Developmental Learning for Intelligent Tutoring Systems

Manuel Lopes, Benjamin Clément, Didier Roy, Pierre-Yves Oudeyer

Developmental Learning for Intelligent Tutoring Systems

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

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

Didier Roy
Inria
Bordeaux, France
didier.roy@inria.fr

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

***Abstract*—We present an approach to Intelligent Tutoring Systems which adaptively personalizes sequences of learning activities to maximize skills acquired by each student, taking into account limited time and motivational resources. At a given point in time, the system tries to propose to the student the activity which makes him progress best. We introduce two algorithms that rely on the empirical estimation of the learning progress, one that uses information about the difficulty of each exercise RiARiT and another that does not use any knowledge about the problem ZPDES.**

The system is based on the combination of three approaches. First, it leverages recent models of intrinsically motivated learning by transposing them to active teaching, relying on empirical estimation of learning progress provided by specific activities to particular students. Second, it uses state-of-the-art Multi-Arm Bandit (MAB) techniques to efficiently manage the exploration/exploitation challenge of this optimization process. Third, it leverages expert knowledge to constrain and bootstrap initial exploration of the MAB, while requiring only coarse guidance information of the expert and allowing the system to deal with didactic gaps in its knowledge.

I. Introduction

Intelligent Tutoring Systems (ITS) have been proposed to make education more accessible, more effective and simultaneously as a way to provide useful objective metrics on learning ([1], [21]). Recently, online learning systems have further raised the interest in these systems [11], and several recent projects started on Massive Open Online Course (MOOC) for web-based teaching of university level courses. Research in ITS includes many different aspects such as: interaction, instructional design, generation of exercises, modeling of student learning, optimizing teaching, among many other aspects. For a broad coverage on the field of ITS see [20] and [26].

According to [26], there are four main components of an ITS: i) a *cognitive model* that defines the domain knowledge or which steps need to be made to solve problems in a particular domain; ii) a *student model* that considers how students learn, what is the evolution of their cognitive state depending on particular teaching activities; iii) a *tutoring model* that defines, based on the cognitive and the student model, what teaching activities to present to students and iv) a *user interface model* that represents how the interaction with the students occurs and how problems are proposed to the learners.

A large body of work considered the *Cognitive/Student Model*. A seminal work for defining the cognitive model is the *Knowledge Tracing* method introduced by [13] which builds a detailed cognitive model of the student, of its learning processes and of the relation between this cognitive model and the involved activities and KCs. This is a very difficult problem because the knowledge level of the students and their learning approach is hidden.Recent results include methods to simultaneously discover the relation between activities and KC ([18], [3]), an alternative formulations such as the Additive Factor Model [12].

In this work we are more focused on the *tutoring model*, that is, how to choose the activities that provide a better learning experience based on the estimation of the student competence levels and progression, and some knowledge about the cognitive and student model. We can imagine a student wanting to acquire many different skills, e.g. adding, subtracting and multiplying numbers. A teacher can help by proposing activities such as: multiple choice questions, abstract operations to compute with a pencil, games where items need to be counted through manipulation, videos, or others. The challenge is to decide what is the optimal sequence of activities that maximizes the average competence level over all skills.

There are several approaches to develop a *Tutoring Model*. A first approach is based on hand-made optimization and on pedagogical theory, experience and domain knowledge. There are many works that followed this line, see the recent surveys on the field by [20], [26]. A second approach considers particular forms of knowledge to be acquired and creates didactic sequences that are optimal for those particular classes of problems [4], [17], [10]. Other approaches try to construct examples and/or exercises that optimize the training process allowing to create new questions, or other activities, aimed directly at solving the mistakes observed [28], [15], [10]. The third approach, and more relevant for our work, is that the optimization is made automatically without particular assumptions about the students or the knowledge domain. The framework of partial-observable Markov decision process (POMDP) has been proposed to select the optimal activities to propose to the students based on the estimation of their level of acquisition of each KC [27]. In general the solution to a POMDP is an intractable problem and approximate solutions have been proposed using the concept of envelope states [8]. Other approaches consider a global optimization of the

pedagogical sequence based on data from all the student using ant colony optimization algorithms [31], but can not provide a personalized sequence.

Our ITS system aims at providing to each particular student the activities that are giving the highest learning progress. We do not consider that these activities are necessarily the ones defined a-priori in the cognitive and student model, but the ones that are estimated, at runtime and based on the students results, to provide the maximum learning gain. This approach has three main advantages:

Weaker dependency on the cognitive/student model In most cases the tutoring model incorporates the student model inside. For instance, in approaches based on POMDP, the optimization of teaching sequences is made by using as dynamics the student model. Given students' particularities, it is often highly difficult or impossible for a teacher to understand all the difficulties and strengths of individual students and thus predict which activities provide them with maximal learning progress. Also, typically, 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, the intractability of the problem and the lack of identifiability of many parameters [5]. This often results in models which are inaccurate in practice [6]. It has been shown that a sequence that is optimal for the average student is often suboptimal for most students, from the least to the most skilled [23].

We consider that it is important to be as independent as possible of the cognitive and student model when deciding which activities to propose. This requires that the ITS explores and experiments various activities to estimate their potential for learning progress for each student. The technical challenge is that these experiments should be not just sufficiently informative about the student's current competence but also to evaluate the effectiveness of each exercise to improve those competences (a form of stealth assessment [32]). This boils down to what has been called the "exploration/exploitation" trade-off in machine learning, where we have to simultaneously try new activities to know which ones are the best, but also select the best ones so that the student actually learns.

Efficient Optimization Methods We will rely on methods that do not make any specific assumptions about how students learn and only require information about the estimated learning progress of each activity. We make a simple assumption that activities that are currently estimated to provide a good learning gain, must be selected more often. A very efficient and well studied formalism for these kind of problems is Multi-Armed Bandits [9]. Following a casino analogy, at each step we can choose a slot-machine and we get to observe the payback we get, the goal it to find the best arm, but while we are trying to discover it we have to bet to test them.

More Motivating Experience Our approach considers that exercises which are currently providing the higher learning progress must be the ones proposed. This allows not only to use more efficient optimization algorithms but also to provide a more motivating experience to students. Several strands of work in psychology [7] and neuroscience [19] have argued that the human brain feels intrinsic pleasure in practicing activities of optimal difficulty or challenge, i.e. neither too easy nor too difficult, but slightly beyond the current abilities, also known as the zone of proximal development [22]. Such activities typically create positive psychological states of flow [14], themselves fostering learning, as exploited in several educational approaches [30], [16].

Our main contributions, when compared to other ITS systems, are: the use of highly performing Multi-Armed Bandit algorithms [9]; a simpler factored representation of the cognitive model that maps activities to the minimum necessary competence levels; and considering that the acquisition of a KC is not a binary variable but defined as the level of comprehension of that KC. The advantage of using MAB is that they are computational efficient and require a weaker dependency between the tutoring and the cognitive and student models. Other contributions include an algorithm to estimate student competence levels; and the empirical learning progress of each activity. An extended version of this article is available at [24] including an initial user study.

II. Teaching Scenario

To make the discussion more clear, we will describe a specific teaching scenario. This activity will show how, even in simple learning objectives, there are several activities parameters that might allow different students to learn at different rates. This scenario is about learning how to use money, typically targeted to students of 7-8 years old. The parameters of the activities are commonly used in schools for acquiring these competences and there are already well studied teaching sequences validated in several studies [29].

In each exercise, one object is presented with a given tagged price and the learner has to choose which combination of bank notes, coins or abstract tokens need to be taken from the wallet to buy the object, with various constraints depending on exercises parameters. The five Knowledge Components aimed at in these experiments are:

KnowMoney: Global skill characterizing the capability to handle money to buy objects in an autonomous manner;
SumInteger: Capability to add and subtract integer numbers;
DecomposeInteger: Capability to decompose integer numbers into groups of 10 and units;
SumCents: Capability to add and subtract real numbers (cents);
DecomposeCents: Capability to decompose real numbers (cents);
Memory: Capability to memorize a number which is presented and then removed from visual field;

The various activities are parameterized in order to allow students to acquire a greater flexibility in using money, and to slowly increase the number of KCs that are already mastered. The parameters are the following:

Exercise Type We consider a parameterization of exercise types depending on the values[1] that can be read directly by

[1]In the euro money system the money items (bills and coins) have the values 1, 2 and 5 for the different scales.

making the correspondence to a real note/coin $a = (1, 2, 5)$ and those that need a decomposition that requires more than one item $b = (3, 4, 6, 7, 8, 9)$. The exercises will be generated by choosing prices with these properties in a set of six levels of increasing difficulty and picking an object that is priced realistically.

Price Presentation: i) written and spoken; ii) written; iii) spoken

Cents Notation: i) x€x; ii) x, x€

Money Type: i) Real euros; ii) Money Tokens

III. ITS WITH MULTI-ARMED BANDITS

A. Relation between KC and pedagogical activities

In general, activities may differ along several dimensions and may take several forms (e.g. video lectures with questions at the end, or interactive games or exercises of various types). Each activity can provide opportunities to acquire different skills/knowledge units, and may contribute differentially to improvement over several KCs (e.g. one activity may help a lot in progressing in KC_1 and only little in KC_2). Vice versa, succeeding in an activity may require to leverage differentially various KCs. While certain regularities of this relation may exist across individuals, it will differ in detail for every student. Still, an ITS might use this relation in order to estimate the level of each student. We will latter show how to further simplify this assumption.

First, we model here the competence level of a student in a given KC as a continuous number between 0 and 1 (e.g. 0 means not acquired at all, 0.6 means acquired at 60 percent, 1 means entirely acquired). We denote c_i the current estimation of this competence level for knowledge unit KC_i. In what we call a R Table, for each combination of an activity $\mathbf{a}$ and a KC_i, the expert then associates a $q-$value ($q_i(\mathbf{a})$) which encodes the competence level required in this KC_i to have maximal success in this activity $\mathbf{a}$. This in turn provides a upper and lower bound on the competence level of the student: below $q_i(\mathbf{a})$ in case of mistake; 2) above $q_i(\mathbf{a})$ in case of answering correctly. As explained in the following section, this information will be useful to calibrate the measure of impact of each activity over each KC.

We start by assuming that each activity is represented by a set of parameters $\mathbf{a} = (a_1, ..., a_{n_a})$. The R Table then uses a factorized representation of activity parameters, where instead of considering all $(\mathbf{a}, KC_i)$ combinations and their corresponding $q_i(\mathbf{a})$, we consider only (a_j, KC_i) combinations and their corresponding $q_i(a_j)$ values, where $q_i(a_j)$ denotes the competence level in KC_j required to succeed entirely in activity $\mathbf{a}$ which $j-th$ parameter value is a_j, as shown in Table I. This factorization makes the assumption that activity parameters are not correlated. This assumption is not valid in the general case, but has appeared true in particular applications we considered and may not significantly harm the dynamics of the whole system. The alternative would require a larger number of parameters and would also require more exploration in the optimization algorithm. We use the factorized R Table in the following manner to heuristically estimate the competence level $q_i(\mathbf{a})$ required in KC_i to succeed in an activity parameterized with $\mathbf{a}$:

$$q_i(\mathbf{a}) = \prod_{j=1}^{n_a} q_i(a_j)$$

B. Estimating the impact of activities over students' competence level in knowledge units

Key to the approach is the estimation of the impact of each activity over the student's competence level in each knowledge unit. This requires an estimation of the current competence level of the student for each KC_i. We do not want to introduce, outside activities, regular tests that would be specific to evaluate each KC_i since it would have a high probability to negatively interfere with the learning experience of the student. Thus, competence levels need to be inferred through stealth assessment [32] that uses indirect information coming from the combination of performances in activities and the R Table specified above.

Let us consider a given knowledge unit i for which the student has an estimated competence level of c_i. When doing an activity $\mathbf{a} = (a_1, ..., a_{n_a})$, the student can either succeed or fail. In the case of success, if the estimated competence level c_i is lower than $q_i(\mathbf{a})$, we are underestimating the competence level of the student in KC_i, and so should increase it. If the student fails and $q_i(\mathbf{a}) < c_i$, then we are overestimating the competence level of the student, and it should be decreased. Other cases provide little information, and thus c_i is not updated. For these two first cases we can define a reward:

$$r_i = q_i(\mathbf{a}) - c_i \tag{1}$$

and use it to update the estimated competence level of the student according to:

$$c_i = c_i + \alpha r_i \tag{2}$$

where α is a tunable parameter that allows to adjust the confidence we have in each new piece of information. Accordingly, this also encodes that being always successful at a given activity a_j cannot increase the estimated competence level c_i above $q_i(\mathbf{a})$.

A crucial point is that the quantity $r_i = q_i(\mathbf{a}) - c_i$ is not only used to update c_i, but is used to generate an internal reward $r = \sum r_i$ to be cumulatively optimized for the ITS (details below). Indeed, we assume here that this is a good indicator of the learning progress over KC_i resulting from doing an activity with parameters $\mathbf{a}$. The intuition behind is that if you have repeated successes in an activity for which the required competence level is higher than your current estimated competence level, this means you are probably progressing.

C. RiARiT Algorithm: Right Activity at Right Time

To address the optimization challenge for ITS, we will rely on state-of-the-art multi-arm bandit techniques (MAB)[9]. These techniques were first developed in machine learning to solve the so-called "gambling machines" problem, where

TABLE I

R TABLE INDICATING THE REQUIRED COMPETENCE LEVEL FOR EACH OF THE n KNOWLEDGE COMPONENTS TO SUCCEED AT EACH PROPOSED ACTIVITY. EACH ACTIVITY IS PARAMETERIZED BY A SET OF m PARAMETERS a WITH DIFFERENT POSSIBLE VALUES. AN ACTIVITY IS THUS REPRESENTED AS A VECTOR $a = \{a_1, \ldots, a_m\}$ OF PARAMETERS.

		KnowMoney	IntSum	IntDec	DecSum	DecDec	Memory
Exercise Type	1	0,3	0.4	0	0	0	0.3
	2	0,5	0.6	0.3	0	0	0.5
	3	0,6	0.7	0.6	0	0	0.5
	4	0.8	0.7	0.6	0.4	0.3	0.7
	5	0.9	0.9	0.8	0.7	0.6	0.8
	6	1	1	1	1	1	1
Price Present.	S	0.9	1	1	1	1	1
	W	1	1	1	1	1	0.6
	S&W	0.8	1	1	1	1	0.2
Cents Not.	x.x€	0.8	1	1	1	1	1
	x€x	1	1	1	1	1	1
Money Type	Real	1	-	-	0.9	0.9	1
	Token	0.1	-	-	1	1	1

Algorithm 1 Update competence level

Require: Teaching activity a
Require: Student Solution S
Compute activity required competence level $Q(a)$
$r = 0$
for $i = 1, \ldots, n_c$ **do**
  $r_i = q_i(\mathbf{a}) - c_i$
  if CORRECT(S) and $r_i > 0$ OR WRONG(S) and $r_i < 0$ **then**
    $c_i = c_i + \alpha r_i$
    $r = r + r_i$
  end if
end for
return Vector of estimated competence level C^L
return Total reward r

a player has to optimize his strategy for investing his own money in gambling machines, whose money redistribution rate is initially unknown. We here use and adapt such approaches to the problem of optimal teaching, where the gambler is replaced by the teacher, the choice of machines is replaced by a choice of learning activity, and money is replaced by learning progress (which is a proxy for maximizing acquired skills).

A particularity here is that the reward (learning progress) is non-stationary, which requires specific mechanisms to track its evolution. Indeed, here a given exercise will stop providing reward, or learning progress, after the student reaches a certain competence level. Also we cannot assume that the rewards are i.i.d. as different students will have different preferences and many human factors, i.e. distraction, mistakes on using the system, create several spurious effects. Thus, we rely here on a variant of the EXP4 algorithm [2], [9] that considers a set of experts and chooses the actions based on the proposals of each expert. For our case the experts are a set of filters that track how much reward each bandit is giving. Then the algorithm selects stochastically the teaching activities proportionally to the expected learning progress for each bandit.

In order to harness the combinatorial explosion of parameter values, we do not use one MAB for each activity but a set of simultaneous MAB for each parameters j. The alternative of considering a given arm for each specific combination $\mathbf{a}$ of parameters would increase the number of arms that would impact on the number of parameters, the amount of trials required to estimate learning progress and thus the learning time.

This follows the factorization already described before. Each simultaneous MAB uses a bandit algorithm, derived from EXP4, presented in [25]. Each bandit expert tracks how much reward is provided by each activity parameter over all KC_i. Precisely, for each parameter value a_j we define the quantity $w_j(a_j)$ that tracks the recent rewards (correlate of learning progress) provided by activities using this parameter. Each time that such parameter is used, we update this value as follows $w_j(a_j) \leftarrow \beta w_j(a_j) + \eta r$, where β and η allow to define the tracking dynamics of the filter.

At any given time, we will sample the value of each parameter i according to: $p_i = \tilde{w_i}(1-\gamma) + \gamma\xi_u$, where $\tilde{w_i}$ are the normalized w_i values to ensure a proper probability distribution, ξ_u is a uniform distribution that ensures sufficient parameter exploration and γ is the exploration rate. The resulting algorithm is shown in Algorithm 2.

Expert knowledge can also be used by incorporating *coarse* global constraints on the ITS. Indeed, for example the expert knows that for most students it will be useless to propose exercises about decomposition of real numbers if they do not know how to add simple integers. Thus, the expert can specify minimal competence levels in given KC_i that are required to allow the ITS to try a given parameter a_j of activities.

D. ZPDES Algorithm: Zone of Proximal Development and Empirical Success

We will now present a variant of the algorithm that aims at reducing even further the dependency on the R Table. The table is used to compute the reward for the bandit system and it is only domain dependent and not student dependent. Nevertheless, our goal is to reduce the dependency on the

Algorithm 2 Right Activity at Right Time (RiARiT)

Require: Set of n_c competences C
Require: Set of exercise with n_a parameters
Require: Set of n_a experts w_i
Require: γ rate of exploration
Require: distribution for parameter exploration ξ_u
Initialize estimated competence level c^L
Initialize value of experts w_i uniformly.
while *learning* **do**
  {Generate exercise:}
  for $i = 1 \dots n_a$ **do**
    $\tilde{w_i} = \frac{w_i}{\sum_j w_i(j)}$
    $p_i = \tilde{w_i}(1-\gamma) + \gamma\xi_u$
    Sample a_i proportional to p_i
  end for
  Propose exercise $\mathbf{a} = (a_1, ..., a_{n_a})$
  Get Student Answer
  $C^L, r \leftarrow$ Update competence level using Alg. 1
  {Update greedy expert}
  for $i = 1 \dots n_a$ **do**
    $w_i(a_i) \leftarrow \beta w_i(a_i) + \eta r$
  end for
end while

cognitive and student models and so we will try to simplify further the algorithm. Our simplification will take two sources of inspiration: **zone of proximal development** and the **empirical estimation of learning progress**.

As discussed before focusing teaching in activities that are providing more learning progress can act as a strong motivational cue. Estimating explicitly how the success rate on each exercise is improving will remove the dependency on the table. For this we replace Eq. 1 with:

$$r = \sum_{k=1}^{t} \frac{C_k}{t} - \sum_{k=1}^{t-d} \frac{C_k}{t-d} \tag{3}$$

where $C_k = 1$ if the exercise at time k was solved correctly. The equation compares the recent success (the last $d+1$ samples) with all the previous past, providing an empirical measure of how the success rate is increasing. We no longer estimate the competence level of the student (as in Eq. 2), and directly use the reward estimation.

The use of the zone of proximal development will provide three advantages. Improve motivation as discussed before; further reduce the need of quantitative measures for the educational design expert; and provide a more predictive choice of activities. We considered that the design expert would define a set of thresholds on the activity parameters that would be used to allow a new parameter value to be used. This number are still dependent on the R Table and might be difficult to define or, if there are too many activities, too costly to define. Another point is that there are some parameters that have clearly relation of increasing complexity (such as the parameter exercise type) and should be treated differently than other parameters that do not have such ordering (for instance the complexity in the modality presentation will change depending on each student and not on the problem itself). A final point is that we are choosing exercises based on the estimated (recent) past learning progress, and if we know which exercise is next in terms of complexity then we can use that one. This information, if correct, allows the MAB to propose the more complex exercises without requiring to estimate their value first. Providing a more predictive behavior and not just relying on the recent past.

This algorithm is identical to RiARiT but we treat the parameters with an identified complexity order differently. For the parameter i, when the bandit level of parameter value j is below the level of the more complex parameter value, $w_i(j) < w_i(j+1)/\theta$, and the success rate is higher than a pre-defined threshold : $\sum_{k=1}^{t} \frac{C_k(j)}{t} > \omega$, we activate the parameter value $j+3$ with the following rule $w_i(j) = 0$ and $w_i(j+3) = w_i(j+2)$.

IV. Simulations

We start by presenting a set of simulations with virtual students to test systematically different properties of our algorithm. We define two different virtual populations of students to see how well the algorithm is able to select exercises adequate for each particular student. We note that the algorithm itself is not provided with any a-priori information about the properties of the students.

A. Student Simulation Model

We consider two populations. A population "Q" where the students have different learning rates and maximum comprehension levels for each KC and another population "P" where, in addiction, the students have limitations in the comprehension of specific parameterizations of the activities. We expect that in the population "Q" an optimization will not provide big gains because all students are able to use all exercises to progress. On the other hand, the population "P" will require that the algorithm finds a specific teaching sequence for each particular student.

Our student model is a generalization of the one used in Knowledge Tracing theory [13]. In our formulation, the competences are not just acquired or not, but are represented as a continuous variable between 0 and 1 that shows how much the KC is mastered. For each student population, Q and P, we describe how we compute the probability of acquiring the next level of competence p^T, and the probability of solving an exercise p^C that, in our case just depends on the current competence level of the student, i.e. assuming that $p^C = p(G) = 1 - p(S)$. In the standard KT formulation this means that the probabilities of guessing or of not making a mistake are defined in the same way.

1) Student Q:

a) Probability of solving an exercise: We define a population "Q" where the maximum achievable competence level of each student follows a normal distribution. For a given KC i, a student with competence level c_i^Q, and an exercise a with

required competence level $c_i(a)$, as defined in Section III-A. We can compute the probability of solving exercise a correctly: $p_i^C(a) = \psi(\pi^{-1}\arctan(\beta(c_i^Q - c_i(a) + \alpha)) + 0.5)$ with α, β and ψ tunable variables with values of 0.1, 30 and 0.7 in our simulations. To take into account all KC we follow the following rule: $p^C(a) = \sqrt[n_c]{\prod_{i=1}^{n_c} p_i(a)}$. As the previous rule has a non-zero probability of solving even the most difficult exercises we include a tunable threshold γ, between 0 and 1 (0.1 in our simulation), that if $p^C(a) < \gamma$, then probability of success becomes zero, $p^C(a) = 0$.

b) Acquisition of KC: The probability of learning is defined as the probability of jumping to the skill level of the exercise being proposed. This probability depends on the difference between the current student's level and the required competence level of the exercise: $p_i^{QT}(a) = \eta(1 - 2(c_i(a) - c_i^Q))$ with η tunable variable, 0.6 in our simulation. We note that the competence level for each different KC evolves independently.

2) Student P:

a) Probability of solving an exercise: Another population "P" provides a more sophisticated simulation of the cognitive behavior of students. Here, the understanding of an exercise, depends on the parameters of each specific activity. For example, this corresponds to simulate a student that has difficulty understanding a given type of representation but can still achieve the most difficult exercise if a different representation is used. For every parameter a_i, each student has a fixed level $l_i(a_i) \in [0 \dots 1]$. The probability that a student correctly solves an exercise is defined as $p^P(a) = \sqrt[m]{\prod_{i=1}^{m} l_i(a_i)}$. To take into account their competence level on each KC we combine the two probabilities as $p^C = p^Q p^P$. For this we combine the probability of success based on their competence levels p^{QT} but taking also into account their comprehension on the particular exercise parameterization p^P. For the experiments, we defined several profiles representing students who: do not understand money or token representation; do not understand written prices; and do not understand fractional numbers.

b) Acquisition of KC: The acquisition of KC follows the principle of the previous modeling for "Q" students for acquiring each competence: $p_i^{PT}(a) = p^P(a) p_i^{QT}(a)$ Another dimension has been added to the "P" students, allowing them the possibility to learn how to understand the different parameters. The understanding of each parameter may increase every time the student solves an exercise with that parameter. This probability is as follows $p^{parT}(a) = \sqrt[n_c]{\prod_{i=1}^{n_c} p_i^{QT}(a)}$. The understanding of parameter evolves independently but with the same probability. When they increase, the new value is: $l_i(a_i) = l_i(a_i) + v_i(1 - l_i(a_i))$, with v_i a tunable variable, 0.001 in our simulations.

B. Results

We present here the results showing how fast and efficiently our algorithms estimate and propose exercises at the correct level of the students. Each experiment considers a population of 1000 students generated using the previous methods and lets each student solve 100 exercises. For all populations the different initial, maximum final level of understanding of each KC is sampled from a truncated gaussian distribution.For the population "P" the values of parameter's understanding are sampled from four different profiles.

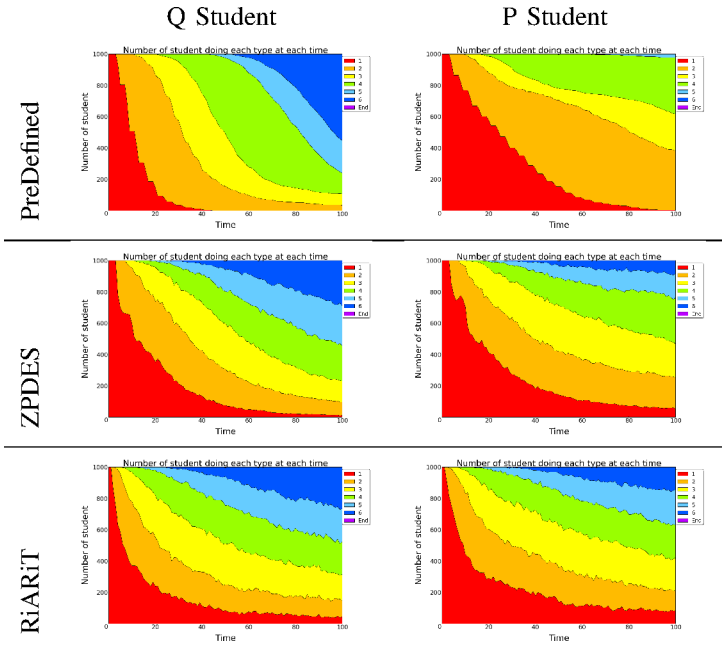

Fig. 1. For each time instant, the curve show the total number of students estimated at being at the level required for the corresponding Exercise Type.

In order to evaluate our algorithms, we use as baseline an optimized sequence created based on instructional design theory, whose reliability has been validated through several user studies, see [29]. This sequence grows in terms of complexity of the problem and simultaneously in terms of the difficulty of interaction.

Figure 1 shows the evolution of the estimation of the students' competence level, corresponding to the exercise that is being proposed to the learners (only showing the parameter Exercise type). We can see that in general, RiARiT and ZPDES starts proposing more difficult exercises earlier while at the same time keeps proposing the basic exercises much longer, matching the actual students level. Figure 2 shows the skill's levels evolution during 100 steps. For Q student, learning with RiARiT and ZPDES is faster than with the predefined sequence, but at the end, Predefined catch up with ZPDES. For P simulations, as students can not understand particular parameter values, they block on stages where the predefined sequence does not propose exercises adequate to their level, while ZDPES, by estimating learning progress, and RiARiT, by considering the estimated level on all KC and parameter's impact, are able to propose more adapted exercises.

Figure 3 shows the competence level of the students after 100 steps, represented as a standard boxplot. For "Q" and "P" students, differences are statistically significative for almost all KC. RiARiT gives better results than Predefined due to its greater adaptation to the students' levels. We can not distinguish between Predefined and ZPDES. In the case of

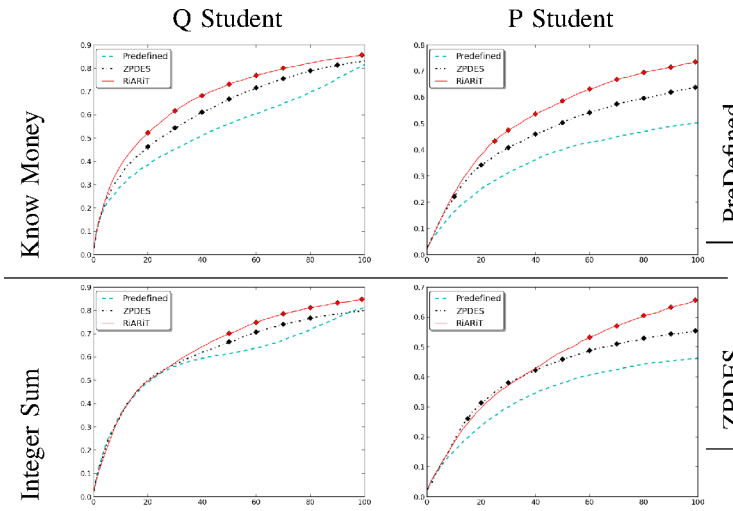

Fig. 2. The evolution of the comprehension of two KC with time for population "Q" and "P". Markers on the curve mean that the difference is significative (red : RiARiT/ZPDES, black : ZPDES/Predefined).

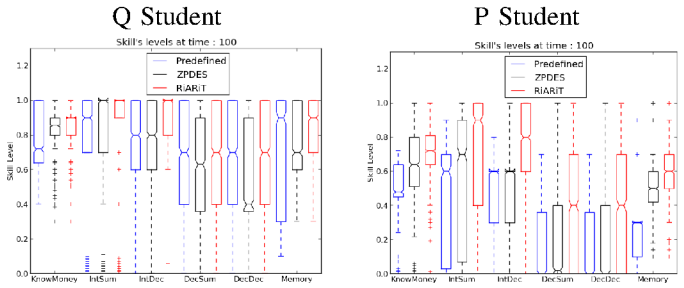

Fig. 3. Distribution of the acquired competence levels after 100 steps represented as a boxplot indicating mean and the 4 quartiles. A statistical significant difference exists if the notchs do not overlap.

students of type "P", RiARiT and ZPDES are both better than Predefined. This is explained because when the student is not able to understand exercises with a specific parameter, a pre-defined sequence can not adapt and propose an alternative path. We can also analyze the errors that the students make during learning because if the exercises are too difficult to solve there will be many errors and this can be a source of frustration. Figure 4 shows that for both types of students, at the beginning, the number of errors is equal among methods but with time, predefined sequence gives rise to more errors than when using RiARiT or ZPDES, in particular for "P" students, providing a less enjoyable learning experience. And for "P" simulation, students have less errors with RiARiT than with ZPDES, showing that RiARiT has a better adaptation than ZPDES.

V. Conclusions and Future Work

In this work we proposed a new approach to intelligent tutoring systems. We showed through simulations and empirical results that a very efficient algorithm, that tracks the learning progress of students and proposes exercises proportionally to the learning progress, can achieve very good results. Using as baseline a teaching sequence designed by an expert in

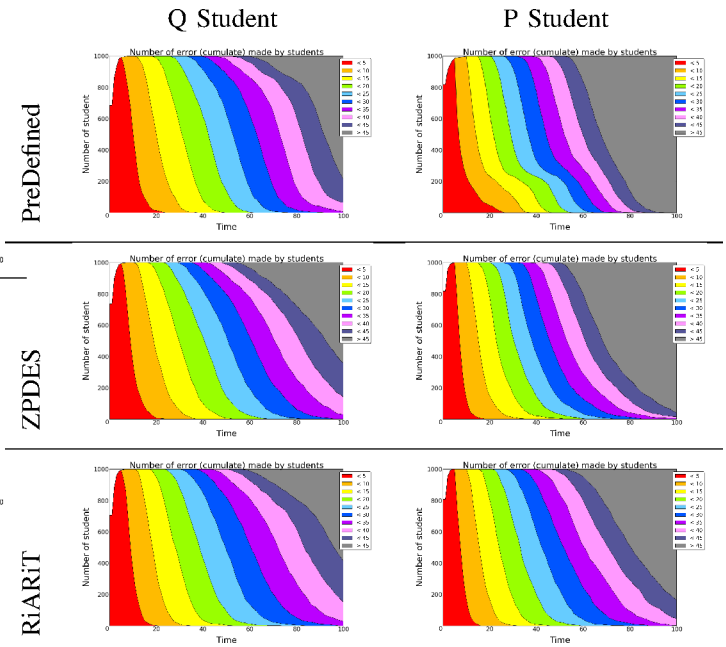

Fig. 4. For each time instant, the curve shows the total number of students that made the number of cumulative number of error (indicated in the colors).

education [29], we showed that we can achieve comparable results for homogeneous populations of students, but a great gain in learning for populations of students with larger variety and stronger difficulties. In most cases, we showed that it is possible to propose different teaching sequences that are fast to adapt and personalized. We introduced two algorithms RiARiT that uses some information about the difficulty about the task, an another algorithm ZPDES that does not use any information about the problem. It is expected that RiARiT, as it uses more information, behaves better when the assumptions are valid, while ZPDES, without any information can not achieve as high performance in well behaved cases but is surprisingly good without any information. Even when compared with a hand optimized teaching sequence ZPDES shows better adaptation to the particular students' difficulties.

There are several directions of future work for this research including: a better validation of the impact of personalization, in particular using our algorithm, for students; a comparison with other methods; and exploiting other sources of information. An initial user study is reported in [24].

Currently we are studying different teaching scenarios to better identify in which situations our method provides higher gains and where it can be easily deployed. The advantage of our system is that it has much less assumption in relation to the cognitive/student model but for this it requires to empirically evaluate the teaching gain of each activity. For this we expect it to be useful in situations where there are many interactions with the tutoring system and with simpler exercises. It will be more suited to the *inner loop*, i.e. within-activity, of the ITS than to the *outer loop*, i.e. across-activity, see definitions in [20]. The comparison with other methods is very difficult due to the different assumptions made by each of them. If

we have access to a well-identified cognitive/student model for populations of students with large variations, we might expect approaches based on POMDP to work best. But, for the contrary case, we expect our approach to better address the identifiability problems and the variations in the student population.

We took a different approach than the one commonly used in KT by considering a continuous level for which a given KC is acquired and not just a boolean variable. This extension reduces the number of KC to define. It is important to see the impact of this new approach and compare it with the standard KT model.

Another interesting direction would be to exploit the clustering that our algorithm implicitly produces in the teaching sequences. We could transfer information from one student to another based on similarities detected at runtime. We see that our algoritms provide sequences that cluster students based on their characteristics [24].

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] P. Auer, N. Cesa-Bianchi, Y. Freund, and R.E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2003.

[3] 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.

[4] Frank J. Balbach and Thomas Zeugmann. Recent developments in algorithmic teaching. In *Inter. Conf. on Language and Automata Theory and Applications (LATA'09)*, 2009.

[5] Joseph E Beck and Kai-min Chang. Identifiability: A fundamental problem of student modeling. In *User Modeling 2007*, pages 137–146. Springer, 2007.

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

[7] D.E. Berlyne. *Conflict, arousal, and curiosity.* McGraw-Hill Book Company, 1960.

[8] Emma Brunskill and Stuart Russell. Rapid: A reachable anytime planner for imprecisely-sensed domains. *arXiv preprint arXiv:1203.3538*, 2012.

[9] Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Stochastic Systems*, 1(4), 2012.

[10] M. Cakmak and M. Lopes. Algorithmic and human teaching of sequential decision tasks. In *AAAI Conference on Artificial Intelligence (AAAI'12)*, Toronto, Canada, 2012.

[11] H. Cen, K.R. Koedinger, and B. Junker. Is over practice necessary?-improving learning efficiency with the cognitive tutor through educational data mining. *Frontiers in Artificial Intelligence and Applications*, 158:511, 2007.

[12] Hao Cen, Kenneth Koedinger, and Brian Junker. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems*, pages 164–175. Springer, 2006.

[13] 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.

[14] Isabella Selega Csikszentmihalyi. *Optimal experience: Psychological studies of flow in consciousness*. Cambridge University Press, 1992.

[15] Jodi Davenport, Anna Rafferty, Michael Timms, David Yaron, and Michael Karabinos. Chemvlab+: evaluating a virtual lab tutor for high school chemistry. In *Inter. Conf. of the Learning Sciences (ICLS)*, 2012.

[16] Stefan Engeser and Falko Rheinberg. Flow, performance and moderators of challenge-skill balance. *Motivation and Emotion*, 32(3):158–172, 2008.

[17] S.A. Goldman and M.J. Kearns. On the complexity of teaching. *Computer and System Sciences*, 50(1):20–31, February 1995.

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

[19] Jacqueline Gottlieb, Pierre-Yves Oudeyer, Manuel Lopes, and Adrien Baranes. Information-seeking, curiosity, and attention: computational and neural mechanisms. *Trends in Cognitive Sciences*, 17(11):585–593, 2013.

[20] Kenneth R Koedinger, Emma Brunskill, Ryan SJd Baker, Elizabeth A McLaughlin, and John Stamper. New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine*, 2013.

[21] 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.

[22] Carol D Lee. Signifying in the zone of proximal development. *An introduction to Vygotsky*, 2:253–284, 2005.

[23] 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.

[24] Manuel Lopes, Benjamin Clement, Didier Roy, and Pierre-Yves Oudeyer. Multi-armed bandits for intelligent tutoring systems. *arxiv*, 2013.

[25] Manuel Lopes and Pierre-Yves Oudeyer. The strategic student approach for life-long exploration and learning. In *IEEE International Conference on Development and Learning (ICDL)*, San Diego, USA, 2012.

[26] Roger Nkambou, Riichiro Mizoguchi, and Jacqueline Bourdeau. *Advances in intelligent tutoring systems*, volume 308. Springer, 2010.

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

[28] A.N. Rafferty, M.M. LaMar, and T.L. Griffiths. Inferring learners knowledge from observed actions. In *Inter. Conf. on Educational Data Mining (EDM)*, 2012.

[29] Didier Roy. Usage d'un robot pour la remédiation en mathématiques. Technical report, Université de Bordeaux, 2012.

[30] Richard M Ryan and Edward L Deci. Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary educational psychology*, 25(1):54–67, 2000.

[31] Y Semet, Y Yamont, R Biojout, E Luton, and P Collet. Artificial ant colonies and e-learning: An optimisation of pedagogical paths. In *International Conference on Human-Computer Interaction*, 2003.

[32] Valerie J Shute. Stealth assessment in computer-based games to support learning. *Computer games and instruction*, 55(2):503–524, 2011.