Learning Content Sequencing in an Educational Environment According to Student Needs 

Ana Iglesias, Paloma Martínez, Ricardo Aler, and Fernando Fernández

Computer Science Department
University Carlos III of Madrid
Avda. de la Universidad, 30, 28911- Leganés (Madrid), SPAIN
Tel: 34-91-624{9917, 9454, 9418, 8842} Fax: 34-91-6249430.
{aiglesia, pmf, aler, ffernand}@inf.uc3m.es

Abstract. One of the most important issues in educational systems is to define effective teaching policies according to the students learning characteristics. This paper proposes to use the Reinforcement Learning (RL) model in order for the system to learn automatically sequence of contents to be shown to the student, based only in interactions with other students, like human tutors do. An initial clustering of the students according to their learning characteristics is proposed in order the system adapts better to each student. Experiments show convergence to optimal teaching tactics for different clusters of simulated students, concluding that the convergence is faster when the system tactics have been previously initialised.

1 Introduction

Web-based education (WBE) is currently a hot research and development area. Traditional web-based courses usually are static hypertext pages without student adaptability, providing the same page content and the same set of links to all users. However, since the last nineties, several research teams have been implementing different kinds of adaptive and intelligent systems for WBE.

The Web-based Adaptive and Intelligent Educational Systems (Web-based AIES) use artificial intelligence techniques in order to adapt better to each student. One of the AIES main problems is to determine which is the best content to show next and how to do it.

RLATES (Reinforcement Learning Adaptive and intelligent Educational System) is a Web-based AIES. This proposal provides intelligence and student adaptability in order to teach *Database Design* topics. RLATES is able to adapt the hypermedia pages contents and links shown to the students (adaptive presentation and adaptive navigation support) based on the Reinforcement Learning (RL) model [5] in order to provide the student an “optimal” curriculum sequence according to his/her learning characteristics in each moment of the interaction. This approach forms part of the

PANDORA project [3], whose main goal is to define methods and techniques for database development implemented in a CASE tool.

System learning begins with a clustering of the students according to their characteristics. A different pedagogical tactical (action policy from the Reinforcement Learning point of view) is learned for each of those clusters. The goal of this paper is to study how many students are necessary to interact with the system until an optimal pedagogical sequence of contents is learned. Experiments will show that it depends on the student clusters homogeneity, so if all the students belonging to the same cluster have similar learning characteristics, the system adapt his curriculum sequence faster and better for each student. Furthermore, we will show that initializing the action policy improves the learning convergence to a very reduced number of students, even when the initialization is done assuming student with different learning characteristics.

The paper is organized as follows: first, the proposal architecture is described in Section 2. Section 3 shows how to apply Reinforcement Learning to educational systems. Next, the functional phases are defined in Section 4. Experiments and main results are presented in Section 5. Finally, the main conclusions and further research of this work are given in Section 6.

2 Proposal Architecture

RLATES is composed of four well differentiated modules shown in Figure 1. The *student module* contains all important information about the student in the learning process: goals, student background knowledge, personal characteristics, historical behavior, etc. Experiments in section 5 show the importance of constructing a good student model and clustering the learners according their critical learning characteristics. A great variety of student models and techniques have been studied [8], and any clustering technique could be used to assort students according to their learning characteristics.

The *domain module* contains all characteristics of the knowledge to teach. For the experimentation analysed in this paper the Database Design domain has been used. The hierarchical structure of topics is used for the domain knowledge, where each topic is divided in other topics and tasks (sets of definitions, examples, problems, exercises, etc.) in several formats (image, text, video, etc.).

In the *pedagogical module*, the educational system finds the best way (the sequence of contents) to teach the *knowledge items*, corresponding with the internal nodes of the tree (topics), to the current student. The definition of this problem as a Reinforcement Learning problem is explained in Section 3.

Finally, the *interface module* facilitates the communication between the AIES and the student. This module applies intelligent and adaptive techniques in order to adapt the content and the navigation to the students, leaning on the pedagogical module, that decides which is the next task to be shown to the student and in which format the knowledge is going to be taught.

3 Application of Reinforcement Learning

Reinforcement learning problems [7] treat agents connected to their environment via perception and action. On each step of the interaction, the agent observes the current state, *s*, and chooses an action to be executed, *a*. This execution produces a state transition and the environment provides a reinforcement signal, *r*, that indicates how good the action has been to solve a defined task. The final goal of the agent is to behave choosing the actions that tend to increase the long-run sum of values of the reinforcement signal, *r*, learning its behavior by systematic trial and error, guided by a wide variety of algorithms [7].

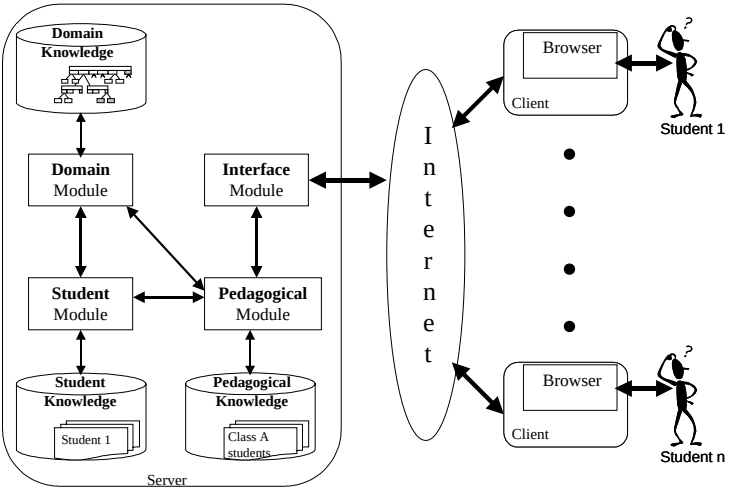

Fig. 1. Proposal Architecture

In RLATES, a *state* is defined as the student knowledge. It is represented by a vector of values related to domain knowledge items (internal nodes of the domain knowledge tree). The i-th value of the vector represent the knowledge level of the student about the i-th topic. In this work we present a simple example and, in order to limit the size of the state space, possible values of the student knowledge are limited to 0 (the student does not know the item) and 1 (the student knows the item). Sometimes, educational systems need to know how good the student learns a topic. This information can be easily added to our system by extending the possible values of the knowledge vector. Furthermore, RLATES perceives the current student state (the values of the knowledge vector) by evaluations (*tests*). The system *actions* correspond with the hypermedia pages that the educational system shows to the student (the *tasks* of the knowledge tree: *definition*, *exercise*, *problem*, etc.). The *reinforcement signal (r)*, supplies a maximum value upon arriving to the goals of the tutor; i.e., when the student learns the totality of the contents of the educational system. This signal is used to update the system's action policy. The system behavior, *B*, should choose the

actions that tend to maximize the long-run sum of values of the reinforcement signal, choosing in this way the optimal tutoring strategy (what, when, and how to teach; the best sequence of contents and how to teach them) to coach the current learner. The action-value function, *Q(s,a)*, estimates the usefulness of executing one action, a_i (showing leaves of the knowledge tree to a student) when the system is in given knowledge state, *s*. This function provides an action policy, defined as is shown in equation (1).

$$\Pi(s)=\arg\max_{a_i} Q(s,a_i). \quad (1)$$

The goal of the learning process is to find the policy such that it maximizes this function, i.e., to obtain the optimal value-action function, denoted by $Q^*(s,a)$, such that $Q^*(s,a) \geq Q(s,a) \ \forall a \in A, s \in S$. There are several ways to learn this function, from dynamic programming [2] to model-free methods [9]. The algorithm implemented in RLATES is the Q-learning algorithm [10], where its value-action function is defined in the equation (2).

$$Q(s,a)=(1-\alpha)\, Q(s,a)+ \alpha\{r+ \gamma \max_{a'} Q(s',a')\} \quad (2)$$

This equation requires the definition of the possible states, *s*, the actions that the agent can perform in the environment, *a*, and the rewards that it receives at any moment for the states it arrives to after applying each action, *r*. The γ parameter controls the relative importance of future actions rewards with respect to new ones, and α parameter is the learning rate, that indicates how quickly the system learns.

In Table 1, how the *Q-learning* algorithm has been adapted to educational system is shown.

The Boltzmann exploration policy has been used in RLATES, because it has been demonstrated previously that it improves the system convergence in relation to the *e-greedy* exploration policy [5]. This exploration policy estimates the probability of choosing the action *a* according to the function defined in equation (3), where τ is a positive parameter called the *temperature*. If the temperature is high, all the probabilities of the actions have similar values and if the temperature is low, it causes a great difference in the probability of selecting each action.

$$P(a) = \frac{e^{\frac{Q_t(a)}{\tau}}}{\sum_{b=1}^{n} e^{\frac{Q_t(b)}{\tau}}} \quad (3)$$

4 System Functional Phases

The use of RLATES requires three phases in order to adapt better to each student in every moment of the interaction: *student clustering*, *system training*, and *system use.*

4.1 Student Clustering

RLATES is able to adapt to each student individually, updating its *Q(s,a)* function according to the interaction with the student.

If the system maintains only one *Q* table for all the students that could interact, RLATES adapts to the set of all the students. However, they could have very different learning characteristics and the adaptation could be low.

Table 1. Adaptation of the *Q-learning* algorithm to Educational Systems

Q-learning adapted to AIES domain
- For each pair (s∈ S, a∈ A), initialize the table entry *Q(s,a)*. - Do for each student of the same cluster o Test the current student knowledge, obtaining s o While the student has not finished learning (*s* is not a goal state) • Select a knowledge tree leaf, *a*, to show to the student, following a exploration strategy. • Test the current student knowledge, *s'* • Receive the immediate reward, *r*. A positive reward is received when the AIES goal is achieved. A null reward is obtained in any other case. • Update the table entry for Q(s,a), that estimates the usefulness of executing the *a* action when the student is in a particular knowledge state: • $Q(s,a)=(1-\alpha)\ Q(s,a)+ \alpha\{r+ \gamma\ max_{a'}\ Q(s',a')\}$ • Let us *s* the current student knowledge state, *s'*.

The solution is to cluster the students according to their learning characteristics before the interaction with RLATES. In this sense, the system maintains one *Q* table for each cluster of students. This allows the system to adapt better to each student cluster.

This phase is not necessary, but it is recommended for better adaptation to each user interacting with the system.

In this paper, a comparison of the system convergence when students of different clusters interact is presented. Two different kind of clusters have been defined based on the homogeneity of the students in the cluster (how similar their learning characteristics are). Only two learning characteristics have been used in order to define the population of students: the kind of task (*introduction* or *definition*) and the format of the hypermedia page (*video* or *text*) that they require to learn, as defined in Table 2.

In the *cluster1*, all the students learn only with *definition* tasks. They will learn the task with a probability of 0.95 if the task has *video* format, while if the task has text format, it will be learnt only with a probability of 0.05.

The *cluster2* students, on the other hand, require *definitions* and *introductions*, both with a percentage of 50%. They will learn the task in the *video* format with a percentage of 75% and the tasks in the *text* format with a probability of 0.25.

It is hypothesized that the RLATES would adapt better to the *cluster1* students individually, because they have very similar learning characteristics and the *cluster2* students are more heterogeneous.

Table 2. Students Cluster Types

Student Clusters	Tasks	Formats
Cluster Type 1 (more heterogeneous cluster)	Definitions (100%)	Video (95%) & Text (5%)
Cluster Type 2 (the most heterogeneous cluster)	Definitions (50%) & Introductions (50%)	Video (75%) & Text (25%)

4.2 System Training

In this phase, the proposal explores new pedagogical alternatives in order to teach the system knowledge, sequencing the domain content in different ways. At the same time the system interacts with the students and updates the appropriate *Q* table.

In this way, the system is able to converge to good teaching strategies based only in previous interaction with other students. This is related to the exploration/exploitation strategies in Reinforcement Learning. In [6], the advantages of the *Boltzmann* exploration strategy are demonstrated. That is why the RLATES system uses this exploration policy.

In this phase, the students that interact with the system could not be learning in the best way, because the system has not learned yet the optimal pedagogical policy. So, it is desired to minimize this phase as possible. This phase finishes once an near optimal action policy has been learned.

4.3 System Use

When the system has converged to a good pedagogical strategy, it is time to use this information to teach other students with similar learning characteristics. From the *Q*-learning algorithm point of view, that means to set the *learning rate* (α) parameter to zero and to select a greedy strategy for action selection. These students will achieve their knowledge goals in the best way the system has learned.

Although the system interaction with students has been divided in two phases (*System Training* and *System Use*), the system never stops learning, adapting its *value-action function, Q(s,a),* in each step of the interaction with each student. It is said that the system is in this phase when it has learned a near optimal teaching sequence of contents.

5 Experimentation

The motivation of the experimentation in this work is to analyze how quickly the system is able to adapt its behavior to the students needs (the size of the *System Training* phase, measured in number of students needed to converge).

All the experimentation has been performed over simulated students in order to theorize about the system behavior when interacting with real students. How the behavior of the students has been simulated is explained in section 5.1. We have empirically tested with a variety of clusters types, from very homogeneous to heterogeneous. Therefore, we believe our conclusions are quite general.

5.1 Simulated Students

In order to obtain general conclusions, a great amount of experiments are necessary, and, then, lots of students are needed to interact with the system.

Most of researchers in educational systems use simulated student in order to prove the applicability of their systems, as was done in Beck's seminal work [1], based in two important motives:

First, it is difficult to persuade one person to use an application that could not be optimized, moreover when the application is an educational system that requires a high concentration (the application tests the knowledge of the person). To persuade a hundred of persons is unthinkable.

Second, the experimentation with real students has a high cost, because they spend a lot of time at the interaction and sometimes they abandon when they get bored or they notice that the system does not teach the content in the best way.

In this paper we want to test the system under very different conditions in order to draw general conclusions. We have use simulated students taking into account only two parameters in order to define the behavior of the simulated students: the student hypermedia format preference (text, video, image, etc.) and the student type of content preference (definition, introduction, etc.) as it has been defined in section 4.1.

Moreover, expert knowledge (knowledge of a database design teacher at the Carlos III University of Madrid) has been used in order to define the prerequisite relationships between the topics and the elements (tasks) of the topics. In this sense, when an hypermedia page of a certain topic, *A*, is shown to a student that do not knows the prerequisite topics of *A*, the student is not able to pass the exam associated to this topic (he could not learn this topic). Every simulated student uses this prerequisite table so that simulation results are reasonable and similar to what could be obtained from current students.

5.2 Experimentation Method

Although simulated students are used in these experiments, real situations are going to be studied: when the system interacts with different students and learns in each interaction according to the *Q-learning* algorithm.

The experiments are going to study three important issues: First, the number of students needed in order for the system to learn the action policy for a kind of cluster without initial pedagogical knowledge, i.e. initializing all the entries of the *Q* table to zero. Second, if the number of student required could be reduced by initializing the action policy with pedagogical knowledge. This could be done by training the system with simulated students which are supposed to model real students. The third issue is what should happen if the model does not represent the real student characteristics. This is equivalent to a situation where the system has been trained with students assumed to belong to *Cluster Type 1* and the real students belong to *Cluster Type 2*.

For the experiments, the *learning rate* parameter (α) of the *Q-learning* function has been fixed to 0.9 and the *Boltzmann* exploration/exploitation policy is followed. Notice that initially the system has no knowledge at all about teaching strategies.

Due to the stochasticity of the domain, experiments have been carried out ten times, and the average produced when the system learns to teach with simulated students has been shown.

5.3 Results

In Figure 2.A, the system convergence is shown when 200 *cluster1* students interacts with RLATES without initializing the *Q* table, for different *temperature* values. It is shown that the system only needs around 50 students in order to learn the optimal policy, obtaining the best result when the *temperature* parameter value is 0.01. With this value, an average of only 20 actions are required to teach the 11 topics of the domain module when *cluster1* students interact with RLATES.

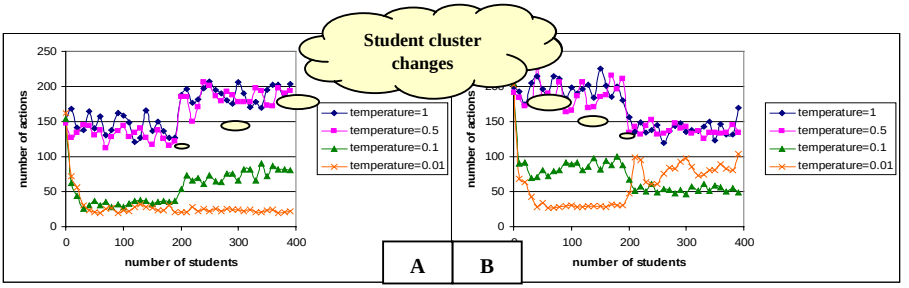

Fig. 2. A: *cluster1* to *cluster2* system convergence; B: *cluster2* to *cluster1* system convergence.

Figure 2.B shows the system convergence when 200 *cluster2* students interact with the system. The convergence property is only achieved when the *temperature* parameter is 0.01, requiring around 60 students. RLATES needs an average of 30 ac-

tions to teach only 11 topics, given the high heterogeneity of the students learning characteristics of this cluster type.

Although this results are reasonable, we are interested in reducing the number of students needed in the *System Training* phase by initializing the *Q* table with the pedagogical knowledge learned interacting with other students. If the initialization is correct according to the learning characteristics of the real students, *Training* phase will be eliminated. Obviously, this assumption is very strong and usually, a pedagogical policy adaptation could be required in order to achieve the best curriculum sequence for current users. This is illustrated in Figure 2.A, where *cluster2* students begin to interact with RLATES when it has previously learned the optimal teaching policy for *cluster1* students, and vice versa in Figure 2.B. In these figures, it can be observed how only 20 students are needed for the *Training* phase, even when this initialization was not good for the current students' learning characteristics. In figure 2.B, we can observe the system does not converge when the temperature is 0.01, because it behaves almost greedy.

Figure 3 summarizes all these results when the temperature is 0.1, showing the *Training* phase length for each cluster of students when no initialization is performed and when a bad initialization is done. Notice that if a correct initialization is performed, *Training* phase length is zero. This situation is the best situation for RLATES.

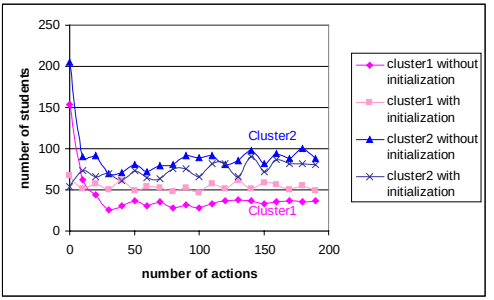

Fig. 3. Experiments summary

6 Conclusions and Further Research

This paper presents the application of the reinforcement learning model in an intelligent and adaptive educational environment. Moreover, experiments that prove the convergence of RLATES, the importance of a previous student clustering and the initialization of the pedagogical strategies have been presented and analyzed.

Simulated students have been used because it is necessary to tune all the learning parameters before using the system with real students. Also, we intended to draw general conclusions for RLATES acting under different conditions.

The experiments prove three important issues. First, the system automatically adapts its teaching tactics according to the current student learning characteristics,

whatever the initial situation was. Second, RLATES adapts to the set of all the students, adapting better to the students when all of them have similar learning characteristics. This motivates a previous student clustering, although it is not necessary for the system. And third, the system reduces its *Training* phase (interacting with fewer number of students) when a initialization of the pedagogical strategies has been done, even for bad initializations (with simulated students with different learning characteristics than real ones). This can be achieved by using simulated students, before the system is put to real use with actual students. Even bad initializations (with students with different learning characteristics) will be better than no initialization at all.

Nowadays, we are involved in the evaluation of the proposal with real students, initializing the pedagogical knowledge with simulated students. It is believed that this initialization will reduce the interaction time with real students until the system convergence. For this validation we are designing an experiment with students of our Faculty belonging to different courses (intermediate and advanced courses of database design) in the Computer Science degree.

References

1. Beck, J. *Learning to Teach with Reinforcement Learning Agent*. In Proceedings of the Fifteenth National Conference on Artificial Intelligence, 1998.
2. Bellman, R. (1957). Dynamic Programming. Princeton University Press. Princeton, NJ.
3. Castro, E., Cuadra, D., Martínez, P., and Iglesias, A. Integrating Intelligent Methodological and Tutoring assistance in a CASE platform: the PANDORA experience. In Proceedings of the Informing Science & IT Education Conference. Cork, Irland, 2002.
4. De Bra, P. and Calvi, L. *An open Adaptive Hypermedia Architecture*. The New Review of Hypermedia and Multimedia, pp. 115-139 (4), 1998.
5. Iglesias, A., Martínez, P. & Fernández, F. (2003). An experience applying Reinforcement Learning in a Web-based Adaptive and Intelligent Educational System. Informatics in Education International Journal. Vol. 2, Pp. 1-18.
6. Iglesias, A., Martínez, P., Aler, R. & Fernández, F. (2003). Analising the Advantages of Using Exploration and Exploitation Strategies in an Adaptive and Intelligent Educational System. Second International Conference on Multimedia and Information & Communication Technologies in Education (m-ICTE 2003), Pp. 489-493, Vol. 1.
7. Kaelbling, L.P., Littman, M.L. & Moore, A.W. (1996) Reinforcement learning: A survey. *Int. J. of. Artificial Intelligence Research*, 237-285.
8. Sison, R. & Shimura, M. (1998). Student Modeling and Machine Learning. *International Journal of Artificial Intelligence in Education*, 9, 128-158.
9. Sutton, R.S. (1988). Learning to Predict by the Method of Temporal Difference. *Machine* Learning. Vol. 3, Pp. 9-44.
10. Watkins, C.J.C.H. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, 1989.
11. Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. Los Altos, CA: Morgan Kaufmann.