Between Clones and Snow-Flakes : Personalization in Intelligent Tutoring Systems

Francisco Azeiteiro[1] and Manuel Lopes[1][0000−0002−6238−8974]

INESC-ID Instituto Superior Técnico
Lisbon, Portugal

Abstract. This work improves intelligent tutoring systems by combining the benefits of online personalization of contents with methods that have strong non-personalized long-term optimized policies.
Our hypothesis is that students are very diverse but they are not all completely different from each other. We will generalize previous algorithms by creating a new approach that (1) creates profiles of students based on historical data, (2) in real time is able to recognize the type of student that is being encountered, (3) personalizes their experience taking into account the information of similar students.
We perform several simulations to study the impact on teaching of the amount of data, the diversity of students, and errors in the estimation of parameters.

1 Introduction

In a perfect world students would have a learning experience perfectly personalized to their needs and interests. With so many student per class such personalization is very hard to achieve. With the help of Intelligent Tutoring Systems (ITS)[7] this might be possible as each student can have a set of exercises proposed and ordered according to their particular needs.

A great deal of work has considered how students learn [19] taking into account the human-factors, psychology, classroom dynamics, memory, among many other factors. A seminal work on this topic was the *Knowledge Tracing* framework [10] which builds a detailed cognitive model of the student, of its learning processes by considering a set of independent skills, the probability of learning them and the probability of correct or wrong answer in exercises that relies on those skills. Many recent extensions exist that try to improve the estimation of the student knowledge and also on how to learn the learning parameters of each student [6, 18, 12, 1, 5, 13, 11].

Researchers argue that if we know exactly how students learn then providing teaching materials and experiences would be trivial. Unfortunately requiring an accurate model of the students is too strong an assumption and might even be the wrong approach. Firstly, having an accurate model is very difficult. The amount of data required is too large, after acquiring the data of a particular student for a particular learning problem the ITS is no longer needed as the student finished the curriculum, and there are even computational problems in the identifiability of the parameters [2–4]. And, even if that was possible we would be assuming that all the students are similar and several recent results already showed that using the optimal policy for the average student is not good on average [14, 8] and some degree of personalization is always needed.

Without personalized models, the main approach has been to consider an approximate model of a population of students and compute an optimal policy for that population [15, 17]. This perspective assumes that students are clones of each other in that they share almost all the learning parameters. On the other side some researchers have considered that all students are different from each other [8, 9]. This snow-flake perspective considers that each student has particular difficulties, learning parameters and initial knowledge. These methods have a potential of high personalization but without a model they cannot do long term planning and require too many interaction between the students and the ITS, which is not possible in many applications. Also, instead of relying on computational prohibitive methods such as POMDP [17] they rely on model-free efficient methods such as multi-armed bandits [9].

Hypothesis The hypothesis followed in that students have differences but they also share some similarities. If we can estimate in real time which type of students profile a particular student belongs, it is possible to personalize the learning experience. We are thus in between assuming that all students are similar and that all students are different. Challenges of this approach include accurately placing students on their correct group and making sure that the model parameters for each group of students are optimal.

2 Background

Formalization of a Learning Scenario In a typical ITS we have a set of skills that we want to teach (sometimes also called knowledge components), a set of activities - that can include exercises, reading materials, videos or any other - that a teacher can choose from. For clarity we will use the following example during this article. We consider a list of exercises/activities A1 (Skill1); A2 (Skill1, Skill2); A3 (Skill1, Skill2, Skill3); B1 (Skill4); B2 (Skill4, Skill5); B3 (Skill4, Skill5, Skill6; C1 (Skill7); C2 (Skill7, Skill8) . Each of these exercises requires different skills in order for the student to succeed in solving them. Since there are typically too many activities to be explored it is expected to have a previously defined ZPD, which connects different activities via dependence relationships. Figure 1 shows the activity hierarchy used for the tests. An arrow indicates a prerequisite relationship - A2 is only added to the ZPD when the success rate in A1 passes the expansion threshold. Connections without arrows indicate activities without prerequisites between them which will be proposed when the leftmost activity gets a reward value lesser than or equal to zero - if the reward obtained for performing A3 is ever zero or less then the ZPD expands to include B2, but the opposite cannot happen.

There are some activities that are under a clear sequence of difficulty while others cannot be directly compared. A student might succeed in exercises of type A but not of type B while with another student the opposite may happen.

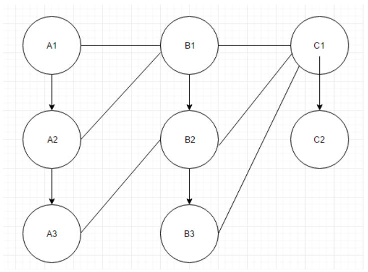


Fig. 1. Relationships between a series of different activities used for testing.

Knowledge Tracing (KT) [10] has been a popular approach to the student model. The original work uses a typical HMM [16] whose hidden states are whether or not the student has acquired a given skill. By solving exercises the students might acquire the associated skill. Whether or not the skill is acquired can be inferred by asking them to solve exercises and measuring the correctness of the answer. It is assumed that once a student learns a skill it cannot be forgotten.

A KT model for a given skill has four parameters: i) $P(L_0)$ - the probability that the student possesses a particular skill prior to the first opportunity to apply it; ii) $P(G)$ - the probability that the student will answer correctly to the question if the skill is in the unlearned state; iii) $P(S)$ - the probability that the student will answer incorrectly if the skill is in the learned state; iv) $P(T)$ - the probability that the skill being learned transitions from the unlearned to the learned state after an opportunity to apply it.

Corbett and Anderson calculate how likely it is that a student understands a particular subject matter using the following formula:

$$P(L_n) = P(L_{n-1}|evd) + (1 - P(L_{n-1}|evd)) \times P(T) \quad (1)$$

Where *evd* is the evidence consisting of the exercise proposed and if it was solved correctly or not.

The main application of KT has been to track if a student has acquired a given skill or not, while the problem of deciding what to present is always considered with other methods.

ZPDES We will extend an approach based on multi-armed Bandits (MAB) due to its reduced computational cost and limited requirements in terms of student model. The ZPDES algorithm [9] is a tutoring model algorithm which requires little information from the student and cognitive models by attempting to estimate student competence online via a reward function. The subjects being taught are organized in skills, and the objective of this algorithm is to make sure that the students learn every skill. As in a typical MAB approach, the quality of each option (in this case each exercise) will increase if it gets a positive reward and reduce if it gets a negative reward. The reward is computed calculating the difference in amount of correct answers C_k of the last $d/2$ samples and the previous $d/2$ samples as follows:$r = \sum_{k=t-d/2}^{t} \frac{C_k}{d/2} - \sum_{k=t-d}^{t-d/2} \frac{C_k}{d-d/2}$

This allows to track if a given exercise is providing improvement in learning or not. An exercise that is always solved correctly does not provide a learning experience.

The following ZPDES parameters directly influence the number of time steps and probability of knowing every skill that the students have at the end of the ZPD: i) β and η values, determining the confidence in a new reward value for $w_a \leftarrow \beta w_a + \eta r$ ii) γ parameter, which determines the exploration rate when calculating the probability of selecting an activity to perform. iii) The d parameter, which determines the number of most recent answers used to calculate the reward value from an activity, as well as the minimum amount of times that an activity needs to be proposed before the ZPD can expand from it. iv) removal threshold - the success rate that an activity needs to have before it can leave the ZPD. v) expansion threshold - the success rate that an activity needs to have before its dependent activities may join the ZPD.

Simulation of Student Populations Different profiles of students will correspond to students that have different initial knowledge and different learning rates. We implement this through two attributes called baselearn (bl) and baseinit (bi), with bl influencing the probability that the student has of learning skills and bi influencing the probability of the student already knowing each skill before solving an exercise containing said skill.

Students are modeled with a KT formalism [10] as follows. Given a skill sk and a student st, the probability of the student learning that skill after each exercise, assuming that the student didn't know the skill before, is given $P(T_{st}) = b_1 bl_{st} + b_2 P(T_{sk})$, where bl_{st} is the student's baselearn attribute, $P(T_{sk})$ is the skill's transition probability, and b_1 and b_2 are two factors that influence the weight of $baselearn_{st}$ and $P(T_{sk})$, with $b_1 + b_2 = 1$. As a result from tests created with the objective of determining the b_1 and b_2 values to be used while making sure that both the student's learning speed and the skill's learning difficulty are taken into consideration, the used values are $b_1 = 0.55$ and $b_2 = 0.45$. A similar equation was used to determine the student's probability of knowing the skill: $P(L_{0st}) = c_1 bi_{st} + c_2 P(L_{0sk})$ In this case the values used for c_1 and c_2 are 0.1 and 0.9 respectively, focusing on how well known the skill's domain is in general. Where $P(L_{0sk})$ is a skill specific probability of initially knowing the skill. The probability of student st learning a skill sk after performing an activity (or exercise) A which includes it is given by: $P(T) = (0.55 \times baselearn_{st} + 0.45 \times P(T_{sk})) \times w_{skA}$ Where w_{skA} is the weight that the skill has in the activity. This equation can be solved for *baselearn* as shown by $baselearn = \frac{P(T) - 0.45 P(T_{sk}) w_{skA}}{0.55 w_{skA}}$ Thanks to the Baum-Welch algorithm, it is possible to estimate the value of $P(T)$, which can then be used to recalculate the student's *baselearn* attribute.

3 The GOZPDES Algorithm

Our new approach contains several components: i) a method to compare students and detect to which group a particular student belongs; ii) a method to provide teaching examples for a particular group. A preliminary step requires information about previous students to create representative profiles of students and their respective optimized teaching experience. The algorithm is summarized in Alg. 1.

Algorithm 1: GOZPDES

```
Offline Phase of Profile Creation from Historical Data;
begin
    Data: K skills to be learned
    Data: Graph of activities
    Data: N exercises from M students
    Result: Cluster of M students into P profiles
    Optimize Teaching parameters for each profile;
    Result: Optimized parameters for each profile
Online Phase with New Student;
begin
    new student;
    $\theta = \theta_{average}$;
    while exercises in ZPD do
        if enough exercises to classify then
            Estimate student parameters;
            Select most similar profile;
            $\theta = \theta_{profile}$
        activity ←ZPDES($\theta$);
        propose activity;
        observe answer and update ZPDES;
```

3.1 Measures of Similarity between Students

Being the goal of this work to provide algorithms that personalize education as much as possible (all students are different) while simultaneously being able to use data from similar situations, (students are similar) we need to have some measures of similarity between students.

One constraint on this measure of similarity between students is that it needs to be data efficient. A trivial way would be to let the student go through the whole learning process and then compare the evolution and results, but then the knowledge learned about the student would not be needed anymore as the student already went through the learning process.

We have 3 main options for comparing students: perform a pre-test to identify the previous knowledge of the student; give an initial set of exercises and compare the success and errors; or give an initial set of exercises but compare based on learning parameters estimated from this initial and limited data. The first option requires a specific assessment period while the other two do not consider such different phase and can work while teaching. As we want to be data efficient we prefer methods that do not require pre-tests.

There are more fundamental differences between the three approaches, the first method compares previous knowledge, the second compares acquired knowledge, but the last one compares the learning process itself. In the second metric (based on success rates) we have a metric that compares students by measuring the distance between two vectors. Each vector contains the percentage of success in each type of exercise. The third metric (based on estimated parameters) uses the percentage of success in each type of exercise but applies an EM algorithm and then compares on the space of the estimated parameters. For our study we used a variant of the EM algorithm to estimate the baselearn (bl) and baseinit (bi) parameters.

The metric based on the parameters is expected to better compare the second but might require too many data points to be useful. To test this, We made several simulations to verify if similar students are classified as similar with each of the methods. With an increasing number of students we noticed that as the number of exercises increases comparing at the parameter level is able to do a better distinctions than comparing directly on the success rate of the exercises.

3.2 Creation of Student profiles and Corresponding Teaching Policy

A profile of students is no more than a group of students that has very similar learning parameters that results in very similar teaching policies. If students are so similar that they will learn with the same sequence then they do not need to be individualized and can be considered clones of each other.

These profiles can be created automatically using clustering methods on the individual learning parameters. For each student in the dataset we can learn its learning parameters. Then a clustering method is done on the learned parameters. As the data is very sparse we do not learn all parameters in the typical BKT scenario but consider only the *baselearn* attribute that influences the learning rate.

Even in a small class of 30 students empirically we observe that there are between 5 to 10 different groups of students. The teaching policy could be optimized in different ways. To ensure further personalization and to make it consistent with the rest of the approach we will used ZPDES[9].

3.3 Teaching Policy Optimization

Each group of students will have a set of parameters that represent their learning behavior. For each student, or group of students sharing the same learning model, the optimal policy for teaching is different. This policy determines how the GOZPDES algorithm will function for each group. Within each group the parameters are chosen as the optimal values that minimize the average number of time steps for that group's student profile while maintaining the probability of learning all skills above a certain threshold. Since the parameters are optimal for the profile, then as long as the classification step is done correctly all students will have approximately optimal parameters.

3.4 Online Identification of Student Profile

With all the previous components we are now ready to accomplish the goal of our work. After a short number of exercises we will be able to identify to which profile each particular student belongs and provide each one with the best teaching strategy available.

Below are different conditions tested for deciding when to perform the *baselearn* estimation. Note that in some methods such estimation is not made.

1. Avg params - A set of average parameters obtained from previous tests are used from start to finish
2. A priori - A priori classification, performing a series of pre-tests and use observations obtained from there to estimate *baselearn* before placing the student into the ZPD. A time penalty is applied for this method to compensate for the time taken in pre-tests
3. A1 - Uses the observations obtained from activity A1 (as shown in Figure 1) once the activity has been removed from the ZPD
4. A1/B1 - Uses the observations obtained from activities A1 and B1 once they have both been removed from the ZPD
5. 20 steps - Uses the observations obtained after 20 time steps (a time step is an attempt at solving an exercise, regardless of correctness)
6. 30 steps+ - Uses the observations obtained after 30 time steps, along with artificially increasing the estimated value slightly

4 Results

Creation of Student Profiles from Data We performed several simulations to evaluate if we can create the student profiles from historical data. To do a systematic study, we generated 1100 students with *baselearn* parameters uniformly from 0.0 to 1.0. Then we let them go through the ITS until the termination condition and the results they have in each exercise will serve as database for the results.

Using a standard k-means for different numbers of classes we can see in Table 1 the obtained results. We can see that for any number of clusters the obtained means are very uniformly spread along the real distribution of data. Also, with the results we will see in the next section small differences in the estimation have little impact on the parameters of the policy, so in our case 3 to 5 student profiles would be ideal.

Table 1. Obtained clusters when running K-means for the estimated baselearn values with k = 3, 5, 7 or 9.

Num clusters	c1	c2	c3	c4	c5	c6	c7	c8	c9
3	0.2061	0.5296	0.8491	-	-	-	-	-	-
5	0.0942	0.3045	0.4913	0.6912	0.9425	-	-	-	-
7	0.0558	0.2031	0.3369	0.4721	0.6127	0.7571	0.9589	-	-
9	0.0545	0.1922	0.3081	0.3964	0.4787	0.5794	0.6878	0.8147	0.9773

GOZPDES Online grouping and Teaching using ZPDES This test measures the average number of time steps and probability of knowing all skills for all the conditions described. Depending on the condition, after a student is classified according to a given profile it will start using the optimal parameters for that profile, before that the average parameters are used. Detailed results are shown in Table 2 while results summarizing the main conclusions can be seen in Figs. 2 and 3.

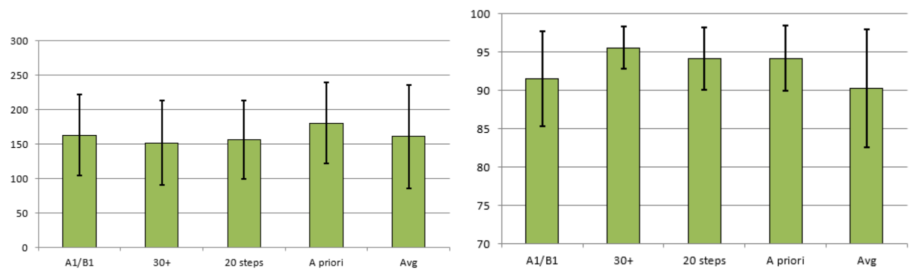
Table 2. Average number of time steps and probability of knowing all skills, starting with default parameters and updating them dynamically at certain points through the model's execution. The conditions tested are performing the estimation after a certain number of time steps, a priori classification, after A1 has been removed from the ZPD, after both A1 and B1 are removed from the ZPD, using the average parameters throughout and the time taken if the classification had been perfect while ensuring a 97% probability of learning all skills.

Condition/Profile	$P_{0.1}$	$P_{0.2}$	$P_{0.3}$	$P_{0.4}$	$P_{0.5}$	$P_{0.6}$	$P_{0.7}$	$P_{0.8}$	$P_{0.9}$
20 steps time	283.9	210.1	177.4	150.6	137.7	123.0	116.4	107.7	102.5
20 steps %skills	98.5	98.2	97.1	96.2	95.8	93.3	91.1	90.5	86.5
A priori time	341.6	255.0	199.0	178.2	154.8	144.7	126.7	118.1	109.7
A priori %skills	99.3	97.5	92.5	97.3	90.6	86.6	98.4	93.8	91.6
A1 time	285.0	216.2	182.4	157.0	141.2	125.7	116.9	106.3	101.0
A1 %skills	98.5	98.1	96.8	96.0	94.5	92.9	88.4	88.5	86.1
A1/B1 time	285.0	223.4	181.0	162.4	143.7	130.4	120.3	114.5	107.0
A1/B1 %skills	99.0	98.0	96.9	95.1	92.5	89.2	86.5	85.0	81.7
Avg params time	284.2	211.8	181.2	157.5	140.6	129.6	120.2	114.3	108.7
Avg params %skills	98.0	97.9	96.8	95.9	91.4	89.2	84.2	81.8	77.1
30 steps+ time	282.5	208.6	173.9	147.4	130.9	117.2	109.0	101.9	95.3
30 steps+%skills	97.5	98.5	97.4	97.7	95.9	96.8	94.1	91.5	90.9
Optimal time	271.9	202.9	160.7	133.5	116.9	105.0	89.5	81.9	76.0

In order to obtain a better idea of how these methods behave on average, the mean and standard deviation for the average number of time steps and probability of learning all skills for each method was plotted using the values obtained for every student profile. Out of the conditions tested, the best one in terms of time and probability of learning is doing online classification after 30 time steps (the 30 steps+ condition).

The results, visible in Figure 2, show that in terms of learning quality the "30+" condition has both a higher mean and lower standard deviation on average than every other method, as well as showing that the average parameters have the lowest mean and highest standard deviation value. This proves that treating all students equally leads to a suboptimal learning experience for most students. It is worth mentioning that every method ensured an average probability of knowing all skills of at least 90%, which is lower than the 97% threshold. This was expected since that threshold was defined for perfect classification of student profiles as opposed to this case's occasionally wrong classification of students which only use approximately optimal parameters as a natural approach of the grouping paradigm.

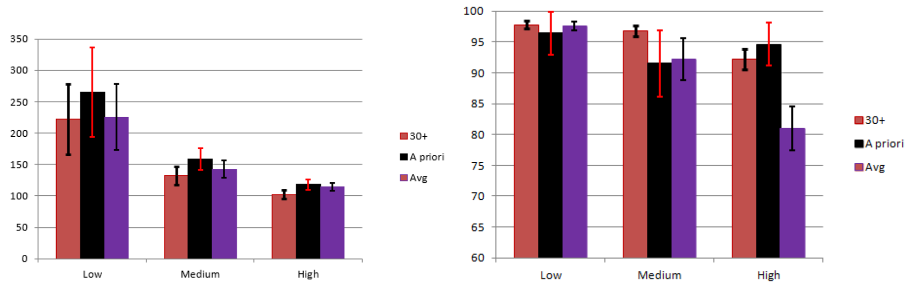
Analysis of the results shows that while the a priori classification does have the highest mean number of time steps, with the 30+ condition having the lowest one, the standard deviation is quite high for every method. This happens because there is a very large difference between the average number of time steps that a low *baselearn* student takes to go through the algorithm in comparison to a high *baselearn* student, ranging from approximately 280 to 100 in Table 2.



a) Time b) Learning Quality

Fig. 2. Mean and standard deviation for the average number of time steps and probability of knowing all skills for each reestimation condition. The "30+" conditions yields the lowest mean number of time steps, along with the highest mean and lowest standard deviation in probability of learning all skills.

When grouping all students together, the time differences between the different methods are not statistically significant. For the learning quality the only difference that is statistically different is the condition 30+ being better than the avg condition (one-tailed t-test, p = 0.035). This proves that an online classification of student is not only possible but also provide gains in learning without



a) Time b) Learning quality

Fig. 3. Comparison between the time and probability of knowing all skills in terms of mean and standard deviation for three conditions. Our method ("30+") guarantees a faster learning and more efficient for all student levels.

any cost in terms of learning time. Also, doing an a-priori classification does not seem to provide enough benefit for the extra cost paid.

Impact on each type of student We do not want any potential gains for one type of student being dependent on leaving others behind. We summarize the results in Figure 3 where we show the time it takes to learn all the skills and for a fixed time how much will they learn.

Although the probability of knowing all skills are all very similar for low *baselearn* students, our method "30+" clearly has the highest probability and lowest standard deviation for the students classified as "Medium". For the high *baselearn* group, however, the a priori method shows a higher average probability of knowing all skills. This happens due to the fact that faster learning students spend less time in the ZPD, meaning that the relative number of time steps that they are using the optimal parameters obtained by methods which require a certain number of time steps for their condition to trigger is greater than for slower students. This would also be the case for a system that treats all students differently, as it would require a much greater number of observations than those used by the online grouping estimation conditions which were attempted, meaning that it would not learn the optimal model parameters in time for them to be useful.

The improvement in terms of the average probability of knowing all skills by the end of the model's execution in comparison to using the average parameters from start to finish - the "Avg" values for this figure - are statistically significant for "High" *baselearn* for both the "30+" (one-tailed t-test, $p = 0.004$) and a priori (one-tailed t-test, $p = 0.005$) methods, along with for the "30+" online grouping estimation condition for "Medium" *baselearn* classifications (one-tailed t-test, $p = 0.043$).

In terms of the average number of time steps to go through the exercises, the results, also shown in Figure 3, show that both the mean and standard deviation decrease for all methods as the average *baselearn* that the students were classified as having increases. The standard deviation for low *baselearn* students remains quite high, as even with this separation of students there is a large difference between the average number of time steps between students with *baselearn* values of, for example, 0.1 and 0.3 (approximately 280 to 180 time steps according to Table 2). No improvement in terms of mean was found to be statistically significant between methods on the same student type (for example, the 30+ method does not have a statistically significant improvement on the average number of time steps in comparison to the a priori method when comparing exclusively either the low, medium or high *baselearn* students). However, every method's transition to a better student type shows statistically significant improvements in comparison to the results on the previous students. The obtained p-values for the one-tailed t-tests were the following: 30+ low to medium: $p = 0.027$; 30+ medium to high: $p = 0.018$; A priori low to medium: $p = 0.034$; A priori medium to high: $p = 0.010$; Avg low to medium: $p = 0.0029$; Avg medium to high: $p = 0.016$.

5 Conclusions and Future Work

In this work we merge the advantages of approaches that provide optimal long-term teaching policies to students (but need to rely on considering all students as clones) with approaches that are fast to adapt to particular student characteristics (but cannot do long-term planning). We also wanted to see if the use of historical data of student could allows us to merge these approaches.

Our results showed that, given sufficient data, we can classify student in less than 30% of the time that they take to go through all the curriculum and then provide an optimal teaching experience after that. We also showed that teaching to the average model, or doing an a-priori test are worse strategies. Our results are stronger when we consider low, medium, and high learning rate students and see that the greater gains are obtained for the high learning rate students, but no group of student loses for being in this systems. For all student they either learn more and faster or are no worse than the standard approach of considering them all equal.

This work was made in simulation to ensure that we could control all the parameters of the process. Knowledge Tracing systems have been used very much in the research and in the real world to the point that we already trust that it is a good model for real students. Considering our results we expect that our system will provide greater gains when used in more heterogeneous classes but without any cost when the classes are homogeneous. Another important aspect is that this system is an extension of the ZPDES system that does not need prior information, but with our new system prior information can be used.

Acknowledgements

This work was partially by the Fundação para a Ciência e a Tecnologia with grants `FCT: UID/CEC/50021/2019` and `PTDC/CCI-COM/32378/2017`.

References

1. Baker, R.S., Corbett, A.T., Aleven, V.: More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In: Intelligent Tutoring Systems. pp. 406–415 (2008)
2. Beck, J.E., Chang, K.m.: Identifiability: A fundamental problem of student modeling. In: User Modeling 2007 (2007)
3. Beck, J.E., Xiaolu Xiong: Limits to Accuracy: How Well Can We Do at Student Modeling? EDM (2013)
4. Beck, J.E., Xiong, X.: Limits to accuracy: How well can we do at student modeling? In: Educational Data Mining (2013)
5. Cen, H., Koedinger, K., Junker, B.: Learning factors analysis–a general method for cognitive model evaluation and improvement. In: Intelligent Tutoring Systems (2006)
6. min Chang, K., Beck, J., Mostow, J., Corbett, A.: A bayes net toolkit for student modeling in intelligent tutoring systems. In: Intelligent Tutoring Systems (2006)
7. Clancey, W.J.: Intelligent Tutoring Systems: A tutorial Survey p. 58 (1986)
8. Clement, B., Oudeyer, P.Y., Lopes, M.: A comparison of automatic teaching strategies for heterogeneous student populations. In: EDM (2016)
9. Clement, B., Roy, D., Oudeyer, P.Y., Lopes, M.: Multi-Armed Bandits for Intelligent Tutoring Systems. Journal of Educational Data Mining **7**(2), 20–48 (2015)
10. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge (1994). https://doi.org/10.1007/BF01099821
11. Dhanani, A., Lee, S.Y., Phothilimthana, P., Pardos, Z.: A comparison of error metrics for learning model parameters in bayesian knowledge tracing. In: Inter. Conf. on Educational Data Mining Workshops (2014)
12. González-Brenes, J.P., Mostow, J.: Dynamic cognitive tracing: Towards unified discovery of student and cognitive models. In: EDM. pp. 49–56 (2012)
13. González-Brenes, J., Huang, Y., Brusilovsky, P.: General features in knowledge tracing: Applications to multiple subskills, temporal item response theory, and expert knowledge. In: Inter. Conf. on Educational Data Mining (2014)
14. Lee, J.I., Brunskill, E.: The Impact on Individualizing Student Models on Necessary Practice Opportunities. Educational Data Mining p. 8 (2012)
15. Psotka, J., Massey, L.D., Mutter, S.A.: Intelligent tutoring systems: Lessons learned. Psychology Press (1988)
16. Rabiner, L.R., Juang, B.H.: An introduction to hidden {M}arkov models. IEEE Signal Proc. Magazine **3**(1), 4–16 (1986)
17. Rafferty, A.N., Brunskill, E., Griffiths, T.L., Shafto, P.: Faster Teaching via POMDP Planning. Cognitive Science **40**(6), 1290–1332 (2016)
18. Villano, M.: Probabilistic student models: Bayesian belief networks and knowledge space theory. In: Intelligent Tutoring Systems (ITS'92) (1992)
19. Wang, M.C., Haertel, G.D., Walberg, H.J.: What Helps Students Learn? Spotlight on Student Success. Educational Leadership (1997)