Investigating Knowledge Tracing Models using Simulated Students

Qiao Zhang, Christopher J. MacLellan
College of Computing and Informatics
Drexel University
3675 Market Street,
Philadelphia, Pennsylvania 19104
{qiao.zhang, christopher.maclellan}@drexel.edu

Abstract

Intelligent Tutoring Systems (ITS) are widely applied in K-12 education to help students to learn and master skills. Knowledge tracing algorithms are embedded in the tutors to keep track of what students know and do not know, in order to better focus practice. While knowledge tracing models have been extensively studied in offline settings, very little work has explored their use in online settings. This is primarily because conducting experiments to evaluate and select knowledge tracing models in classroom settings is expensive. We explore the idea that machine learning models that simulated students might fill this gap. We conduct experiments using such agents generated by Apprentice Learner (AL) Architecture to investigate the online use of different knowledge tracing models (Bayesian Knowledge Tracing and the Streak model). We were able to successfully A/B test these different approaches using simulated students. An analysis of our experimental results revealed an error in the implementation of one of our knowledge tracing models that was not identified in our previous work, suggesting AL agents provide a practical means of evaluating knowledge tracing models prior to more costly classroom deployments. Additionally, our analysis found that there is a positive correlation between the model parameters achieved from human data and the parameters obtained from simulated learners. This finding suggests that it might be possible to initialize the parameters for knowledge tracing models using simulated data when no human-student data is yet available.

Introduction

Intelligent Tutoring Systems (ITS) are used within K-12 education to improve learning outcomes. In addition to providing students with scaffolding and feedback, tutoring systems utilize an approach called *knowledge tracing* to estimate what students know and do not know. When combined with a problem selection policy (e.g., Rollinson and Brunskill 2015), knowledge tracing enable tutors to support mastery learning and to focus students practice where it is most needed (i.e., on the skills they do not yet know rather than the skills they already know). While many studies have explored knowledge tracing for offline evaluation (e.g., fitting knowledge tracing models to large, existing data sets), there is comparatively little work on evaluating these algorithms in online settings (e.g., evaluating how well these algorithms estimate students' mastery from just a few data points and decide when to stop giving them additional problems).

We aim to understand which knowledge tracing models yield the greatest mastery learning efficiency in online settings. Additionally, for more complicated models like Bayesian Knowledge Tracing (BKT, Corbett and Anderson 1994), we want to find out how model parameters can be selected before any data is collected. To investigate knowledge tracing models in online settings, the conventional approach would be to run A/B testing in a K-12 classroom. However, this is a time consuming and labor intensive process.

To meet our need for multiple A/B experiments to investigate our knowledge tracing questions, we introduce a novel way of using computational learning models, or simulated student models that learn from interactions with a tutor just like human students do, to simulate our knowledge tracing experiments. We use the Apprentice Learner (AL) architecture (MacLellan et al. 2016), a machine-learning framework that aims to model how humans learn from examples and feedback. By leveraging AL agents to conduct simulated A/B experiments, we can compare different knowledge tracing approaches in online settings.

To explore the feasibility of this approach, we conducted a simulated A/B experiment where we compare BKT to the Streak model, a much simpler model that estimates that students have mastered a skill when they get it right three times in a row. We also compared these approaches with a baseline approach that gives students all available problems (no knowledge tracing). Our simulation experiments show that both BKT and Streak model stop before giving all the problems, but that the BKT model is much more aggressive than the Streak model and seems to assume students have mastered skills much earlier than expected. Upon further inspection, our simulation studies revealed a bug in our underlying implementation of BKT. This finding demonstrates that these simulation studies might serve a valuable role in testing knowledge tracing models before more costly classroom deployments.

In addition to evaluating different knowledge tracing approaches, we also explored the use of simulated data from these experiments to estimate initial parameters for the BKT model. Prior to collecting human data, knowledge tracing parameters are often set to reasonable hand-picked

defaults—an approach that is guesswork at best. A slightly better approach is to run a pilot study where no knowledge tracing is used to gather data to estimate knowledge tracing parameters for subsequent deployments. However, this also requires additional time and labor. In our analysis of simulation data, we found that the BKT parameters estimated from the simulated data have a significant correlation to the BKT parameters estimated from real human data, suggesting that our simulated student approach might provide a more informed way of estimating knowledge tracing model parameters when no human data is available.

Background

Knowledge Tracing

The main purpose of knowledge tracing is to keep track of students' learning on several skills and record time series data. Knowledge tracing models take the Knowledge Components (KCs) and students' response results (correct or incorrect) to predict the possibility that students can correctly solve a new problem containing the same skills.

BKT (Corbett and Anderson 1994) is a well-known knowledge tracing algorithm that can predict if students' have learned a skill given their current knowledge state. For each skill, a student can be in one of the two possible knowledge states: "unknown" or "known". A binary response (correct or incorrect) is generated at each opportunity a student practices a skill (Fancsali, Nixon, and Ritter 2013). It is typically assumed that students never forget what they have mastered (Rollinson and Brunskill 2015). If a student reaches 95% probability being in the known state of a skill, the skill can be marked as being mastered by the student. With these assumptions, the BKT model has four parameters.

- P(L_0): initial probability that a skill is already known by students.
- P(T): probability that students learn the skill.
- P(G): probability that students produce a correct response despite not knowing the skill ("guessing").
- P(S): probability that students produce an incorrect response despite knowing the skill ("slipping").

Researches have created multiple variants of the original BKT model. (Yudelson, Koedinger, and Gordon 2013) introduced an approach to building individualized BKT models that can take student differences in initial mastery probabilities and skill learning probabilities into account. (Nedungadi and Remya 2015) suggested an enhanced BKT model called the PC-BKT (Personalized and Clustered) with individual priors for each student and skill, and dynamic clustering of students based on changing learning ability. Both of the above-mentioned procedures aimed to improve the prediction results comparing to the original BKT model, but limited comparisons have made between BKT and other student models.

In addition to the BKT models, another popular model is the Streak model. Also known as "three-in-a-row" (Heffernan and Heffernan 2014), it is a relatively simple and intuitive model since it only has one parameter (how many correct answers in a row equates to mastery). It was first applied in ASSISTments and the key idea was to keep giving the student questions until some proficiency threshold was reached. The default setting was "three correct in a row" but this could be manipulated by teachers.

Finally, it is worth noting that knowledge tracing is not a required procedure in designing and building tutors. Many studies and tutors just have a fixed problem sequence. However, we hypothesized that knowledge tracing (specifically the "When-to-stop" strategy) is one of the main components of tutoring systems that makes them effective.

Computational Models of Learning

The Apprentice Learner (AL) Architecture is a framework for modeling human learning from demonstrations and feedback in digital environment (Weitekamp et al. 2020). The simulated students we employed in the experiments were implemented within AL. Simulated students (AL agents) can learn as human students do through demonstrations and feedback provided by intelligent tutor systems. AL agents can support tutor technology development (MacLellan et al. 2016). We aim to explore their use for testing different knowledge tracing models.

There are several mechanisms included in the modular design of AL. The *how-learning* mechanism is the first learning mechanism employed when an AL agent receives a training example. This mechanism enables the agent to search through compositions of primitive and prior skills (e.g. addition, subtraction, multiplication and division) to explain the worked examples it receives. In then generalizes these explanations into new skills. In this study we are using the fraction arithmetic tutor, which covers skills such as "converting the denominator" and "converting the numerator" in addition to singular arithmetic skills. Secondly, the *where-learning* mechanism is responsible for generalizing values in the demonstrations it receives to identify relational patterns for binding inputs from tutor interface elements. For example, an AL agent would learn patterns for extracting the first and second numbers from the tutor interface when there is an operator sign between them (MacLellan 2017). Lastly, the AL agents use the *when-learning* mechanism to learn when it is appropriate for a skill to be applied. This mechanism is built upon classification algorithms and the agent updates learned classifiers based on the feedback (correct or incorrect) it receives on its problem solving attempts. For a complete description of AL, see the work by MacLellan and Koedinger (2020).

The majority work in the field of educational data mining focuses on building mathematical, predictive models of learning. In contrast, the Apprentice Learner Architecture can actually perform the tasks by learning from demonstrations and feedback. Therefore, one approach we explore in this paper is applying the computational learning models to simulated students in order to test the algorithms prior to more costly humans studies.

Simulation Studies

In this study, we used the "WhereWhenHowNoFoa" agent in the AL architecture to run the simulations (MacLellan

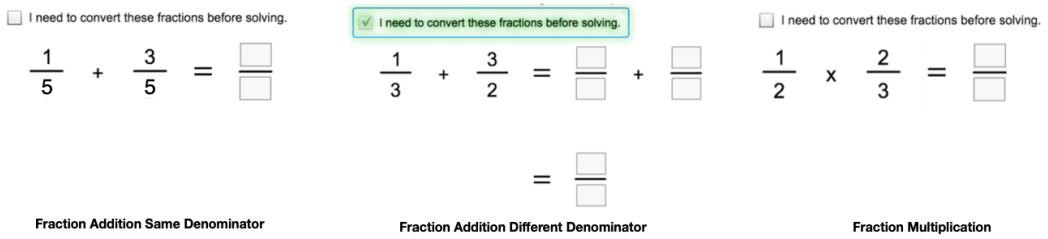

Figure 1: The HTML interface for the fraction arithmetic tutor that the Apprentice Learner agents interact with.

and Koedinger 2020). We created 30 simulated students (AL agents) to solve fraction arithmetic problems (see Figure 1). Three different types of problems were included in the experiment. For each question type, there are 80 unique questions.

- Add Different (AD): add when denominators are different
- Add Same (AS): add when denominators are the same
- Multiplication (M): multiply the two fractions together

We applied five models in our simulation: BKT_default, Random, Streak, BKT_random and BKT_human. The first three models BKT_default, Random, and Streak differ in the way they select the next problem to give to a simulated student. For the parameters in BKT_default, they were set to reasonable defaults based on our prior experience with BKT models. We set P(L_0) to 0.1, P(T) to 0.05, P(G) to 0.05 and P(S) to 0.02. These parameters are identical for each KC. The last two models BKT_random and BKT_human are similar to BKT_default, but have different parameters. BKT_random and BKT_human parameters were obtained from fitting BKT model over "Fraction Addition and Multiplication" dataset accessed via DataShop (Koedinger et al., 2010). and data log generated by Random model in LearnSphere (Koedinger et al. 2010).

During the experiment process, we ran each of the five models over 30 simulated students and analyzed the results. For these experiments, we created KC models with a combination of "Problem Type" and "Selection". There are 14 KC models in our analysis, 8 for type AD, 3 for type AS and 3 for type M (each field for each problem type constitutes a unique skill). As the Additive Factors Model (AFM) is often used to examine learning curves from existing data (Cen 2009), we used pyAFM (MacLellan, Liu, and Koedinger 2015) (a python implementation of AFM) to predict the probability that students can get next step correct with the respective skill at the end of their practice. This provided an independent means for us to estimate how well each knowledge tracing approach did at appropriately recognizing when students had achieved mastery.

Discussion

For each of the five models, we first looked at the numbers of questions included in the training, see Figure 2. The Random model administers all of the 80 problems for each type. The Streak model gives around 24 problems for AD, 19 problems for AS and 16 problems for M. BKT_default model

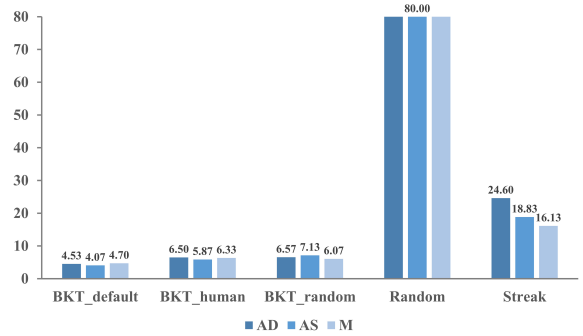

Figure 2: An overview of the number of problems included in the training for each of the five models.

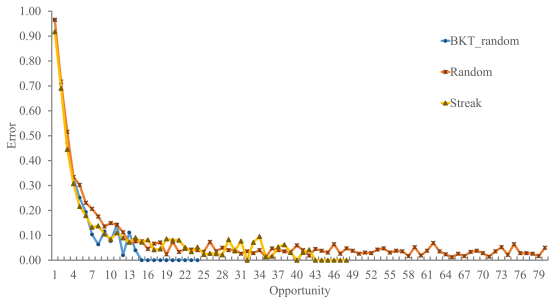

Figure 3: The Learning Curve for BKT_random model, Random model and Streak model. X-axis is the number of opportunities and Y-axis is the error rate.

gives less than 5 problems in each type, while BKT_human model and BKT_random model give around 6 to 7 problems in each type. The number of problems given by the BKT_random model is slightly higher than those given by the BKT_human model given the fact that simulated students do not have prior knowledge as human students do. These findings suggest that BKT is stopping much earlier than the streak model, indicating that it might have better efficiency.

To get a better sense of the overall differences between BKT (BKT_random), Streak, and Random, we plotted the overall learning curves for the data from these models, see Figure 3. We can see from this figure that the BKT model stops giving practice earlier than the Streak model, which subsequently stops giving practice earlier than the Random model. This graph also suggests that BKT seems to stop

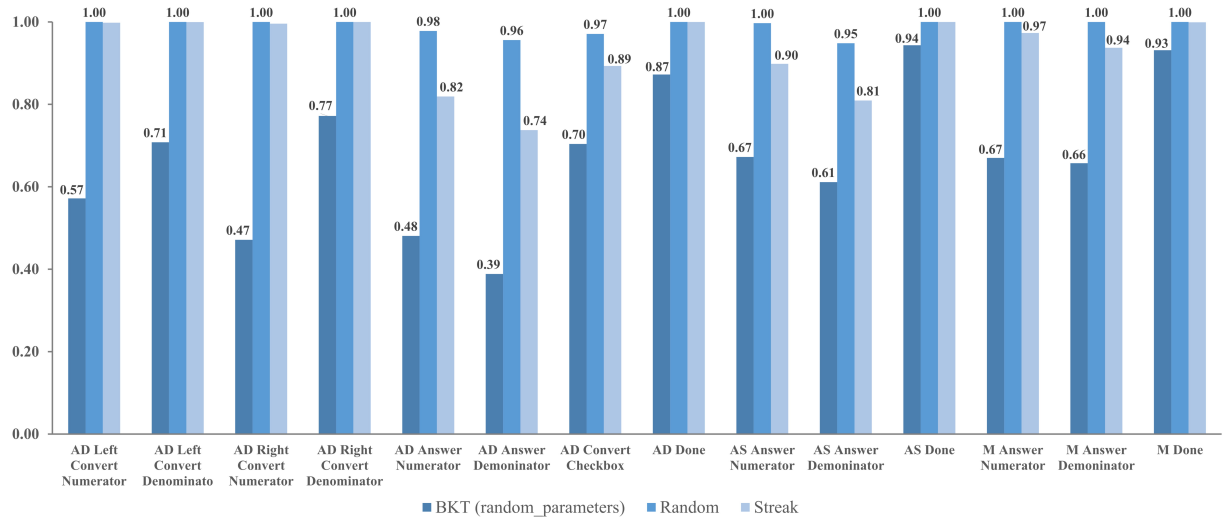

Figure 4: An overview of the predicted probability that students can give correct response when the training ends. X-axis listed each KC and Y-axis is the predicted probability.

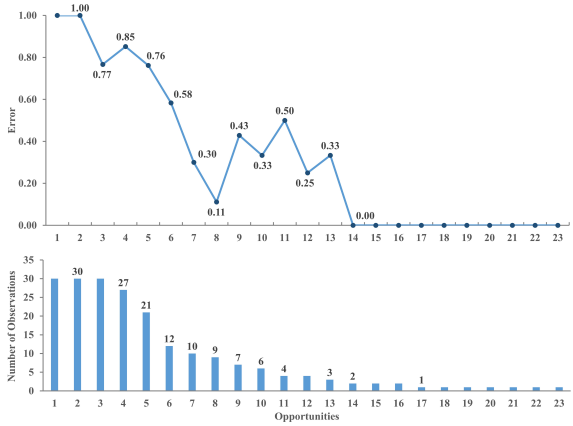

Figure 5: Students' process in completing the training and the related learning curve for the skill "Answer Denominator" in "Add Different" (AD) type. The bars correspond to the number of students at each opportunity and the line corresponds to error rate.

giving students practice right at the point where learning starts to plateau. To evaluate the probability students will get each skill correct at the point where the model assumes they have reached mastery, we applied the AFM model (a type of mixed-effect regression model) to predict performance at the final opportunity. Figure 4 shows the average predicted correctness (across students) at the final opportunity for each skill. Unfortunately, this graph shows that the BKT model appears to be assuming students have mastered skills well before the AFM model thinks they should.

To investigate further, we plotted a learning curve for the skill where BKT did the worst at estimating mastery, the "AD Answer Denominator" skill. Figure 5 shows the learning curve for this skill paired with a bar graph showing how many students have still not achieved mastery at each point. What this graph displays is that BKT is assuming some students achieve mastery starting at the fourth opportunity, even though the students error rates are still quite high. When we investigated this issue further, we discovered that there was an error in the AL architecture's implementation of the BKT model that would incorrectly update *all* the skills related to a particular problem type (AD, AS, M) whenever a any step was taken; e.g., the AD answer denominator step was updated when the AD answer numerator was taken (or any other AD step). While we did not expect this underlying issue with our BKT implementation, we argue that this demonstrates the utility of using simulated agents to test these knowledge tracing approaches prior to a more costly classroom deployment.

To further investigate whether generating BKT parameters from simulation when no human-student data is yet available, we did a correlation analysis between the parameters of BKT_random and BKT_human. Figure 6 shows that there's a positive correlation of around 0.65 in the "Learn" parameter, which means the simulated students generated by Apprentice Learner have a similar learning rate as the human students. We argue that this is one of the harder parameters to set (it determines roughly how much practice each problem should receive). The "pInit" parameter ($P(L_0())$) was near 0 for all skills in the simulated data set because all simulated agents start off without any prior knowledge of fractions. In human data, this parameter will vary based on the learning context (e.g., students in different grades will likely result in different estimates of $P(L_0())$. The "Guess" and "Slip" parameters based on the simulated data were reasonable (both greater than 0), but exhibited no notable correlation with the human guess and slip values. Taken together,

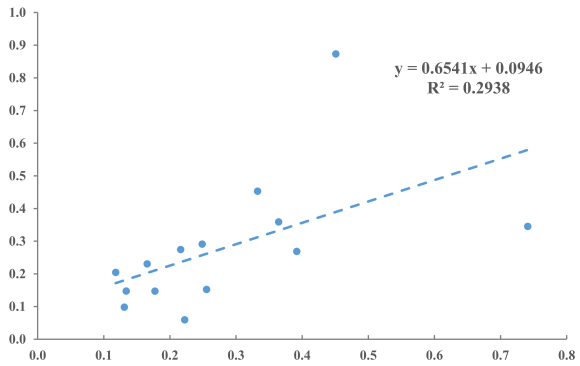

Figure 6: There's a positive correlation in the "Learn" parameter for BKT model between Random-based BKT parameters and Human-based BKT parameters.

we argue that this approach shows promise as a way to identify initial parameters values for the BKT model, but more research is needed to explore this idea.

Related Works

The closest work to ours is the simulation studies conducted by Doroudi and Brunskill (2019). They noted that while the adaptivity provided by knowledge tracing makes it substantially more equitable than treating all the students with the same training load, the knowledge tracing (BKT) algorithm can still be inequitable. They argued that when the student model makes incorrect assumptions about student learning, regardless of the model predicting almost all the students had mastered the skill, lower performing students still tended to achieve lower posttest scores. This result partially coincides with our findings. Figure 5 shows that some students started to drop off after the third opportunity, where the error rate is 77%. While when the algorithm decides training can stop, two students haven't yet mastered the skill. One of the major differences between their approach and ours is that they use statistical models that predict correctness rather than computational models of learning that actual perform the task, as we do with AL agents.

Conclusions and Future Work

From the experiments we have conducted, we found that we were able to successfully apply simulated students to A/B test different knowledge tracing models. When we compared the two knowledge tracing models (BKT and Streak) to a no-knowledge-tracing baseline (Random), we found that BKT gave the fewest problems, streak gave the second fewest, and Random gave the most. Our results suggested that BKT was estimating that students had achieved mastery even though their error rates were still high. Upon further inspection, our simulated experiments revealed that there was an implementation error in our BKT model that was causing it to incorrectly estimate student mastery. We argue that this is a positive outcome for our simulated student approach, suggesting that it was able to successfully highlight issues in our BKT implementation. It is worth noting that this BKT implementation has been used in CMU LearnLab summer school workshops for the past two years and has not yet been detected. In subsequent work, we plan to re-run our analyses with a corrected BKT implementation, to further investigate the BKT model and how it compares to the Streak model in online settings.

Our analysis also found evidence to support the idea that simulated student data might be used to initialize BKT parameters when no human-student data is available. In particular, we found that BKT learning rates estimated from simulated data have a significant correlation to the learning rates estimated from human data. While these initial results are promising, more work is needed to further explore these ideas. In particular, we would like to try running human-subject experiments to compare BKT models initialized using simulated student data to those with default parameters.

We have a number of additional future directions we would like to explore. First, we intend to individualize the AL models to make it better mimic the behaviors of different kinds of students (e.g., high vs. low performing students). Second, we would like to explore ways outputting skill dependency models from AL agents (Maclellan et al. 2020), such as identifying which skills should be practiced before others. Finally, we should move beyond simulation and explore how well our simulated students predict which knowledge tracing approaches yield the best learning (by running comparable human studies).

References

Cen, H. 2009. *Generalized learning factors analysis: improving cognitive models with machine learning*. Ph.D. thesis, Citeseer.

Corbett, A. T.; and Anderson, J. R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge.

Doroudi, S.; and Brunskill, E. 2019. Fairer but not fair enough on the equitability of knowledge tracing. *ACM International Conference Proceeding Series* 335–339. doi: 10.1145/3303772.3303838.

Fancsali, S. E.; Nixon, T.; and Ritter, S. 2013. Optimal and worst-case performance of mastery learning assessment with Bayesian knowledge tracing. *Proceedings of the 6th International Conference on Educational Data Mining, EDM 2013* .

Heffernan, N. T.; and Heffernan, C. L. 2014. The ASSISTments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education* 24(4): 470–497. ISSN 15604306. doi:10.1007/s40593-014-0024-x.

Koedinger, K. R.; Baker, R. S.; Cunningham, K.; Skogsholm, A.; Leber, B.; and Stamper, J. 2010. A data repository for the EDM community: The PSLC DataShop. *Handbook of educational data mining* 43: 43–56.

MacLellan, C. i. J.; Liu, R.; and Koedinger, K. R. 2015. Accounting for Slipping and Other False Negatives in Logistic

Models of Student Learning. *Proceeding of the 8th International Conference on Educational Data Mining, EDM15* 53–60.

MacLellan, C. J. 2017. *Computational models of human learning: Applications for tutor development, behavior prediction, and theory testing*. Ph.D. thesis, Carnegie Mellon University.

MacLellan, C. J.; Harpstead, E.; Patel, R.; and Koedinger, K. R. 2016. The apprentice learner architecture: Closing the loop between learning theory and educational data. *Proceedings of the 9th International Conference on Educational Data Mining, EDM 2016* 151–158.

MacLellan, C. J.; and Koedinger, K. R. 2020. Domain-General Tutor Authoring with Apprentice Learner Models. *International Journal of Artificial Intelligence in Education* 1–42.

Maclellan, C. J.; Maclellan, C.; Edu, D.; and Stowers, K. 2020. Optimizing Human Performance using Individualized Computational Models of Learning. *Advances in Cognitive Systems* (2013): 1–6.

Nedungadi, P.; and Remya, M. S. 2015. Predicting students' performance on intelligent tutoring system - Personalized clustered BKT (PC-BKT) model. *Proceedings - Frontiers in Education Conference, FIE* 2015-February(February). ISSN 15394565. doi:10.1109/FIE.2014.7044200.

Rollinson, J.; and Brunskill, E. 2015. From Predictive Models to Instructional Policies. *International Educational Data Mining Society* .

Weitekamp, D.; Ye, Z.; Rachatasumrit, N.; Harpstead, E.; and Koedinger, K. 2020. Investigating differential error types between human and simulated learners. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 12163 LNAI: 586–597. ISSN 16113349. doi:10.1007/978-3-030-52237-7_47.

Yudelson, M. V.; Koedinger, K. R.; and Gordon, G. J. 2013. Individualized bayesian knowledge tracing models. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7926 LNAI: 171–180. ISSN 03029743. doi:10.1007/978-3-642-39112-5-18.