the summation of the difficulty of the original question and the difficulties of the required skills in that step. In this way, the computational cost is greatly reduced and an approximate difficulty for the step can be estimated. The corresponding equation is shown Equation 3.

$$m(i, j, q \in questions, s, f) = \beta_q + \sum_{j \in required\_skills} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}) \quad \textbf{(3)}$$

2.3 Experiments

For all studies in this chapter, including the ones presented in this section and in Section 4 and Section 5, we did 4-fold cross-validation at the level of students, and tested the models on held-out students. We chose to hold out at the student level since it results in a more independent test set. We focused on a student model's accuracy in predicting those held-out students' performances.

All work in this chapter focus on predictive accuracy. Predictive accuracy is the measure of how well the instantiated model fits the test data. For studies in this chapter, we used two metrics to examine the model's predictive performance on the test data set: Efron's R^2 and AUC of ROC curve (Area under the curve of Receiver Operating Characteristic). Efron's R^2 is a measure of how much error the model makes in predicting each data point, compared to a model that uses the mean of the those data to predict. A 0 indicates the model does no better than simply predicting the mean; a 1 indicates prefect prediction. A negative value of Efron's R^2 indicates that the model has more error than a model that just simply guesses the mean for every prediction. AUC of the ROC curve evaluates the model's performance on classifying the target variable which has two categories. In our case, it measures the model's ability to differentiate students' positive and negative responses. AUC of 0.5 is the baseline, which indicates random prediction.

When presenting results in this chapter, we report the comparative results by providing the R^2 and AUC measurements across all four folds. To test the differences of the means, we also performed paired two-tailed t tests using the results from the cross-validation with degrees of freedom of N-1, where N is the number of folds (i.e. df=3).

As all the experiments in this chapter were designed in the same way, in the next two sections, I skip how to conduct experiments

2.4 Results

We examine the predictive power provided by different student model components, including item difficulty, skill difficulty and student proficiencies on the skills in the transfer model. Since each of our models only consider a single feature, the results of testing the model can be attributed to that component.

Table 4 shows the comparative results of models, each of which was fit by a single student model component. First, we found that compared to the other student model components, the model using item difficulty results in higher predictive accuracy and the differences in the means are significant. In the comparison of item difficulty vs. skill difficulty, the t-tests resulted in p=0.02 in R^2 and p=0.005 in AUC. In the comparison between the model using item difficulty and the model using transfer models, the t-tests yielded p=0.006 in R^2 and p=0.48 in AUC. The p-value in AUC suggests that there is not enough evidence to show that the two models have different classification abilities for the student performances,

while the predictive error made by the model using item difficulty is significantly smaller than its counterpart.

Table 4 Comparative performance on unknown students

Student model component	R^2	AUC
Item difficulty	0.149	0.739
Skill difficulty	0.139	0.720
Student proficiencies on the transfer model	0.132	0.738

The results concerning item difficulty suggest that contrary to the traditional belief that student proficiencies on the transfer model (required skills) are the most important predictor; instead item difficulty is an even more powerful predictor of student performance. This finding is also consistent with the finding in the study using the data gathered from ASSISTments (Gong and Beck 2011), suggesting that item difficulty can cover more variance of student performance is a general phenomenon across different computer tutors and different populations.

Table 4 also shows the results of comparing skill difficulty and student proficiencies on the transfer model. The results of the two metrics do not agree with each other, but both differences are found to be reliable: $p=0.03$ in R^2 and $p=0.02$ in AUC; therefore, it is still uncertain about whether skill difficulty or student proficiency is more important for predicting student performance.

3. Modeling Student Overall Proficiencies to Improve Predictive Accuracy

3.1 Background

We observed that most student models are using transfer models to predict. Using transfer models to predict refers to the use of a specific predictor, student proficiencies on the skills required by the question. Since skills required for a question are designated by a transfer model, the term is also called "student proficiencies on the transfer model".

In theory, cognitive scientists believe that students are learning individual skills, and might learn one skill but not another (Anderson and Lebiere 1998). In practice, directed by the theory, student model designers believe that when predicting student performance on a question, student proficiencies on non-required skills are having little impact to the target, so often not being considered in a student model. Consequently, most major student models use transfer models to predict.

Specifically, the knowledge tracing model (Corbett and Anderson 1995) uses student performance to estimate student knowledge, and based on the estimated knowledge to predict student future performance. The KT model has no ability to handle multi-skill questions, i.e. a question requires multiple skills to be answered correctly. Naturally, the model uses a series of student historical performances on a single skill as the observations to predict a student response to a question requiring the same skill. As a result, the model is never able to see student performances on any other skills, and so characterized as "using transfer models to predict".

Another class of are discriminative models, such as Learning Factors Analysis (LFA)(Cen, Koedinger et al. 2006) and a variant of LFA, Performance Factors Analysis (PFA) (Pavlik, Cen et al. 2009). The LFA model uses transfer models to predict. It counts how many practices a student has done for a skill, and uses the count as a predictor. This predictor captures the effect of the student practicing on the series of problems. The PFA model modifies LFA in tracking the numbers of both correct responses and incorrect response separately. Accordingly, the model estimates the effects of those successful and unsuccessful practices. It is important to know, no matter in LFA or PFA, when they count the number of

practices, the models only consider the number of prior practices on the required skills of the problem to be predicted. Therefore, this class of models is also characterized as "using transfer models to predict".

Using transfer models to predict becomes one of the common characteristics of LFA/PFA and KT, in spite of their markedly different functional forms (logistic regression vs. HMM). Our question arises at this point. We want to give more exploration to the assumption of using transfer models to predict.

The common use of transfer models assumes that student proficiencies on, and only on, the required skills, as specified by a transfer model, have impact on solving the question. Note that the assumption only holds when the following corollary is also true: student performance on the problem is independent of student proficiencies on non-required skills. However, the corollary could fail to be true, perhaps due to the possibility that there are relationships between required skills and non-required skills that are not well captured by the transfer model. Or perhaps problems involve a broader range of skills than the subject matter expert believed and encoded in the transfer model. Therefore, it is reasonable for us to relax the assumption and design a model acknowledging that the probability a student successfully solves a problem might also depend on his proficiencies on skills, which were considered not required in the transfer model. Accordingly, we propose a model where student proficiencies on *all* skills are considered as possibly relevant for making predictions. We refer to student proficiencies on all skills as student overall proficiencies.

Aside from the hypothesis that using transfer models to predict is not sufficient for producing an accurate predictive model, there is another reason for us to believe that incorporating student overall proficiencies could result in higher predictive accuracy. Student overall proficiencies reflects student ability about the domain, and student ability is an important predictor, being used by some student models for producing higher accuracy. LFA has an independent variable to capture student ability by estimating a parameter for each individual student based on examining the student's overall proficiencies. An individualized knowledge tracing model was proposed recently. It enhances the traditional knowledge tracing model by considering student's individual difference and leads to higher predictive accuracy than the classic KT model (Pardos and Heffernan 2010). Thus, it appears that considering the student's individual ability is reasonable to other researchers. Since student proficiencies across all skills is a reasonable proxy for student ability, we suspect it will likewise be a useful predictor. In a sense, it is reasonable to assume that an overall stronger student is more likely to produce a correct response than a weaker student, even if neither has practiced the skills required for the problem.

Moreover, it is worth pointing out that there is a thorny problem with the approaches that utilize an explicit parameter to represent student ability (such as LFA): in those approaches, a student's ability is represented as a specific value based on examining all of the student's performances, so the value cannot be applied to a new student. This leads to the model's lack of ability to adapt to new incoming students. Nevertheless, the requirement of handling new students is not negligible in applications of intelligent tutoring systems, as findings should generalize to new students. Our model can accommodate new students as, rather than trying to estimate student ability, it instead estimates the *effects* of student proficiencies on all skills. Therefore, it is able to reuse those estimated effects when making predictions for new students. In this way, since the student parameter is no longer necessary, the model doesn't require peeking into the future at all of the student's performances.

3.2 Approach

We used the performance factors analysis as our framework, for the reason that it has been shown to work well on our data (Gong, Beck et al. 2010), as well as it takes the form of logistic regression, so it is straightforward to incorporate more (or different) variables.

3.2.1 The Overall Proficiencies Model

The overall proficiencies model is built based on the assumption that student proficiencies on certain specific skills are not more important than his overall proficiencies. We reconfigured the PFA model's

predictors, keeping question difficulty, yet replacing the student proficiencies on required skills to those on all skills. Its formula is shown as follows, to contrast to the formula of PFA, shown in Equation 1.

$$m(i, j, q \in questions, s, f) = \beta_q + \sum_{j \in ALL\_KCs} (\gamma_j s_{i,j} + \rho_j f_{i,j}) \qquad \textbf{(4)}$$

The skills taken into account by the model differentiate our proposed model from the original PFA model (note: the set which skill *j* is drawn from—all KCs vs. required KCs). In this new model, student proficiencies on all skills are believed to have effects on student performance. This modification enables the model to break the limitations due to the potential failure of the assumption underlying transfer models, namely that student performance is independent of non-required skills. Furthermore, it also incorporates student overall ability as a predictor of student performance.

Table 5 shows the factors used in the PFA model and the overall proficiencies model. Suppose there are two skills in the data set. Table 5 shows a sequence of performances, extracted from the middle of the input file. These questions are answered by a single student and organized in chronological order. In each row, the counts of prior correct responses and incorrect responses, achieved by the student in the past for the corresponding skills, are shown in the last four columns.

In the PFA model, the counts for a skill are only non-zero when that skill is required in the question. Consequently, as a correct data format for the PFA model, all the cells with two numbers separated by a slash should be set to 0s (the number preceding the slash), as the transfer model does not believe performance on that skill impacts performance on the question. For example, in the second row, even though the student has generated 5 correct and 3 incorrect responses for skill 1 in the past, when the model deals with the question with ID = 53, since this question requires no ability about skill 1, the student proficiency on skill 1 is ignored, thus two zeros should be assigned for the number of prior success and failures (columns 4 and 5). In this way, the model follows the assumption of using transfer models to predict: student proficiencies on non-required skills are irrelevant.

In the overall proficiencies model, the data format is different from that of the PFA model in using the underlined values to the right of the "/" in those cells with two numbers—it considers a student's historical performances for all skills.

Table 5 Input data formats of the PFA model and the <u>overall proficiencies model</u>

Question ID	skills	correct	prior successes skill 1	prior failures skill 1	prior successes skill 2	prior failures skill 2
1004	1	Yes	4	3	0 / <u>10</u>	0 / <u>4</u>
53	2	No	0 / <u>5</u>	0 / <u>3</u>	10	4
5	1,2	Yes	5	3	10	5
214	2	No	0 / <u>6</u>	0 / <u>3</u>	11	5

3.2.2 A Hybrid Model – The Overall Student Proficiencies Model Emphasizing the Transfer Model

The original PFA model solely pays attention to the skills in the transfer model, as it follows the assumption that student proficiencies on non-required skills are not helpful. The overall proficiencies model takes the opposite approach and makes no assumption about which skills are more important for a particular problem. Compared to the well-established models, this model acknowledges the effects of student overall proficiencies, yet overlooks the importance of transfer models in prediction. Ignoring the transfer model could be an issue, as empirically almost all existing student modeling techniques make use of it, suggesting its effectiveness in prediction. Furthermore, it is reasonable to believe that student proficiencies on those required (at least according to the transfer model) skills would be more important predictors than an average skill. Towards this issue, we designed a hybrid model which considers both student overall proficiencies and his proficiencies on the required skills. The model is built based on the

overall proficiencies model, meanwhile combining the idea of emphasizing the skills noted in the transfer model.

$$m(i, j, k, q, s, f) = \beta_q + \sum_{j \in ALL\_KCs} (\gamma_j s_{i,j} + \rho_j f_{i,j}) + \sum_{k \in required\_KCs} (\gamma'_k s_{i,k} + \rho'_k f_{i,k}) \quad \textbf{(5)}$$

As shown in Equation 5, the first part remains the same as the overall proficiencies model, while the effects of student's proficiencies on skills in the transfer model are included in the second part of the equation. The problem with this model is that when there are a large number of skills, the number of estimated parameters is also very large. There are two parameters for each skill in the original PFA model (γ and ρ), while in this hybrid model the number increases to 4 for each skill (γ, ρ, γ' and ρ'). The first two parameters, γ and ρ captures the effects of practices on a skill, when those practices are treated as evidence of student overall proficiencies, while the other two, γ' and ρ', are corresponding to the effects of student proficiency on the required skill. Considering that if we add additional 2*n (n=# of skills) columns in the input data, most cells in a single row would be 0s, as among n skills, only a small number of skills are required in a question, to reduce the sparseness we compressed the 2*n columns to 2*x columns, where x is the maximum number of required skills of a question across all questions in our data set. For the second part of the model, for each row, the spaces for non-required skills are removed and all the followings are moved forward, until the preceding cell has been filled in and corresponding to another required skill, so that all effective counts are maintained in those 2*x columns.

Table 6 shows the data format under the scenario where there are n skills and at most a question requires x skills. Due space limitations, we use abbreviations for the titles: s-s_1 is short for the number of prior successes of skill 1; the counterpart is f-s_1. Req-s-s_1 is short for the number of prior successes of the first required skill; while for failures, the abbreviation is req-f-s1.

Table 6 Input data format of the hybrid model

Question ID	skills	correct	s-s_1	f-s_1	…	s-s_n	f-s_n	req-s-s_1	req-f-s_1	…	req-s-s_x	req-f-s_x
1004	1	Yes	4	3	…	0	0	8	7	…	0	0
53	2	No	0	0	…	10	4	15	24	…	10	4
5	1,2	Yes	5	3	…	10	5	15	8	…	10	5
214	2	No	0	0	…	11	5	17	8	…	11	5

Note that for those x columns, the counts in a single column could correspond to different skills in different rows. For example, suppose in the first row, the values of 8 and 7 in the cells of req-s-s_1 and req-f-s_1 are of the skill of Addition; in the second row, the values in the corresponding cells, 15 and 24 could be the counts of the same, or any other skill, such as Subtraction, Multiplication, etc. Thus, this model has an issue where the model parameters of γ' and ρ' lose the meanings of the effects of practices on a specific, named skill, but acquires the interpretation of the effects of practices on a skill with a specific position (first, second, third, …).

In order to preserve semantic meaning for a particular position in the table, and thus have interpretable model parameters, we need some way to order the required skills. There are several reasonable approaches we can take. If we assume that in a multiple skill question, all the required skills are equally important in terms of contributing an accurate prediction of student performance, then we could use a random ordering. However, in the case where even if multiple skills are required, if the proficiency on one skill is more important than the others, we could put the more important skill earlier. In such a model, the first skill is the most important, and presumably the most difficult, skill required in the question. To determine difficulty, we could use student initial knowledge of skills, or the grade when the skill is taught, based on the assumption that an easier skill is taught earlier. We used the latter in this study; specifically the highest grade-level skill is req-s1, the second highest level skill is req-s2, etc. Our subject matter expert provided, as part of the domain model, the grade level where different skills are typically introduced. Thus, the coefficient for req-s1 is not interpretable in terms of a particular skill, but instead refers to the impact of the most advanced skill related to the problem.

3.3 Data and Results

This study used data from ASSISTments, a web-based math tutoring system. The data are from 445 twelve- through fourteen- year old 8th grade students in urban school districts of the Northeast United States. They were from four classes. These data consisted of 113,979 problems completed in ASSISTments during Nov. 2008 to Feb. 2009. Performance records of each student were logged.

It is worth pointing out that the results of this study might be sensitive to the transfer model we used. Imagine that if the transfer model has many mistakes in associating skills to questions, it could lead to opportunities for the all skills or hybrid models being a better classifier than the original PFA model built with student proficiencies on the skills in the transfer model. Therefore, in order to reduce the possibility of using a poor transfer model, we used two transfer models with different grain sizes. The fine-grained transfer model has 104 math skills, including area of polygons, Venn diagram, division, etc. The other has 31 coarser math skill categories, such as Data-Analysis-Statistics-Probability: understanding-data-generation-techniques, Data-Analysis-Statistics-Probability: understanding-data-presentation-techniques, Geometry: understanding-polygon-geometry, etc. It is much less likely for a problem to be mistagged in the coarse- than in the fine-grained model since there are fewer possible skills with which to tag it.

A source of bias could be how affected our data are by the transfer model itself. For example, if ASSISTments is making pedagogical decisions based on the transfer model, it could impact how students perform. For this dataset, ASSISTments did not make use of the transfer model for any adaptation techniques (e.g., no mastery learning, although this feature has been since added to ASSISTments). For this study, the only way the transfer model was used was to group questions into problems sets that contained related questions. The impact of such problem grouping is probably minimal, as it is also the most common method of assigning math problems to students both in computer tutors and for school work.

We did a 4-fold cross validation at the level of students, and tested our models on unknown students. We report the comparative results by providing mean test-set performance across all four folds, and use R^2 and AUC to evaluate.

3.3.1 Student Proficiencies on Required Skills vs. Student Overall Proficiencies

We proposed that estimating the effects of student overall proficiencies might contribute to more accurate predictions. To test that, we compared the proposed student overall proficiencies model against the original PFA model, which, in order to predict student performance on a question, only uses the skills in the transfer model.

Table 7 shows the comparative results with the models sorted by predictive accuracy. For the models using the coarse-grained transfer model, the results in the first and the fifth rows, the mean values of the two metrics suggest that the overall proficiencies model is superior to the PFA model. The t-tests yielded p values for R^2 and AUC less than 0.005, indicating that the differences are reliable.

Table 7. Comparisons between the original and our proposed PFA models

	Transfer model	Overall proficiencies	Grain Size	R^2	AUC
PFA-Coarse	Yes	No	Coarse	0.162	0.740
PFA-Fine	Yes	No	Fine	0.167	0.745
Overall proficiencies-Fine	No	Yes	Fine	0.181	0.756
Hybrid-Fine	Yes	Yes	Fine	0.189	0.760
Overall proficiencies -Coarse	No	Yes	Coarse	0.191	0.762
Hybrid-coarse	Yes	Yes	Coarse	0.194	0.763

For the models using the fine-grained transfer model, the second and third rows, the overall proficiencies model seems to outperform the PFA model in both metrics, but we failed to find any reliable differences between these two models, even though there is a suggestive trend in the mean values that the proposed model is probably better than PFA. We have encountered this problem previously (Gong, Beck

et al. 2010), as the issue is one of relatively low statistical power of the t-tests, as we only have four independent observations (one for each fold of the cross validation).

Given that the statistical tests might not be sensitive to detect differences due to small number of observations, increasing the sample size is a cure. We grouped the measurement values from the models with fine and coarse grain size together. For instance, for the R^2 values, the number of observations increased to 8 (4 from each model). Taking the 8 observations, we were able to conduct paired two-tailed t-tests (df=7) with a larger sample size. The p values of 0.005 in R^2 and 0.001 in AUC suggest that the overall proficiencies model is reliably better.

One interesting pattern in the data is summing the R^2 values of the Question Difficulty and Transfer Models in Table 4 is approximately equal to the R^2 of a model that uses both components (as seen in the second row of Table 7 for the fine-grained PFA model and the first row for the one using coarse granularity). With the fine-grained model, 0.101+0.075=0.176 is fairly close to 0.167, while for the coarse-grained model, 0.101+0.061=0.162 equals to that of the PFA model. This fact suggests that the variance covered by question difficulty and the variance covered by the transfer model contain little overlap. In other words, estimating question difficulty can provide unique coverage of variance in student problem-solving performance.

3.3.2 A Hybrid Model: Combining Overall Proficiencies and Transfer Models

Our results showed that the overall proficiencies model is reliably more accurate than the original PFA model. However, the overall proficiencies model treats skills that are peripherally related to solving the problem as having equal importance as those most likely to be helpful in solving the problem. Since focusing on relevant skills might be able to improve model accuracy, we combined the transfer and all proficiencies into a hybrid model.

We compared the overall proficiencies and the hybrid models, showing the results in the last four rows of Table 7. For both model granularities and for both performance metrics, the hybrid model is more accurate on unknown test data. P-values from paired two-tailed t-tests confirmed that the differences are reliable: p=0.043 in R^2 for the fine-grained transfer model, while the value of the coarse-grained model is 0.01. P values in AUC for both comparisons are both less than 0.005.

It is worth noticing that the improvement from incorporating transfer models into the overall proficiencies model is fairly small, less than 1%. Thus, once the model knows question difficulty and student overall proficiencies, student proficiencies on required skills contain little predictive power in terms of modeling student performance. Therefore, we question whether student proficiencies on required skills in the transfer models are overrated in the traditional student modeling approaches.

4. Modeling Multiple Distributions of Student Performances to Improve Predictive Accuracy

4.1 Background

Our prior work examined KT and PFA, two popular student modeling techniques(Gong, Beck et al. 2010). When visualizing their classification performances in confusion matrices, we found a common characteristic of both: a large number of false positives in the confusion matrix. A confusion matrix, seen in Table 8, is a generic metric used to visually understand a classifier's misclassifications. It summarizes the number of instances predicted correctly or incorrectly by the classification model. It has four elements: true positive (TP), false negative (FN), false positive (FP) and true negative (TN). Traditionally, for binary classification, the rare class is often denoted as the positive class, while the majority class is denoted as the negative class (Tan, Steinbach et al. 2005). In our case, however, the class of correct student performances is denoted as the positive class, as conveys more semantic meaning (i.e., positive indicates the student responded correctly).

Table 8. The confusion matrix of PFA

		Predicted class	
		Positive	Negative
Actual class	Positive	16206 (TP)	2399 (FN)
	Negative	5899 (FP)	3965 (TN)

Table 8 shows the confusion matrix of the PFA model on the data set used in the previous (Gong, Beck et al. 2010) and this study. There are two types of errors: false positive and false negative. The bottom-left cell, FP, corresponds to the number of incorrect responses wrongly predicted as correct (5899) by the classification model; while FN, 2399, denotes the number of correct student responses misclassified as incorrect by the model. Consequently, FP is much higher than FN. We also found this trend to be true for KT, as well as for KT's and PFA's variants (Gong, Beck et al. 2010). This result inspired us with an idea that a more promising move for improving accuracy could be to reduce FP, as FP has larger room to work on than FN.

One thing worth pointing out is that we acknowledge that the high FP we observed was possibly due to the specific data set used in the study. In particular, the data set does have imbalanced class distributions, where the class of correct responses was the majority. In the opposite case, the model would tend to generate prediction biased towards an incorrect response, and so would produce a high FN instead. However, the phenomenon that correct responses are the majority is not unique to our data set, but is fairly common in most of the student performance data sets that are being used in the field (e.g. (Baker, Pardos et al. 2011), (Pardos and Heffernan 2011), and (Pavlik, Cen et al. 2009)). This imbalance makes sense, as in most learning environments students will get more than half the items correct in order to prevent frustration. Consequently, we believed that placing our efforts on decreasing FP is meaningful.

We used PFA, rather than KT, as the modeling approach for this study. The rationale is that we have observed that PFA has been the most accurate at predicting student step-level performances on our data (Gong, Beck et al. 2010). Using this model prevents the improvement, if found in this study, from being attributed to a less fair comparison, where a weak model is used as the baseline.

4.2 Approach

4.2.1 Rationale: Modeling Multiple Distributions of Student Performances

We have established our goal as reducing the error rate, by reducing the FP rate, of student models. In order to find a means of *how* to reduce FP, a reasonable first step is to analyze *why*. In particular, what possibly causes high FP?

We hypothesized that high FP could be due to the insufficiency of using a single classification model to classify student performances. We proposed that a solution could be to learn multiple classification models, with the rationale of modeling multiple distributions of student performances (MMD-SP).

Using a single classification model implies that instances were sampled from a single distribution and thus can be modeled with a single classifier. Contrariwise, using multiple classification models assume that instances were sampled from multiple distributions and thus should be modeled separately representing each of the distributions.

If there are multiple distributions, while using a single classification model to fit, then a high false positive is not unexpected. More specifically, suppose we have a naïve student model, where the target is the correctness of a student performance and the only independent variable is the question the student was solving. We then learn a single classification model based on the naïve model. As a result, all instances would be mapped to *correct* or *incorrect* using the same function. As long as it deals with the same question, the model believes that its difficulty perceived the same across all students, even though the

question could be harder to a subgroup of students. If on the question, the majority of student response happens to be *correct*, the model tends to predict correct for every instance of the question. For those students who have high difficulty in answering this question, a false positive occurs.

Therefore, our hypothesis was that due to the possible existence of multiple distributions of student performances, modeling them separately reduces false positives. The pseudo code of implementing MMD-SP is listed in below.

4.2.2 Distinguish Samples of Multiple Distributions

In order to accomplish MMD-SP, we need to first identify samples of each of those multiple distributions. We used the k-means cluster analysis to partition student performances into clusters, each of which represents the sample of a distribution. The corresponding pseudo code is from line 2 to line 12.

We assumed that being sampled from the same distribution, student performances should share common characteristics and be different from student performances from anther distribution. That is to say, student performances from a distribution should be able to form a mathematically meaningful group. We used unsupervised classification as we did not know what characteristic could reasonably feature the groups. We chose k-means because the algorithm is straightforward and prominent for being a beginning clustering method.

Pseudo code of the MMD-SP algorithm

```
1:Let D denote the training data, D[i] denote the i^th
    student's data in D, D[i][j] denote the j^th instance in
    D[i], CM[i] denote the i^th student's confusion matrix, T
    denotes the test data, T[i] denote the i^th student's data,
    T[i][j] denote the j^th instance in T[i], and k denote the
    number of clusters specified.
2:  PFA0 = train_PFA(D).
3:  for i=1 to D.length do (i.e., for each student)
4:    Initialize CM[i]. //CM[i].TN=0,CM[i].FP=0,CM[i].FN=0,
                        CM[i].TP=0
5:    for j=1 to D[i].length do
6:      NCM[i] = normalize (CM[i]).
7:      Attributes[i][j] = {NCM[i].TN, NCM[i].FP, NCM[i].FN}.
8:      apply_PFA(PFA0, D[i][j]).
9:      update CM[i] according to the result from line #8.
10:   end for
11:  end for
12: Clusters[] = K-means(Attributes, k).
13: for c=1 to k do
14:   Dc = instances D[][] ∈ Clusters[c].
15:   PFAc = train_PFA(Dc).
16: end for
17: for i=1 to T.length do
18:   for j=1 to T[i].length do
19:     PFAx=select model from {PFA0...PFAk} for T[i][j].
20:     apply_PFA(PFAx, T[i][j]).
21:   end for
22: end for
```

Choose an attribute set for a student performance.

To classify a student performance, a set of attributes describing that performance is needed. We used normalized confusion matrices. In Table 8, the counts can be normalized, so that all elements of the matrix sum to 1. The proportion of FP in the data is 0.21, FN is 0.08, TP is 0.57, and TN is 0.14.

Rather than using a single confusion matrix to summarize a model's overall classification performance, for each student performance, we calculated a confusion matrix that summarized the model's classification performance so far on that student. More specifically, a base classier, PFA, was induced from training data. Before a student's first instance, the student's confusion matrix is initialized to be four zeros, indicating no observations so far in his TN, FP, FN or TP. Then the algorithm computes the normalized confusion matrix. Since the four normalized values sum up to 1, the dimensions of the attribute set can be reduced to 3, and so we used the tuple <TN, FP, FN> as the attributes of the instance. Then the algorithm applies the base classifier to the instance, resulting in either a TN, FP, FN or TP, and the algorithm updates the confusion matrix to maintain it for use in the next iteration. For example, suppose that our algorithm is about to generate a confusion matrix for the jth performance of the student i. It looks at his performances from 1 to j-1, and calculates the normalized confusion matrix. We use this normalized confusion matrix as the attribute set to perform the clustering.

Although using confusion matrices are an odd choice for features for clustering, it was not a haphazard decision. We chose confusion matrices for two reasons.

First, we prefer generic attributes that require nothing beyond the binary response data normally required to train a student model. Our proposed approach is designed to be widely applicable to solve the problem of high false positives. Using confusion matrices as attributes perfectly matches the approach, as it can be calculated on any sequential user data. Therefore, our approach can be easily applied to any other modeling techniques and data sets, without requiring certain attributes exclusive to a specific data set (such as in (Trivedi, Pardos et al. 2011)).

Second, we think that using confusion matrices as the attributes helps distinguish samples of multiple distributions. A confusion matrix is informative in reflecting the model's performance and capturing a student's proficiency, and thus represents exactly the constructs we are interested in analyzing.

In the aspect of capturing a student's proficiency, a confusion matrix shows how well the student performed previously, and shows which instances the base classifier confuses and how it misclassifies them. For example, if the confusion matrix of an instance shows large FP, it suggests that the instance is not suitable to be modeled by the base classifier; rather it might be sampled from a distribution where the class of negative is the majority, perhaps reflecting a relatively weaker student.

4.2.3 Learn Multiple Classification Models

Applying k-means, we partitioned the training data into K portions, one for each cluster, which presumably represents each of the multiple distributions. Now for each distribution, we learn a separate classification model. The corresponding pseudo code is from line 13 to line 16.

All classification models were learned on the basis of the same approach, PFA. In particular, we fit each portion of the data to a PFA model and learned a classification model. As a result, we had K classification models.

We decided to use PFA as the student model for all classification models, as we wanted to test the effectiveness of the proposed approach, MMD-SP, in isolation. We controlled other factors that possibly result in improvement, especially the use of another student modeling approach that may benefit accuracy improvement. In this way we can ensure that the parameter estimates of K classification models capture differences between different distributions. For example, if a question's difficulty parameter is estimated large by one model, while it is estimated considerably smaller by another model, this could indicate that there are two distributions of student performances that respond to the same question very differently.

4.2.4 Select a Classification Model for an Unknown Instance

For each instance in the test data, we need to estimate from which distribution it was drawn, or equivalently, select the best model to use for predicting this instance. The corresponding pseudo code is

from line 17 to line 22. We implemented two methods for selecting which model to use when making a prediction.

Least distance. We think that a test instance should be similar to the training instances sampled from the same distribution. Following the k-means cluster analysis, the instance should be assigned to a cluster whose centroid is closest to this instance's attributes. We took a similar procedure as we did for the training data. We used the base classifier to generate a confusion matrix for each unknown instance and compared it to each of the cluster centroids. We then selected the classification model corresponding to the cluster having the least distance from its centroid to the instance.

Least error. We select a classification model depending on its error rate. In particular, for an unknown instance of a student, we computed which classifier, so far, has performed the best for this student. Presumably the best-performed classifier should also work best on the current instance. In this method, no confusion matrices are needed.

In addition, to overcome the cold-start problem, for the first three instances of each student, we used the base classifier.

4.3 Data and Results

We used data from ASSISTments (http://www.assistments.org), a web-based math tutoring system,. The data are from 445 8th-grade (generally twelve- through fourteen- year old) students in urban school districts of the Northeast United States. These data consisted of 113,979 problems completed in ASSISTments during Nov. 2008 to Feb. 2009. There are 31 skills involved in the data set, such as Data-Analysis-Statistics-Probability: understanding-data-generation-techniques, Data-Analysis-Statistics-Probability: understanding-data-presentation-techniques, Geometry: understanding-polygon-geometry, etc. ASSISTments logged performance records of each student chronologically.

We did a 4-fold cross validation at the level of students, and tested our models on unknown students. We report the comparative results by providing mean test-set performance across all four folds, and use R^2 and AUC to evaluate.

We evaluate our proposed approach. We compared the predictive accuracy of the multiple classifiers induced by the approach and the predictive accuracy of the base classifier. We used the k-means cluster analysis in SPSS. We used the value of *K* from 2 to 5 without specifying initial cluster centers.

Table 9 Cross-validated of predictive accuracy of the base and multiple classifiers

No. of classifiers	R^2		AUC	
	Least distance	Least error	Least distance	Least error
Base (PFA)	16.2%		0.740	
2	19.6%	20.5%	0.765	0.770
3	19.7%	20.1%	0.766	0.769
4	19.5%	19.8%	0.766	0.768
5	18.5%	19.3%	0.761	0.765

Table 9 compares predictive accuracy of multiple classifiers against the base classifier. The first row shows the predictive accuracy of the base classifier, a single PFA model on the test data. From the second row downwards are the multiple classifiers induced by our proposed approach with the number of classifiers varying from 2 to 5, one for each cluster. In order to address the model selection problem for an unknown instance in test data, we report results for least distance and least error.

We noticed that multiple classifiers induced by our approach all outperformed the base classifier, with a 4.3% absolute improvement in R^2 (20.5% - 16.2% = 4.3%) and 0.03 absolute improvement in AUC (0.770 - 0.740 = 0.03) achieved with the best setting. Based on the paired-sample t-tests (df=3) using the

results from the crossvalidation, all differences in two metrics using multiple classifiers and the base classifier are significant with $p<0.01$.

We also found that the two model selection methods performed fairly consistently. Both resulted in similar predictive accuracy, though using least error to select generally achieved slightly higher predictive accuracy, but not noticeably so. Furthermore, least error is superior to least distance due to its low complexity, as, unlike least distance, this method does not require building confusion matrices for each instance in the test data.

Interestingly, we found that introducing more classifiers does not help for boosting predictive accuracy further. Two classifiers resulted in the peak when using least error, while three classifiers did the best when using least distance. Three possible reasons could cause these results. First, the student performances are not from *many* distributions, but rather from a small number of distributions. Therefore, modeling 2 or 3 distributions of student performances is sufficient, while modeling extra distributions causes over-fitting. Second, the presence of more classifiers confuses the model selection methods, no matter using least error or least distance. Assigning an instance an improper classifier caused the drop of predictive accuracy. The third possibility is we do not have sufficient training data to train 5 classification models with well-estimated parameters. In fact, a classification model may have not seen some questions at all while being trained, and also could be required to predict an unknown instance involving the question. Perhaps more training data would enable the use of additional classifiers? To resolve these issues, we examined the classifiers' predictive accuracy on training data, as shown in Table 10 .

Comparing the two values of the base classifier, 18.0% vs. 16.2%, we found that the PFA model generalizes well to unknown students. We also noticed that the presence of more classifiers doesn't help much on the training data either, with only a 2.4% improvement from using 2 classifiers to using 5 classifiers. Thus the third explanation is not plausible, as sparse training data should produce strong model-fit statistics on the training data. The second explanation is less plausible, since no heuristic is required to determine which cluster to use. Thus, our results suggest that a small number (2 or 3) of distributions, at least as derived by k-means, are the more likely explanation for the asymptote in performance relative to the number of clusters.

Since the accuracy measure treats every class as equally important, it may not be suitable for analyzing imbalanced data sets, such as the one we used. Therefore, we used confusion matrices to evaluate the classifiers derived from our approach. In paper, due to limited space, we chose to show two errors in charts, rather than all elements of a confusion matrix.

Table 10 Comparisons of predictive accuracy of the classifiers on the training and test data

No. of classifiers	R^2	
	Training	Test (least distance)
Base (PFA)	18.0%	16.2%
2	23.2%	19.6%
3	25.1%	19.7%
4	25.5%	19.5%
5	25.6%	18.5%

Figure 7 shows the percent of false positives and the percent of false negatives, generated from the test data and using least error to select the classifiers. The x axes in the two charts represent the number of classifiers. The first one, with label 1, is corresponding to the base classifier, PFA. The y axes in the two charts have the same unit of 1%, so that it is fair to compare the lines across the graphs. The percentages are also listed for each point. Take 21.10% in the first chart as an example, the value indicates 21.10% of the entire data were misclassified as correct responses, while in fact they are incorrect responses. As we can see, the PFA model produced over 2 times ($21.10\% / 8.36\% \approx 2.53$) as much FP as FN. In addition, we showed that our proposed approach, targeting the goal of reducing FP, works fine. Meanwhile, the

error of FN increases much less than the decreases of FP. Finally, we found that from 3 classifiers afterwards, the two errors did not change much. This result suggests there is little benefit to adding additional clusters, as since neither error is improved, ensembling models with more clusters will probably not be beneficial in improving the error rate.

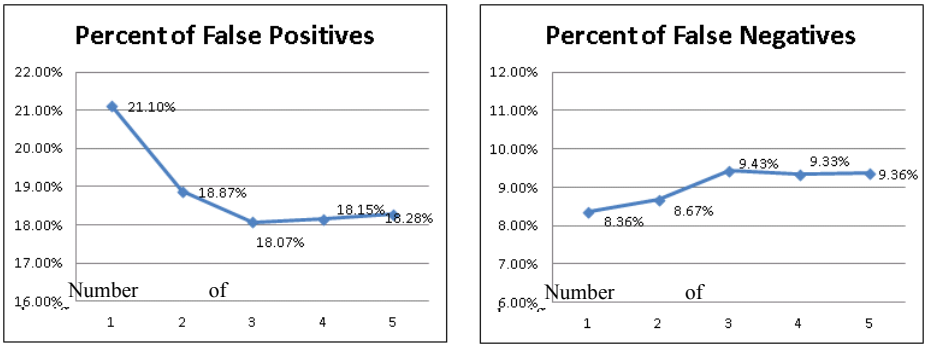


Figure 7 False positive and false negative percentages across a different numbers of classifiers.

Chapter 5. Wheel-Spinning: Student Future Failure in Mastery Learning

1. Introduction

The core of tutoring modeling is its decision mechanism. du Boulay and Luckin pointed out that there are three sources of tutor knowledge that operationalize the computer tutor's decision making: the observation of human teachers, the study of learning theories and the observation of real students interacting with online systems (du Boulay and Luckin 2001). Towards the study of learning theories, a line of research has been conducted and thrived, such as Bloom's mastery learning (Bloom 1984) and Anderson's ACT-R (Anderson, 1993).

Applying those theories, ITS researchers designed and engineered their intelligent tutoring systems. A line of ITSs grounded in the ACT-R theory of cognition bloomed, as the theory is well supported by a rich collection of research and has resulted in an impressive track record of educational successes (Koedinger, Anderson et al. 1997). The ITSs in this series include LISP/Geometry/Algebra Tutors (Farrell, Anderson et al. 1984; Corbett 2001) and Ms. Lindquist (Heffernan 2003). The essence of the ACT-R theory is the distinction between declarative and procedural knowledge – between merely knowing an algebraic rule and being able to apply it in a problem (Nkambou, Bourdeau et al. 2010). It assumes that procedural knowledge can only be acquired with progressive integration, through problem solving, of what is initially declarative knowledge (Anderson, Corbett et al. 1995). This significant assumption characterizes the design of the ITSs backed up by the theory and also leads to a common framework adopted by the ITSs: mastery learning.

There has been a long history of work on mastery learning with computer-based education (Frick 1990; Corbett and Anderson 1992; Shute 1995; MacLaren and Koedinger: 2002; Fancsali, Nixon et al. 2013). The core idea is to trace a student's cognitive step and to assume that the mastery of a skill is able to gradually increase through problem solving. This rationale is also well supported by the theory of "learning-by-doing", the spirit in ITSs' designs and implementations. The concept of "learning-by-doing" refers to the capability of learners to improve their efficiency by regularly repeating the same type of action and the increased efficiency is achieved through practice (Wikipedia). Similarly, in order to achieve increased proficiency, students are required to practice a number of problems in ITSs.

In the systems adopting mastery learning, domain knowledge is typically broken down into a set of skills (some authors use the term *knowledge components* or *topics*; for purposes of this paper these terms may be considered synonyms), and each problem is associated with one or multiple skills. The system presents the student problems as learning opportunities. When the system acknowledges the student's mastery, more practices may not be needed. Therefore, in such systems, the student does not see a fixed number of problems, but continues to solve problems until he achieves mastery of the skill. In other words, the student may possibly see additional problems if he has not yet mastered the skill.

The use of the mastery learning framework is driven by the desire of providing students efficient practice, namely, avoiding giving too many problems to solve, which could waste valuable learning time (Cen, Koedinger et al. 2007; Li, Matsuda et al. 2011; Lee and Brunskill 2012) and possibly jeopardize student motivation to learn, while ensuring there are not too few practice problems, which poorly prepare students for learning future content (Baker, Gowda et al. 2011) due to the lack of mastery.

An application of mastery learning is that if a student is not mastering a skill, he should receive additional practice. Figure 8 shows the simplified mastery learning workflow, which corresponds to how mastery learning is implemented in many ITSs. When a student starts practicing a skill, the tutoring system presents a problem that requires that skill. If the student answers the problem correctly, the system checks whether he has mastered the skill. If so, the mastery learning process for this skill is terminated as the skill is determined to have been mastered. If the student does not respond to the question correctly, the tutoring system typically provides various types of assistance, eventually culminating in supplying the answer to the question, to help the student finish the problem. At this point, presumably the system will decide the student has not mastered the skill (as he responded incorrectly), and so the student will be presented with a new problem as an additional opportunity to master the skill.

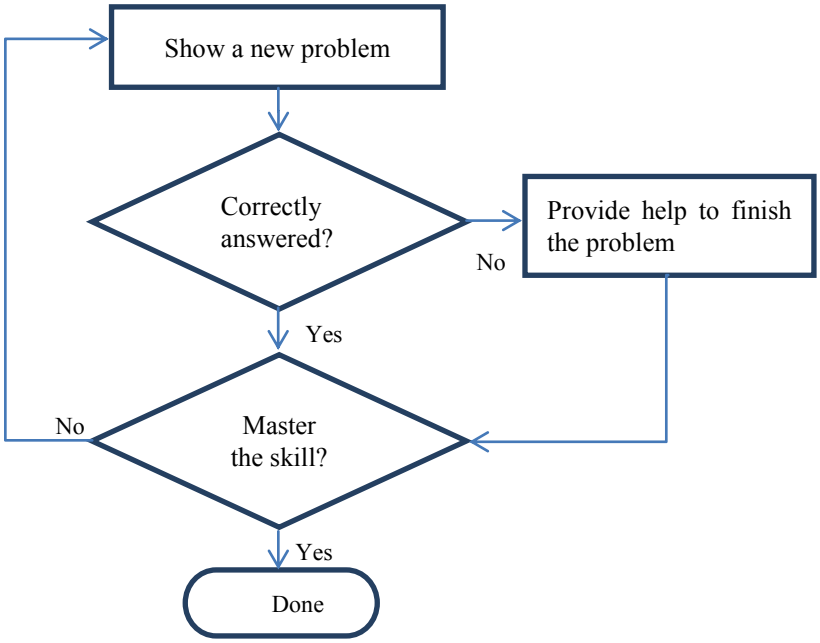


Figure 8. Simplified mastery learning workflow

It seems that the mastery learning framework has been strongly supported, both by prior research work and the theory of learning-by-doing. In this work, we want to point out three reasons with regard to why the model is possibly flawed, theoretically and pragmatically.

First, the ACT-R theory emphasizes that procedural knowledge can be acquired through problem solving. However, we should not neglect that the precondition is that the student has sufficient declarative knowledge (Self 1988; Ohlsson 1996), which is not guaranteed in the mastery learning framework.

Second, pragmatically, the framework's weakness is visible without requiring too much in-depth thinking. If a student requires assistance to solve the first couple of problems, e.g. 2-3 problems, presenting a next problem with the hope that the student will learn the skill could very well be a sensible strategy. Nevertheless, if the student has been unable to solve a number of problems, e.g. 20+ problems, and (or) requested considerable help on them, it is probably rather optimistic to believe that the next problem will enable the student to suddenly acquire the skill (readers will see the this tiny likelihood in the later analyses of the paper)

Third, mastery learning succeeds under the assumption that, given sufficient practice opportunities, the student will eventually manage to learn the skill. This assumption is sound when we can assume a non-zero probability of learning the skill on each problem-solving attempt (Corbett and Anderson 1995). Otherwise, the single termination criterion in this process is problematic: the student can possibly become trapped if he repeatedly fails to achieve mastery. As a consequence, the system keeps giving the student more problems to practice with the hope that he might seize these new opportunities and master the skill. The student however could keep failing the problems or the mastery learning condition, which triggers the system to present more problems to the student.

We refer to this endless loop in the mastery learning cycle as *wheel-spinning*, analogous to a car stuck in mud or snow: its wheels are spinning rapidly, but it is not going anywhere. Similarly, students are being presented with many problems, but are not making progress towards mastery.

2. Research Questions

We are interested in exploring the answers to the following research questions regarding wheel-spinning.

- Research Question 1: What is the wheel-spinning problem? How to instantiate the definition, particularly in the context of particular tutoring systems?

Wheel-spinning could be a consequence of a simplified mastery learning framework in many computer tutors. The details of how mastery learning is instantiated influences our observation of wheel-spinning. Given a tutoring system, we need a means to find out how the students approach mastery and meanwhile how they could possibly be trapped by wheel-spinning. Attempting to answer this research question, we will establish an approach to concretizing the wheel-spinning problem from real tutor log data.

- Research Question 2: What is the scope of the problem? Is wheel-spinning a real problem worth our attention?

Estimating the scope of the wheel-spinning problem is very important to us before any further attention or efforts are suggested. We will approach this question from two aspects: the percent of student-skill instances which led to wheel-spinning and the actual learning time wasted on wheel-spinning. These two aspects can also be used to evaluate a tutoring system. A successful tutoring system should have a low percentage in the number of student-skill instances which led to wheel-spinning, and also help students use their learning time efficiently. To better reveal the scope of the problem and also draw a clear visualization to the researchers who are interested in evaluating the effectiveness of a tutoring system, we will present the optimistic and pessimistic estimations of the wheel-spinning problem.

- Research Question 3: What are the relationships between wheel-spinning and other constructs of interest in ITSs?

We are particularly interested in investigating the relationships between wheel-spinning and two other constructs: efficiency of learning and gaming – a non-productive student behavior. The reasons for us to place special focus on these two constructs are the following.

Mastery learning is supposed to help students in a way that efficient learning is prioritized. Would wheel-spinning break that? Namely, would wheel-spinning wreck efficient learning? The answers to this question inform teachers, educational researchers and system designers what wheel-spinning has actually done to students. To answer the question, we looked into the number of problems done by the students working on the skills that they wheel-spun on, as well as the time spent on a problem that led to wheel-spinning.

The other construct is non-productive behaviors in learning, which is termed "gaming". Mastery learning values efficient learning. Part of the reasons is too much, presumably useless, practice might be harmful to student learning motivation. The non-productive behaviors named "gaming" are commonly believed to associate with negative learning attitude. The exploration of the relationship between wheel-spinning and the non-productive behaviors enable us to understand whether wheel-spinning companies with behaviors due to negative learning attitude.

- Research Question 4: Is wheel-spinning random? Can it be modeled and detected? Can we detect it quickly?

The last, yet the most interesting, research question is whether wheel-spinning is random. Can we detect its future occurrence? If so, how quickly can detect? It is easy to see the benefits from detecting wheel-spinning quickly. From the system's point of view, this preventive early detection leaves it relatively ample time to adjust its tutoring strategy or learning contents to provide students better and more suitable assistance. From the student point of view, the early detection of wheel-spinning and possible tutors' proactive interventions insure the maximal efficient use of their learning time. To facilitate an easy implementation of the wheel-spinning model for various learning systems, we aim to model wheel-spinning and explore useful features, which are generic and easily obtained from most tutoring systems. Then, we attempt to generalize the model to future students, evaluating the model's predictive ability. Finally, we refine our estimations of the scope of the wheel-spinning problem using the estimates of model whose features are generic, that is not specific to any particular tutoring system.

3. Data

To investigate the wheel-spinning problem in a broader range, we collected data from two tutoring systems: the Cognitive Algebra Tutor (CAT) and ASSISTments. There are substantial differences between the two systems (Koedinger, Anderson et al. 1997; Razzaq, Feng et al. 2007) but both apply the similar framework of student practicing. Domain knowledge is broken down to skills (knowledge components. A problem (step) is associated with one or multiple knowledge components. To learn a skill, a student is required to solve a series of problems associated with the skill. The problems could be presented successively or randomly among other problems associated different skills. These similarities facilitate our analyses and comparisons of the wheel-spinning problem in both systems.

3.1 Data descriptions

For the Cognitive Algebra Tutor, we used the data set from the KDD 2010 cup competition: algebra_2005_2006. The provided data were split into a training data set and a test data set. The test data set contains a small number of problems done by students in the training data. That is to say, for some student, the problems he practiced for a skill are mainly scattered in the training data set, and the last problem he practiced for the skill is placed in the test data set. We merged the two sets of data together and instituted our own training/testing methodology. There are 813, 661 problems solved by 575 students. The problems were performed by the students from 2005 to 2006. The data set contains some problems without labelled skills. We removed those problems from the data set. There are 111 Algebra skills encoded in the problems in the data set, such as Setting-the-y-Intercept, Labelling-the-Axes, etc. Since the public data set was lacking many common pieces of information, we have no further detailed information about problems such as the maximum number of hints the problem has, action-level information about problem-solving, or how quickly the student responded.

For ASSISTments, we tracked all ASSISTments students when they used the system to practice Math problems for a full school year from September 2010 to July 2011. For this study, we randomly selected 5997 ASSISTments students, who completed in total 303, 950 Math problems during examined period of time. These students were primarily from the northeast United States. We have student self-reported ages, and 75% of the students asserted they were 12 to 15 years of age on January 1, 2011. Since the students spread across a wide range of grades, the solved problems include a large range of skills as well. The problems cover 190 math skills, such as Equation-Solving-More-Than-Two-Steps, Area-Irregular-Figure, etc. Since we have access to the ASSISTments system's database, we can reach find-grained information, such as every action the student made while he was solving the problem. This allows us to analyze the relationship between wheel-spinning and non-productive "learning" behaviors induced by these find-grained data.

3.2 Data pre-processing

There are 5 rules we used for data pre-processing for the two data sets. Different parts of the work used data obtained by applying a single rule or a combination of multiple rules. Pre-processing rule #1 is applied to all parts of the work. The others will be mentioned where they are used.

1. Multiple skill problems.

There are multiple skill problems in both data sets. A multiple skill problem is a problem that requires more than one skill to solve. For this study, we need to treat a multiple skill problem as an observation for each of the required skills. Following the convention, we split a multiple-skill question to multiple instances of the question, each of which is associated with a single skill. This approach is commonly used in the literature (e.g., Pardos, Beck et al. 2008; Gong, Beck et al. 2011; Li, Matsuda et al. 2011).

2. Indeterminate problems.

An indeterminate problem is a problem done by a student, whose wheel-spinning status on the required skill cannot be determined. The existence of indeterminate instances is due to too few problems attempted by the student for the skill in the data set, and moreover he did not demonstrate mastery for the skill within those practices. No adequate evidence can be gathered to ascertain whether he would end up with mastering the skill or not.

3. Excessive problems after mastery.

An excessive problem is a problem done by a student who has demonstrated mastery for the skill. In some system settings, students are required to complete a fixed number of questions. Even if he has mastered a skill, he may also be presented more problems of the skill to practice. Another case is that we may have applied a different mastery determination criterion from the one actually used by the system, which causes more additional problems in the data set even after mastery. These additional problems are considered excessive practices after mastery.

4. Excessive problems after wheel-spinning

An excessive problem is a problem done by a student who has been determined to have started wheel-spinning for the skill. After the first N practice opportunities (N is the number of practice opportunities required for ascertaining wheel-spinning, to be defined later), if the student is identified as wheel-spinning for the skill, any additional problems done further after that are considered excessive problems after wheel-spinning.

5. Problems with extreme long/short duration

For some problems in the log files, the length of time used to solve a problem is improbably short or long, e.g. 0 seconds or several hours. Such anomalies could possibly be due to the system's logging process or the student's incomplete practice. To limit the impact of these outliers, when we conduct analyses regarding time, we removed 5% of the problems with minimal durations and 5% of the problems with maximal durations from the two data sets.

4. Research Question 1: The wheel-spinning problem

To define the wheel-spinning problem, we first examine some states of student learning in the mastery learning framework. We classify student learning states into three categories.

The "already known" state: Some students will begin working with an ITS already fully understanding the material. For those students, we would observe that within a minimum required number of questions, the students have already achieved mastery. Although they seem not to get benefits from the computer tutor for the particular topic, they are allowed to proceed to next topic quickly and thus they can learn in an efficient way and not spend unnecessary time solving problems they already understand.

The "learning" state: Some students will need to work with an ITS. They are packed with some level of declarative knowledge of the topic. Through practice, they would be able to master the topic gradually, especially with the tutor's assistance. Although the speed of learning varies, we would observe eventually that they meet the system's mastery criterion after a reasonable amount of practices with interleaving correct and incorrect responses.

The "wheel-spinning" state: Some students start working with an ITS, possibly with little declarative knowledge about the topic. They have practiced a seemingly sufficient amount of problems without having mastered the skill. Following the simplified mastery learning workflow, the computer tutor continues to present more problems to the student, hoping that he could achieve mastery at some point. The student wheel-spins in this situation.

Neither of the first two groups is problematic. We are seriously concerned with the third type as we consider the "wheel-spinning" state as a non-progressive state in mastery learning.

4.1 Two factors defining wheel-spinning

This seems a vague definition just to say that wheel spinning is the student getting stuck in the mastery learning loop. It is necessary to establish a definition for wheel-spinning. We define wheel-spinning as a non-progressive state in learning that leads to no mastery or costs too much time to master a skill. There could be a variety of instantiations of wheel-spinning. Two factors in the definition are the key.

1. Mastery

Wheel-spinning is a problem subsidiary to the simplified implementation of mastery learning. What does it mean to say someone "masters" a skill? The definition of mastery and its implementation matter, as different ways to determine mastery naturally lead to different instantiations of wheel-spinning.

2. Time

We are worried about students being stuck in the mastery learning loop. A clear sign of it is the student has spent too much time without mastering the skill. "Too much time" is a vague descriptor. It should be abstracted as a threshold of some notion of time, e.g. the number of problems having been attempted, or the amount of time having spent. After the threshold, wheel-spinning is determined to have occurred.

For the first factor, there are two typical ways to determine mastery. One is based on student knowledge. Some tutoring systems use student models to make inference about student knowledge (e.g., Corbett and Anderson 1995; Fancsali, Nixon et al. 2013). When the inferred student knowledge exceeds a pre-defined probability, such as 0.95 in (Corbett and Anderson 1995), the system considers the student has mastered the skill. The other way is based on overt characteristics of student performance, such as three correct responses in a row (e.g., Hawkins, Heffernan et al. 2013). Such approaches are theoretically less accurate, but have the benefit of being easily explainable to students and eliminating side step issues related to identifiability when inferring student knowledge (Beck and Chang 2007). Once some condition or pattern of student performance has been met, the system asserts that the student has mastered the skill.

In this work, we selected overt student characteristics as the basis of our mastery criterion. For a skill, if a student has answered three consecutive problems correctly, we assert him having mastered the skill. To answer a problem correctly, the student has to respond to the problem correctly on his first attempt, and requests no assistance from the tutor prior to that. This definition of correctness is the same as used in most student modeling work (Corbett and Anderson 1995; Pavlik, Cen et al. 2009; Gong, Beck et al. 2011).

There are two reasons for us to use such mastery criterion. First, observing student overt performance, rather than inferring a latent variable as with knowledge tracing, keeps the work easier to understand methodologically and enables the work to more cleanly focus on wheel spinning. In particular, it is easier for others to replicate our work without requiring complex statistical machinery to estimate mastery. Second, we selected three as the threshold, because it is a relatively easy condition for a student to reach. In fact, this criterion is probably too low for most definitions of mastery. For example, in a study of using inferred knowledge in mastery criterion, to let the probability of knowing the skill greater than 0.95, students have to practice averagely 30+ problems (Rai, Gong et al. 2009). In this work, we purposefully chose a weak mastery criterion to avoid creating a problem which is purely an artefact of an overly strict mastery criterion that few students could ever reach.

With regard to the second factor, how much time should be considered too much?, we did not have a good answer and therefore resorted to looking at the data. We used learning curves to examine how students progress to mastery, expecting to find insights with regard to learning time used in mastery learning. Learning curves have become a standard tool for not just measuring student's learning, but also evaluating ITSs (e.g., Anderson, Bellezza et al. 1993; Koedinger and Mathan 2004; Martin, Koedinger et al. 2005). We generated (a form of) learning curves for students of the Cognitive Algebra Tutor and ASSISTments based on real data.

4.2 Determining the threshold of time

Figure 9 is a visual representation of students progressing to mastery in the Cognitive Algebra Tutor (CAT) and in ASSISTments. The x-axis represents the number of practice opportunities (PO), (i.e., problems) students have practiced on a skill. The y-axis represents the percent of student-skill instances which have demonstrated mastery (i.e., gotten three problems in a row correct for the practiced skill).

A student-skill instance (a student-skill pair) is the unit to talk about mastery or wheel-spinning. When we say mastery or wheel-spinning, it implicitly involves a student and a skill. For example, Ann practiced two skills: Addition and Subtraction. She mastered Addition and wheel-spun on Subtraction. In this example, there are two student-skill instances: Ann-Addition and Ann-Subtraction. Ann-Addition is a mastery instance, and Ann-Subtraction is a wheel-spinning instance.

The data used for generating the curves were pre-processed by applying rule # 1 (splitting a multiple skill problem to multiple single skill problems), and rule #3 (removing excessive instances after mastery). After the pre-processing, for the CAT data, there are 202, 631 algebra problems solved by 575 students left, forming 23,517 student-skill pairs. For the ASSSITments data, there are 220, 539 Math problems solved by 5997 students, forming 45,787 pairs. As an example of interpretation, in the chart, if a curve of ASSISTments reaches 100%, it means that all 45,787 student-skill pairs achieved mastery. Equivalently, all students mastered all skills they practiced.

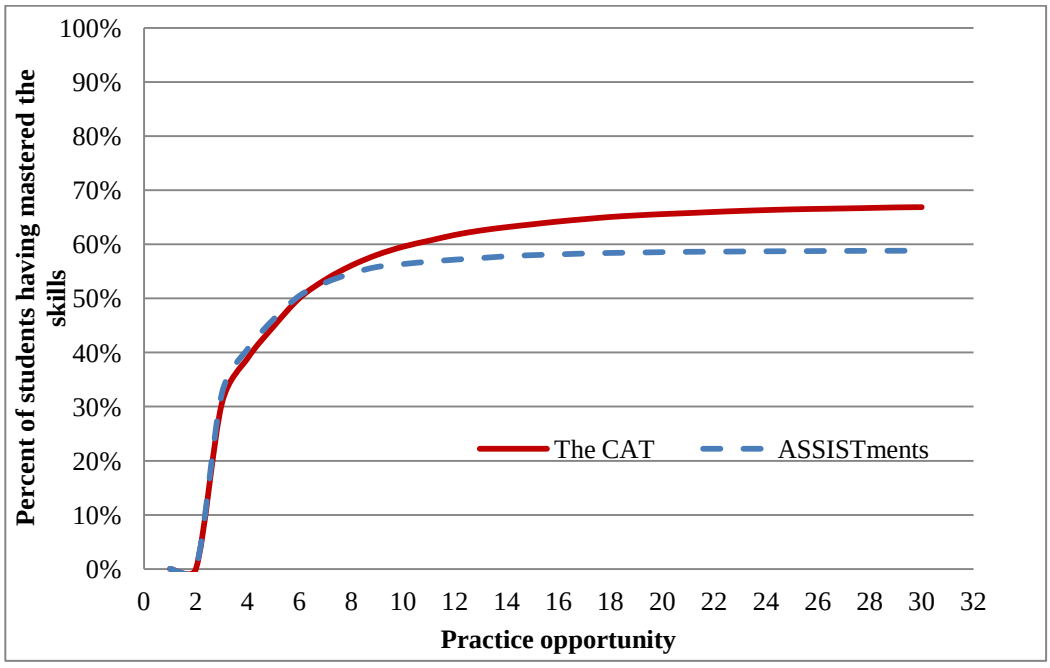


Figure 9 Wheel-spinning in the Cognitive Algebra Tutor and ASSISTments

The performance of students of CAT and ASSISTments both follow the "power law of practice" approaching mastery (Newell and Simon 1972; Mathan and Koedinger 2005). By our definition of mastery, three correct responses in a row, tautologically no student has mastered a skill until after his first two practice opportunities.

After the third practice opportunity, approximately one-third of student-skill pairs of the CAT and ASSISTments respectively demonstrated mastery. These students are, and probably were, in the "already known" state for the practiced skills. The tutor cannot claim credit for helping these students, as they already knew the material before working with the tutor, or learned in the process of problem solving without assistance from the tutor.

We can observe the effectiveness of the CAT and ASSISTments systems. After 6 practice opportunities, 50% of student-skill pairs have been demonstrated mastery in the CAT and ASSISTments.

Students lying in the gradually-increasing sections of the curves can be viewed as in the "learning" state of mastery learning. They benefitted from the tutoring systems, and the ITSs deserve credit for having helping these students.

For ASSISTments, the percentage of student-skill pairs achieving mastery asymptotes around the 10th practice opportunity, and does not increase noticeably further past that point. The CAT curve continues to improve until around the 15th practice opportunity. The CAT has about 35% student-skill pairs remaining non-mastery at the end, and the number for ASSISTments is about 40%.

Two interpretations can be made from the curves. First, there are a significant portion of student-skill instances in both systems that failed to demonstrate mastery within as many as 30 practice opportunities. Second, a student who did not demonstrate mastery within a certain number of initial opportunities would probably never demonstrate mastery. For example, in ASSISTments, after 10 problems, 45% student-skill instances (= 100%- 55%) have not demonstrated mastery. Examining the curve for the 30th problem, after practicing 20 more problems, only an additional 4% of the student-skill instances were actually able to achieve mastery. If a student did not demonstrate mastery after 10 problems, then the conditional probability he will master the skill with more practice is 0.04 / 0.45 = 0.089, rather grim odds. In this regard, the students are probably spinning their wheels in their mastery learning process.

Based on these observations, we pinned down our threshold of time for determining wheel-spinning. If a student who uses the CAT does not master a skill within 15 problems, we categorize him as wheel spinning. For ASSISTments, we set the threshold to 10 problems. We term this threshold as the wheel-spinning determination checkpoint. Although both numbers are subject to discussion, an inspection of Figure 9 indicates that if we changed the threshold for ASSISTments to 20, the percentage of student-skill instances achieving mastery is approximately the same. A similar trend also applies to the CAT.

5. Research Question 2: The scope of the wheel-spinning problem

The visualization of students progressing to mastery, shown in Figure 9, seems to suggest that wheel-spinning is a real problem beyond its theoretic existence in the mastery learning framework. It is obvious from the curves that some students were in a non-progressive state that led to no mastery or cost too much time to master the skills. Besides its existence, the wheel-spinning problem appears to be generic and big. We found similar trends in the curves of the CAT and ASSISTments, indicating that the wheel-spinning problem occurs generically in the two widely-used computer tutors with different student populations, pedagogical approaches and learning environments. Checking on the lags between where the curves are located and the top lines representing 100%, we had no hard time to see that the wheel-spinning problem hurt a substantial number of student-skill instances.

We refined our analysis and were dedicated to explore the scope of the wheel-spinning problem. We approached to this research question from two aspects: percent of student-skill instances which led to wheel-spinning and the amount of time cost by wheel-spinning.

There are two types of problems in the data sets. One is determinate problems, which are further divided into two types: mastery problems and wheel-spinning problems. A problem is a mastery problem, if the student masters the required skill in the problem. Similarly, a problem is a wheel-spinning problem, if the student wheel-spins on the skill.

The other type is indeterminate problems. When a student practiced few problems for a skill, fewer than the number of problems needed to determine wheel-spinning, and he did not master the skill within those problems, this student-skill instance is called an indeterminate student-skill instance. Suppose Ann practiced 4 problems for Multiplication. From the observations, we are very sure that Ann did not master Multiplication. Did Ann wheel-spin on Multiplication? Since she never made it to the wheel-spinning determination checkpoint, we are not sure about it. Therefore, Ann-Multiplication is called indeterminate instance, and all the associated problems (4 problems in this example) are called indeterminate problems.

To estimate the scope of the wheel-spinning problem, we need a means to handle indeterminate problems. Should we just ignore them? Or should we believe that they would lead to mastery at the end

and treat them as mastery instances accordingly? Or should we stick on "no evidence of mastery equals to wheel-spinning" and treat them as wheel-spinning instances?

These three options lead to different results of analyzing the scope of the wheel-spinning problem. If we choose to ignore indeterminate instances, we only have a sub set of the original data which is not randomly sampled. This could result in a less accurate estimate of the degree of the wheel-spinning problem. If we classify all indeterminate problems as mastery, we might ease the wheel-spinning problem by optimistically believing those students would end up mastering the skills. If we treat them as wheel-spinning, we artificially add more evidence of wheel-spinning, and thus the problem appears more severe than it actually is. Although none of the three options seems a good solution if our goal is to acquire the true picture of wheel-spinning, taking the last two options to analyse wheel-spinning is a reasonable method to establish an insight for the boundaries of the scope of wheel-spinning.

For the analyses in Chapter 5, Section 5, we did data pro-processing by applying rule # 1 (splitting a multiple skill problem to multiple single skill problems), and rule #3 (removing excessive problems after mastery). Table 11 and Table 12 show the distributions of determinate and indeterminate instances in the CAT and ASSISTments data sets. Table 11 shows the distribution for problems and Table 12 is for student-skill pairs.

Table 11 Number of determinate and indeterminate problems in the CAT data

	Determinate problems	Indeterminate problems	Total problems
The CAT	109,113 (74%)	37, 366 (26%)	146,479
ASSISTments	145,272 (66%)	75,267 (34%)	220,539

Table 12 Number of determinate and indeterminate problems in the ASSISTments data

	Determinate student-skill pairs	Indeterminate student-skill pairs	Total student-skill pairs
The CAT	16,049 (68%)	7,468 (32%)	23,517
ASSISTments	28,259 (62%)	17,528 (38%)	45,787

5.1 Percent of student-skill instances which let to wheel-spinning

We have shown how the students of the two systems approach mastery in Figure 9. It is worth pointing out that in the way we generated Figure 9, it implicitly treats indeterminate student-skill instances as wheel-spinning instances. At each practice opportunity, we plotted the percent of student-skill instances having demonstrated mastery. After all 30 practice opportunities, since indeterminate instances never demonstrated mastery, they are all left in the data set, because of which the wheel-spinning problem appears more severe that it probably is.

For example, Ann attempted only 4 problems for Multiplication without mastering it. We are using 10 problems as the wheel-spinning determination checkpoint. Since Ann never got to the checkpoint, this instance, Ann-Subtraction is an indeterminate instance. When we plotted the curve in Figure 9, Ann-Subtraction was not counted in for the 1st and 4th practice opportunity, because it is for sure that Ann has not mastered Multiplication yet. From the 5th practice opportunity and afterwards, due to no more observations, Ann-Multiplication still remains non-mastery, which contributes the amount of student-skill instances which seem to end up with non-mastery.

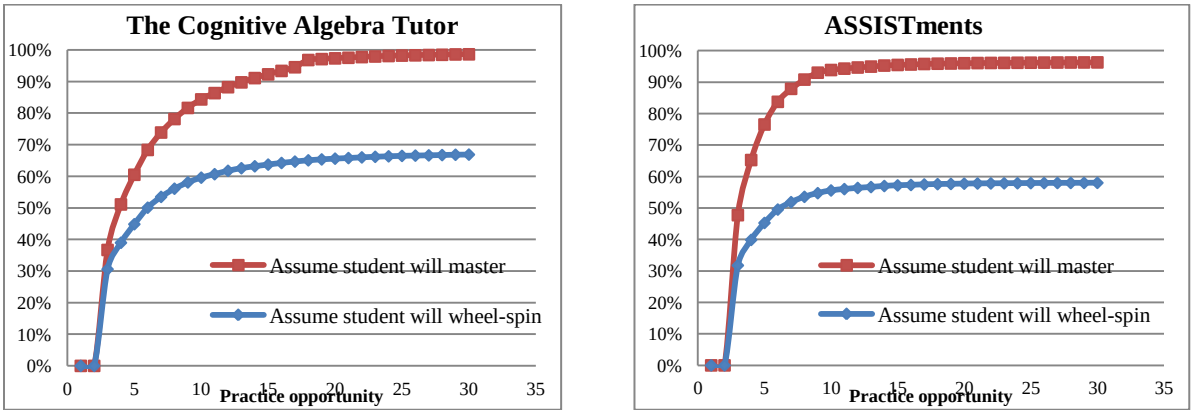
It is easy to see that with this approach, in Figure 9, the proportion of mastery instances is probably under-estimated, because some indeterminate instances could end up with mastery if more observations were collected. Contrariwise, the proportion of wheel-spinning instances is over-estimated, as it is not likely that all the indeterminate would end up with wheel-spinning.

By treating the indeterminate as mastery or wheel-spinning, we estimated the scope of the wheel-spinning problem from an optimistic or pessimistic point of view. When we leave the indeterminate as they are and plot a learning curve, we implicitly treat the indeterminate as wheel-spinning. Shown in Figure 9, the curves depict the pessimistic perspective of viewing the wheel-spinning problem.

The other option is to treat indeterminate problems as mastery problems. The approach in this study is, for an indeterminate student-skill pair, the last problem the student practiced for the skill is assumed where mastery occurs.

An argument is that it is evident that for some cases, the student cannot master the skill at the last practice opportunity even if we assume he correctly answered the last problem. This is due to how he performed in the previous two problems. Our mastery criterion is three consecutive correct answers. If the previous two responses prior to the last problem are both correct, it is sensible to flip the correctness of the answer to the last problem, and assume mastery for this indeterminate student-skill instance. If in any of other cases, e.g. neither of the previous two responses is correct, it is impossible for the student to master the skill, even if we change his answer to the last problem to correct. An alternative to address this problem is to add in virtual problems based on how student performed in the last three problems. However, to make it easier to understand, we decided to apply the straightforward method. We assumed mastery at the last practice opportunity regardless of previous responses. Through considering indeterminate instances as mastery instances, we see the wheel-spinning problem optimistically, and this method gives us an insight as to the least server condition of wheel-spinning, i.e. wheel-spinning's lower bound.

A better visualization presenting the refined scope of the wheel-spinning problem is shown in Figure 10, where we plotted the bounding curves of the wheel-spinning problem. For a clearer view, we showed the curves of a tutoring system separately. Figure 10 (a) shows the bounding curves for the CAT , and Figure 10 (b) contains curves for ASSISTments. Similar to Figure 9, the x-axis represents the number practice opportunity and the y-axis represents the percent of student-skill pairs which have demonstrated mastery. In both charts, the top curve represents the case where indeterminate instances are assumed mastery. This curve shows the optimistic side of mastery, and thus corresponds to the lower bound of wheel-spinning. The bottom curve represents the pessimistic perspective of mastery, assuming that indeterminate students will wheel-spin on the skills, which corresponds to the upper bound of wheel-spinning.



(a) (b)

Figure 10 Wheel-spinning's upper bound and lower bound. (a) of the CAT; (b) ASSISTments

Inspecting the two top curves, we found that the two systems appear to have similar lower bounds of the wheel-spinning problem. The wheel-spinning problem looks quite trivial. There are over 90% student-skill instances having mastered the skill beyond the cut-off points we selected for determining wheel-spinning (i.e. 15 problems for the CAT and 10 problems for ASSISTments). The curves continue to evolve and slowly approach 100% at the 30th practice opportunity.

It is an important understanding that the true degree of wheel-spinning should lie in between the upper and lower bounds. Is the wheel-spinning problem worth further attention? We believe so. In percentage, the wheel-spinning problem could affect 8% - 35% of skill practicing of the CAT students and 8% - 45% of skill practicing of the ASSISTments students. In absolute, the number of possibly wheel-spinning student-skill instances lies in between 1881 and 8230 for the CAT and 3590 and 20,193 for ASSISTment. Even when we look at the minimal numbers, they are still a considerable amount of instances which nether of the two systems should not ignore. In addition, we only know that the true extent of wheel-spinning lies in between the bounding curves, we have no means to locate it. It could be as bad as near the problem's upper bound, or as mild as near the problem's lower bound. Due to this uncertainty, we argue that further research is needed.

5.2 The amount of time spent on wheel-spinning

Mastery learning thrives due to its ability to efficiently use students' learning time resulting in maximum learning out of it. The wheel-spinning problem leads to the opposite outcome, which is a tremendous amount of time is spent without any learning progress. To explore the scope of the wheel-spinning problem, it is necessary, and interesting, to find out in an absolute sense, how much time is used for progressive learning and how much is cost by wheel-spinning. In short, what has wheel-spinning really done to the students?

In the data sets, each problem is logged with time duration, representing the amount of time the student spent on it. We added up the total amount of time spent respectively on three categories of problems: mastery, wheel-spinning and indeterminate. Following the similar logic as we did for analysing the upper bound and lower bound of wheel-spinning in the number of student-skill pairs, we calculated the upper bound and lower bound of time spent on wheel-spinning. When we treat all indeterminate problems as mastery problems, we increase the number of mastery instances. Accordingly, the amount of time on mastery is counted more, probably than its true value. This is an optimistic perspective to see the wheel-spinning problem, and thus yields the lower bound. Contrariwise, when we treat all indeterminate problems as wheel-spinning problems, we see the problem pessimistically, and thus it is corresponding to the upper bound of wheel-spinning on time spent.

Table 13 Summed amount of time cost in different types of problems of the CAT

	In hours	**In percentage**
Mastery	543	42.3%
Wheel-spinning	351	27.4%
Indeterminate	389	30.3%
Total	1283	100%
Wheel-spinning after the determination point (15 problems)	166	12.9%

For this analysis in Chapter 5, Section 5.2, other than the data pre-processing rule #1 and #3, which were also used in Section 5.1, we applied an extra rule: rule # 5 (removing problems with extreme

durations). Table 13 shows the summed amount of time used by the CAT students in the data set. Table 14 shows the upper bound and lower bound of time spent by the CAT students.

In Table 13, the second column shows the summed amount of time in hour for each category, and the last column shows the corresponding percent. Note that the amount of time of mastery, wheel-spinning and indeterminate problems sums up to total (i.e. 543 + 351 + 389 = 1283). In this data set, 1283 hours were spent by the students practicing algebra problems in the CAT. In the last row, we zoomed in onto the time spent on problems after wheel-spinning has determined. For example, Ann practiced 12 problems on Subtraction and did not master the skill. Ann-Subtraction is a wheel-spinning instance, if 10 problems is the checkpoint of wheel-spinning. The last two problems were done after wheel-spinning is determined and observed. Students have started wheel-spinning, and the time was used purely on problems with little or no chance of progressing to mastery. This part of time should be minimized as much as possible.

Surprisingly time-wise, it turns out that merely less than a half of the total hours (42.3%) were actually spent on problems leading to mastery, shown in the row of mastery. When we use a more conservative metric, the amount of time spent after the wheel-spinning determination checkpoint (15 problems), we found that still 166 hours were consumed, over one-tenth of the time in the data set. This is almost 1 month of learning time, if we assume a student spends 2 hours per day on the system and works every day. This finding confirms our concern about the wheel-spinning problem that after a number of unsuccessful practices, giving the student more problems to solve and hoping he can master the skill causes a waste of time.

351 hours on wheel-spinning is just a part of the picture of the wheel-spinning problem. To have a better estimate of time spent on wheel-spinning, we followed the similar approach used in the previous section. Indeterminate problems cost 389 hours of learning time, which is 30.3% of the entire time. We handled them from two perspectives: optimistic and pessimistic, and estimated the upper and lower bounds of wheel-spinning on time spent in Table 14.

Table 14 Summed amount of time, optimistic and pessimistic perspectives for the CAT data

Group	**Optimistic**		**Pessimistic**	
	in hour	%	in hour	%
Mastery	932	72.6%	543	42.3%
Wheel-spinning	351	27.4%	740	57.7%

The two major columns are entitled optimistic and pessimistic. In optimistic, the numbers describe the lower bound of wheel-spinning, the least severe situation. In absolute, it is 351 hours and 27.4% of the entire time in the data set. The upper bound of wheel-spinning is shown in the last row of the pessimistic column, 740 hours, and 57.7% of the entire time in the data set. It seems that the wheel-spinning problem hurt the CAT students badly. At its mildest influence, 27.4%, almost 1/3 of the entire learning is consumed by wheel-spinning. Besides, the true extent of wheel-spinning is likely to cost even more time, when the problem does not appear as we optimistically hoped. The true amount of time cost by wheel-spinning ranges from 351 hours (27.4%) to 740 hours (57.7%). In pessimistic, it is more than half of the entire learning time.

Table 15 and Table 16 are similar to Table 13 and Table 14, containing summed amount of time spent by the ASSISTments students. From the distributions of learning time spent by the ASSISTments students, the wheel-spinning problem seems in a moderate condition. The wheel-spinning problem only cost 18.3% of the entire time in the data set. The excessive time spent on wheel-spinning after the first 10 problems also seems better than in the CAT. We compared the amount of time spent after the first 10 problems, 159 hours, and the amount of time on all wheel-spinning problems, 584 hours. The fraction is 27.2%. This number of the CAT is 47.3%. It suggests for ASSISTments, even though wheel-spinning

occurred, it did not last too long or swallow too much learning time. Not too far did the system push the students who wheel-spun.

Table 15 Summed amount of time cost in different types of problems of ASSISTments

	In hours	**In percentage**
Mastery	1454	45.5%
Wheel-spinning	584	18.3%
Indeterminate	1157	36.2%
Total	3195	100%
Wheel-spinning after the determination point (15 problems)	159	5.0%

Table 16 Summed amount of time, optimistic and pessimistic perspectives for the ASSISTments data

Group	**Optimistic**		**Pessimistic**	
	in hour	%	in hour	%
Mastery	2611	81.7%	1454	45.5%
Wheel-spinning	584	18.3%	1741	54.5%

The upper bound and lower bound of learning time spent on wheel-spinning are 54.4% and 18.3% of the entire learning time spent by these students. It is critical for the system designers that the users wasted considerable time, which in its worst case could be over 1,500 hours. Suppose that averagely a student works 8 hours of Math per week and 32 weeks per year, 1,500 hours wheel-spinning equals to 6 students spending their entire class and homework math time just wheel spinning.

6. Research Question 3: Wheel-spinning and other constructs

This research question examines the relationships between wheel-spinning and other constructs of interest in tutoring systems. We are particularly interested in two constructs: efficiency of learning and seriousness of learning attitude.

Mastery learning is good at keep students to learn efficiently. We are concerned that the occurrence of wheel-spinning has its linkage to hindering efficient learning. To examine efficiency of learning, we used two metrics: the average number of problems solved by wheel-spinning student-skill instances and the average amount of time spent on a wheel-spinning problem. The study is presented in Section 6.1.

Another hypothesis is forcing students to solve too many problems, especially leading to no progress, might hurt their motivation to learn, and thus lean them to some non-productive "learning" behaviors, which is typically associated with negative learning attitude, termed "gaming". We wanted to explore the connection between wheel-spinning and gaming, focusing on how these two constructs interact with each other. The study is presented in Section 6.2.

6.1 Wheel-spinning vs. efficiency of learning

To explore the relationship between wheel-spinning and efficiency of learning, we examined how students behaved differently on problems leading to mastery and problems leading to wheel-spinning. We chose two metrics: the average number of problems practiced by a student for a skill and the average time spent on a problem. Despite a non-precise indicator of time, the number of practiced problems provides a rough sense of time. The other metric is the average time spent per problem. It provides an accurate measure of time.

For this analysis in Chapter 5, Section 6.1, we applied data pre-processing rule #1 (splitting a multiple skill problem to multiple skill problems), rule #2 (removing indeterminate problems) and rule #3 (removing excessive instances after mastery) and rule #5 (removing problem with extreme time duration). The CAT data set after the pre-processing contains 98,435 problems done by 565 students. The ASSISTments data set after the pre-processing contains 143,963 problems done by 5103 students.

We wanted to figure out if there is any difference in time between problems leading to wheel-spinning and problems leading to mastery. Therefore, we applied the data pre-processing rule #2, removing indeterminate problems. The problems remained in the data sets are either mastery instances or wheel-spinning instances. To understand the relationship between wheel-spinning and learning efficiency, we grouped the problems to the mastery group and the wheel-spinning group. We compared the two groups against the two metrics: average number of practiced problem per student-skill instance and average time spent per problem.

We show the comparisons in Table 17 and Table 18 for the CAT and ASSISTments. The two tables have the same format. There are three major columns: group, number of attempted problems and time spent. Each row represents a group.

The first major column is entitled "Number of Attempted Problems", where we show the average number of attempted problems by the two groups. The first sub-column shows the number of student-skill instances that belong to each of the two groups. The second sub-column shows the average number of attempted problems per student-skill instance. We further broke the average to correct and incorrect responses, denoting the average numbers of correct and incorrect responses obtained by the group.

In the second major column, entitled "Time Spent", we showed the number of problems in each category and the average time spent by the students on each problem in the category.

Table 17 Number of problems and time breakdown for the CAT

	Number of Attempted Problems				**Time Spent**	
	Number of student-skill pairs	Avg. # of attempted problems per student-skill pair			Number of problems	Avg. time per problem (in secs)
Mastery	14798	5	Correct Response	4	67,056	29
			Incorrect Response	1		
Wheel-spinning	1676	26	Correct Response	8	31,379	40
			Incorrect Response	18		

Table 17 shows that the wheel-spinning problem hurt the CAT students badly. Comparing the numbers 5 and 26 below the sub-column "Avg. # of attempted problems per student-skill pair", we found that students practiced remarkably fewer questions for the skills they can master. For the CAT, the wheel-spinning determination checkpoint is the 15th problem. However, the students who mastered the skills

needed far fewer practice opportunities, 5 problems on average. It suggests that if students can master the skills, they can do it quickly. Moreover, students made very few mistakes for the skills they can master. We see that among the 5 responses, 4 of them were correct responses, suggesting that either most of these students were in the "already known" state for the skills, or they learned from the CAT quickly.

On the other hand, in the case of wheel-spinning, it seems troublesome. Aside from the fact that these students never mastered the skills, they averagely practiced considerably more problems for a skill. Averagely, a student practiced 26 problems for a skill that he did not master, which is 5 times more than the counterpart. When we break the problems based on correctness, we found an average 30.7% correct rate. It is evident that rather than a strict mastery criterion that makes mastery difficult, the low correctness rate is the reason that these students failed to achieve mastery. In other words, it is not that the students just happened not to make 3 correct responses *in a row*. Rather, they hardly made correct responses.

The last major column, entitled "Time Spent," shows time duration per problem in seconds. The mastery group solved 2 times more problems in total than the wheel-spinning group, but each problem cost less, averagely 29 seconds vs. 40 seconds. The result of this comparison concurs that when a student wheel spins on a skill, a loss of learning time is substantial, as not only he practices more problems, also each problem costs him longer to solve.

Table 18 Number of problems and time breakdown for ASSISTments

	Number of Attempted Problems				**Time Spent**	
	Number of student-skill pairs	Avg. # of attempted problems per student-skill pair			Number of problems	Avg. time per problem (in secs)
Mastery	25,296	4	Correct Response	3	103,431	55
			Incorrect Response	1		
Wheel-spinning	2,810	14	Correct Response	5	40,531	59
			Incorrect Response	9		

Table 18 shows the results for ASSISTments in the same table format as Table 17. In general, the trends are similar to those of the CAT. Averagely, the mastery group practiced 4 problems prior to mastery. It seems that for the skills that were mastered, the students understood them very well and thus can achieve mastery very quickly. Similar to the CAT students, the success rate of this group is also high. With 3 correct responses and 1 incorrect response, the success rate is 75%.

When wheel-spinning occurred, students performed poorly. They on average practiced 14 problems on the skills they could not master, which is 3 times more than on the skill they mastered. Besides, the students had fewer correct responses. This is consistent with what we found for the CAT, that the students' failures to mastery is due to a low correctness rate, rather than some misfortune that they did not get correct answers clustered to make 3 consecutive correctness.

In the last major column for "Time Spent", we found a divergence that time-wise ASSISTments students did not perform differently on problems leading to mastery and wheel-spinning.

We think that two findings are particularly interesting, based on the results across systems. First, students typically either master the skill quickly or do not get it at all. Second, while wheel-spinning, students were not racing through questions, at least not superficially. Instead, they were spending more or comparable time. This contradicts our initial conjecture that students in wheel-spinning might tend to avoid serious efforts and intent to get through questions quickly. The opposite observation implies that

either students indeed have no rushing-through problem, or restricted by system settings, they are not possible to rush through problems.

6.2 Wheel-spinning vs. gaming

"Gaming the system", also called "gaming", refers to a type of student non-productive behaviors. A student is gaming if he or she is attempting to systematically use the tutors' feedback and help methods as a means to obtain a correct answer with little or no work (Baker, Corbett, Koedinger, & Wagner, 2004; Walonoski & Heffernan, 2006). There have been many prior works showing that gaming behavior is generally associated with a reduced learning rate, both immediately and aggregately (e.g. Baker, Corbett, Koedinger, & Wagner, 2004; Walonoski & Heffernan, 2006; Cocea, Hershkovitz and Baker (2009); Beck & Mostow, 2008). In particular, Beck and Mostow 2008 pointed out that the apparent immediate impact of gaming, at the step level, appears to be due to a lack of learning at that very step where the gaming occurred; in other words, through gaming, an opportunity to learn is wasted.

Wheel-spinning is defined as a non-progressive state in learning, which is also, associated with negative learning outcomes, such as poor performance, inability to master and poor efficiency in use of learning time. It is interesting for us to explore whether there are any connections between these constructs. In particular, mastery learning values student learning time and is designed to avoid laying too many problems to students. We hypothesized that too many problems might hurt students' motivation to learn, and the situation is probably worsened when students realize the additional practice is not helping them. Wheel-spinning is such state in learning that students are solving more problems, shown in Section 6.1, which lead to little or no progress. Would they experience negative learning attitude and exhibit non-serious learning behaviors? For this research question, we analyzed the relationship between wheel-spinning and student gaming for ASSISTments. We were not able to attempt this research question using the Cognitive Algebra Tutor data, as we did not have access to detailed information which allows us to analyze gaming.

6.2.1 Implementing a student gaming detector

What types of student behaviors are considered gaming is rather subjective. Different researchers, standing on different viewpoints, could certainly have various opinions. Some behaviors, however, have gotten researchers' consensus, such as rapidly asking for system help without even thinking about the question at all, intentionally seeking the answer to the question so as to finish the work as quickly as possible, etc(Baker, Corbett et al. 2006; Gong, Beck et al. 2010; Muldner, Van de Sande et al. 2011). As the main focus of this work is not gaming, we hereby only briefly introduce how we identify gaming behaviors based on three gaming criteria and how accordingly we derived a unified score, named "gaming score", for each problem the student attempted. More details related to gaming can be found in our prior work (Gong, Beck et al. 2010).

Three gaming criteria are the following.

Rapid Guessing.

Rapid guessing concerns students submitting responses too quickly, especially after having given incorrect responses. We think of this kind of behavior as a guess action. To be called "rapid guessing", two conditions need to be met. 1) The student guesses: A student submits two attempted answers less than 2 seconds apart in a question, and 2) The student's behavior is repetitive: This pattern is repeated on two successive questions.

Rapid Response.

Rapid response refers to students responding too quickly after seeing a new material, such as a new question or a hint message. We define a reasonable amount of time for a material as the time needed to read through the material. We chose a reading rate of 400 words per minute as the threshold, as this rate is faster than most college readers, and is a plausible upper bound for middle-school students. Therefore,

given a question with 200 words, the student should at least take 30 seconds to finish reading the question. Any action done within 30 seconds would be considered rapid response.

Repeatedly Bottom-out Hinting.

In each question, there is a special hint called a bottom-out hint, where the answer to the question is given directly by some words like "Type in 20". It is understandable and reasonable for students having considerable difficulty in solving questions to reach the bottom out hints. However, repeatedly bottom-out hinting suggests that the student may be being engaged in just seeking the answer. Thus, if a student reaches bottom out hints on three consecutive questions, we mark the third question as repeatedly bottom-out hinting.

Based on the three criteria, we developed a gaming detector. On each student action, the gaming detector triggers if at least one of the three criteria is matched. The gaming score ranges between 0 and 1, where 0 represents no gaming detected and 1 means gaming occurred. At the very beginning, there are no previous records of a student, so we assume that the student performs seriously in learning, and thus the gaming score is set to be 0. The score is increased straight to 1 if the student is detected as having done a gaming action. The student later can "recover" from a gaming state to a serious learning state by performing any non-gaming actions (any action in which none of the three criteria is satisfied). When a non-gaming action detected, 0.5 is subtracted from the gaming score until it recovers to 0 again. In this way, the student would be considered to be serious again when two non-gaming actions are performed in a row.

There are usually multiple actions performed by a student while he is solving a problem, each of which has its own gaming score. Afterwards, we averaged these scores and derived a unified gaming score assigned to the problem. Therefore, each problem a student attempted is associated with a single gaming score, and its range is from 0 to 1, indicating being completely serious to being completely gaming.

6.2.2 Overall gaming behaviors in wheel-spinning

The data set for this analysis was pre-processed by applying rule #1 (splitting a multiple skill problem to multiple skill problems), rule #2 (removing indeterminate problems) and rule #3 (removing excessive instances after mastery). As information about some problems' actions is missing, the data set used for this analysis is slightly smaller, containing 140,362 problems done by 5026 students.

One of our conjectures is that when a student is working on problems leading to wheel-spinning, he is probably aware of him having practiced many problems and having little hope to master the skill. This experience is reasonable to be very frustrating, and could be worsened if the student is forced to continue with more problems. The behaviors our gaming detector captures reflect student seriousness to learning. It is possible that wheel-spinning and gaming have some relationships. Although it requires much more work to state the causality between the two, if there is any, it is still meaningful to investigate whether, and in what way, they are associated.

We examined the general relationship between the two constructs in a coarse-grained level. We categorized problems based on the wheel-spinning status, so two groups were formed: mastery and wheel-spinning. Foe each group, we calculated the average gaming scores across all problems of the group. We show the results in Table 19. A smaller gaming score indicates fewer gaming behaviors and possibly more serious learning attitude.

Table 19 Average gaming scores by mastery and wheel-spinning

Group	# of problems	Avg. gaming score (AGS)
Mastery	101071	0.01
Wheel-spinning	39291	0.11

We found a clear divergence between the mastery group and the wheel-spinning group. The average gaming score (AGS) of the mastery group is 0.01, and the AGS of the wheel-spinning group is 0.11, than times bigger than the former. The mastery group appears to be more serious in learning. Students tended to be more serious when they were working on problems leading to mastery. Contrariwise, problems in the wheel-spinning group are found to have high gaming scores, yielding a high average, indicating that gaming occurred remarkably often on problems leading to mastery failure.

Note that the unit to talk about mastery and wheel-spinning is student-skill instance, which means that a student could belong to both groups, differing on the skills he worked on. This is exemplified by the previous mockup scenario, where Ann mastered Addition, but wheel-spun on Subtraction. Ann is a cross-group student: Ann-Addition is in the mastery group and Ann-Subtraction is in the wheel-spinning group.

The above analysis whose results are shown in Table 19 blurs the relationship between wheel-spinning, gaming and individual students. How who the student is interacts with gaming and wheel-spinning? Is gaming a trait of student, so that it does not vary much by skill? Or does gaming vary even for the same student by skills he mastered or wheel-spun on?

Looking closer to students' wheel-spinning, we found three types of students. They are characterized by how often they exhibited wheel-spinning. The first type of students is the "Master all skills" type. The students of this type, as suggested by the group name, mastered all skills they are assigned to practice. They are stronger students and did not wheel-spin on a single skill. The second type is "Wheel-spin on all skills" type of students. They are the opposite of the "Master all skills" type. None problems they practiced led them to mastery. They are worrisome and failed to learn anything. The third type of students lies in between. We called them "Master some skills and wheel-spin on other skills". They are average students in performance. In the example we have been using, Ann is one of them, as she mastered Addition and wheel-spun on Subtraction.

By this kind of separation, we split students into groups. No students can be across groups, which allows us to answer two questions. First, is there any difference in gaming between students who mastered all skills and students who wheel-spun on all skills? Which one is more towards non-serious learning behaviors? Second, is there any difference in gaming for the same set of students, while they were practicing for skills they would master and for skills they would wheel-spin on?

In Table 20, we showed these three groups of students in rows. We reported the number of students in each group, and we broke their average gaming scores by mastery and wheel-spinning.

Table 20 Gaming scores breakdown at the student level

Frequency of wheel-spinning	**# of students**	**Avg. gaming score (AGS)**	
		Mastery	**Wheel-spinning**
Master all skills	3452	0.01	N/A
Master some skills and wheel-spin on other skills	1363	0.02	0.10
Wheel-spin on all skills	211	N/A	0.16

It is a very good finding that the majority of the students never wheel-spun. There are 3452 students in the group of "Master all skills". This suggests the effectiveness of ASSISTments, as most students learned well in the system and thus succeeded on every skill they practiced. There are a very small number of students falling in the category of "Wheel-spin on all skills". It is 4% of the students in the data set, which seems not too bad in terms of amount. However, with regard to the extent of the problem, we are deeply concerned. They failed in every single skill they practiced, meaning these 200+ students

learned nothing at all from the system. Why has the system not helped them at all? It seems that there is a blind spot where the system fails completely, and provides no effective help to students.

Comparing these two groups' gaming, we found that the students in the "Master all skills" has lowest gaming score, 0.01 indicating very rare gaming. Since they never wheel-spun on any skills, the gaming score for wheel-spinning problems is null. It is similar for the "Wheel-spin on all skills" group in their gaming score for mastery. It is noticeable that the "Wheel-spin on all skills" students exhibited the most gaming behaviors. Their AGS, 0.16, is higher than the overall AGS of wheel-spinning reported in Table 19. This result suggests that when a student is having a terribly difficult time mastering skills, intensified gaming co-occurs.

To answer the second question - does gaming vary for the same students while they were working on problems leading to mastery and problems leading to wheel-spinning? - we focus solely on the middle group. When they were working on problems leading to mastery, the AGS is 0.02. When they were working on problem leading to wheel-spinning, the AGS drastically increases to 0.1, 5 times of their AGS in mastery (= 0.10/0.02). Note that these two scores were both obtained by the same set of students. The only difference is mastery or wheel-spinning. This remarkable increase suggests that wheel-spinning and gaming are positively associated.

6.2.3 Gaming behavior's trend in wheel-spinning

We have shown that more gaming occurs on problems leading to wheel-spinning. We conjecture that students would probably not realize him wheel-spinning at a sudden moment, and it is even more unlikely to realize it in the earlier practice opportunities. It is sensible for us to conjecture that the realization is probably gradually emerged along with practicing more problems. Since we hypothesized that wheel-spinning is positively associated with gaming, we wanted to examine whether students exhibit more gaming, i.e. non-serious learning behaviors, when they are more clearly aware of them having little hope to understand the skill and inevitably moving towards wheel-spinning.

.

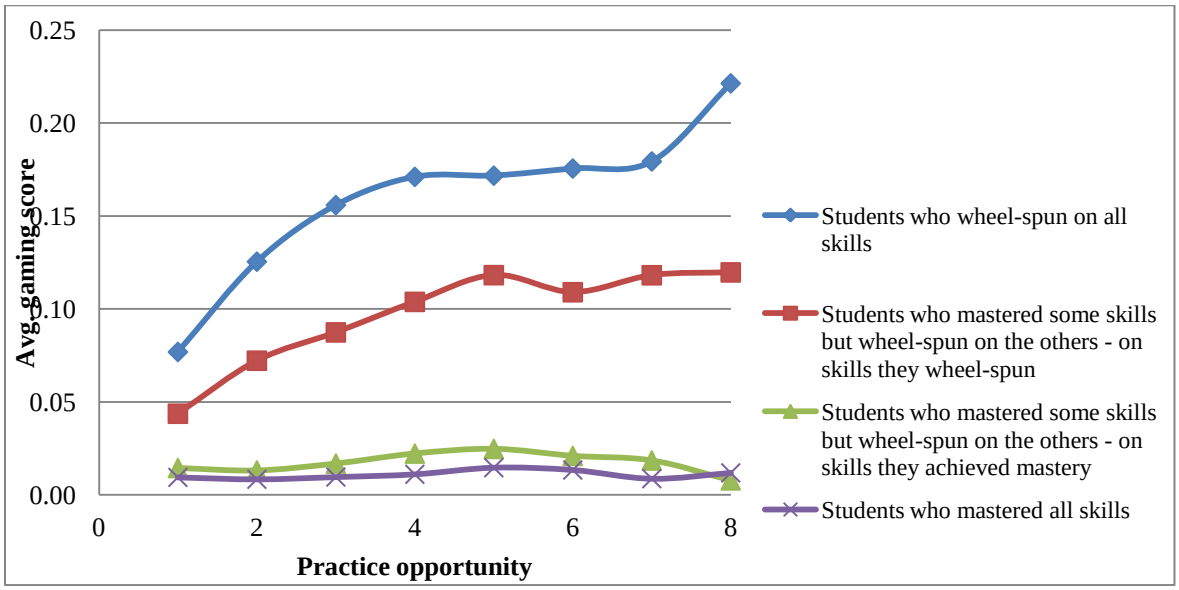


Figure 11 Gaming changes over practice opportunity

Figure 11 shows how the gaming score changes when students practiced more problems. The x-axis represents practice opportunity, ranging from 1 to 8. We did not show gaming scores at the 9th and 10th practice opportunity. The reason is that we found that data became very sparse (e.g. less than 2% data points left at the 9th practice opportunity) for mastery, as most mastery occurs much earlier. The y-axis represents the gaming score averaged across all problems at the practice opportunity.

The top curve was generated based on the problems done by the "Wheel-spin on all skills" students. The bottom curve with X markers was generated based on the problems done by the "Master all skills" students. For the third group of students, who mastered some skills and wheel-spun on the other skills, their problems were divided to two groups based on whether they led to mastery or wheel-spinning. Accordingly, there are two curves generated respectively using the problems leading to mastery and the problems leading to wheel-spinning. The curve with square markers on the top was generated based the problems done by those students and the required skills were not mastered. The curve with triangle markers below that was generated based on the problems in which the required skills were mastered.

Students who mastered all skills have a very steady and flat curve in gaming score, locating at the bottom of the graph, marked with X. In general, these students resulted in the lowest gaming score at each practice opportunity, and they maintained lightest gaming behaviors over time. Another curve at the bottom marked with triangles was generated by using the mastery problems done by the students who had some mastery skills and some wheel-spinning skills. This curve appears a similar trend: flat and steady. The two curves show that when students were progressing to mastery, they were not sensitive to being given more problems. This is probably due to the realization of progression. Feeling more motivated and hopeful to understand the topic might have kept them from gaming.

Different trends are shown in the rest two curves. These curves were generated by wheel-spinning problems. The top curve, with diamond markers, was generated by the wheel-spinning problems done by the students who mastered no skills. The other curve, with square markers, was generated by the wheel-spinning problems done by the students who mastered some skills and wheel-spun on other skills. The two curves appear increasing trends. Within the first 4-5 practice opportunities, the gaming scores of the two groups increase rapidly. We are not certain about the reason. Perhaps, the student quickly senses the difficulty of the skill, and knows that he would have difficulty mastering it, so he gives up and exhibits increasingly more non-serious learning behaviors. Or perhaps, the causality is the other way around. These students have made their mind to treat the practice less seriously and so ended up with wheel-spinning. The two curves experience a section of plateau after the rapid climbing. This shows the students continued to maintain in a stable amount of gaming behaviors. For students who only wheel-spun on some skills, this trend remains to the end. On the contrary, there appears to be another increase occurring to the students who wheel-spun on all skills. We prefer not to over-interpret this difference due to its little significance depending on only one data point.

6.2.4 Two way factorial of gaming behaviors and wheel-spinning

We present a two-way factorial to show how problems are distributed based on gaming and wheel-spinning in Table 21. Each factor is broken down to two cases respectively indicating the behavior occurs and does not occur. The values of the problems are shown with the corresponding percents in parentheses.

Table 21 Two-way factorial of gaming vs. wheel-spinning

		Wheel-Spinning	
		Yes	No
Gaming	Yes	4,288 (3%)	1,278 (1%)
	No	34,583 (25%)	99,671 (71%)

In Table 21, the rightmost column represents the problems leading to mastery. The measurement in the right bottom cell represents the problems where neither wheel-spinning nor gaming occurs. Effective and serious learning is the majority. The cell above it represents the problems in which despite exhibiting some gaming behaviors, students in this sub-category still managed to learn effectively. We can see that

this case is very rare, only 1% of problems. It is possible that the result is due to misclassifications of the gaming detector. Or perhaps, these rarely occurred cases were what is so called non-harmful gaming (Baker, Corbett et al. 2006).

The other sub-category represents problems leading to wheel-spinning. The number of the problems in which no gaming occurred is 34,583, one-fourth of the entire data set. Of all the wheel-spinning problems, these problems are the majority, meaning that when students are struggling mastering skills, they did not obviously give up and start to game. The worst case is when students were gaming and wheel-spinning, the left top cell. There are 4288 problems in this category, which is 3% of the data set.

Another interesting thought is that compared to gaming, we suggest that wheel-spinning deserves more attention. Based on numbers in Table 21, we can calculate the margins, representing how frequently these two problems occur. 28% (= 3% + 25%) problems are wheel-spinning, whereas only 4% (= 3% + 1%) problems are detected as gaming. Wheel-spinning apparently has a much broader scope, affecting a larger number of problems.

7. Research Question 4: Modeling and detecting wheel-spinning

Wheel-spinning seems to be a generic problem that occurs across tutoring systems and hurts a substantial number of students. We have great interests in understanding this phenomenon in terms of whether it occurs randomly or whether it is associated with some factors which could allow us to model wheel-spinning, and hopefully detect it in an early stage. It would be very helpful for students if we could predict whether a student will wind up wheel-spinning in an early stage, as we could provide some coping tactics to prevent wheel-spinning, or at least we could not waste students' time. Detecting wheel-spinning in an early stage is also useful for the system. From an engineering point of view, it gives the system ample time to do decision making and self-adjustment for better tutoring. Our framework is not that wheel spinning is an intractable problem with mastery learning. Rather, wheel spinning is a consequence of a simplified mastery learning framework in many computer tutors, and additional tutoring interventions (possibly including human intervention), should suffice to get the student unstuck in the loop.

Three correct responses in consecutive questions of a skill is used as the mastery criterion. We chose 15 practice opportunities for the CAT and 10 for ASSISTments as the wheel-spinning determination checkpoints. A student could wheel-spin on one skill, while performing fine on another. The unit of classifying wheel-spinning is a student-skill instance. If for a student-skill pair, the student has successfully solved three questions of the skill in a row within his first 15 (10 for ASSISTments) practice opportunities, this student-skill pair is classified mastery. All the problems associated with this pair, i.e. problems done by the student for the skill, are classified as mastery problems. Otherwise, if the student practiced more than 15 problems (10 for ASSISTments), yet did not demonstrate mastery, the student-skill pair and associated problems are classified wheel-spinning.

For this part of work in Section 7, we applied data pre-processing rule #1 (splitting a multiple skill problem to multiple single skill problems), rule #2 (removing indeterminate problems), rule #3 (removing excessive instances after mastery), and rule #4 (removing excessive instances after wheel-spinning determination). We removed excessive problems after mastery and wheel-spinning. This is because that for those problems, the wheel-spinning/mastery state is already determined and no prediction is ever needed after.

Since indeterminate problems do not have mastery/wheel-spinning labels, we removed them from the data set to avoid having too many data points with missing dependent variable values. It is interesting to think about whether the removal would result in selection bias. And if so, in what way does it bias the model? We investigated the reasons why in both systems there were students who only attempted a small number of problems without mastery?

In ASSISTments, there was no hard control on how students should navigate in the system. Students could quit the system before achieving mastery and never came back. Or maybe they attempted several problem sets of different skills around the same time, and jump around among them leaving the skills they thought difficult un-mastered. One sensible conjecture about ASSISTments students is that these

students are less likely to be excellent in Math. Being weaker students, they chose to give up early. If this is the case, removing them seems to introduce selection bias. Without seeing these pieces of evidence, the model may tend to predict more mastery class instances.

In the CAT, due to different tutoring settings, there are a few different reasons resulting in indeterminate problems. The CAT will "promote" the student to the next section without reaching mastery. Promotion is at the section level. There are two cases leading to promotion. The common case is that the student does a certain number of problems without mastery the skills. The rare case is that the system runs out of problems addressing the to-be-mastered skills. Other than promotion, indeterminate problems are also possible due to different mastery and wheel-spinning determination criteria in use. We used 3 correct responses in a row to determine mastery and 15 problems as the wheel-spinning determination checkpoint. Suppose a section contains two skills, and each is associated with 12 problems. The student could be promoted to next section without having 15 attempted problems. In some cases, the CAT used 2 correct responses in a row to determine mastery, and thus the students who demonstrated mastery based on this rule could appear to be indeterminate in our rule. It is possible for them to get 2 correct responses in a row before the 15^{th} problem, and as they have mastered the skills, we would not be able to observe 3 correct responses in a row. The reasons why the CAT data contains the indeterminate have a mixed effect on selection bias. If the promotion is because the student has done a certain number of problems without mastering the skills or the system runs out of problems, it seems reasonable to assume the student does not well understand the skill and would probably end up with wheel-spinning. On the other hand, if the indeterminate is due to different mastery and wheel-spinning criteria, the student could probably do just fine and would demonstrate mastery even under our criteria. Without detailed examination, the effect of removing the indeterminate for the CAT wheel-spinning model on selection bias seems to remain uncertain.

After data pre-processing, the ASSISTments data set contains 133,061 problems done by 5,103 students. There are 104,961 mastery problems and 28,100 wheel-spinning problems. The ratio of the two classes is 3.7 : 1. The CAT data set contains 96,919 problems done by 567 students. There are 76,684 mastery problems and 20,235 wheel-spinning problems. The ratio of the two classes is 3.8: 1.

There are three types of predictions in student modeling (Chi, Koedinger et al. 2011). In this paper, we examined type 1 prediction: how unknown students will perform on the observed problems. We randomly distributed the students in the original data set into three groups of roughly equal size. We used two portions of the data as our training set and the rest portion as our test data. This separation allows us to conduct a three-fold cross-validation. As we separated the data at the student level, in each fold, students in the test data set have no intersection with those in the training data set. Therefore, to the model, these students serve as future students, and the task for the model is to predict whether the student will master a problem or wheel-spin on it. The results averaged across three folds are reported in the following sections.

7.1 Feature Engineering

Feature engineering is crucial in student modeling. Since learning and problem-solving are complex cognitive and affective processes, many automated student models succeeded due to using, manually or automatically, extracted features (e.g. (Nguyen, Horváth et al. 2011), (Baker, Gowda et al. 2011), (Yu, Lo et al. 2010), (Mostow, González-Brenes et al. 2011)).

In our prior work, we built a model specifically for ASSISTments(Gong and Beck (in preparation)). Some features used in the model are specific to ASSISTments and only obtainable by accessing detailed data in the system database. From a scientific point of view, it is worth making efforts to create a model with generality with regard to having the ability to accommodate features generally available in most tutoring systems. In the prior work, we selected features from three aspects: student in-tutor performance, learning attitude and generic factors, such as skill difficulty. We showed that the model can successfully

detect wheel-spinning for ASSISTments. Therefore, we follow the same logic to select features for the generic model.

7.1.1 Student in-tutor performance

Intuitively, whether a student is able to master a skill has much to do with how well the student understands the skill. Therefore, student in-tutor performance is an important aspect to count for. Five features reflect student in-tutor performance in the model, corresponding to the first five features shown in Table 22. These features are skill-oriented. That is to say, observations associated with other skills do not affect the calculations of feature values associated with this skill.

- Correct_Response_Count - The number of problems correctly responded by the student on this skill prior to the current problem. This feature is used in the Performance Factors Analysis model (PFA) (Pavlik, Cen et al. 2009), and found very useful in predicting student performance (Gong and Beck 2011).
- Correct_Response_In_A_Row_Count - The number of problems correctly responded in a row by the student on the skill prior to the current problem. For example, if a student has consecutively correctly answered two problems, for the third problem, the value of this feature is 2. If his answer to the third problem happens to be incorrect, for the fourth problem, the value of this feature is reset to 0.
- Exp_Mean_Response_Time_Z-Score - This variable is derived based on response times of prior questions of the skill. Response time of a question is the length of time spent by the student after being presented the question and before taking the first action. This feature was acknowledged by Anderson et al. that according to the theory of practice curves that fast speed of responding to a question often comes after high accuracy(Anderson 1993). In practice, the winner of the KDD cup 2010 also applied this feature in their comprehensive model (Yu, Lo et al. 2010). Instead of using an absolute value of time, we used exponential mean of the z-scores of response times. The method of calculating this variable is as follows.

We first converted a response time to a z-score. For a problem, we collected all response times across all students who attempted that problem. Then, we calculated the mean and standard deviation of the response times. According to the z-score formula, we calculated a z-score for each problem. Next, we aggregated all prior problems done by the student for the skill together and organized them in chronological order. On this series of z-scores, we used the following formula to calculate the exponential mean, $\gamma * prior\_average + (1- \gamma) * new\_observation$, with $\gamma = 0.7$. The exponential mean is a method of summarizing sequential data, but provides lesser weight to older observations, as prior observations are decayed by γ at each time step.

- Prior_Problem_Count_With_Hint_Request - The number of prior problems of the skill for which the student requested any number of hints.
- Prior_Problem_Count_With_At_Least_5_Hint_Requests - The number of prior problems of the skill for which the student requested 5 or more hints.

The above two features are substitutes of the feature "the number of prior problems for which the student requested a bottom-out hint". This feature has a wide usage in student modeling as an predictor of student performance (Feng, Heffernan et al. 2009; Baker, Goldstein et al. 2010). However, the CAT data do not indicate whether for a problem the student has reached a bottom out hint, nor does it contain the information about the number of hints a problem is equipped. As a result, there is no way to know whether a bottom-out hint is reached, and thus we are not able to get this feature for the generic model. We experimented with a few alternatives. One attempt is that for each problem, we used the maximum number of hints we ever observed in the data set as bottom-out. For example, if for a problem, there is a student who asked 5 hints and no other students ever asked more than 5 hints for the problem, we think of the 5^{th} hint as the bottom-out hint. This attempt did not work expectedly, because using this approach, bottom-out hinting rarely occurs. The problems with bottom-out hinting are very sparse. Therefore, we decided to relax the condition of counting as bottom out hinting. Two alternatives are used

simultaneously. One assumes that any number of hints could be counted as bottom-out hinting, corresponding to the feature "Prior_Problem_Count_With_Hint_Request". The other one is stricter, only greater than or equal to 5 hint requests within a problem are counted as bottom out hinting, corresponding to the feature "Prior_Problem_Count_With_At_Least_5_Hint_Requests".

7.1.2 Learning attitude

Given that a rich number of literatures have demonstrated the connections between learning and student attitude (e.g. (Arroyo and Woolf 2005), (Baker, Corbett et al. 2008), (Craig, Graesser et al. 2004), (Gong, Beck et al. 2010; Liu, Pataranutaporn et al. 2013)), we think of learning attitude as another type of factors possibly associated to wheel-spinning. When building a model specifically for ASSISTments(Gong and Beck (in preparation)), to capture the effects of negative learning attitude, we used three features which were investigated thoroughly for ASSISTments (Gong, Beck et al. 2010). Due to the inability to get access to the action-level information for the CAT data, it is challenging for us to find similar alternatives. Based on what we have in the data set, we focused on two aspects: the speed of responding to a problem and asking for hints in consecutive problems. We designed 6 features reflecting response speed, and 2 features for consecutive hinting. They are shown in the 6th to 13rd row of Table 22. Since these features are intended to represent learning attitude, they are calculated across skills. It is because we believe that student attitude tends to be temporarily stable and does not fluctuate easily along with the skills students are working on.

- Prior_Problem_Count_Fast_Correct - the number of prior problems done by the student across skills which cost too little time to respond and was correctly answered.
- Prior_Problem_Count_Normal_Correct - the number of prior problems done by the student across skills which cost normal time to respond and was correctly answered.
- Prior_Problem_Count_Slow_Correct - the number of prior problems done by the student across skills which cost too long to respond and was correctly answered.
- Prior_Problem_Count_Fast_Incorrect - the number of prior problems done by the student across skills which cost too little time to respond and was incorrectly answered.
- Prior_Problem_Count_Normal_Incorrect - the number of prior problems done by the student across skills which cost normal time to respond and was incorrectly answered.
- Prior_Problem_Count_Slow_Incorrect - the number of prior problems done by the student across skills which cost too long time to respond and was incorrectly answered.

The key to the above first 6 features is to determine what a normal speed response should look like, and accordingly to determine too fast or too slow. We have no access to the content of a problem, so did not calculate an absolute speed based the number of words a problem has and the normal speed of English reading. Therefore, we used a relative standard. As we've already calculated z-scores for each problem. We determined the normalness of the response speed by the value of a z-score. If the student spends an exact average amount of time responding to the problem (i.e. equal to the mean of the response times of the problem), the z-score is 0. If the student spends too little time responding to the problem, the z-score is a negative value. Otherwise, if the student spends too much time, the z-score is a positive value. We picked -1 and 1 as the cut-points to determine a fast (or slow) response. They indicate the actual response time is 1 standard deviation below or above the mean. Problems with z-scores smaller than and equal to -1 are considered costing too little time. Problems with z-scores ranging openly from -1 to 1 are considered normal. Problems with z-scores greater than and equal to 1 are considered too long.

One intuitive method is to simply track the number of prior responses of each category. We grew our idea based on this root too. After a series of experiments, we found that a combined feature representing both response speed and correctness of the response seems most helpful for detecting wheel-spinning. We created 6 bins to categorize problems according to the problem's relative response speed and correctness. The number of past problems of a bin is accumulated across skills, because we considered these features

as learning attitude indicators rather than academic strength indicators, even though the features seem to capture the effects of both.

- Prior_Problem_Count_With_Hint_Request_In_a_Row - The number of consecutive prior problems across skills for each of which the student requested at least 1 hint
- Prior_Problem_Count_With_At_Least_5_Hint_Requests_In_a_Row - The number of consecutive prior problems across skills for each of which the student requested at least 5 hints.

Another type of features is consecutive bottom-out hinting. This feature is not clearly easy to categorize, as it seems to have its linkage across both student in-tutor performance and learning attitude, which blurs the category to which it belongs. Apparently, continuously requesting bottom-out hints in a sequence of problems is a strong indicator, suggesting the problems are beyond the student's ability, and the required skill is not well acquired by the student. Meanwhile, it could also be an attitude issue, as the student perhaps refuses to place any of his own efforts to those problems. We used the italic font for this variable to explicitly acknowledge the lack of our certainty as to where it should be placed in Table 22. For the similar reason, we faced a challenge determining bottom out hinting. We solved this issue by using two alternatives. Instead of sticking on using the last hint, we relax the condition to any number of hints and at least 5 hints. As the two features are primarily for capturing student learning attitude, we calculated them across skills.

7.1.3 Generic factors

We used two independent variables that track generic information about problem solving. They are corresponding to the last eleven rows in Table 22.

- Prior_Problem_Count - The number of problems that have been practiced by the student for this skill. This features is used in the Learning Factors Analysis model (LFA) (Cen, Koedinger et al. 2006) and the Item Response Theory (Hernando 2011). Although the feature is believed to have a linear effect on the correctness of student performance on next problem, we informed our model this variable as a factor instead of a covariant. This is because we did not see a strong reason to assume that there is a linear relationship between this variable and our target, wheel-spinning. It could be true that wheel-spinning does not occur gradually. Rather, maybe after a certain point, the probability of the student being wheel-spinning increases remarkably. Therefore, for each possible value (0 to 14 for the CAT and 0 to 9 for ASSISTments) our model estimates a unique value, indicating the effect to wheel-spinning given that many practices opportunities have passed.
- Skill_ID - the skill identification. This feature reflects that different skills may have differing effects on wheel-spinning. It is treated as factor. Each skill id is taken in the model and a parameter is estimated, which could conceptually be interpreted as how difficult the skill is. Skill difficulty is a widely used feature in almost all types of student models. Research found that skill difficulty is very useful (Gong and Beck 2011) in predicting student performance. Since there are many skills, we do not list their estimated values in the table.

7.2 Model Fitting

We used a logistic regression model to fit the data. The dependent variable is binary, representing mastery or wheel-spinning. The 15 independent variables are as described as above. We showed model fitting coefficients in Table 22. In the first column are the variable names, the same as we described in text. The second and third columns are exponentiations of the coefficients estimated for the CAT and ASSITments data. We categorized the independent variables and showed the labels of the categories in the last column.

Table 22 Exponentiated parameter estimates of the wheel-spinning models

Feature		The CAT	ASSISTments	Feature Type
Correct_Response_Count		1.25*	1.51*	Student In-tutor Performance on *this* skill
Correct_Response_In_A_Row_Count		2.05*	2.76*	
Exp_Mean_Response_Time_Z-Score		0.98	1.14*	
Prior_Problem_Count_ With_Any_Hint_Request		0.92*	0.89*	
Prior_Problem_Count_ With_At_Least_5_Hint_Requests		0.93*	1.05	
Prior_Problem_Count_Fast_Correct			1.09*	Learning Attitude and Academic Strength across *all* skills
Prior_Problem_Count_Normal_Correct		1.01*	1.01*	
Prior_Problem_Count_Slow_Correct		1.01*	1.00	
Prior_Problem_Count_Fast_Incorrect		1.62*	0.97*	
Prior_Problem_Count_Normal_Incorrect		0.99*	0.99*	
Prior_Problem_Count_Slow_Incorrect		0.99	0.99*	
Prior_Problem_Count_ With_Any_Hint_Request_In_a_Row		0.90**	0.83*	
Prior_Problem_Count_ With_At_Least_5_Hint_Requests_In_a_Row		0.85**	0.82*	
Prior_ Problem _Count	0	103.1*	489.6*	Prior Number of Problems
	1	70.7*	256.3*	
	2	49.8*	147.1*	
	3	36.3*	87.3*	
	4	28.1*	49.6*	
	5	21.2*	28.0*	
	6	16.3*	14.9*	
	7	12.5*	7.6*	
	8	9.6*	3.3*	
	9	7.4*	0	
	10	5.3*		
	11	3.8*		
	12	2.7*		
	13	1.8*		
	14	0		
Skill_ID				Skill Difficulty

After a logistic regression, coefficient estimates describe the relationship between the independent variables and the dependent variables. The estimates mean that if we increase 1 unit (or decrease) in the predictor (the independent variable), how much increase (or decrease) in the predicted log-odds of the target can be resulted in, holding all other predictors constant. To facilitate interpretation, instead of showing the coefficients in log-odd units, we showed the odds ratios for the predictors by exponentiating the coefficients in the second and third columns of Table 22. They are the exponentiation of the

coefficients. Estimates greater than 1 indicate positive relationships between the independent variables and the dependent variable. Estimates smaller than 1 indicate negative relationships and estimates equal to 1 indicate no effect. Note that these estimates are obtained for the dependent variable of being mastery. Therefore, a larger estimate greater than 1 can be interpreted as the independent variable has a large positive effect on getting mastery. We used asterisks to denote whether the independent variables are significant at the significance level of 0.05, i.e. the coefficients are significantly different from 0.

Among the features of student in-tutor performance, "Correct_Response_In_A_Row_Count" is recognized as the most important feature. For both models, it is positively associated with mastery, which in opposite has a negative effect wheel-spinning. It is reasonable that more consecutive correct responses is associated with a higher chance to master the skill. Our mastery criterion is 3 correct responses in a row. A value close to 3, for example 2, indicates the student has a good chance to get 3 correct responses, as he has already got 2 correct responses in a row.

For the feature "Exp_Mean_Response_Time_Z-Score", the two models' estimates are opposite in terms of whether the effect is positive or negative. This feature describes how fast the student solved problems recently. A larger value of this independent variable means longer time. The estimated value for the ASSISTments data is 1.14. The greater-than-1 value suggests that when the student does not rush through problems, he is more likely to master the skill. The estimated value for the CAT data is negative, but not significant.

In general, the two features representing bottom-out hinting are estimated to have negative effects to mastery. A student reaching more bottom-out hints generally indicates weaker knowledge on the skill, and thus he is more likely to fail mastery and prone to wheel-spinning. The independent variable "Prior_Problem_Count_With_At_Least_5_Hint_Requests" is not significant in the ASSISTment model. This is probably due to sparse data. In ASSISTments, there are few problems with 5+ hints, and it is even rarer to have consecutive problems with 5+ hint requests.

The 6 ResponseTime-Correctness bins are shown in the 7th to 12th row of Table 22. Although some of the bins are not significant, in general, we still see that incorrect responses are negatively associated with mastery. For the CAT model, the coefficient estimate of the feature "Prior_Problem_Count_Fast_Correct" is null, as no problems were actually in this category. The two features representing "bottom-out hinting in a row" are both estimated as having significant negative effects on mastery in both models.

For the generic factors, we found that there is a clear, yet unexpectedly, linear trend that more problems the student has attempted, more likely he will be wheel-spinning. This is, to some degree, an artefact of our data set, where we removed the data corresponding to the problems practiced after mastery. Therefore, when a student has practiced a large number of problems for a skill and is still left in the data set, he is probably a wheel-spinning student for the skill; while his classmates who have mastered the skill within a small number of problems will not have any data at this moment.

One point worth mentioning is that the much larger values in "Prior_Problem_Count" – 103.1 for 0 prior problem for the CAT model and 489.6 for 0 prior problem for the ASSISTments model – do not necessarily mean they are much more important than the other independent variables. These independent variables are treated as factors; therefore each of them is associated with a coefficient as we see in Table 22. The coefficient estimates are used as in a lookup table. For all problems with the same number of prior problems, say *n* prior problems, the effects of the independent variable are the same, which equal to the value of the estimate. Contrariwise, for the other variables, they are co-variants. Take "Prior_Problem_Count_Normal_Correct" as example, the effect of this variable should be calculated by multiplying the coefficient estimate and the value of this variable together. The resultant product is the one comparable to the coefficient estimate of an independent variable treated as a factor. Therefore, we should not assert that "Prior_Problem_Count" seems to be a dominantly important independent variable.

7.3 Model Evaluation

To evaluate our wheel-spinning model, we created a baseline model. We acknowledged that as more problems were practiced, more likely the student would end up with wheel-spinning. This is probably an artifact of our data set, as we removed excessive data after mastery, data pre-processing rule #3. Specifically, at the last practice opportunity, most data remained at this point are likely to be wheel-spinning problems. Students who have mastered the skills should not have any problems left at this point. The baseline model is a model that just contains the information about the number of prior. The dependent variable remains intact, while the only independent variable is "Prior_Problem_Count". To be fair, this independent variable is also treated as a factor in a logistic regression.

We investigated how the models performed in fitting training data. How well a model fits training data describes how strong the associations are between the extracted features and wheel-spinning. We also evaluated the models from the perspective of predictive accuracy. Model accuracy describes the model ability to generalize on unknown instances from unknown students.

7.3.1 Model accuracy

We evaluated the baseline and wheel-spinning models on the training and test data set of the CAT and ASSISTments data respectively. We present results of the CAT in Table 23 and the results of ASSISTments in Table 24. The two tables are in the same format. We used three metrics, percent correct, AUC and R^2 to measure model performance.

The metric, Percent Correct, is straightforward. It is the percent of correct model predictions. We chose 0.5 as the cut-off point. For instance, if the model's predicted value, the probability of being mastery, is greater than or equal to 0.5, we marked the instance's estimated category as mastery; otherwise we marked the estimated category as wheel-spinning.

AUC is area under curve of the ROC. AUC measures how well the model is able to classify an instance of a binary category. The possible range of this metric is from 0.5 to 1. 0.5 means a random classification, and 1 means a perfect classification.

Table 23 Model performance on the training and test data set of the CAT

	Baseline Model		Wheel-spinning Model	
	Training	Test	Training	Test
Percent Correct	82%	84%	86%	85%
AUC	0.78	0.77	0.89	0.88
R^2	0.28	0.22	0.40	0.38

We used correlation R^2, calculated based on the residuals between the dependent variable's actual value - a binary value of 0 or 1 - and its predicted value - a decimal value ranging from 0 to 1. This metric, compared to the above two metrics, focuses less on the classification ability, but more on the magnitude difference between the predicted values and true values. The R^2 value ranges between 0 and 1, with 1 indicating a perfect fit between the data and the model estimates and 0 indicating no correlation between the two.

Table 23 shows the measurements of the CAT models. The wheel-spinning model outperforms the baseline model, with evident improvements in AUC and R^2. As AUC measures the model's classification ability and R^2 measures the model's ability to produce predictions close to the target's true values in

magnitude, the wheel-spinning model has superior performances in both aspects. This finding reveals the importance of the selected features. Although our data sets have an artificially-resulted characteristic that more prior problems is positively associated with more wheel-spinning instances, this result shows that only using the number of prior problems to model wheel-spinning does not suffice.

We found the wheel-spinning model fit the training data very well, yielding decent measurements in all three metrics. The AUC value is close to 0.9, suggesting high model accuracy in classification. A good fit on training data also indicates that the extracted features used in the model are very helpful to describe wheel-spinning. Note that it is relatively easy to create a model having a good performance fitting training data well. It is also required that the model must accurately classify records it has never seen before. A good classification model must have low training error as well as low generalization error. Model over-fitting is a common phenomenon that a model fitting the training data too well could have a poor performance in reducing generalization error.

However the wheel-spinning model well accommodated to unknown students. Traditional classification tasks in data mining are targeted to unknown instances, which are not seen by the model in the training process. Our task differs from that in a way that not only the instances in test data sets are not seen by the model before, so are the students who generated those instances. In each of the three metrics, the measurement on the test data is just slightly lower than the one on the training data. For example, the model correctly classified 86% problems in the training data set, and correctly classified 85% problems in the test data set. The good performance of the wheel-spinning model on the test data suggests that the model is not overly complicated favoring little training error. Rather, it has strong generalization ability which allows it to be used for unknown students.

Notice that the measurements in R^2 appear to be less satisfying at the first glance. However, we argue that the model's performance on this metric is acceptable. This metric focuses on magnitude differences between the predicted values and the actual values. The dependent variable, wheel-spinning, has a binary value. We used 1 to represent wheel-spinning and 0 to represent mastery. Suppose we have two predictions for one wheel-spinning data point: 0.95 and 0.75. Both predictions could correctly classify the data point to wheel-spinning, but the former produces higher R^2 than the latter. From this perspective, we can see that for a classification model, getting a high R^2 is very challenging. Moreover, student behavioral models generally have poor performance in this metric (Gong, Beck et al. 2010; Baker, Pardos et al. 2011).

Table 24 Model performance on the training and test data set of ASSISTments

	Baseline Model		Wheel-spinning Model	
	Training	Test	Training	Test
Percent Correct	83%	83%	87%	86%
AUC	0.76	0.76	0.89	0.88
R^2	0.23	0.23	0.42	0.40

Table 24 shows the measurements of the ASSISTments models. The models' performance is similar to the CAT models. Most of the conclusions of how the CAT models perform also apply to the ASSISTments models. Consistent performance of the wheel-spinning model on two systems suggest that, despite without specific tailoring, the wheel-spinning model using features generally obtainable in most tutoring systems is applicable to other systems (model parameters would need to be refit). We call this model a generic model, in contrast to a model using features specifically extracted accommodating to a

system's unique characteristics. We want to be clear that with regard to the wheel-spinning model's generality, we are not over-claiming that a wheel-spinning model trained on a tutoring system's data can be generalized to another tutoring system, although this could form interesting future work.

7.3.2 Model misclassifications

Typically, it is necessary to examine a classifier's misclassifications for each class of the dependent variable. This is particularly important for data where one class is of more interest than the others, or with imbalanced classes, i.e. one or more classes dominate the data set. In our context, we are particularly interested in investigating how the model performs for the wheel-spinning problems, as the ultimate goal of using a student model is to support the tutoring module and drive tutor intervention. The ratios of mastery problems and wheel-spinning problems for the two data set are around 4 : 1, which makes the two data sets have imbalanced class distributions. The accuracy measure, conventionally used to compare classifiers, may not be well suited for evaluating models derived from imbalanced data sets (Tan, Steinbach et al. 2005). Therefore, besides accuracy, we used an alternative metric, confusion matrix, to evaluate the models. A confusion matrix is a generic metric used to visually understand a classifier's misclassifications. It summarizes the number of instances predicted correctly or incorrectly by the classification model. We used 0.5 as the cut-off point to classify test data points. If a problem's predicted probability of being mastery is greater than or equal to 0.5, we marked this problem's estimated class as mastery. Otherwise, we marked the problem as a wheel-spinning problem.

We present the confusion matrix of the CAT model in Table 25 and that of the ASSISTments model in Table 26. The confusion matrices are generated on test data only. The numbers in the tables are obtained by averaging the corresponding numbers of the three folds in cross validation. Specifically, we calculated a confusion matrix for each fold by counting the number of instances belonging to each of the four categories. Three confusion matrices are then averaged to get the reported table. The correct classifications are in the left top cell and the bottom right cell. The misclassifications are in the right top cell and the left bottom cell. We highlight the numbers of misclassifications in italic. We also show the percent of each category in parentheses.

Table 25 The CAT's confusion matrix based on model performance on test data

		Predicted Category	
		Mastery	Wheel-spinning
Actual Category	Mastery	24,128 (74.7%)	*1,433 (4.4%)*
	Wheel-spinning	*3,172 (9.8%)*	3,572 (11.0%)

Table 26 The ASSISTment's confusion matrix based on model performance on test data

		Predicted Category	
		Mastery	Wheel-spinning
Actual Category	Mastery	33,469 (75.5%)	*1,518 (3.4%)*
	Wheel-spinning	*4,390 (9.9 %)*	4976 (11.2%)

The confusion matrices in the Table 25 and Table 26 show that the wheel-spinning model performs very well on the majority category: mastery problems, with around 75% of data correctly classified for both models. The major misclassification occurs to wheel-spinning problems being mistakenly classified as mastery problems. For both models, this type of misclassification is about 10% of the data. The other type of error seems minor, which is round 4% of the data. Poorer model classification ability on minority class is commonly seen in classification tasks dealing with imbalanced data (Tan, Steinbach et al. 2005). The wheel-spinning model seems in lack of sensitivity to distinguish wheel-spinning from normality.

Table 27 Precision and recall on the test data sets of the CAT and ASSISTments data sets

	Precision on mastery	**Recall on mastery**	**Precision on Wheel-spinning**	**Recall on Wheel-spinning**
The CAT	88.4%	94.4%	71.1%	52.7%
ASSISTments	88.4%	95.7%	76.6%	53.1%

To find out how the misclassifications occurred, we used two additional metrics, precision and recall. They are widely used metrics for the applications where successful detection of one of the classes is considered more important than detection of the other classes. Precision determines the fraction of instances that are actually wheel-spinning problems and the classifier has declared as a wheel-spinning problem. Recall measures the fraction of wheel-spinning instances correctly detected by the classifier. We show the measurements in Table 27.

On the majority class, although there is often a trade-off between precision and recall, the classifier resulted in high measurements in both metrics. Precision is 88.4 for both models and recall is around 95%. On the other hand, for the minority category, the wheel-spinning class, the model's classification performance is weaker. The models have moderate precision rates: 71.1% for the CAT model and 76.6% for the ASSISTments model. This means that when being used as a real-time wheel-spinning detector, the model has around 23% - 29% of chance to send a false alarm to the tutoring module. The recall rates are poorer in both models, which are just slightly above 50%, indicating that the model's ability of distinguishing the wheel-spinning problems out of normal problems is weaker.

However, we argue that compared to precision, a low recall rate is more acceptable, especially considering using the classifier as a real-time detector to drive tutor intervention. First, a high precision implies that when the detector is triggered, it has a good chance to be accurate. The number of the unnecessary interruptions it mistakenly delivers, such as proactively providing help to the student, should be minimized. Second, a 50% recall indicates that the detector is possible to provide a preventive strategy to a fair number of students who will become stuck in mastery learning. We do not decline that recall is probably a critical metric, as missing early intervention for students who will really have trouble with mastering a skill later is also a big mistake.

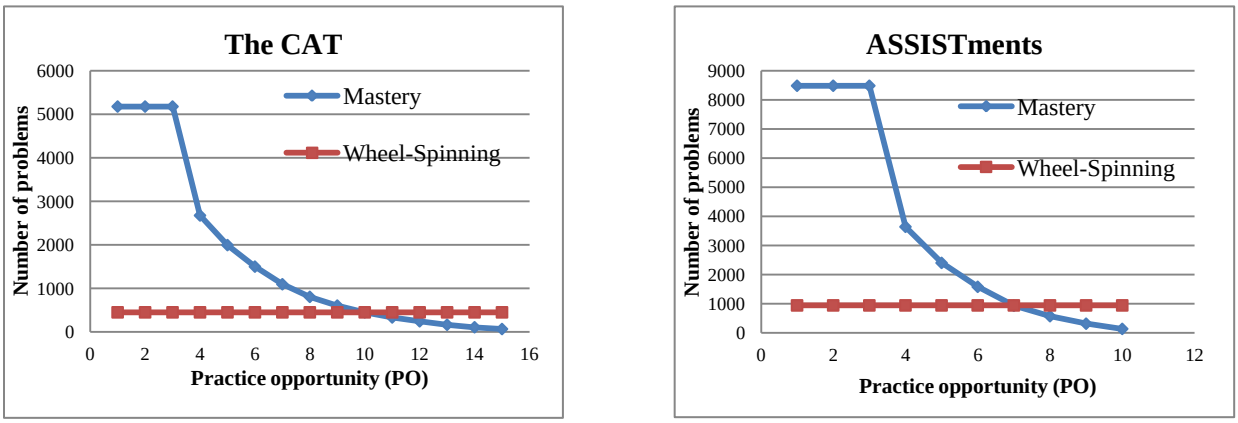
7.3.3 Speed at detecting wheel-spinning

Unlike most behavioral detectors which aim to detect the student's state at present or in a short time, a wheel-spinning detector is supposed to detect something that happens in the long term: will the student end up with wheel-spinning in the future? The most wanted characteristic of this kind of detectors is its ability to detect the construct of interest at an early stage. We want to detect wheel-spinning at an early stage of learning process, even when little information is provided. It is intriguing for us to zoom in on examining how quickly the wheel-spinning detector starts to work. Apparently, our wish is to accurately detect a student's future failure in mastery as soon as possible. We did fine-grained measures of the detector performance at each practice opportunity (we use PO for short in the rest part of this chapter), and plotted the measurements showing how they evolved.

Changes in data size and balance

One point worth emphasizing is that due to data pre-processing rule #3, removing excessive problems after mastery, the size of the data processed by the model keeps shrinking over practice opportunities. For example, suppose in a simple data set, there are only two students, Ann and Tom, and one skill, Addition. Ann practiced 4 problems and achieved mastery after the last problem. Tom practiced 12 problems without mastery. Wheel-spinning is determined after the 10th practice opportunity, and thus Tom wheel-spun on Addition. When the data set is split by practice opportunity, there are 10 sub-data sets corresponding to PO1 – PO10. Each data set contains only the problems done for that practice opportunity. There would be two problems in each of the sub-data sets of PO1 – PO4, where one problem is from Ann and the other is from Tom. From PO5, there is only one problem in the sub-data set until the end. This is because Ann has mastered Addition and she had no more problems remained in the sub-data sets.

Through a closer look, we can figure out that over practice opportunities, mastery problems become fewer and fewer, and wheel-spinning problems remain the same amount all the time from the beginning to the last PO. To facilitate the later interpretations on the model's speed at detecting wheel-spinning, we plotted the changes in class distributions of the data in Figure 12, (a) for the CAT data set and (b) for the ASSISTments data set.



(a) (b)

Figure 12 Test data broken down by practice opportunity (a) the CAT (b) ASSISTments

In Figure 12, the x-axis is practice opportunity (PO). For the CAT data, the values are from 1 to 15, and for the ASSISTments data, the values are form 1 to 10, denoting the first to the last problem done by the students. We plotted the amount of data of two classes, mastery and wheel-spinning, at each practice opportunity. The numbers are averaged across three test data sets used in the three-fold cross-validation. The line with diamond markers represents the amount of mastery problems in the sub-data sets. The other line with square markers represents the amount of wheel-spinning problems in the sub-data set. As our mastery threshold is "3 correct consecutive responses", no students can master a skill before PO3. The amount of mastery instances remains the same for the first three POs. After that, students start to progress to mastery. Due to the data pre-processing rule #3, we see that the number of mastery problems after the 3rd PO starts to decrease gradually. For the wheel-spinning problems, the number of problems remains constant. This is because that according to the wheel-spinning determination criterion, a student is determined as wheel-spinning only if he has done at least 15 (10 for ASSISTments) problems and did not mastered the skill. Therefore, for a wheel-spinning student-skill pair, there must be 15 problems in the data set, and each of which corresponds to a practice opportunity in PO1 – PO15.

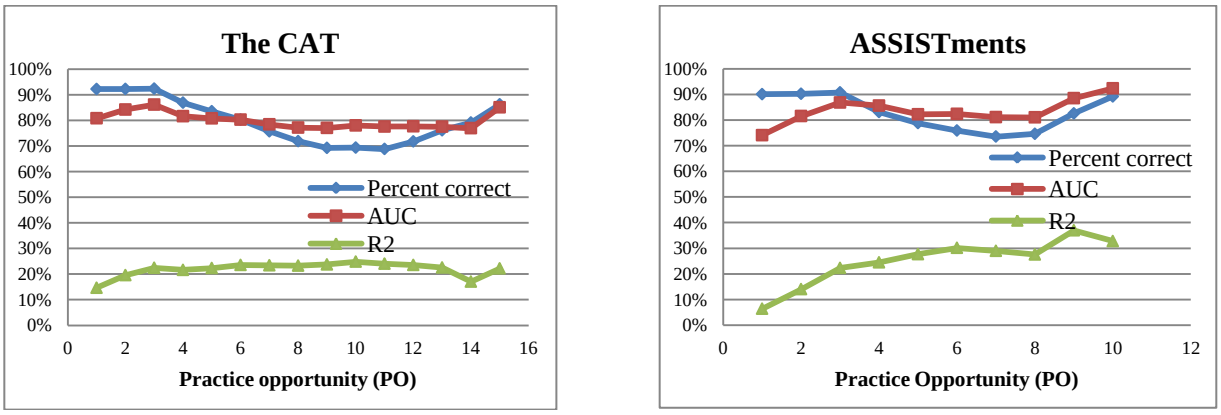
As we can see, a key characteristic of the detector is the data it operates on changes over time. Based on the quantitative relationship between mastery problems and wheel-spinning problems, we discuss the detector's performance in three chronological phases. Phase 1 is the initiation phase, from PO 1 to PO 3,

in which the number of mastery problems remains constant and significantly more than the number of wheel-spinning problems. Phase 2 is the evolution phase, from PO 4 to PO 10 for the CAT data and from PO 4 to PO 7 for the ASSISTments data, in which the number of mastery problems decreases, and gradually drops to the level of wheel-spinning problems. Phase 3 is the termination phase in the rest practice opportunities, in which mastery problems becomes minority.

Model accuracy per PO

We present the measurements of detector performance in Figure 13 and Figure 14, where the x axis is practice opportunity (PO), and the y-axis is the measurement. Figure 13 visualizes the overall accuracy three metrics: percent correct, AUC and R^2.Although conventionally AUC and R^2 are not represented in percentage, we do so for the purpose of showing them in the same chart. Figure 14 focuses only on wheel-spinning. We plotted the changes in precision and recall per PO.

The trends in Figure 13 (a) and (b) are fairly similar, indicating that when used as a detector, the wheel-spinning model performs consistently on the two systems' data. We use the figure for the CAT as example to do interpretations except where the two models perform divergently. We interpret the curves by phase.



(b)

Figure 13 The model overall performance on accuracy, broken down by practice opportunity on test data. (a) The CAT (b) ASSISTments

Phase 1: initiation (PO1 to PO3)

The detector experiences a cold start problem in this phase. In all three metrics, the measurements at the first PO are lower than those at PO2 and PO3. For the readers who are puzzled by why at the very first practice opportunity our detector is still able to make around 90% accurate detections (based on the metric: percent correct), we think that two explanations sound plausible. First, although a problem could be the first PO for a skill, it is not necessarily the first problem ever done by the student. Referring to

Table **22**, we designed the model which contains learning attitude features, which are calculated across skills. A problem being PO1 for a skill may have non-zero values in these features. The model, therefore, can take advantage of the past observations on other skills. Generic factors, such as skill id, could also help prediction. Second, we reason that high accuracy is probably due to the extremely imbalanced data at early POs. Mastery problems significantly dominate the data set at this point. We see that percent correct is even higher than AUC, indicating that the model's actual ability to distinguish the two classes is not as good as suggested by percent correct.

The detector starts to work after PO1. The evidence of the detector's effectiveness is shown by clear climbing in the metrics, AUC and R^2, from PO 1 to PO 3. As more information collected by the detector, the performance climbs up quickly within three observations, as shown in AUC and R^2. In percent correct, the 0.5 cut-off point seems too subtle to respond to changes in classification, therefore the measurements

seem not to change at all. Note that data size remains constant during this period; so does the balance between the two classes. In other words, no data changes should be responsible for the improvement in the detector's accuracy. Therefore, we credit the detector for the improvement.

Phase 2: Evolution (PO4 to PO10 for the CAT, PO4 to PO7 for ASSISTments)

In Phase 2, when we compare Figure 13 (a) and Figure 13 (b), we found similar trends in "Percent Correct" and AUC, but R^2 appears a disagreement. "Percent Correct" and AUC both experience a drop as entering Phase 2. "Percent Correct" deceases continuously and bottoms out at the end of this phase. In AUC, there are some up and downs, but in general it appears a plateau and remains a constant measurement. On the contrary, R^2 of the CAT model has a slightly drop at the beginning of this phase. After that, the measurements start to increase slowly and slightly. R^2 of the ASSISTments model interestingly has an evident increase from the beginning of this phase except a drop at PO7.

During this phase, data-wise, data balance changes drastically. When the data becomes balanced, the detector can no longer take advantage of dominant majority, which typically favors good overall accuracy. As a result, classicisation confusions probably occur more easily. The detector performs satisfyingly in this phase. Although "Percent Correct" drops, we interpret it as losing the benefit of operating on an imbalanced data, picking 0.5 as the cut-off point is no longer sufficient in distinguishing the classes. We see that AUC's value becomes greater than the value of "Percent Correct" at the middle of this phase, PO7. Constant measurements in AUC suggest that during this phase the ability of the model to accurately classify wheel-spinning and mastery is stable. The R^2 values suggest that the model works (slightly for the CAT, and remarkably for ASSISTments) better along with seeing more problems. It can generate probabilities which are increasingly closer in value to the actual value of the dependent variable.

Phase 3: Termination (PO11 to PO15 for the CAT, PO8 to PO10 for ASSISTments)

The CAT model and the ASSISTments model appear differing trends in this phase. For the CAT model, the trend at the end of Phase 2 extends to this phase until the last PO. Before that, the model appears to perform steadily in accuracy, which is suggested by gentle changes in the measurements in AUC and R^2. Notice that during this period - the last half of Phase 2 and the entire part of Phase 3 - data balance keeps changing. The stable measurements in AUC suggest that the model's classification ability is not sensitive to how the data set is composed.

For the ASSISTments model, we found that the performance of the model in accuracy appears to improve immediately after entering Phase 3. All metrics confirm this trend. We also found this similar trend but only at the last PO for the CAT model.

One explanation is that at this point, the data become imbalanced again with wheel-spinning problems as the new majority. The models were also trained using data with this characteristic, and thus could tend to predict problems in this phase towards wheel-spinning. We referred back to

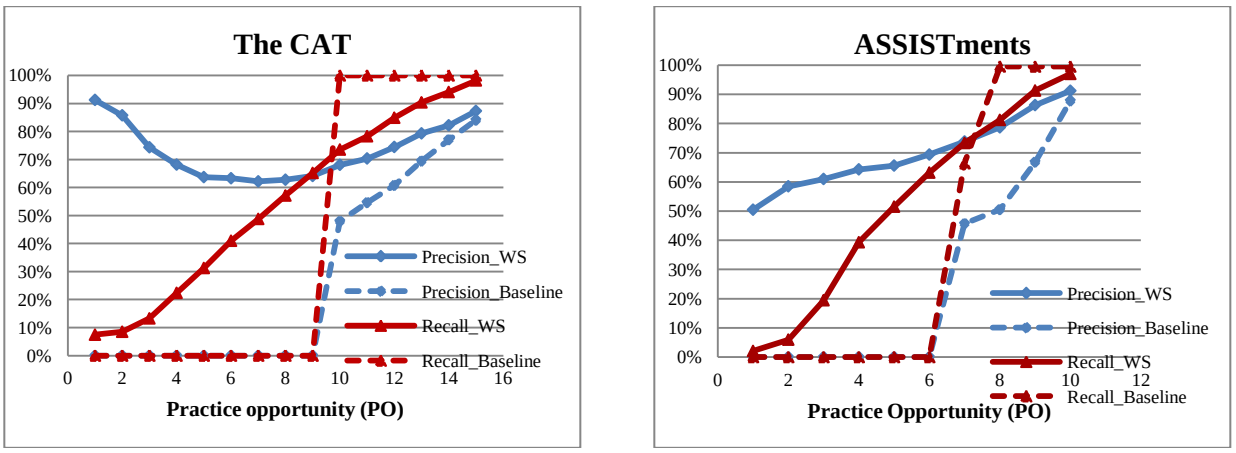
Table **22** and examined "Prior_ Problem _Count". The estimates of this feature are smaller and smaller over practice opportunity. Closer to the last problem where wheel-spinning is determined, greater positive effect on predicting wheel-spinning. To see how quickly the model is towards predicting wheel-spinning with more problems, we calculated the average rate of descend of the estimates of this feature for each phase. For the CAT, the rates of descend in the three phases are 8%, 9%, 41%. For ASSISTments, the rates are 10%, 14% and 55%. We found that the rates of descend in Phase 3 are remarkably higher than the other two, meaning when a problem serves as a practice opportunity in Phase 3, the model has a remarkably increased tendency to predict it as wheel-spinning.

"Percent Correct" seems to benefit from the imbalance data. This metric responds to the changes in data immediately with a better measurement right after the beginning of Phase 3 and a continuing improvement after that. It seems that picking 0.5 suffices to distinguish the two classes when one class dominates the other in quantity. The symmetry that appears at the two ends of the curves in "Percent Correct" is a by-product of the data set with extremely imbalanced classes.

Precision and recall on wheel-spinning per PO

We plotted precision and recall on wheel-spinning at the practice opportunity level in Figure 14. Besides how the wheel-spinning model performs, we also plotted the measurements from the baseline model, which uses the number of prior problems as the sole predictor. The data in use has a clear trend that there are more wheel-spinning problems at larger POs. We set up this baseline model to find out whether the ability to detect wheel-spinning can be acquired from merely knowing which practice opportunity the student is at. The wheel-spinning model has more predictors than just that, and we need to understand whether they are helpful detecting wheel-spinning.

In Figure 14, (a) shows the measurements for the CAT data, and (b) shows them for the ASSISTments data. The x axis is practice opportunity, ranging from 1 to 15 for the CAT and 1 to 10 for ASSISTments. The y axis is the value of the measurements. Take Figure 14 (a) as example, there are four lines: two solid lines and two dash lines. The solid lines are generated by the measurements from the wheel-spinning model. The dash lines are generated by the measurements from the baseline model. The lines with diamond markers are for precision, and the lines with triangle markers are for recall.



(b)

Figure 14 Model performance on wheel-spinning, broken down by practice opportunity on test data. (a) the CAT. (b) ASSISTments

We compared the wheel-spinning model against the baseline model. We found that the CAT and ASSISTments have similar patterns, so we present our interpretations based on the CAT models in Figure 14 (a). From PO1 to PO9, both precision and recall are 0. Note that although we plotted precision as 0, the fact is no single problem was predicted as wheel-spinning and thus precision could not be calculated (due to 0 as denominator). In other words, the baseline model has not ability to detect wheel-spinning, as it predicts all problems of PO1 to PO9 as mastery problems. From PO10, the recall rate of the baseline model directly jumps to 100%, meaning that all true wheel-spinning problems are retrieved. Based on low precision at the same time, we know that at PO10 the baseline model turned suddenly, from predicting 0 wheel-spinning problems to predicting many wheel-spinning problems, which not only cover all true wheel-spinning problems, also mis-include lots of mastery problems. Referring to Figure 12 (a), we can see that PO10 is where data balance biases towards wheel-spinning. The baseline model's precision inclines after PO10, and catches up with the precision of the wheel-spinning model at PO15, where the number of wheel-spinning problems becomes dominant, facilitating higher precision. The baseline model relies on the majority of class to predict, which did not provide reliable performance until PO13, with 100% recall and 70%+ precision. However, we must notice that there are only 2 problems left before wheel-spinning is determined, and thus the baseline model seems not helpful in detecting wheel-spinning at the early stage.

For the wheel-spinning model, we see a similar trend in recall and a very different trend in precision in the CAT and ASSISTments. For recall, as the two curves are similar, we present our interpretations based on the CAT model. The cold start problem is more apparent on wheel-spinning and hurts recall

badly. The detector starts, it has a close-to-0 recall rate. Data balance at this time is poor with mastery problems 8 to 10 times more than wheel-spinning problems. The detector has no ability to identify wheel-spinning problems. However, the detector is able to adapt quickly. At the end of the initiation phase, at PO3, when the detector gathered information from only two prior problems, the recall rate of the CAT model increases to around 13%, and the recall rate of the ASSISTments model rapidly increases to 20%. Note that during this phase, data balance has not changed at all. The models should be credited for the improvements. In the following phases, the recall rates of the two models rise gradually and firmly.

Precision looks very surprising to us. First, it is not hurt as badly as recall by the cold start problem. For ASSISTments, the initial measurement of precision is around 50%. This could be explained by the fact that the model is also informed by across-skills features. Even for a problem at PO1 for a skill, the model might have collected information from other prior problems done by the student for other skills.

More surprisingly, precision is around 90% at PO1 for the CAT model. The corresponding curve has a decreasing trend in the entire initiation phase and the first half of the evolution phase. We were curious about what could have caused this divergence. We examined two objects: the model coefficient estimates and the data, and we found this divergence is due to a dominant skill.

Given all features representing student in-tutor performance are calculated based on the skill, we are sure that none of these features should be responsible for the divergence, as their values at PO1 must be 0. Next, we examined the model estimates of the features representing learning attitude in the CAT and ASSISTments models, as well as data distributions in these features. We found no clear difference worth special attention. In the generic features, we know that "Prior_ Problem _Count" should be 0 for problems at PO1, but the corresponding coefficient estimate denotes its positive effect on predicting mastery, instead of wheel-spinning.

The last feature is "skill_id". We found that in the ASSISTments data set, among all problems declared as wheel-spinning at PO 1, the skills are distributed randomly. In opposite, in the CAT data set, there is a single dominant skill. The problems of this skill are 79% at PO 1, 55% at PO 2 and 41% at PO 3 (the percents are obtained by averaging the corresponding numbers of the three folds in cross validation). We found that most of the problems of this skill are wheel-spinning problems. And, this skill's estimated coefficient is 2.19E-10, representing the model's predicted probability of mastery. This extreme small number, even combined with effects from other independent variables, results in a prediction of wheel-spinning. Therefore, for these problems, the model detects wheel-spinning correctly. Given these problems account for a great proportion in problems declared as wheel-spinning, it results in a high precision. The fraction of the declared wheel-spinning problems associated with this particular skill decreases over practice opportunities, and meanwhile the effects of other features start to kick in, which makes this particular skill lose its domination, causing the precision rate drops.

Both precisions and recalls continuingly improve after the middle of the evolution phase. When the detectors enter the termination phase, at PO10 for the CAT and PO7 for ASSISTments, we see reliable performances. Both precision and recall are close to or above 70%. Note that at this point, the data sets have a good balance where two classes have roughly equal instances. The detectors should get the least impacts from any class being a dominant majority. There are still a couple of problems left before the wheel-spinning determination checkpoints, which means we can still accurately capture a good number of students and intervene to possibly prevent wheel-spinning.

7.4 Refining the scope of the wheel-spinning problem

We have shown the estimated scope of the wheel-spinning problem, which is bounded by the upper and lower bounds. We only know that the true extent of wheel-spinning should lie in between the two bounds, but have no means to locate it. However, we think it is very important to get the knowledge about how serious the wheel-spinning problem is. Is it close to its lower bound, that over 90% student-skill instances in the two systems can achieve mastery, which trivializes the wheel-spinning problem? Or is it close to its upper bound, that the number of student-skill instances failing mastery is over one-third?

We propose to use our wheel-spinning model to estimate the true extent of the wheel-spinning problem. In order to do so, we need a means to handle indeterminate student-skill instances and indeterminate problems. A student-skill pair and its associated problems are identified indeterminate due to lack of observations associated with the skill from the student. This means that this student-skill pair only has problems that are done at early practice opportunities. The evaluations in the previous section showed that even at early practice opportunities, our wheel-spinning detectors works well on classifying mastery and wheel-spinning problems. Therefore, we applied the two models for the CAT and ASSISTments data. Since the goal is to predict for indeterminate problems, we trained the model using the whole data sets without splitting into training and test data. The data sets are the same, as used for generating the upper bound and lower bound in Section 5.

7.4.1 Percent of mastery per practice opportunity

We present our approach by using a concrete example. Suppose there are only two student-skill pairs in a data set: Ann-Addition and Mary-Subtraction. Ann did 5 problems for Addition and mastered the skill at the 5th PO. Mary did 5 problems for Subtraction and did not master the skill. We use 10 problems as the wheel-spinning checkpoint. Ann-Addition is a determinate instance, as mastery has been achieved. Mary-Subtraction is an indeterminate instance due to insufficient observations to get to the checkpoint.

When we calculate the percent of students having mastered the skills, at PO 1 to PO 4, since nether of Ann and Mary has achieved mastery, the percent value is 0. At PO 5, Ann demonstrated mastery, while Mary has not. The percent increases to 50%. From PO 5 to PO 10, the percent value remains 50% constantly, indicating that half of the student-skill pairs have achieved mastery. This approach leaves indeterminate student-skill pairs as they are, resulting in them being counted as wheel-spinning instances. This method takes the pessimistic perspective to estimate wheel-spinning, as all indeterminate instances are counted as wheel-spinning, which yields the upper bound (the worst case) of the wheel-spinning problem.

To calculate the lower bound, we assume that at the last PO of an indeterminate student-skill pair, the student masters the skill. If an indeterminate student-skill pair contains fewer than three POs, mastery is assumed to be demonstrated at PO3. For the above example, From PO 1 to PO 4, the percent is 0, as no mastery has occurred. At PO 5, Ann demonstrated mastery, and Mary is *assumed* to demonstrate mastery. The percent increases to 100%, and the value lasts for the rest POs. This is the lower bound of wheel-spinning, as we optimistically assume that all indeterminate instances would end up with mastery.

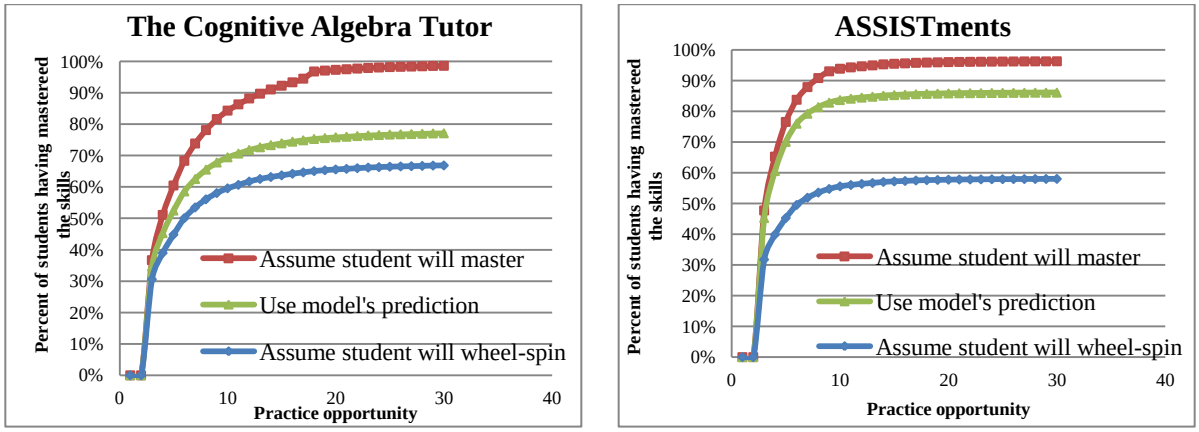
We use model predictions to estimate the true extent of wheel-spinning. We used the prediction for the last problem due to higher accuracy there. From PO1 to PO4, the percent is 0. At PO 5, suppose the prediction of the last problem of Mary-Subtraction is 0.3, meaning a 30% chance to demonstrate mastery. We use this formula $\frac{P_{Ann-Addition}(mastery) + P_{Mary-Subtraction}(mastery)}{\text{number of student-skill pairs}}$ to calculate the percent. $P_{Ann-Addition}(mastery)$ is 1, as it is determinate that Ann mastered Addition. $P_{Mary-Subtraction}(mastery)$ is 0.3 as assumed. There are 2 student-skill pairs in the data set. The percent at PO5 is 65% and has no change after.

As the probability of wheel-spinning complements the probability of mastery, at PO5 the upper bound of wheel-spinning for this example is 50%; the lower bound is 0%; and the model estimated extent is 35%. It suggests that in the best case, wheel-spinning did not occur at all; in the worst case, wheel-spinning occurs to half of the student-skill instances; however its true extent is estimated as 35%.

We plotted how students progress to mastery in Figure 15, (a) for the CAT and (b) for ASSISTments. The x-axis represents the number of practice opportunities and the y-axis represents the percent of student-skill instances having demonstrated mastery.

There are three curves in each chart. The top and the bottom ones respectively depict the lower bound and upper bound of the wheel-spinning problem in the CAT and ASSISTments. The upper bound of the wheel-spinning problem is obtained by treating indeterminate problems pessimistically as wheel-spinning problems, so it is labelled as "Assume student will wheel-spin". The lower bound problem is obtained by treating indeterminate problems optimistically as mastery problems, so it is labelled as "Assume student

will master". The middle curves are generated by using the estimates of the wheel-spinning models for indeterminate problems. The label is "Use Model's Prediction."



(a) (b)

Figure 15 The bounding and estimated curves of (a) the CAT (b) ASSISTments

In Figure 15 (a) for the CAT, we see that the middle curve is closer to the bottom curve, the upper bound of the wheel-spinning problem. At PO 15, at which we determine wheel-spinning, the middle curve reaches 75%. As the complement, the wheel-spinning problem is estimated to affect 25% student-skill pairs in the CAT. In Figure 15 (b), the wheel-spinning problem seems less severe in ASSISTments. The middle curve is closer to the top curve, the lower bound of the wheel-spinning problem. At PO 10, the wheel-spinning determination checkpoint, the middle curve climbs up to around 82%, which leaves 18% of the student-skill pairs wheel-spinning.

There are two points that we want to make in particular. First, the number of student-skill instances suffering from wheel-spinning is substantial in absolute. According to 25% of the CAT data and 18% percent of the ASSISTments data, there are 23,517 student-skill pairs in the CAT data set and 45,787 student-skill pairs in the ASSISTments data set which failed to achieve mastery and fall in wheel-spinning. Second, we reason that wheel-spinning is underestimated by the model. Indeterminate student-skill instances have few problems, which end at early POs. The interpretations to Figure 14 show that at early POs the detectors recall rates are low, and precisions are above 50%. Suppose that there are n problems declared as wheel-spinning. With a precision rate, P, there are $n*P$ problems that are actually wheel-spinning. Given a recall rate, R, the number of all wheel-spinning problems in the data set should be $n*P/R$. Our detector is evaluated having a fair P and a low R at early POs. Take the ASSISTments model as example, at PO 3 in Figure 14, precision is around 60% (P = 60%), and recall is around 20% (R = 20%). According to the formula, the adjusted number of wheel-spinning problems should be 3n, three times of the problems having been declared wheel-spinning.

7.4.2 The amount of time spent on wheel-spinning

Similarly, we applied two detectors for the CAT and ASSISTments data used in Section 5.2 for analyzing the estimated amount of time cost by wheel-spinning. When we calculated time spent on a determinate problem with duration t, if the problem is a mastery problem, the summed amount of time on mastery is increased by t and the summed amount of time on wheel-spinning remains the same; if the problem is a wheel-spinning problem, the summed amount of time on wheel-spinning is increased by t and the summed amount of time on mastery does not change. With an indeterminate problem with duration t, if its probability of mastery is p, we add $t*p$ in the summed amount of time on mastery and $t*(1-p)$ in the summed amount of time on wheel-spinning.

Table 28 shows time spent by the CAT students, including the upper bound, the lower bound and the estimated time. The upper bound is corresponding to the major column entitled as "Pessimistic", and the

lower bound is corresponding to the major column with "Optimistic". In the middle major column, we presented estimated time spent on problems in each group.

Table 28 Time spent by the wheel-spinning group and the mastery group in the CAT

Group	Optimistic		Model Estimated		Pessimistic	
	Time (in hours)	%	Time (in hours)	%	Time (in hours)	%
Mastery	932	72.6%	881	68.7%	543	42.3%
Wheel-spinning	351	27.4%	402	31.3%	740	57.7%

Same as shown in Table 14, the lower bound of time on wheel-spinning is 351 hours and the upper bound is 740 hours. In the middle major column, we see that the estimated time of mastery is 881 hours and the estimated time of wheel-spinning is 402 hours. It seems that the estimated time of wheel-spinning is not terribly worse than its optimistic case. In fact, it is only a 13% deviation on the scale from the optimistic case to the pessimistic case. We calculate this ratio using the difference between the estimated time and the optimistic case (704-351=53) to divide the difference between the pessimistic case and the optimistic case (740-351=389). However, we should see that this small deviation does not make the wheel-spinning problem any less serious. For the CAT data, the optimistic case is 27.4% of the learning time was cost by wheel-spinning. On top of that, based on the model estimates, time spent on wheel-spinning takes 31.1% of the entire student learning time. This stunning high percentage indicates that 1/3 of learning time is wasted on wheel-spinning.

Table 29 shows learning time distributions in ASSISTments students. The left and right major columns are the same as in Table 18. The model estimated time is shown in the middle major column. Similar to the CAT, estimated time spent on wheel-spinning problems is closer to its optimistic case. We calculated the ratio between two differences: the difference between the estimated time and the optimistic case (811-584 = 227), and the difference between the pessimistic case and optimistic case (1741-584 = 1157). The result is 19.6%, meaning the model estimates that students spent 19.6% more time on wheel-spinning than in the optimistic case. Overall, the wheel-spinning problem cost the ASSISTments students 25.4% of their entire learning time and thus they seem to use their learning time a bit more efficiently than the CAT students.

Table 29 Time spent by the wheel-spinning group and the mastery group in ASSISTments

Group	Optimistic		Model Estimated		Pessimistic	
	Time (in hours)	%	Time (in hours)	%	Time (in hours)	%
Mastery	2611	81.7%	2384	74.6%	1454	45.5%
Wheel-spinning	584	18.3%	811	25.4%	1741	54.5%

Notice that for both systems, the models' estimated time is closer to the optimistic cases. We should keep in mind one thing, that the model has modest performance on yielding accurate predictions in magnitude. The measurements in R^2 are around 0.4 for the two models. Two predictions varying in magnitude, for example 0.9 and 0.6, may yield similar classification ability, but can result in very different estimates in time distribution.

8. Future work: Mediating wheel-spinning by WEBsistments

The most important question we are interested to address is what actions we can take about wheel-spinning. It is very meaningful to think about what the effective interventions would look like. The CAT and ASSISTments are shown to be effective tutoring systems and both have applied tutorial strategies that are believed to work very well on students. Nevertheless, it seems inevitable that student wheel-spinning still has substantially negative impact. We are eager to know what else will work. A new exploration might take a long time but certainly worthwhile. In addition, we may acknowledge that the accuracy of the detector is allowed to vary based on the intervention. For example, if the intervention is to suggest an option of providing extra instruction for the student to review, the accuracy of the detector can be set a low bar, as little harm could be resulted in by mistakenly detecting wheel-spinning. To contrast, if the intervention requires the participation of teachers, the detector should be with considerable certainty.

We hypothesize that the failure of the mastery learning framework might be due to the dissatisfaction of the prerequisite in applications of mastery learning. The ACT-R theory emphasizes that procedural knowledge can be acquired through problem solving. In this context, mastery learning can perfectly fit in. However, the prerequisite is that the student have had sufficient declarative knowledge (Self 1988; Ohlsson 1996). As traditional tutoring systems, such as the CAT and ASSISTments, mainly provide coaching help, aiming to help students acquire procedural knowledge, we think a system design providing instructional assistance might be a good start to explore for helping students acquire declarative knowledge and succeed in mastery learning.

We are working on a project, WEBsistments, which is designed and implemented for complementing ASSISTments. Other than providing traditional text-based help, such as hints, scaffolding questions, in this project, we enhanced the system with a new feature. With the new feature, students are allowed to request instructional help delivered from web-based educational sources. Students are allowed to request a web page in any stage of problem-solving, even before their first attempts. When a student clicks the request button, WEBsistments displays a web page associated with the skill tested by the problem. When there are multiple skills required in a problem, the web pages associated with the most advanced skill will be used to select a web page. A student cannot ask for multiple web pages while solving a problem, but he can use original assistance (hints and scaffolding) for the problem.

We did a pilot study where we used machine learning techniques to analyze the effect of viewing webpages on student learning in WEBsistments (Gong, Beck et al. 2012). Our results suggested that when web-based resources were used to in problem-solving, they are associated with more gains in their performances in next problems. In a model where multiple factors were considered simultaneously, the positive effect of web pages on better student performance on next problem is still acknowledged. This result looks promising. It is interesting to link WEBsistments and wheel-spinning together and to find out whether WEBsistments could also provide a good solution to the wheel-spinning problem.

References

Anderson, J. R. (1993). Rules of the mind. Hillsdale, NJ, Lawrence Erlbaum Associates.

Anderson, J. R., Bellezza, et al. (1993). The Geometry Tutor and Skill Acquisition. Rules of the Mind. J. R. Anderson. Hillsdale, NJ: Erlbaum.

Anderson, J. R., F. S. Bellezza, et al. (1993). The Geometry Tutor and Skill Acquisition. Rules of the Mind, Chapter 8. Hillsdale, NJ: Erlbaum.

Anderson, J. R., A. T. Corbett, et al. (1995). "Cognitive tutors: Lessons learned." Journal of the Learning Sciences 4(2): 167-207.

Anderson, J. R. and C. Lebiere (1998). The atomic components of thought. Mahwah, NJ, USA, Lawrence Erlbaum Associates.

Anderson, J. R. and B. J. Reiser (1985). "The LISP Tutor." Byte 10: 159-175.

Arroyo, I., D. G. Cooper, et al. (2009). Emotion Sensors Go To School. The 14th International Conference on Artificial Intelligence in Education: 17-24.

Arroyo, I. and B. Woolf (2005). Inferring learning and attitudes from a Bayesian Network of log file dat. the 12th International Conference on Artificial Intelligence in Education. Amsterdam.

Arroyo, I. and B. Woolf (2005). Inferring learning and attitudes from a Bayesian Network of log file data the 12th International Conference on Artificial Intelligence in Education. Amsterdam.

Ashby, F. G., A. M. Isen, et al. (1999). "A neuropsychological theory of positive affect and its influence on cognition." Psychological Review 106: 529-550.

Baker, R. S., A. T. Corbett, et al. (2006). "Responding to Problem Behaviors in Cognitive Tutors: Towards Educational Systems Which Support All Students." National Association for the Dually Diagnosed (NADD) Bulletin 9 (4): 70-75.

Baker, R. S., A. Goldstein, et al. (2010). Detecting the Moment of Learning. the 10th International Conference on Intelligent Tutoring Systems. V. Aleven, J. Kay and J. Mostow, Springer. Part 1: 25-33.

Baker, R. S., S. Gowda, et al. (2011). Automatically Detecting a Students Preparation for Future Learning: Help Use is Key. the 4th International Conference on Educational Data Mining. M. Pechenizkiy, T. Calders, C. Conatiet al: 179-188.

Baker, R. S. J. d., A. T. Corbett, et al. (2008). More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. the 9th International Conference on Intelligent Tutoring Systems: 406-415.

Baker, R. S. J. d., A. T. Corbett, et al. (2010). Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. the 18th Annual Conference on User Modeling, Adaptation, and Personalization: 52-63.

Baker, R. S. J. d., A. T. Corbett, et al. (2006). Adapting to When Students Game an Intelligent Tutoring System. Proceedings of the 8th International Conference on Intelligent Tutoring Systems, Jhongli, Taiwan.

Baker, R. S. J. d., A. T. Corbett, et al. (2008). "Developing a Generalizable Detector of When Students Game the System." User Modeling and User-Adapted Interaction.

Baker, R. S. J. d., S. Gowda, et al. (2011). Towards predicting future transfer of learning. The15th International Conference on Artificial Intelligence in Education: 23-30.

Baker, R. S. J. d., S. M. Gowda, et al. (2011). Automatically Detecting a Student's Preparation for Future Learning: Help Use is Key. the 4th International Conference on Educational Data Mining: 179-188.

Baker, R. S. J. d., S. M. Gowda, et al. (2012). Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. The 5th International Conference on Educational Data Mining: 126-133.

Baker, R. S. J. d., G. Moore, et al. (2011). The Dynamics Between Student Affect and Behavior Occurring Outside of Educational Software. the 4th bi-annual International Conference on Affective Computing and Intelligent Interaction: 14-24.

Baker, R. S. J. d., Z. Pardos, et al. (2011). Ensembling Predictions of Student Knowledge within Intelligent Tutoring Systems. The 19th International Conference on User Modeling, Adaptation, and Personalization: 13-24.

Beal, C. R., S. Mitra, et al. (2007). Modeling learning patterns of students with a tutoring system using Hidden Markov Models. The 13th International Conference on Artificial Intelligence in Education. Marina del Rey, CA: 238–245.

Beck, J. (2006). Using learning decomposition to analyze student fluency development. ITS2006 Educational Data Mining Workshop, Jhongli, Taiwan.

Beck, J. E. (2007). Difficulties in inferring student knowledge from observations (and why you should care). the AIED2007 Workshop on Educational Data Mining. Marina del Rey, CA: 21-30.

Beck, J. E. and K.-m. Chang (2007). Identifiability: A Fundamental Problem of Student Modeling. International Conference on User Modeling, Corfu, Greece.

Beck, J. E., K. Chang, et al. (2008). Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology. The 9th International Conference on Intelligent Tutoring Systems. Montreal, QC: 383-394.

Beck, J. E. and Y. Gong (2013). Wheel-Spinning: Students Who Fail to Master a Skill. The 16th International Conference on Artificial Intelligence in Education. Memphis, TN: 431-440.

Beck, J. E. and X. Xiong (2013). Limits to accuracy: how well can we do at student modeling? the 6th International Conference on Educational Data Mining (EDM2013). R. C. S. D'Mello, & A. Olney (Eds.) Memphis, TN: 4-11.

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(6): 4-16.

Boeck, P. (2008). "Random Item IRT Models." Psychometrika, 73(4): 533-559.

Boekaerts, M. (1993). "Anger in relation to school learning. ." Learning and Instruction 3: 269-280.

Cen, H., K. Koedinger, et al. (2006). Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. Intelligent Tutoring Systems, Jhongli, Taiwan, Springer.

Cen, H., K. R. Koedinger, et al. (2007). Is over-practice necessary? – Improving learning efficiency with the Cognitive Tutor through educational data mining. the 13th International Conference on Artificial Intelligence in Education Los Angeles, USA: 511-518.

Chang, K.-m., J. E. Beck, et al. (2006). A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. Proceedings of the 8th International Conference on Intelligent Tutoring Systems, Jhongli, Taiwan.

Chi, M., K. Koedinger, et al. (2011). Instructional Factors Analysis: A Cognitive Model For Multiple Instructional Interventions. The 4th International Conference on Educational Data Mining: 61-70.

Chi, M., K. R. Koedinger, et al. (2011). Instructional Factors Analysis: A Cognitive Model For Multiple Instructional Interventions. the 4th International Conference on Educational Data Mining. M. Pechenizkiy, T. Calders, C. Conatiet al: 61-70.

Chin, D. B., I. M. Dohmen, et al. (2010). "Preparing Students for Future Learning with Teachable Agents." Educational Technology Research and Development 58(6): 649-669.

Conati, C. and H. Maclaren (2009). Empirically Building and Evaluating a Probabilistic Model of User Affect. User Modeling and User-Adapted Interaction. 19: 267-303.

Corbett, A. and J. Anderson (1995). "Knowledge tracing: Modeling the acquisition of procedural knowledge." User Modeling and User-Adapted Interaction 4: 253-278.

Corbett, A., L. Kauffman, et al. (2010). "A Cognitive Tutor for Genetics Problem Solving: Learning Gains and Student Modeling." Educational Computing Research 42: 219-239.

Corbett, A. T. (2001). Cognitive computer tutors: Solving the two-sigma problem. International Conference on User Modeling.

Corbett, A. T. (2001). Cognitive computer tutors: Solving the two-sigma problem. Proceedings of User Modeling 2001: 8th International Conference, Sonthofen, Germany, New York: Springer.
Corbett, A. T. and J. R. Anderson (1992). Student modeling and mastery learning in a computer-based programming tutor. The 2nd International Conference on Intelligent Tutoring Systems. C. FRASSON, G. GAUTHIER, G. MCCALLA and (Eds.). New York, Springer-Verlag. 413-420.
Corbett, A. T. and J. R. Anderson (1995). "Knowledge tracing: Modeling the acquisition of procedural knowledge." User Modeling and User-Adapted Interaction 4: 253-278.
Corbett, A. T., L. Kauffman, et al. (2010). "A Cognitive Tutor for Genetics Problem Solving: Learning Gains and Student Modeling." Journal of Educational Computing Research 42(2): 219-239.
Craig, S. D., A. C. Graesser, et al. (2004). "Affect and learning: an exploratory look into the role of affect in learning with AutoTutor." Journal of Educational Media 29: 241-250.
Croteau, E., N. T. Heffernan, et al. (2004). Why are Algebra word problems difficult? Using tutorial log files and the power law of learning to select the best fitting cognitive model. the 7th International Conference on Intelligent Tutoring Systems: 240-250.
D'Mello, S. K., S. D. Craig, et al. (2008). "Automatic Detection of Learner's Affect from Conversational Cues." User Modeling and User Adapted Interaction 18((1-2)): 45-80.
D'Mello, S. K. and A. C. Graesser (2010). "Multimodal Semiautomated Affect Detection From Conversational Cues, Gross Body Language, and Facial Features." User Modeling and User-Adapted Interaction 20(2): 147-187.
D'Mello, S. K., B. Lehman, et al. (2014). "Confusion can be beneficial for learning." Learning and Instruction 29: 153-170.
D'Mello, S. K., R. Taylor, et al. (2007). Monitoring Affective Trajectories during Complex Learning. The 29th Annual Cognitive Science Society. D. S. M. J. G. T. (eds.): 203-208.
Desmarais, M. (2011). Performance Comparison of Item-to-Item Skills Models with the IRT Single Latent Trait Model. the 19th International Conference on User Modeling, Adaption and Personalization
Girona, Spain: 75-86.
Desmarais, M., P. Meshkinfam, et al. (2006). "Learned student models with item to item knowledge structures." User Modeling and. User-Adapted Interaction 16(5): 403-434.
Draney, K., P. Pirolli, et al. (1995). A Measurement Model for a Complex Cognitive Skill. Cognitively diagnostic assessment. P. Nichols, S. Chipman and R. Brennan. Hillsdale, NJ: Erlbaum: 103-126.
du Boulay, B. and R. Luckin (2001). "Modeling human teaching tactics and strategies for tutoring systems." IJAIED Special Issue on Modeling Teaching 12(3): 1-24.
Embretson, S. E. and S. P. Reise (2000). Item Response Theory for Psychologists. Mahwah, Lawrence Erlbaum Associates.
Fancsali, S. E., T. Nixon, et al. (2013). Optimal and Worst-Case Performance of Mastery Learning Assessment with Bayesian Knowledge Tracing. the 6th International Conference on Educational Data Mining Memphis, TN.
Fancsali, S. E., T. Nixon, et al. (2013). Optimal and Worst-Case Performance of Mastery Learning Assessment with Bayesian Knowledge Tracing. The 6th International Conference on Educational Data Mining. S. K. D'Mello, R. A. Calvo and A. e. Olney. Memphis, Tennessee: 35-42.
Farrell, R. G., J. R. Anderson, et al. (1984). An Interactive Computer-Based Tutor for LISP. Fourth National Conference on Artificial Intelligence, Austin, Texas.
Feng, M., N. T. Heffernan, et al. (2009). "Addressing the assessment challenge in an Intelligent Tutoring System that tutors as it assesses." User Modeling and User-Adapted Interaction 19: 243-266.
Fiedler, K. (2001). Affective states trigger processes of assimilation and accommodation. Theories of mood and cognition: A user's guidbook. L. L. M. G. L. C. (Eds.). Mahwah, NJ: Erlbaum: 85-98.
Frick, T. W. (1990). "A comparison of three decision models for adapting the length of computer-based mastery tests." Journal of Educational Computing Research 6(4): 479-513.
Gong, Y. and J. E. Beck (2011). Items, Skills, and Transfer Models: Which Really Matters for Student Modeling? . the 4th International Conference on Educational Data Mining: 81-90.

Gong, Y. and J. E. Beck (2011). Looking Beyond Tranfer Models: Finding Other Sources of Power for Studnet Models. The 19th International Conference on User Modeling, Adaptation and Personalization. Girona, Spain.

Gong, Y. and J. E. Beck ((in preparation)). "Modeling and Detecting Wheel-Spinning: When a Student Will Fail to Learn." UMUAI.

Gong, Y. and J. E. Beck ((in preparation)). "Wheel-spinning: An important problem with the mastery learning framework " International Journal of Artificial Intelligence in Education.

Gong, Y., J. E. Beck, et al. (2010). Using Multiple Dirichlet distributions to improve parameter plausibility. Educational Data Mining 2010. R. S. J. d. Baker, A. Merceron and P. I. J. Pavlik: 61-70.

Gong, Y., J. E. Beck, et al. (2010). The impact of gaming (?) on learning at the fine-grained level. The 10th International Conference on Intelligent Tutoring Systems V. Aleven, J. Kay and J. Mostow: 194-203.

Gong, Y., J. E. Beck, et al. (2010). "How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factors Analysis." International Journal of Artificial Intelligence in Education.

Gong, Y., J. E. Beck, et al. (2011). "How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factor Analysis " International Journal of Artificial Intelligence and Education.

Gong, Y., J. E. Beck, et al. (2012). WEBsistments: Enabling an Intelligent Tutoring System to Excel at Explaining Why Other Than Showing How. the 11th International Conference on Intelligent Tutoring Systems.: 268-273.

Gong, Y., J. E. Beck, et al. (2012). Modeling Multiple Distributions of Student Performances to Improve Predictive Accuracy. The 20th International Conference, UMAP 2012. Montreal, Canada: 102-113.

Gonzalez-Brenes, J. and J. Mostow (2012). Dynamic Cognitive Tracing: Towards Unified Discovery of Student and Cognitive Models. The 5th International Conference on Educational Data Mining: 49-56.

Grafsgaard, J. F., J. B. Wiggins, et al. (2013). Automatically Recognizing Facial Expression: Predicting Engagement and Frustration. the 5th International Conference on Educational Data Mining. S. D'Mello, R. Calvo and A. Olney: 43-50.

Harrigan, M., M. Kravcík, et al. (2009). What Do Academic Users Really Want from an Adaptive Learning System? The First and 17th User Modeling, Adpatation and Personalization. 5535/2009: 454-460

Hawkins, W., N. Heffernan, et al. (2013). Extending the Assistance Model: Analyzing the Use of Assistance over Time. The 5th International Conference on Educational Data Mining.

Hawkins, W., N. Heffernan, et al. (2013). Extending the Assistance Model: Analyzing the Use of Assistance over Time. The 6th International Conference on Educational Data Mining, Memphis, TN.

Heathcote, A., S. Brown, et al. (2000). "The Power Law Repealed:The Case for an Exponential Law of Practice." Psychonomics Bulletin Review: 185-207.

Heffernan, N. T. (2003). Web-based evaluations showing both cognitive and motivational benefits of the Ms. Lindquist tutor. 11th International Conference Artificial Intelligence in Education, Sydney, Australia, IOS Press.

Hembree, R. (1988). "Correlates, causes, effects, and treatment of test anxiety." Review of Educational Research 58: 47-77.

Hernando, M. (2011). Student Procedural Knowledge Inference through Item Response Theory. 19th International Conference on User Modeling, Adaption and Personalization: 426-429.

Hershkovitz, A., R. S. J. d. Baker, et al. (2013). Predicting Future Learning Better Using Quantitative Analysis of Moment-by-Moment Learning. the 6th International Conference on Educational Data Mining: 74-81.

Hidi, S. E. (1990). " Interest and Its Contribution as a Mental Resource for Learning." Review of Educational Research 60(5).

Isen, A. M., K. Daubman, et al. (1987). "Positive affect facilitates creative problem solving." Journal of Personality and Social Psychology. 52: 1122-1131.

Kapoor, A. and R. W. Picard (2005). Multimodal Affect Recognition in Learning Environments. The 13th Annual ACM International Conference on Multimedia: 677-582.

Khajah, M., W. R., et al. (2014). Integrating Latent-Factor and Knowledge-Tracing Models to Predict Individual Differences in Learning. The 7th International Conference on Educational Data Mining. London, U.K.: 99-106.

Koedinger, K. and E. A. Mclaughlin (2010). Seeing language learning inside the math: Cognitive analysis yields transfer. the 32nd Annual Conference of the Cognitive Science Society. Austin, TX: 471-476.

Koedinger, K. R. (2002). "Toward Evidence for Instructional Design Principles: Examples from Cognitive Tutor Math 6." PME-NA XXXIII (the North American Chapter of the International Group for thePsychology of Mathematics Education).

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: 30-43.

Koedinger, K. R., A. C. Corbett, et al. (2012). "The Knowledge-Learning-Instruction (KLI) framework: Bridging the science-practice chasm to enhance robust student learning." Cognitive Science 36(5): 757-798.

Koedinger, K. R. and A. T. Corbett (2006). Cognitive tutors: Technology bringing learning sciences to the classroom. Cambridge handbook of the learning sciences. K. Sawyer. New York, Cambridge University Press.

Koedinger, K. R. and S. Mathan (2004). Distinguishing Qualitatively Different Kinds of Learning Using Log Files and Learning Curves. ITS2004 Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes.

Koedinger, K. R. and M. J. Nathan (2004). "The real story behind story problems: Effects of representations on quantitative reasoning." The Journal of the Learning Sciences, 13: 129-164.

Lee, D. M., M. M. Rodrigo, et al. (2011). Exploring the Relationship Between Novice Programmer Confusion and Achievement. the 4th bi-annual International Conference on Affective Computing and Intelligent Interaction.

Lee, J. I. and E. Brunskill (2012). The impact of individualizing student models on necessary practice opportunities. the 5th International Conference on Educational Data Mining Chania, Greece: 118-125.

Leibowitz, N., B. Baum, et al. (2010). "The exponential learning equation as a function of successful trials results in sigmoid performance." Journal of Mathematical Psychology 54: 338-340.

Li, N., N. Matsuda, et al. (2011). A Machine Learning Approach for Automatic Student Model Discovery. The 4th International Conference on Educational Data Mining: 31-40.

Linnenbrink, E. A., A. M. Ryan, et al. (1999). "The role of goals and affect in working memory functioning." Learning and Individual Differences 11: 213-230.

Litman, D. J. and K. Forbes-Riley (2006). "Recognizing Student Emotions and Attitudes on the Basis of Utterances in Spoken Tutoring Dialogues with both Human and Computer Tutors." Speech Communication 48(5): 559-590.

Liu, Z., V. Pataranutaporn, et al. (2013). Sequences of Frustration and Confusion, and Learning. the 6th International Conference on Educational Data Mining. S. D'Mello, R. Calvo and A. Olney. Memphis, TN: 114-120.

Liu, Z., V. Pataranutaporn, et al. (2013). Sequences of Frustration and Confusion, and Learning. the 6th International Conference on Educational Data Mining: 114-120.

MacLaren, B. and K. R. Koedinger: (2002). When and Why Does Mastery Learning Work: Instructional Experiments with ACT-R "SimStudents". Sixth International Conference on Intelligent Tutoring Systems, Springer-Verlag.

Martin, B., K. Koedinger, et al. (2005). On Using Learning Curves to Evaluate ITS. Twelfth International Conference on Artificial Intelligence in Education, Amsterdam.

Martin, J. and K. Vanlehn (1995). "Student assessment using Bayesian Nets." INternational Journal of Human-Computer Studies 42: 575-591.

Mathan, S. and K. Koedinger (2005). "Fostering the Intelligent Novice: Learning From Errors With Metacognitive Tutoring " Educational Psychologist 40(4): 257-265.

Mostow, J. and G. Aist (2001). Evaluating tutors that listen: An overview of Project LISTEN. Smart Machines in Education. K. Forbus and P. Feltovich. Menlo Park, CA, MIT/AAAI Press: 169-234.

Mostow, J., J. Gonz´alez-Brenes, et al. (2011). Learning classifiers from a relational database of tutor logs. the 4th International Conference on Educational Data Mining. M. Pechenizkiy, T. Calders, C. Conatiet al: 149-158.

Muldner, K., B. Burleson, et al. (2010). "Yes!": Using tutor and sensor data to predict moments of delight during instructional activities. the International Conference on User Modeling and Adaptive Presentation (UMAP' 10): 159-170.

Muldner, K., B. Van de Sande, et al. (2011). "An analysis of students' gaming behaviors in an intelligent tutoring system: predictors and impacts." UMUAI 21:1-2: 99-135.

Newell, A. and P. S. Rosenbloom (1981). Mechanisms of skill acquisition and the law of practice. Cognitive skills and their acquisition. J. R. Anderson. Hillsdale, NJ, John Wiley and Sons: 1-55.

Newell, A. and H. Simon (1972). Human Problem Solving. Englewood Cliffs, N.J., Prentice-Hall.

Nguyen, T., T. Horváth, et al. (2011). Factorization Models for Forecasting Student Performance. EDM 2011: 11-20.

Nkambou, R., J. Bourdeau, et al. (2010). Advances in Intelligent Tutoring Systems.

Nwana, H. S. (1990). "Intelligent Tutoring Systems: An Overview." Artificial Intelligence Review 4: 251-277.

Ohlsson, S. (1994). Constraint-Based Student Modeling. Student Modeling: The Key to Individualized Knowledge-Based Instruction. J. E. Greer, McCalla, G. (eds.): 167-189.

Ohlsson, S. (1996). "Learning from performance errors." Psychological Review 103: 241-262.

Pardos, Z., J. E. Beck, et al. (2008). The Composition Effect: Conjunctive or Compensatory? An Analysis of Multi-Skill Math Questions in ITS. The First International Conference on Educational Data Mining. B. B. (Eds.). Montreal, Canada.: 147-156.

Pardos, Z. A. and N. T. Heffernan (2010). Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. The 18th Proceedings of the International Conference on User Modeling, Adaptation and Personalization.

Pardos, Z. A. and N. T. Heffernan (2011). KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model. the 19th International Conference on User Modeling, Adaptation and Personalization.

Pavlik, P. I. and J. R. Anderson (2008). "Using a Model to Compute the Optimal Schedule of Practice." Journal of Experimental Psychology: Applied 14(2): 101-117.

Pavlik, P. I., H. Cen, et al. (2009). Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. the 2rd International Conference on Educational Data Mining: pp.121-130.

Pavlik, P. I., H. Cen, et al. (2009). Performance Factors Analysis - A New Alternative to Knowledge Tracing. the 14th International Conference on Artificial Intelligence in Education. Brighton, UK: pp. 531-538.

Pavlik, P. I., H. Cen, et al. (2009). Performance factors analysis—A new alternative to knowledge tracing. Proceedings of the 14th International Conference on Artificial Intelligence in Education. V. Dimitrova and R. Mizoguchi. Brighton, England.

Pekrun, R., T. Goetz, et al. (2004). "Beyond test anxiety: Development and validation of the Test Emotions Questionnaire (TEQ)." Anxiety, Stress and Coping 17(287-316).

Pekrun, R., T. Goetz, et al. (2002). "Academic emotions in students' self-regulated learning and achievement: A program of quantitative and qualitative research." Educational Psychologist 37: 91-106.
Porayska-Pomsta, K., M. Mavrikis, et al. (2013). "Knowledge Elicitation Methods for Affect Modelling in Education." International Journal of Artificial Intelligence in Education 22(3): 107-140.
Rai, D., Y. Gong, et al. (2009). Using Dirichlet priors to improve model parameter plausibility. The 2nd International Conference on Educational Data Mining. Cordoba, Spain: 141-148.
Razzaq, L., M. Feng, et al. (2007). Blending Assessment and Instructional Assistance. Intelligent Educational Machines within the Intelligent Systems Engineering Book Series. N. Nedjah, L. d. Mourelle, M. N. Borges and N. N. Almeida: 23-49.
Ritter, F. E. and L. J. Schooler (2002). The learning curve. International Encyclopedia of the Social and Behavioral Sciences. Amsterdam: Pergamon: 8602-8605.
Sao Pedro, M., R. S. Baker, et al. (2013). Incorporating Scaffolding and Tutor Context into Bayesian Knowledge Tracing to Predict Inquiry Skill Acquisition. The 6th International Conference on Educational Data Mining. S. K. D' Mello, R. Calvo and A. Olney: 185.
Sao Pedro, M. A., R. S. Baker, et al. (2013). "Leveraging Machine-Learned Detectors of Systematic Inquiry Behavior to Estimate and Predict Transfer of Inquiry Skill." User Modeling and User-Adapted Interaction 23: 1-39.
Sao Pedro, M. A., J. Gobert, et al. (2012). Assessing the Learning and Transfer of Data Collection Inquiry Skills Using Educational Data Mining on Students' Log Files. The Annual Meeting of the American Educational Research Association. Vancouver, BC, CA.
Self, J. (1988). Bypassing the intractable problem of student modelling. Intelligent Tutoring Systems.
Shute, V. (1995). SMART Evaluation: Cognitive Diagnosis, Mastery Learning and Remediation. 7th World Conference on Artificial Intelligence in Education, Washington, DC, Springer-Verlag.
Tan, J. and G. Biswas (2006). The Role of Feedback in Preparation for Future Learning: A Case Study in Learning by Teaching Environments. Intelligent Tutoring Systems 2006: 370-381.
Tan, P., M. Steinbach, et al., Eds. (2005). Introduction to Data Mining., Addison-Wesley.
Tellegen, A., D. Watson, et al. (1999). "On the dimensional and hierarchical structure of affect." Psychological Science 19: 297-309.
Thai-Nghe, N., T. Horvath, et al. (2011). Personalized forecasting student performance. the 11th IEEE International Conference on Advanced Learning Technologies (ICALT 2011).
Toscher, A. and M. Jahrer (2010). "Collaborative filtering applied to educational data mining." KDD Cup 2010: Improving Cognitive Models with Educational Data Mining.
Trivedi, S., Z. Pardos, et al. (2011). Clustering Students to Generate an Ensemble to Improve Standard Test Score Prediction. The Aritificial Intelligence in Eduction.
VanLehn, K. (2006). "The Behavior of Tutoring Systems." International Journal AI in Education 16(3): 227–265.
VanLehn, K., C. Lynch, et al. (2005). The Andes Physics Tutoring System: Five Years of Evaluations. 12th International Conference on Artificial Intelligence in Education, Amsterdam, IOS Press.
Wang, Y. and J. E. Beck (2012). Using Student Modeling to Estimate Student Knowledge Retention. The 5th International Conference on Educational Data Mining: 200-203.
Wang, Y. and N. Heffernan (2012). Leveraging First Response Time into the Knowledge Tracing Model. The 5th International Conference on Educational Data Mining: 176-179.
Wang, Y. and N. T. Heffernan (2011). The "Assistance" Model: Leveraging How Many Hints and Attempts a Student Needs. Proceedings of the 24th International FLAIRS Conference. Palm Beach, FL.
Wikipedia. "Learning-by-doing (economics)." from http://en.wikipedia.org/wiki/Learning-by-doing_(economics).
Wixon, M., I. Arroyo, et al. (2014). The Opportunities and Limitations of Scaling Up Sensor-Free Affect Detection. The 7th International Conference on Educational Data Mining: 145-152.

Xu, Y. and J. Mostow (2011). Using Logistic Regression to Trace Multiple Subskills in a Dynamic Bayes Net. the 9th International Conference on Educational Data Mining: 241-246.

Xu, Y. and J. Mostow (2013). Using Item Response Theory to Refine Knowledge Tracing. The 5th International Conference on Eduational Data Mining. S. K. D' Mello, R. Calvo and A. Olney: 356-357.

Yu, H.-F., H.-Y. Lo, et al. (2010). "Feature engineering and classifier ensemble for KDD cup 2010." KDD Cup 2010: Improving Cognitive Models with Educational Data Mining.

Yudelson, M., O. Medvedeva, et al. (2008). "A Multifactor Approach to Student Model Evaluation." User Modeling and User-Adapted Interaction 18(4): 349-382.

Zhang, X., J. Mostow, et al. (2007). All in the (word) family: Using learning decomposition to estimate transfer between skills in a Reading Tutor that listens. AIED2007 Educational Data Mining Workshop.