A Comparison of Automatic Teaching Strategies for Heterogeneous Student Populations

Benjamin Clement
Inria
Bordeaux, France
benjamin.clement@inria.fr

Pierre-Yves Oudeyer
Inria
Bordeaux, France
pierre-yves.oudeyer@inria.fr

Manuel Lopes
Inria
Bordeaux, France
manuel.lopes@inria.fr

ABSTRACT

Online planning of good teaching sequences has the potential to provide a truly personalized teaching experience with a huge impact on the motivation and learning of students. In this work we compare two main approaches to achieve such a goal, POMDPs that can find an optimal long-term path, and Multi-armed bandits that optimize policies locally and greedily but that are computationally more efficient while requiring a simpler learner model. Even with the availability of data from several tutoring systems, it is never possible to have a highly accurate student model or one that is tuned for each particular student. We study what is the impact of the quality of the student model on the final results obtained with the two algorithms. Our hypothesis is that the higher flexibility of multi-armed bandits in terms of the complexity and precision of the student model will compensate for the lack of longer term planning featured in POMDPs. We present several simulated results showing the limits and robustness of each approach and a comparison of heterogeneous populations of students.

1. INTRODUCTION

The current advances and ubiquity of learning and teaching technologies have the potential to improve education accessibility and personalization. Intelligent Tutoring Systems (ITS) have been proposed to make education more accessible, more effective, and as a way to provide useful objective metrics on learning [1].

A major aspect of personalized education is to be able to identify the current level of students and how to address particular difficulties in the student learning process. The goal is to be able to choose online the activity that better addresses the challenges being encountered by each particular student. Even two students with the same knowledge will require different activities to progress further due to their previous experience, cognitive skills or preferences. This is a difficult challenge because as ITS are encountering the students for the first time, it is difficult to know what is the impact of each activity on their progress. A commonly used method is to exploit a population-wide model on how students learn and assume that they are all similar. The personalization in such an approach is limited to adapting to student's knowledge levels but assumes that the impact of each exercise is the same for all students with the same knowledge levels.

Different methods have been proposed to handle this problem. One popular and well-known method is the Partially Observable Markov Decision Process (POMDP) framework which has been proposed in different ways to select the optimal activities to propose to a learner [13]. This framework can find the optimal teaching trajectories for a given teaching scenario model if an accurate student model is provided which is not always possible in practice. The main drawback is the high computational complexity and as a consequence, only the simplest cases can be solved exactly. Another method explored recently to select optimized activities is to use the Multi-Arm Bandit (MAB) framework to personalize sequences of pedagogical activities [6]. These methods optimize learning in the short term (rather than in the long-term) and rely on much simpler student models while being computationally very efficient.

In this paper, we compare the POMDP framework and the MAB framework (specifically the algorithm ZPDES already evaluated in real classrooms [6]). We first introduce a student model used to compare the different algorithms. We then propose ways to model the heterogeneity in classrooms by considering that different students will have not only different learning parameters but also that they might have different dependencies between the different knowledge components (KCs). Our experiments will evaluate how well a MAB can approach the optimal solution of a POMDP, and how the different algorithms behave when encountering a heterogeneous group of student.

2. RELATED WORK

In this work we are interested in the impact of the quality of the student models on the quality of the sequences of activities chosen by online algorithms.

There are several approaches to choose automatically exercises based on the current knowledge level of students. We are here particularly interested by optimization methods that rely on minimal prior assumptions about the students or the knowledge domain.

One option already explored is the use of a partial-observable Markov decision process (POMDP) [13], [14]. PODMPs offer an appealing theoretical framework that guarantees an optimal long-term solution for a planning problem. However, in general, as the computational complexity is high, it is practically impossible to find an exact solution to the problem. Some approximate solutions in the domain of ITS have considered the use of aggregations of states instead of tracking the full knowledge components. Another drawback is that POMDPs require a precise student model for which the policy is optimized. If the real student encountered deviates from this model, then the optimality properties are lost.

A more recent approach is to use the Multi-Arm Bandit (MAB) framework to manage pedagogical activities [6]. MABs have the advantage of being extremely computationally efficient and rely on very weak student models. The main drawback is that there is no long-term planning of the best sequence of activities relying on an exploration-exploitation tradeoff to find the best path. Aware of such problem, authors of one such algorithm considered that standard MAB needs to be complemented with a weakly specified knowledge graph to provide a long-term view on the optimization [6].

As noted, before optimizing the sequence of exercises it is important to have some knowledge about the impact of a given exercise in the learning of the KCs, and also to be able to track what each student already masters. A large part of the results on ITS has been on the modeling aspects of the cognitive and student models. A seminal work on this topic was the *Knowledge Tracing* framework [7] which builds a detailed cognitive model of the student, of its learning processes by considering a set of independent KCs, the probability of learning them and the probability of correct or wrong answer in exercises that relies on those KCs. More recent methods extend this framework to a bayesian probabilistic approach [12, 15] improving the performance and understanding of those methods. Recent methods have started to consider how to learn such models, and variants of it, allowing to simultaneously discover the relation between activities and KC, e.g. [8, 2, 5, 9].

As discussed these methods require an accurate knowledge of how students learn and require to track their mastery of each KC. For this, it is necessary to learn the constraints between different KC, exercises and KC. Given students' particularities, it is impossible for a teacher to understand all the difficulties and strengths of individual students and provide an accurate student model manually. Even with the recent advances on model learning, there are several challenges in identifying parameters that best describe each individual student. These models have many parameters, and identifying all such parameters for a single student is a very hard problem due to the lack of data, often making the problem intractable. In most cases it is even impossible to identify some of the parameters [3, 4]. In the general case, it results in inaccurate models that cannot be exploited for individualized learning. Another problem is that these planning methods are for a population of students and not for a particular student and this has already been proven to be suboptimal [11].

3. STUDENT MODELS

3.1 Student model

In this section, we will present the student model we will use, also called learner model in literature. We want a generative model that can simultaneously be used to predict students behaviour, model their knowledge acquisition and track their mastery level. For this, we built a student model, shown in Fig.1 similar to the Knowledge Tracing framework [10] and its variants. Similarly to [9], we include extra features in our model. We are particular interested in more realistic cases where each KC might depend on other KCs. In most cases it is assumed that each exercise just depends on one KC and that they are independent, this is not realistic most of the time, and such dependencies have a strong impact on the learning sequences generated by the different algorithms.

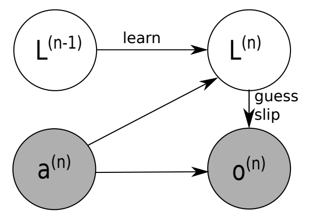

Figure 1: Graphical model of the Student model, with $L^{(n)}$ the hidden state of the student at step n, $a^{(n)}$ activity proposed, and $o^{(n)}$ the result obtained by the student.

We consider a situation where a student has a set of m KCs K_i to learn. A student's state at step n is represented by the state of each KC, $L^{(n)} = K_1^{(n)}, \ldots, K_m^{(n)}$, the global model is described on figure Fig.1. Each KC is defined by his state, mastered ($K_i = 1$) or not mastered ($K_i = 0$). For each KC, there is an initial probability of mastering it $p(K_i^{(0)} = 1)$ which is always null in our experiments to make students learn all the KCs through activities. The emission probabilities are defined by the guess probability, i.e performing correctly without mastering the skill, and the slip probability, i.e performing incorrectly despite mastering the knowledge. Theses probabilities are constant. Finally $p(K_i^{(n)} = 1|L^{(n-1)}, a^{(n)})$ defines the probability of transition from not mastered to mastered K_i while doing activity a at step n and depending of the constraints between KCs and their states. An activity can be represented as a vector $a = \alpha_1, \ldots, \alpha_m$ where $\alpha_i = 1$ if the activity allows to acquire K_i, $\alpha_i = 0$ else. The transition probability to learn a given KC K_i at step n is given by the following formula:

$$p(K_i^{(n)} = 1|L^{(n-1)}, a^{(n)}) = \alpha_i(\beta_{i,i} + \sum_{j \neq i}^{m} \beta_{i,j} K_j^{(n-1)}) \quad (1)$$

Where $\beta_{i,i}$ represent the probability to learn K_i without considering other KCs and $\beta_{i,j}$ represent the impact of the KC K_j on the probability to learn K_i. If a given KC does not need other KCs to be learned, the term $\sum_{j \neq i}^{m} \beta_{i,j} K_j$ is null.

For more simplicity, in our experiments, an activity a can provide an opportunity to acquire only one KC which induces an isomorphism between the knowledge space and the activity space.

3.2 Models of populations

The previous model can be used to describe a single student or an average model of a population. Our goal is to understand the impact that the diversity of students has when the given sequence is optimized considering the same parameters for all students. We will achieve such goal by considering a canonical model and then make two types of disruptions: i) change the probabilities between the variables; ii) change the knowledge graph.

The first way is to disrupt the parameters in the model, i.e. the probability of transition, guess, and slip. To do that, we consider that each parameter is sampled from a gaussian distribution. We can change the variance to increase the heterogeneity of the population. With a variance null, all the population has the same parameters. The second way is to change the knowledge graph that changes the dependencies between the different knowledge. This type of disruption can be small like adding or removing a dependency, or it can be as critical as rearranging completely the organization of the knowledge dependencies. These two types of disruption are combined in our experiments.

$$LM_0 : \xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_6$$

$$LM_1 : \left.\begin{array}{l} \xrightarrow{0.2} K_1 \\ \xrightarrow{0.2} K_2 \end{array}\right\} \xrightarrow[0.1]{0.1} K_3 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_6$$

$$LM_2 : \begin{array}{l} \xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_6 \\ \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_5 \end{array}$$

$$LM_3 : \xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_6$$

$$LM_4 : \xrightarrow{0.2} K_1 \xrightarrow{0.2} K_2 \xrightarrow{0.2} K_4 \xrightarrow{0.2} K_3 \xrightarrow{0.2} K_5 \xrightarrow{0.2} K_6$$

Figure 2: Knowledge graphs used in the simulations. LM_0 is the nominal knowledge graph, with LM_1 and LM_2 introducing small disruptions in the prerequirements between KCs. LM_3 and LM_4 represent more critical disruptions that change the overall order of KCs.

We used multiple knowledge graphs, shown in Fig.2. The arrows represent the dependencies between KCs. For example, LM_0 represents a graph where the constraints between the different KC are ordered in a linear way. Here, $\beta_{1,1} = \beta_{2,1} = \beta_{3,2} = \beta_{4,3} = \beta_{5,4} = \beta_{6,5} = 0.2$ and all the others values of $\beta_{i,j}$ are null. We then created several different transformations and variants to model different needs of the students in terms of the order of the different KC.

LM_1 and LM_2 follow approximately the same overall sequence of KC, but considering two initial branches for the different KC. LM_1 considers that KC_1 and KC_2 are independent and any of them allows to learn KC_3. In these knowledge graphs, we can expect that optimizing for one will also work for the other as the overall sequence of KC is respected, even if the strategy is no longer optimal. We also created more critical disruptions in the knowledge graph. LM_3 and LM_4 present an inversion between two KCs. For LM_3, KC_4 and KC_5 are inverted, what radically change the overall sequence of KCs. For LM_4, it is K_3 and K_4 that are inverted.

4. OPTIMIZING LEARNING POLICIES

4.1 Partially Observed Markov Decision Process (POMDP)

POMDP is a markovian decision process where the state is hidden and can only be inferred indirectly from the observations. A POMDP consists of a tuple $\langle S, A, Z, T, R, O, \gamma \rangle$ with S the state space, A the action space and Z the observation space. T is the transition model, it gives the probabilities $p(s'|s, a)$ of transitioning from state s to state s' with the action a. O is the observation model, it gives the probabilities $p(z|s, a)$ of having the observation z when action a is made in state s. R the cost model, it specifies the cost $r(s, a)$ of choosing action a in state s, and the discount factor γ gives the relation between immediate costs and delayed costs. With all these components, the solution of a POMDP is a policy that optimizes total discounted future reward.

This framework has been already used in the context of ITS [13]. The learner's mastery is the hidden state s, learning is the transition between states, the probabilities that the learner gives a good answer are given by the observation model of the observation {correct, incorrect}. We use Perseus [14] as solver to find the optimal policy for our POMDP problem.

4.2 Zone of Proximal Development and Empirical Success (ZPDES)

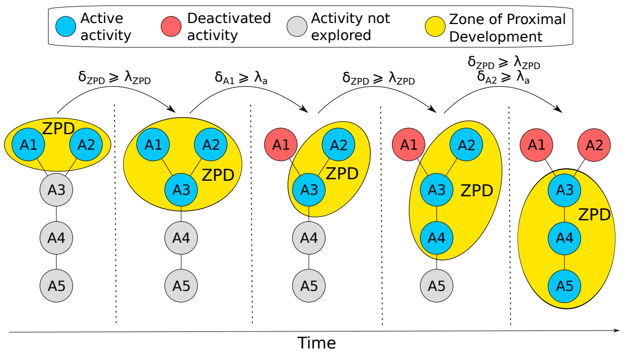

Figure 3: ZPDES exploration of an activity graph, with δ_{ZDP} the success rate over all active activities, λ_{ZPD} the threshold to expand the ZPD, δ_{Ax} the success rate for the activity Ax, and λ_a the threshold to reach to deactivate an activity.

Here we present the recently introduced algorithm Zone of Proximal Development and Empirical Success (ZPDES) that is based on multi-armed bandits [6]. The idea of the algorithm is presented in Fig.3 and summarized in Alg.1. The algorithm follows an activity graph but goes through it in a stochastic way. ZPDES is initialized with a certain number of activities defined as starting activities. At each point in time, ZPDES has a set of activities, called the zone of proximal development, that can be proposed to the student which is adapted depending on student result. In the experiments presented here, we make small changes in the activation/deactivation mechanism of the original algorithm. When the recent student success rate over all active activities δ_{ZPD} reaches a value λ_{ZPD}, the graph is expanded to explore another activity and when the recent success rate for a particular activity δ_{a_i} is higher than a threshold λ_a, this activity can be removed from the active list. This two threshold allow to partially configure the exploration behaviour of the algorithm. Inside the set of active activities, ZPDES proposes exercises proportionally to the recent learning progress obtained by that activity. The activity graph following the same structure than the knowledge graph, we can directly configure ZPDES with the same knowledge graph used to configure POMDP.

Algorithm 1 ZPDES algorithm

```
Require: Set of $n_a$ activities $A$
Require: $\zeta$ rate of exploration
Require: distribution for parameter exploration $\xi_u$
 1: Initialize of quality $w_a$ uniformly
 2: while learning do
 3:     Initialize ZPD
 4:     {Generate exercise:}
 5:     for $a \in ZPD$ do
 6:         $\tilde{w}_a = \frac{w_a}{\sum_j w_j}$
 7:         $p_a = \tilde{w}_a(1-\zeta) + \zeta\xi_u$
 8:         Sample $a$ proportional to $p_a$
 9:     end for
10:     Propose activity $a$
11:     Get student answer $C_t$ and compute reward:
12:     $r = \sum_{k=t-d/2}^{t} \frac{C_k}{d/2} - \sum_{k=t-d}^{t-d/2} \frac{C_k}{d-d/2}$
13:     $w_a \leftarrow \beta w_a + \eta r$ {Update quality of activity}
14:     Update ZPD based on activity graph and success rates
15: end while
```

5. EXPERIMENTS

The goal of our experiments is to compare the impact of the knowledge about the students on the online algorithms for choosing exercises, namely POMDP and ZPDES. We will proceed to change the heterogeneity of the student populations and see how much disruption each algorithm is able to adapt. Our comparative measure of performance is the average skill level overall knowledge and over time, for all the students in the population.

We will compare the results obtained with two algorithms: POMDP and ZPDES. Each algorithm will have different variants based on the knowledge included on each of them. POMDP relies on a knowledge graph and the parameters of such graph. Each variant of $POMDP_x$ is characterized by a specific student model used to find the optimal policy. ZPDES has as information the knowledge graph, and some parameters describing how to traverse this graph, no particular assumption is made about the probabilities of knowledge acquisition. $ZPDES^H_x$ is a variant of ZPDES with the corresponding graph x and using the parameters that were used in an other experiment in a real world situation [6] mostly hand-tuned with the help of a pedagogical expert. $ZPDES^*_x$ will also use the graph x but the parameters to traverse the graph are optimized for that particular graph using a greed search. During the optimization, we saw that the majority of parameters present average results and only extreme parameters gave critical results.

Single model results. The first experiment will do a sanity check to evaluate each algorithm in conditions where each student is the same in the population and each algorithm is configured for this model of student. We expect POMDP to have the best results and we want to see how far ZPDES will be from the optimal solution. A Random strategy which selects one activity randomly among all possible is also presented in this first experiment to see the gain of the algorithms.

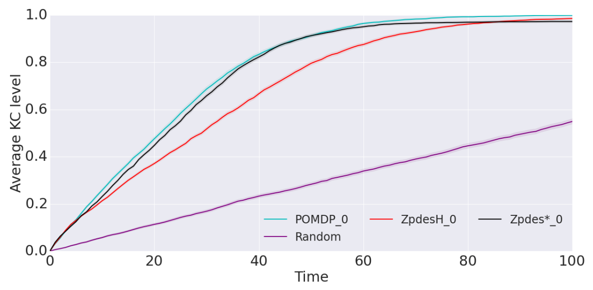

Figure 4: Evolution of the average skill level for 600 students modeled with LM 0 which activity are managed by POMDP, ZPDES*, ZPDES^H configured for LM 0. Shaded area represents the standard error of the mean.

Fig.4 shows the comparison of POMDP, ZPDES*, ZPDES^H and Random with a population of 600 students modelled with the knowledge graphs LM 0. We can see POMDP is the best for all the models, closely followed by ZPDES*. ZPDES^H give a slower learning than the two others. Unsurprisingly, for one particular model, POMDP has the best performance. The optimized ZPDES is very close in performance to POMDP. The results are similar for models 1, 2, 3 and 4, the curves are not presented here for space reason. We can thus verify that the combination of knowledge graphs and the activity exploration rules provides a space of policies that is close to the optimal POMDP one. ZPDES^H present the slowest population learning among the algorithms but has its configuration was not optimized for any particular model we can expect such result.

These results show that the algorithms behave as expected and that ZPDES has the potential to be close to the optimal POMDP solution.

Multi model results. We will now present the main results of this work with the comparison between POMDP, ZPDES* and ZPDES^H when confronted with heterogeneous populations of students. The protocol of the experiments is as follows. First we provide each algorithm with the information about a specific population of students and then we test the capability of the algorithms to address a different and diverse population of students. As described earlier, each algorithm is given information about a particular student model x, POMDP$_x$ receives the graph and the student model parameters, ZPDES$^*_x$ receives the graph and exploration parameters optimized for that same graph, ZPDES^H_x receives the graph and standard parameters for the graph exploration. We test different versions of each algorithm with a population composed of students following 3 differ-

Table 1: Performance position of each algorithm configuration for each setup. The rank of each algorithm configuration, and the average rank of each algorithm is presented for steps 50 and 200.

Students 0,1,2 / Alg config 0,1,2

	Rank t 50		Rank t 200	
Algorithm	Per conf	Average	Per conf	Average
POMDP_0	1		1	
POMDP_1	3	1	2	2
POMDP_2	4		3	
ZPDES_0^H	3		1	
ZPDES_1^H	3	3	1	1
ZPDES_2^H	6		3	
ZPDES_0^*	2		1	
ZPDES_1^*	3	2	2	2
ZPDES_2^*	5		3	

Students 0,3,4 / Alg config 0,3,4

	Rank t 50		Rank t 200	
Algorithm	Per conf	Average	Per conf	Average
POMDP_0	1		2	
POMDP_3	2	1	3	2
POMDP_4	4		5	
ZPDES_0^H	2		1	
ZPDES_3^H	3	2	2	1
ZPDES_4^H	4		3	
ZPDES_0^*	2		2	
ZPDES_3^*	3	2	4	2
ZPDES_4^*	4		4	

Students 2,3,4 / Alg config 0,1

	Rank t 50		Rank t 200	
Algorithm	Per conf	Average	Per conf	Average
POMDP_0	1	1	3	2
POMDP_1	4		6	
ZPDES_0^H	2	1	1	1
ZPDES_1^H	3		2	
ZPDES_0^*	2	1	4	2
ZPDES_1^*	3		5	

ent knowledge graphs. The probabilistic parameters of the student models in the population follow a gaussian distribution. There is 200 students per graphs for a total of 600 students.

On figure 5 we can see the evolution of the average mastery level for all KCs. The table 1 presents the ranking of each version of the algorithms and the average ranking of each algorithm at step 50 and 200 according to the curves comparison for each setup $LM_{0,1,2}$, $LM_{0,3,4}$, and $LM_{2,3,4}$. The table 2 presents the statistical significance tests at step 50 and 200 for each setup and what is the best methods if the results are statistically significant.

By comparing the different p-values, we can see that the differences between POMDP and ZPDES* are never significant, but it's not the case for ZPDES^H. For the models $LM_{0,1,2}$, at step 50, ZPDES^H drops behind the two others, but it catches up rapidly with the two others and present the same results at step 200. So for models which are close to each other, the 3 algorithms present almost the same result.

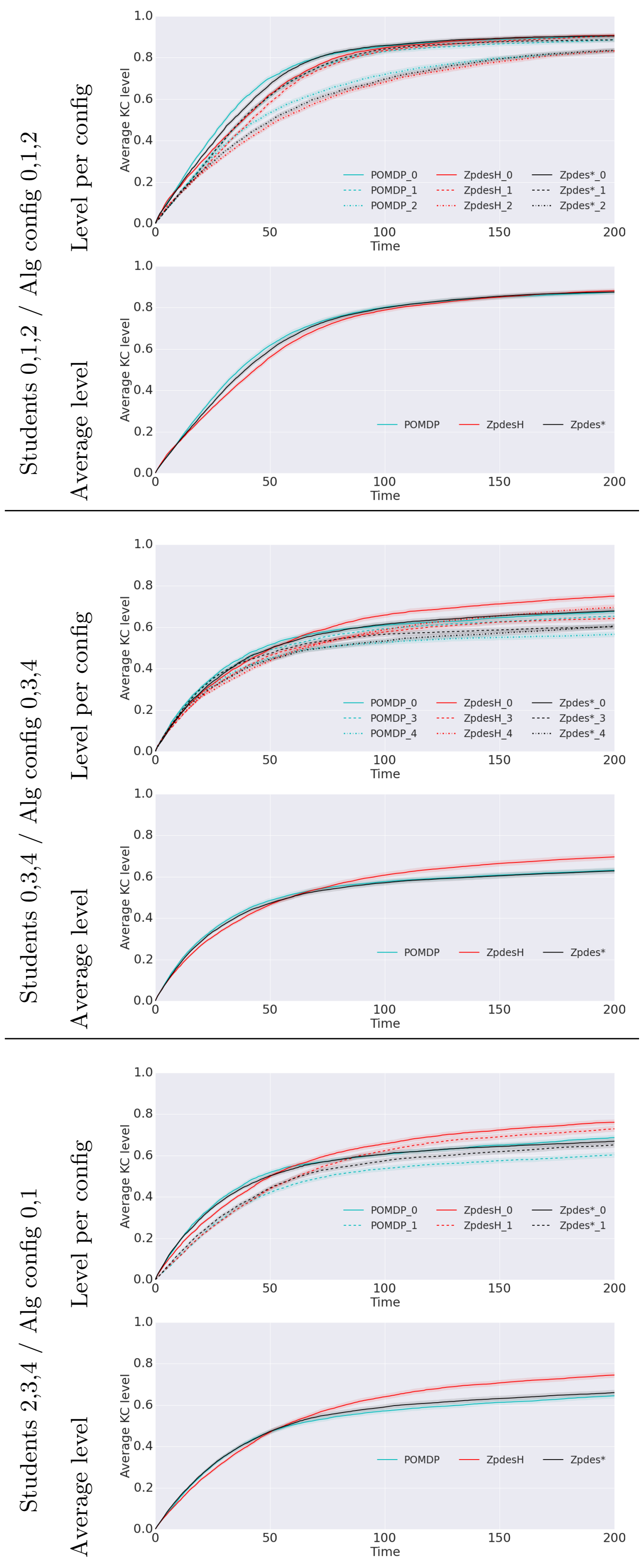

Figure 5: Evolution of the average skill level for 600 students with POMDP, ZPDES*, ZPDES^H. For each curve, the number attached to the algorithm's name indicate what knowledge graph has been used to configure the algorithm. Each curve shows the average KC level of the student population over time for each algorithm configuration. In general ZPDES have better results than POMDP. Shaded area represents the standard error of the mean.

Table 2: ANOVA p-values for each setup to verify if the differences in the KC level distribution according to each algorithm are statistically significant with the best algorithms in parenthesis when it is significant. We note P for POMDP and Z for ZPDES

	P/Z*		P/Z^H		Z*/Z^H	
LM	t 50	t 200	t 50	t 200	t 50	t 200
0,1,2	.075	.95	$\mathbf{10^{-6}}$ **(P)**	.82	**.003** **(Z*)**	.87
0,3,4	.24	.90	.17	$\mathbf{10^{-5}}$ $(\mathbf{Z}^H)$	.89	$\mathbf{10^{-4}}$ $(\mathbf{Z}^H)$
2,3,4	.31	.30	.18	$\mathbf{10^{-5}}$ $(\mathbf{Z}^H)$	.77	$\mathbf{10^{-7}}$ $(\mathbf{Z}^H)$

For the models $LM_{0,3,4}$, observations are different. At step 50, all the algorithms seem to have approximately the same performance, even if ZPDES^H seems a bit behind but it's not significant (p-values at 0.17 and 0.89). But with time, it takes the lead and achieves the best performance at 200 steps. So when there are two models critically different from another, ZPDES^H presents the best results. For the last case, the population is constituted of students following $LM_{2,3,4}$ models, and the algorithms are configured for models $LM_{0,1}$. As for the previous case there is no differences at step 50 but ZPDES^H presents the best results at step 200.

ZPDES^H provides the best result because its exploration parameters were not optimized for any particular knowledge graph, giving it higher adaptability and less constrains in the exploration. For a particular type of student model it will present worse performance than POMDP or ZPDES*, but for a heterogeneous population, ZPDES^H, being more adaptable, has the best performance.

6. CONCLUSION

In this work we considered student models where the knowledge components can have constraints among each other, allowing to model some kind of pre-requisites. Under different student models we can find an optimal teaching sequence using POMDP. Another alternative is the use of the recently proposed method ZPDES that is computationally more efficient but without optimality guarantees. Our goal was to test how robust each of these methods is in relation with ill-estimated parameters of the models, or even wrongly estimated relations between KCs. This corresponds to the more realistic case of heterogeneous classes of students.

We showed that for the trivial situation where the students are perfectly modeled with the student model, ZPDES can achieve the same performance as the POMDP. For heterogeneous populations again ZPDES can achieve solutions similar to POMDP. The best algorithm was using ZPDES that uses parameters that are not optimized for no population in particular. By having more flexibility in the exploration it becomes more robust to changes in the population.

We conclude that multi-armed bandits, when combined with an activity graph, are a best choice in comparison with POMDPs due to its computational efficiency and reliance on simpler student models.

The code to generate the graphics and the results is available at: **github.com/flowersteam/kidlearn/tree/edm2016**, follow the README.

7. REFERENCES

[1] John R Anderson, Albert T Corbett, Kenneth R Koedinger, and Ray Pelletier. Cognitive tutors: Lessons learned. *The journal of the learning sciences*, 4(2):167–207, 1995.

[2] Ryan SJ Baker, Albert T Corbett, and Vincent Aleven. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *Intelligent Tutoring Systems*, pages 406–415, 2008.

[3] Joseph E Beck and Kai-min Chang. Identifiability: A fundamental problem of student modeling. In *User Modeling 2007*. 2007.

[4] Joseph E Beck and Xiaolu Xiong. Limits to accuracy: How well can we do at student modeling? In *Educational Data Mining*, 2013.

[5] Hao Cen, Kenneth Koedinger, and Brian Junker. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems*, 2006.

[6] Benjamin Clement, Didier Roy, Pierre-Yves Oudeyer, and Manuel Lopes. Multi-Armed Bandits for Intelligent Tutoring Systems. *Journal of Educational Data Mining (JEDM)*, 7(2):20–48, June 2015.

[7] A.T. Corbett and J.R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[8] José P González-Brenes and Jack Mostow. Dynamic cognitive tracing: Towards unified discovery of student and cognitive models. In *EDM*, pages 49–56, 2012.

[9] JP González-Brenes, Yun Huang, and Peter Brusilovsky. General features in knowledge tracing: Applications to multiple subskills, temporal item response theory, and expert knowledge. In *Inter. Conf. on Educational Data Mining*, 2014.

[10] K.R. Koedinger, J.R. Anderson, W.H. Hadley, M.A. Mark, et al. Intelligent tutoring goes to school in the big city. *Inter. Journal of Artificial Intelligence in Education (IJAIED)*, 8:30–43, 1997.

[11] J.I. Lee and E. Brunskill. The impact on individualizing student models on necessary practice opportunities. In *Inter. Conf. on Educational Data Mining (EDM)*, 2012.

[12] Kai min Chang, Joseph Beck, Jack Mostow, and Albert Corbett. A bayes net toolkit for student modeling in intelligent tutoring systems. In *Intelligent Tutoring Systems*, 2006.

[13] A. Rafferty, E. Brunskill, T. Griffiths, and P. Shafto. Faster teaching by pomdp planning. In *Artificial Intelligence in Education*, pages 280–287, 2011.

[14] Matthijs T. J. Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

[15] Michael Villano. Probabilistic student models: Bayesian belief networks and knowledge space theory. In *Intelligent Tutoring Systems (ITS'92)*, 1992.