IJICTR International Journal of Information & Communication Technology Research

Volume 9-Number 3-Summer 2017(47-58)

# *A Novel Approach for Learning Improvement in Interactive Classroom Environments using Learning Automata*

Hajar Hajary
Department of Computer Science and Research Branch Islamic Azad University
Tehran, Iran
Hajar_Hajary@yahoo.com

Ali Ahmadi
(Corresponding Author)
Department of Computer Engineering, K.N. Toosi University of Technology
Tehran, Iran
Ahmadi@kntu.ac.ir

Maryam Khani
Department of Computing
Macquarie University
North Ryde, NSW 2109, Sydney, Australia
maryam.khani@hdr.mq.edu.au

**Received: February 16, 2017 - Accepted: June 25, 2017**

***Abstract*- Determining the best way of learning and acquiring knowledge, especially in intelligent tutoring systems has drawn researchers' attention during past years. Studies conducted on E-learning systems and strategies proposed to improve the quality of these systems, indicate that the main learning resources for students in an educational environment are provided by two crucial factors. The first is the teacher who can basically influence students' success through demonstrating her ability and skills, and the second is interaction among students. In this article, a new modeling approach is presented for improving learning/teaching models as well as interaction among learners, from which the most benefit can be derived by learners. The proposed model uses the learning automata for modeling the teacher and her behavior in such a way that she can also learn and teach better at the same time, thus improves her teaching skills. The model also uses cellular learning automata in order to model behavior of the learners as well as interactions between the learners for knowledge acquisition. The results indicate that in addition to teacher's skills, the interaction/communication among learners can significantly improve the quality and speed of learning as compared with previous methods.**

*Keywords: tutorial like system, interactions, learning automata, cellular learning automata*

## I. INTRODUCTION

Intelligent tutoring systems (ITS) are novel metaphors for educational paradigms that employ AI techniques to enhance learners' knowledge acquisition and internalization process, and improve teachers' teaching abilities, simultaneously [1,2]. In general, these systems concern with three main factors, including domain model, student model, and educational model, where the main focus lies on the student model. It is noteworthy that in a few studies, user interface is considered as the fourth factor [3,4]. Domain model is a control center that encompasses the entire domain knowledge, which generates instruction content and evaluates student's performance [5]. Whereas student model represents the student's behavior, attitude and state [6]. Educational model specifies how the student should be taught [7]. Self [6] defined these three factors as

the tripartite architecture for an ITS: the what (domain model), the who (student model), and the how (tutoring model).

The applications of machine learning techniques in ITS systems have been investigated in a number of studies which suggest such techniques can improve teaching and learning quality. These techniques can be utilized in different parts of ITS such as background knowledge [8]. Beck et al. used machine learning to improve tutoring strategy [9]. Sision and Shimura suggested that analogical learning is more appropriate for learning-level analysis, whereas reinforcement learning is more appropriate for tutoring [10]. Reinforcement learning as a semi-supervised machine learning approach can be used to train an agent to comply with the student's needs [6]. Frasson et al. designed the main ITS components (student model, domain knowledge and the tutoring model) in form of intelligent agents[11]. Sision and Legaspi utilized reinforcement learning to model the learning process [12]. Baffes and Mooney implemented ASSERT which exploits reinforcement learning and domain knowledge for student modeling to find the errors that the new students may make [13]. Lelouche devised a series of interactive elements to model the learning process in intelligent educational systems [14]. Finally, Hesham and Oommen [15, 16, 17] and Wang and Jiang, Hoa Ge et al. used learning automata to model the students' learning process as well as the interactions between them [18, 19]. Mostly, computer-aided tutorial systems present the educational material indiscriminately and do not consider the learner's scientific and educational background. Thus, in such systems, the tutorial methods do not suit the learner's needs and interests due to the lack of learner's mental and behavioral models. According to a well-established theory in education, learners follow their self-customized learning pattern through the learning process [15]. Thus, a practical ITS must be able to adapt the learners' needs and provide them with customized educational material. This capabilities can be embedded to the tutorial systems only by applying AI techniques.

Learners and teachers are the main entities that play important roles in training system Teachers are the main source of knowledge acquisition for students, and the teachers' skills profoundly influence the students' success rate. Thus, constructing proper teacher models can positively influence the success rate of an educational system. The teacher model represents the decision making mechanism utilized as teaching strategies and tries to optimally transfer the educational material to the students. In this paper, we propose a learning model for the teacher, so that s/he can adapt to the students' learning model. Using MetaLA model proposed by Oommen and Hashem teacher can distinguish each student's model type [16, 17]. This structure can recognize the student's mental model during learning process. The teacher exploits this model to learn how to help each student and concurrently guides the students toward their best learning performance using a penalty-reward paradigm. Thus, through this learning-while-teaching process, the teacher can increase the students' learning efficiency significantly. Furthermore, Interactions among students are another source of learning in real-life educational environments. Although traditional educational paradigms assume that the students learning highly depends on teachers, in reality, they also adjust their learning curve based on the interactions among them. We generalize the traditional paradigm to let the student to learn from a so-called classroom of students learning at different rates and abilities. One of the main objectives of the proposed system in this study is to introduce a new method based on the cellular learning automata to model the interaction among students in a tutorial like system. In this model a student is a member of a classroom of students, in which s/he learns from the teacher and obtains information from other students. In our system, a student simulator is used to mimic the behavior of real-life students during the learning process. Students are divided into three categories based on their mental model including slow, normal and fast learners. This classification is in accordance with the real educational systems. In this model, each student is considered as a learning automaton within a cell. The interactions among students are modeled as the interactions among different learning automata (i.e. neighbouring cells), and the student-teacher interaction is simulated as the interaction learning automata with their environment. This models aims to accelerate the learning process and enhance the overall quality of the students' learning.

The paper is organized as follows: .in Section II presents an overview on cellular learning automata. The concept of tutorial-like systems is thoroughly discussed in section III. Our proposed intelligent tutorial-like system is elaborated on in section IV. Section V presents the experimental results and evaluations. Finally, section VI concludes the paper.

## II. CELLULAR LEARNING AUTOMATA

Research in Learning automata started with Tsetlin who introduced the use of deterministic and stochastic automata operating in a random environment as learning model [20]. The term "Learning Automata" was first publicized in the survey paper by Narendra and Thathachar [21]. The

goal of LA is to 'determine the optimal action out of a set of allowable actions". These automata are mostly used in the systems with incomplete environmental information [22, 23]. An automaton can select an action among a set of actions as its output. Once the action is selected and executed, it is evaluated by the environment and the corresponding feedback is sent to the learning automata either as a positive feedback signal (i.e. in case the action was done properly) or a negative one (i.e. in case the action was done improperly). The value of this signal determines which actions should be chosen in the following steps. This process makes the automata to gradually converge to the most appropriate action regarding the environmental criteria. The closed-loop interaction between a stochastic automaton and the random environment is shown in Figure 1.

The machine acts randomly in the probabilistic environment, and updates the probabilities of action selection based on the inputs received from the environment. The learning automata are classified into two classes including variable structure automata (VSSA), fixed structure automata (FSSA) [21]. A VSSA is defined as a a quadruple $M=<\alpha,\beta,p,T>$ in which $\alpha=\{\alpha_1,\alpha_2 ,..., \alpha_n \}$ represents the action set of the automaton, $\beta=\{\beta_1,\beta_2 , ..., \beta_r\}$ is the input set, $p=\{p_1,p_2 , ..., p_r\}$ represents the action probability set, and finally $p(n+1)=T[\alpha(n),\beta(n),p(n)]$ represents the learning algorithm [21, 24].

The automaton selects an action $\alpha_i$ regarding the action probability set $p$, and performs it within the environment. Then, the automaton updates its action probability set using equation (1) for favorable responses, and equation (2) for unfavorable responses based on the received reinforcement signal from the environment.

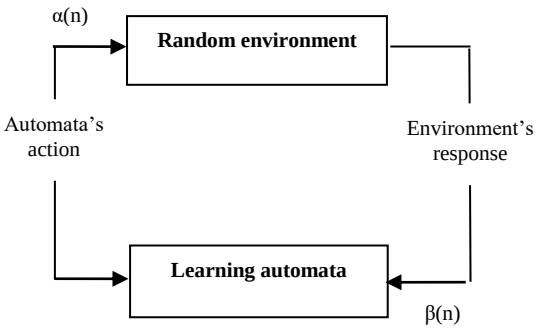


**Fig.1** Closed-loop interaction between a learning automaton and environment

$$\begin{cases} p_i(t+1) = p_i(t) + a.(1 - p_i(t)) \\ p_j(t+1) = p_j(t) + a.p_j(t) \quad \forall j \neq i \end{cases} \quad (1)$$

$$\begin{cases} p_i(t+1) = (1-b)\,p_j(t) \\ p_j(t+1) = \dfrac{b}{r-1} + (1-b)\,p_i(t) \quad \forall j \neq i \end{cases} \quad (2)$$

Where $p_i(t)$ is the probability of selecting action $\alpha_i$ at time $t$. $a$ and $b$ are reward and penalty parameters, respectively. In the case of $L_{R-P}$ learning algorithm the reward and penalty parameters are set equal. $L_{R\varepsilon P}$ algorithm sets the reward parameter significantly smaller than penalty parameter, and in $L_{R-I}$ learning algorithm, the penalty parameter is set to zero. On the other hand, for fixed structure stochastic automata (FSSA), their transitions are determined by state transition probabilities that are fixed with time. The FSSA suffers from slow convergence speed in comparison with VSSA.

Pursuit automata are new models of learning automata that estimates the optimal action was introduced by Thathachar and Sastry [24, 25]. In their novel approach, the updating algorithm improves its convergence results by using the history to maintain an estimate of the probability of each action being rewarded, in what is called the estimate vector. While in nonestimator algorithms the probability vector is updated based on the environment's response, in an estimator algorithm the update is based on both the environment's response and the estimate vector. Thus, it is easy to observe cases where an action is rewarded while the probability of choosing another action is increased [15]. The main advantage of the Pursuit automata over other types is their high speed of learning process.

Cellular automata introduced by Von Neumann are mathematical models for defining systems that consist of a large number of simple identical components with local interactions [26]. The combination of cellular automata and learning automata results in cellular learning automata (CLA) which is superior to cellular automata due to its learning ability and also is superior to single learning automaton due to its distributed processing ability which is provided by employing a set of interacting learning automata.

CLA is a mathematical model for simulating dynamical complex systems that include large number of simple components. These simple components have learning capabilities and act together to produce complex behavioral patterns. In other words, a CLA is a cellular automaton in which a learning automaton is assigned to its every cell [27]. The learning automaton residing inside each cell determine the state of the cell on the basis of its action probability set. The active rule in CLA and the actions selected by the neighbouring cells determine the reinforcement signal to the learning automata residing in that cell. The neighbouring learning automata of any cell constitute its local environment. The state of the cell is determined by the action probability set of the learning automaton residing in that cell. The initial value of the state may be set based on the past experience or randomly. After initializing the states,

the reinforcement signal to each learning automaton is determined by the CLA rule. Then, each learning automaton updates its action probability set based on the reinforcement signal and the chosen action. This process continues until the desired result is obtained. A sample structure of a CLA is depicted in Figure 2 [27, 28].

Formally, a *d*−dimensional cellular learning automata can be defined as A = ($Z^d$, Φ, A, N, F), where: $Z^d$ is a lattice of *d*−tuple of integer numbers, Φ is a finite set of states, A is the set of learning automatons each of which is assigned to each cell of the cellular automata, $N=\{X_1,X_2, \dots , X_m\}$ is a finite subset of $Z^d$ called neighbourhood vector where *m* represents the number of neighbouring cells and $X_i \in Z^d$ and finally F is a set of action functions each of which determines the next action of each automaton. The neighbourhood vector determines the relative position of the neighbouring cells from any given cell u in the lattice $Z^d$. The neighbours of a particular cell u are set of cells Which are located in a neighbourhood radius *r*. We assume that there exists a neighbourhood function N(u) mapping a cell u to the set of its neighbours.

A number of applications for cellular learning automata have been developed recently such as modeling of commerce networks, fixed channel assignment in cellular networks, image processing, and VLSI placement [26].

## III. TUTORIAL-LIKE SYSTEM

Tutorial-like systems are special educational systems that involve artificial intelligence techniques and methods to represent the knowledge, as well as to conduct the learning interaction. These systems represent a student's state through the learning process. In these systems, the student can learn and be tested without the presence of a real person. Even students can be replaced by a simulated student that mimics a real-life student. The teacher attempts to provide the training materials to a set of student simulators.

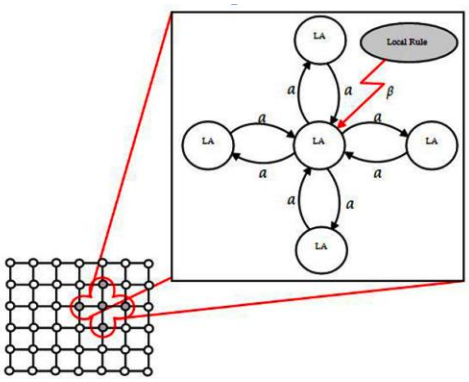


**Fig. 2** A sample structure of cellular learning automata (Meybodi and Beigy 2004)

Moreover, the students are allowed to share their information with each other, so that they can learn from each other which is more realistic than the traditional learning paradigms. In our model, components of the tutorial-like system follow a scholastic model. The students obtain knowledge through multiple choices questions. These questions include several items with different confidence level. The student gradually learns to choose the answer with the highest confidence [15].

Tutorial-like systems have some similarities with the well-established tutorial systems. They both model the teacher, the student, and the domain knowledge. However, they have some main differences as well. These differences include different teacher type, none-real students, uncertain course material, and testing versus evaluation [15]. The first difference is different teacher type. In tutorial systems, the teacher is assumed to have perfect information regarding the material to be taught. Also, the knowledge of teaching and communicating the domain material and interactions with students is embedded into the teacher model. The teacher in our Tutorial-like system possesses different features. First, the teacher in our model is uncertain of the teaching material. Second, the teacher does not initially possess any knowledge of "how to teach" the domain subject. Rather, the teacher himself is involved in a learning process, and s/he learns what teaching material has to be presented to a particular student. To do so, the teacher follows the Socratic learning model by teaching the material using questions that are presented to the students. Then, s/he uses the feedback from the students and their corresponding learning automata to suggest new teaching materials. Although omitting the how-to-teach knowledge from the teacher takes away the bread-and-butter premise of the teaching process in a tutorial system, in a tutorial-like system, it allows the system to be modeled without excessive complications and renders the modeling of knowledge less burdensome.

The second difference is that a tutorial system is used by real students, whereas in our tutorial-like system, there is no need for real students. Thus, the system can be used by either a student simulator which mimics the behaviors and actions of real students using the system, or an artificial entity such as a software component that needs to learn specific domain knowledge. The third difference arises from uncertain course material. Unlike the traditional tutorial systems in which the domain knowledge is well-defined, in our tutorial-like system, the domain-knowledge of teaching material has some degree of uncertainty. The teaching material contains

some questions with the corresponding probability which associates to the certainty of correct answers to the questions. Finally, the last difference is testing versus evaluation. Sanders (2008) differentiated between the concepts of teaching evaluation and teaching testing. The teaching evaluation is defined as an interpretive process in which the teacher determines the students' performance and their needs. In a tutorial system, an evaluation is required to measure the student's performance online. In our tutorial-like system, the student acquires knowledge using a Socratic model, where s/he gains knowledge from answering questions without having any prior knowledge about the subject material. In our model, the testing is based on the performance of the set of student simulators.

## IV. Intelligent Tutorial-Like System

Our proposed model attempts to improve the learning in tutorial-like systems using hybrid techniques, so that slow and normal learners can improve their learning abilities and approach the abilities of fast learners. In this way, the learners' learning efficiency is increased collectively regardless of the group they belong to (i.e. slow, normal, or fast). Similar to the model proposed in (Hashem and Oommen April. 2010, 2013), our proposed model consists of several learning automata connected indirectly to one another. It improves the learning process in three directions. First, the teacher finds the best penalty-reward vector by simultaneous learning and teaching (i.e. teacher's learning scheme). Second, the teacher helps learners to identify their mistakes and correct them by testing learners during teaching (i.e. teacher's test scheme). Third, learners use their classmates' knowledge to improve their own by communicating with them through CLA. Structure of the proposed model is illustrated in figure 3.

As shown in Figure 3, the model represents the structure of interconnection, which can be viewed as being composed of two levels: a lower level automaton, which is the student LA, and a higher level automaton (i.e., the meta-LA*)* which attempts to characterize the learning model of the students (or student simulators), While the latter uses the tutorial-*like* system and consists of following items:

- **Learn teacher:** by learning while teaching, the teacher can learn and teach better at the same time, thus improves her teaching skills.
- **Teacher's test:** by testing learners during teaching the teacher helps learners to identify their mistakes and correct them.

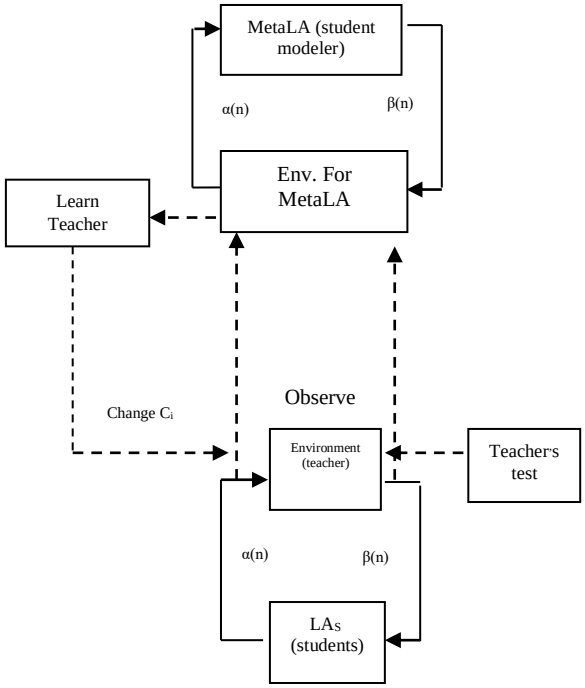


**Fig.3** Structure of proposed model

In some research works, it is mentioned that in tutorial-like systems, regardless of the students' different mental models, all of the students are subjected to a uniform education strategy and exposed to the same penalty-reward method [15]. On the other hand, using the method proposed by Hashem and Oommen the teacher can identify the student's learning trend (i.e. slow, normal, fast) and decide a proper penalty-reward vector regarding the learner's behavior to guide her through the learning process. In this study, we utilized MetaLA to model the students' behavior which recognizes the student's learning trend in time intervals, and enables the teacher to assign different penalty-reward vectors for each student [15, 16].

### A. *Environment Learning Algorithm (teacher's learning)*

The first step in this algorithm is to extract the student' trend from the student simulator using Once the students' learning trend is determined during a specific time interval, the environment learning algorithm tweaks the penalty-reward vector until the best vector is assigned to each student regarding her learning abilities. In this algorithm, the probability vector is modified in both linear and nonlinear manner using equations (3) and (4), respectively.

$$\delta = \begin{cases} \lambda_S & for\ slow\ student \\ \lambda_N & for\ normal\ student \end{cases} \qquad (3)$$

$$\Phi(X) = \delta X^m \quad 0 < \delta < 1, \quad m \in D$$
$$Subject\,to: \begin{cases} C_l(n) + \delta < C_j(n) \\ C_l(n) + \Phi(X) < C_j(n) \quad \forall n, j \neq l \\ \Phi(X) > 0 \end{cases} \quad (4)$$

Where one action $\alpha_l$ continues to have the minimum penalty probability $c_l$

The purpose of our proposed model is to increase the students' collective learning efficiency regardless of their different learning abilities. In this way, students with slow and normal learning abilities get closer to the abilities of fast students.

### B. *Teacher's Test*

We can consider another learning mechanism for students through which they can detect and correct their mistakes, improve their learning abilities, and increase their learning speed. This mechanism is performed by teacher who, periodically test the students to familiarize them with their mistakes and the corrections. Tests play an important role in the learning process by evaluating the students' knowledge and detecting their mistakes. A proper way to model a test is to study each student's (i.e. automaton) behavior during various time periods and identify their mistakes (i.e. wrong knowledge). This process can be modeled using (5).

$$W = Max(\frac{P_i}{N_i}), \quad B = Min(\frac{P_i}{N_i}), \quad i = 1..r \quad (5)$$

Where $W$ refers to the wrong action, $B$ refers to the best action, $N_i$ is the number of times that the action is selected within the time period, and $P_i$ is the number of penalties per action in each period. In other words, the maximum penalty per action selection is assigned to the wrong knowledge or incorrect behavior and thus the probability of selecting that action is reduced. On the other hand, the probability of selecting the best action is increased proportionally. This process is depicted in (6).

$$\begin{cases} p_i(t+1) = (1-h)p_i(t) \\ p_j(t+1) = p_j(t) + hp_j(t) \end{cases} \quad (6)$$

Where i is the index of wrong action and j is the index of best action.

### C. *Tutorial-Like System Based on Cellular Learning Automata*

In this research, we simulate the tutorial-like system using CLA. We employed a student simulator to mimic real-life behavior of the students during their learning process. As aforementioned, each student is considered as a learning automaton in a cell. The interactions between students are simulated through the interactions among different learning automata (i.e. neighbouring cells). These interactions can accelerate the individual and collective learning process. Structure of this model is illustrated in figure 4.

In the simulations, we applied the majority-minority rule for neighbourhood effect [29]. This rule states that if the cell selects action $\alpha_i$ and at least five of its neighbours select the same action, the selected action is likely to be the correct one. In this case, the neighbour's response is considered as a favorable response. On the other hand, if less than five neighbours select the same action, the neighbours' response is considered as undesirable response. We integrate the neighbours' responses with environmental factors in a model shown in Table 1. In this table, the neighbourhood rule presents the responses of the neighbours (i.e. zero in case of favorable response and one in case of undesirable response).

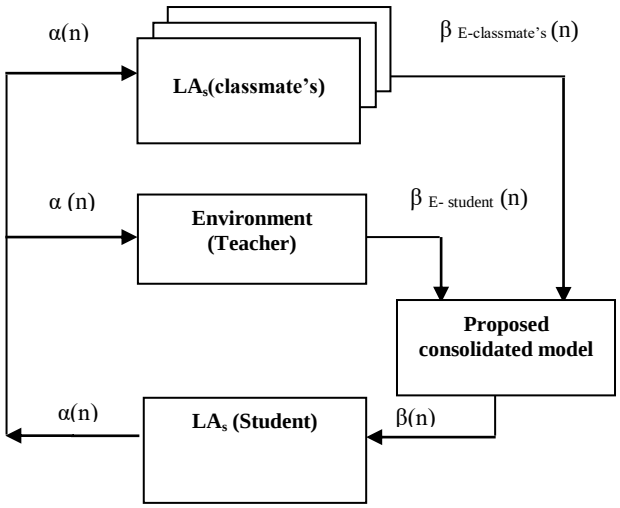


**Fig.4** structure of Tutorial-like System Based on Cellular Learning Automata.

**Table.1** Model for integration neighbourhood and environmental factors

| Neighborhood rule | Environmental factors | Result of $\alpha_i$ |
|---|---|---|
| Neighbor=0 | Reward | Reward |
| Neighbor=0 | Penalty | Reward=0.2 penalty=0.8 |
| Neighbor=1 | Reward | Reward=0.85 penalty=0.15 |
| Neighbor=1 | Penalty | Penalty |

D. *Modeling a Students*

In this system, each student's model indicates the behavior, state of the mind, and knowledge acquisition approach of that student. We selected a slow VSSA to represent slow students, a VSSA to model normal students, and finally an estimator automata (Pursuit) to simulate fast students. In addition, all actions and overall performance of the student are recorded online.

## V. EXPERIMENTAL RESULTS

In order to evaluate the proposed system, we implemented a prototype simulation of the student-classroom interaction system. We defined nine students in our simulations so that we can compare our results with the system proposed in [15]. In the system proposed in [15], the students only learn from the teacher and do not have interaction abilities. Thus, the cellular automaton employed in simulation consisted of nine cells with a learning automaton assigned to each cell regarding the type of the student (i.e. whether he is a slow, normal or fast learner). In order to simulate the fast-learning students, the student simulator used a pursuit $PL_{RI}$ scheme with $\lambda \in [0.0041, 0.0127]$ (based on proposed model in [15]). In this scheme, each LA updates its action probability vector if it obtains a reward. To simulate the normal-learning students, VSSA with the $L_{RI}$ scheme and $\lambda \in [0.0182, 0.0192]$ was utilized. Finally, a VSSA model with $L_{RI}$ scheme and $\lambda \in [0.0142, 0.0152]$ was exploited to simulate the slow-learning students.

The evaluation is based on 75 samples of simulations performed on the proposed system. The teaching materials are represented by two different environments: two four-action environment ($E_{4,A}$ and $E_{4,B}$) and two ten-action environments ($E_{10,A}$ and $E_{10,B}$). Both of these environments are widely used benchmarks for evaluating learning automata [15]. We define a threshold, $T$, as the convergence criterion. As soon as the as probability of selecting an action exceeds the threshold value, we stop the algorithm. In our simulations, we set the threshold to 0.99 which can result in a high accuracy. The value of $\lambda$ set for the mentioned environments is shown in Table 2. Furthermore, the reward probabilities of these environments are set to:

$E_{4,A}$ ={0.7,0.5,0.3,0.2}, $E_{4,B}$ ={0.1,0.45, 0.84, 0.76}, $E_{10,A}$ ={0.7, 0.5, 0.3, 0.2, 0.4, 0.5, 0.4, 0.3, 0.5, 0.2}, $E_{10,B}$ ={0.1, 0.45, 0.84, 0.76, 0.2, 0.4, 0.6, 0.7, 0.5, 0.3}.

**Table.2** The λ of the student simulators LA

| Student type | λ | | | |
|---|---|---|---|---|
| | $E_{4,A}$ | $E_{4,B}$ | $E_{10,A}$ | $E_{10,B}$ |
| Fast-learning student | 0.0127 | 0.0041 | 0.0127 | 0.0041 |
| Normal-learning student | 0.0192 | 0.0182 | 0.0192 | 0.0182 |
| Slow-learning student | 0.0142 | 0.0152 | 0.0142 | 0.0152 |

### A. *Environment Learning Algorithm (teacher learning)*

#### 1) *Simulation Based on the Linear Algorithm*

The simulation results obtained for the linear method are shown in Table 3. As shown, the proposed method leads to a significant improvement in the proficiency level of the slow and normal students compared to the method where all students are treated similarly. By assigning the optimal penalty-reward vector to each student, the algorithm approximates the behavior of fast-learning students which in turn, reduces the needed number of iterations for convergence. For example, it is shown that the number of iterations for normal-learning and slow-learning students in $E_{A,4}$ is decreased from 996 and 1382 to 656 and 760, respectively. As another example, in $E_{B,10}$ which is considered as a difficult environment due to the large number selected actions and close penalty probability vector, it is observed that the number of iterations for normal students for achieving convergence is reduced from 2114 to 1843, and similarly, for slow students it is decreased from 2859 to 2134. Considering that the number of iterations for obtaining convergence for fast students is 1655, we find that when the teacher learns how to deal with students, the students' learning process will improve.

#### 2) *Simulation based on the nonLinear Algorithm*

Moreover, we simulated the system with nonlinear algorithm whose results are depicted in Table 4. Same to the linear algorithm, it is shown that the slow and normal students' learning abilities are improved considerably, and they have managed to approximate the learning trends of fast students. Comparing the results from linear and nonlinear algorithms leads to the conclusion that the nonlinear algorithms can find the optimal penalty-reward vector for each student within fewer iterations due to its ability in modifying the probability vector with more accuracy than the linear technique.

The improvements of students learning process in both proposed model and model introduced in are shown in Figure 5 [15]. It is shown that the

proposed model improves the learning abilities of slow and normal students considerably, Furthermore, it is shown that the learning speed of the slow and normal students closely tracks the learning speed of the fast students.

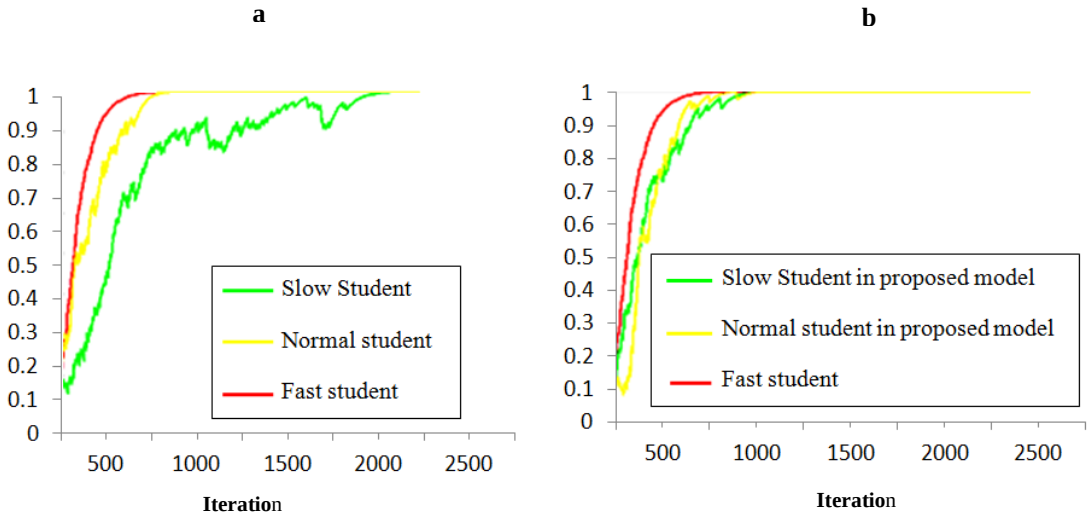


**Fig. 5** The rate of students learning in the reference model (a) compared with the proposed model (b).
(x-axis: Iteration , y-axis: Threshold)

**Table.3** Convergence of linear algorithm and comparison with presented model in [15].

| | | **Fast student** (λ: 0.004 - 0.0127) | | **Normal student** (λ: 0.0182 - 0.0192) | | **Slow student** (λ: 0.0142 - 0.0152) | |
|---|---|---|---|---|---|---|---|
| | | **# of Iteration to converge** | | | | | |
| **Env.** | **# of Actions** | **Old model [15]** | **λ** | **Old model [15]** | **proposed model** | **Old model [15]** | **proposed model** |
| **$E_A$** | 4 | 572 | $\lambda_S$=0.01, $\lambda_N$=0.007 | 996 | 656 | 1382 | 760 |
| **$E_B$** | 4 | 1482 | $\lambda_S$=0.002, $\lambda_N$=0.001 | 2201 | 1669 | 2633 | 1673 |
| **$E_A$** | 10 | 686 | $\lambda_S$=0.009, $\lambda_N$=0.004 | 1297 | 852 | 1804 | 1137 |
| **$E_B$** | 10 | 1655 | $\lambda_S$=0.002, $\lambda_N$=0.001 | 2114 | 1843 | 2859 | 2134 |

Reward probabilities for 4-action environment are :
$E_{A,4}$: 0.7 0.5 0.3 0.2 $E_{B,4}$: 0.1 0.45 0.84 0.76
Reward probabilities for 10- action environment are:
$E_{A,10}$: 0.7 0.5 0.3 0.2 0.4 0.5 0.4 0.3 0.5 0.2 $E_{B,10}$: 0.1 0.45 0.84 0.76 0.2 0.4 0.6 0.7 0.5 0.3

**Table.4** Convergence of nonlinear algorithm and comparison with presented model in [15]

| | | **Fast student** (λ: 0.004 - 0.0127) | | **Normal student** (λ: 0.0182 - 0.0192) | | **Slow student** (λ: 0.0142 - 0.0152) | |
|---|---|---|---|---|---|---|---|
| | | **# of Iteration to converge** | | | | | |
| **Env.** | **# of Actions** | **Old model [15]** | **λ** | **Old model [15]** | **proposed model** | **Old model [15]** | **proposed model** |
| **$E_A$** | 4 | 572 | $\lambda_S$=0.01, $\lambda_N$=0.007 | 996 | 588 | 1382 | 1011 |
| **$E_B$** | 4 | 1482 | $\lambda_S$=0.002, $\lambda_N$=0.001 | 2201 | 1639 | 2633 | 1645 |
| **$E_A$** | 10 | 686 | $\lambda_S$=0.009, $\lambda_N$=0.004 | 1297 | 805 | 1804 | 1041 |
| **$E_B$** | 10 | 1655 | $\lambda_S$=0.002, $\lambda_N$=0.001 | 2114 | 1733 | 2859 | 1838 |
| Reward probabilities for 4-action environment are : $E_{A,4}$: 0.7 0.5 0.3 0.2 $E_{B,4}$: 0.1 0.45 0.84 0.76 Reward probabilities for 10- action environment are: $E_{A,10}$: 0.7 0.5 0.3 0.2 0.4 0.5 0.4 0.3 0.5 0.2 $E_{B,10}$: 0.1 0.45 0.84 0.76 0.2 0.4 0.6 0.7 0.5 0.3 | | | | | | | |

**Table.5** Convergence of teacher's test model and comparison with presented model in [15]

| | | **Fast student** (λ: 0.004 - 0.0127) | | **Normal student** (λ: 0.0182 - 0.0192) | | **Slow student** (λ: 0.0142 - 0.0152) | |
|---|---|---|---|---|---|---|---|
| | | **# of Iteration to converge** | | | | | |
| **Env.** | **# of Actions** | **Old model [15]** | **proposed model** | **Old model [15]** | **proposed model** | **Old model [15]** | **proposed model** |
| **$E_A$** | 4 | 572 | 522 | 996 | 713 | 1382 | 1187 |
| **$E_B$** | 4 | 1482 | 1371 | 2201 | 1301 | 2633 | 1657 |
| **$E_A$** | 10 | 686 | 630 | 1297 | 1042 | 1804 | 1434 |
| **$E_B$** | 10 | 1655 | 1502 | 2114 | 1474 | 2859 | 1842 |
| Reward probabilities for 4-action environment are : $E_{A,4}$: 0.7 0.5 0.3 0.2 $E_{B,4}$: 0.1 0.45 0.84 0.76 Reward probabilities for 10- action environment are: $E_{A,10}$: 0.7 0.5 0.3 0.2 0.4 0.5 0.4 0.3 0.5 0.2 $E_{B,10}$: 0.1 0.45 0.84 0.76 0.2 0.4 0.6 0.7 0.5 0.3 | | | | | | | |

3) Teacher's Test

Results indicated in Table 5 indicate that test design can help different students to converge faster towards their learning goal. For example, in $E_{A,4}$, the number of iterations required for the convergence of slow-learning students is reduced from 1382 to 1187, which indicates that the tests can help students learn from their mistakes and stop repeating them. As another example, in $E_{A,4}$ and $E_{A,10}$ , the number of iterations required for convergence of normal students is reduced from 996 to 713, and from 1297 to 1042, respectively. Similarly, the number of iterations required for convergence of normal students in $E_{B,4}$ and $E_{B,10}$ is reduced from 2201 to 1301, and from 2114 to 1474, respectively.

*4) Student Interaction with Cellular Learning Automata*

Diversity of students with different learning abilities in an educational environment, according to different knowledge level of the learners and their interactions, the benefits related to each learner or a group of learners is different from each other. For example, when the number of fast learners is more than the number of slow learners in the learning group, there is a high probability that the slow learner interacts with faster individuals. Thus, he will have a significant progress in his relevant process of learning providing the interactions he has with smarter students and vice versa.

The results of simulating this theory are presented in table 6. The experimental results show that the knowledge of the slow student has a significant progress in the proposed model compared to the model of the students only interacted with his teacher and only learnt from him. Moreover, the number of iterations required to reach to convergence was decreased. For example, in the four-action $E_{4,A}$ environment, the number of iterations needed for the slow-learning student LA to converge decreased to 1110 from 1382. This indicates the effective relationship of the slow student with his other classmates because here three slow students communicated with six other students who had superior knowledge and learn faster than the three slow ones. Also, the learning process of the fast student has slowed down. This deterioration in the learning process of the fast student is due to the fact that there are 3 fast students and 6 normal and slow students in this experiment. Therefore, when the fast student seeks help for improvement, he may find eight other students two of whom are in the same knowledge as himself and six others knowledge are lower than him including three normal students and three slow learner students. In other words, there are no genius help for the fast student in this group, so it is clear that his learning process slows down in this case. For normal students, the interaction with their other fast and slow learner students can be beneficial. For example, in the four and ten-action $E_{4,A}$ and $E_{10,A}$ environments, the number of iterations needed for convergence has decreased from 996 to 696 and from 1297 to 1059. On the other hand, in the four and ten-action $E_{4,B}$ and $E_{10,B}$ environments, the number of iterations needed for convergence has decreased from 2201 to 1286 and from 2114 to 1419. The results of this simulation suggest that the difference between iterations as well as the rate of improvement for slow students is more than normal students. This is due to the fact that there are two superior groups of students for slow students whose knowledge are higher than slow students (to get help from in order to improve in their learning process). However, there are one superior group and one lower group of students for the normal students. Therefore, while this interaction may improve learning status of the normal student, oscillations between these two groups will slow down at some points. Furthermore, the results shown in the table 5 and 3 indicate that the convergence time in $E_{4,B}$ and $E_{10,B}$ environments is greater than of $E_{4,A}$ and $E_{10,A}$ environments. This reflects the fact that the set of $E_B$ environments was more difficult because of the proximity of the underlying penalty/reward probabilities.

Also, the results showed that the ten-action environments were more difficult than the four-action environments. The iterations required for the LA convergence increased from the four-action environments to the ten-action Environments. The rates of learning for a slow, normal and fast student are shown in Figure 6.

**Table.6** Convergence of proposed model and comparison with presented model in [15].

| | | **Fast student** (λ: 0.004 - 0.0127) | | **Normal student** (λ: 0.0182 - 0.0192) | | **Slow student** (λ: 0.0142 - 0.0152) | |
|---|---|---|---|---|---|---|---|
| | | **# of Iteration to converge** | | | | | |
| **Env.** | **# of Actions** | **Old model [15]** | **proposed model** | **Old model [15]** | **proposed model** | **Old model [15]** | **proposed model** |
| **$E_A$** | 4 | 572 | 662 | 996 | 696 | 1382 | 1110 |
| **$E_B$** | 4 | 1482 | 1564 | 2201 | 1286 | 2633 | 1442 |
| **$E_A$** | 10 | 686 | 704 | 1297 | 1059 | 1804 | 1424 |
| **$E_B$** | 10 | 1655 | 1642 | 2114 | 1419 | 2859 | 1479 |

Reward probabilities for 4-action environment are :
$E_{A,4}$: 0.7 0.5 0.3 0.2 $E_{B,4}$: 0.1 0.45 0.84 0.76
Reward probabilities for 10- action environment are:
$E_{A,10}$: 0.7 0.5 0.3 0.2 0.4 0.5 0.4 0.3 0.5 0.2 $E_{B,10}$: 0.1 0.45 0.84 0.76 0.2 0.4 0.6 0.7 0.5 0.3

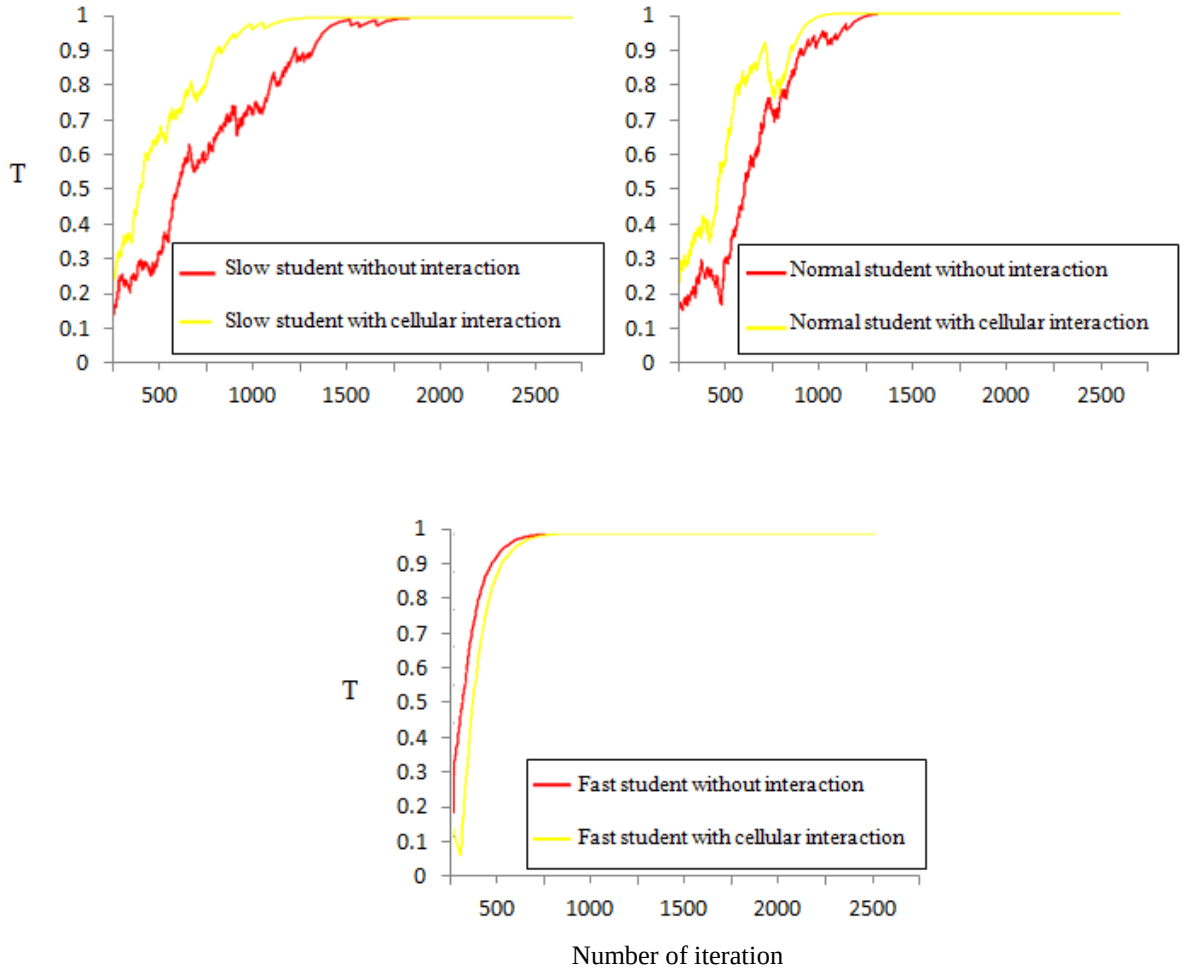


**Fig. 6** Rate of Learning for Students. (x-axis: Iteration , y-axis: Threshold)

## 1. CONCLUSION

Considering the studies conducted on electronic tutorial systems and the strategies presented for improving their quality, we can conclude that interactions play a crucial role in these systems. In addition to being influenced by the teacher, learners in a tutorial environment learn by interacting with other learners. Moreover, studying educational trends in people's real life shows that the way learners are treated plays an important role in their academic progress. In this article, a new approach was presented for modeling tutorial-like systems and improving the student modeling method. We managed to design a teacher model with the ability to learn while teaching how to approach the learners and guide them through the learning process towards faster learning. Also, through testing the learners during their learning period, the method helped them to identify and correct their mistakes. In addition, the students' interactions through cellular learning automata were simulated. As is shown in the results, the proposed model is a suitable mechanism for executing the learning process, thus providing maximal benefits for learners in general. As for our future work, we will try to address the experimental problems listed in previous section.

## REFERENCES

[1] V.J.Shute ; J.Psotka., "Intelligent Tutoring Systems: Past, Present, and Future", Handbook of Research on Educational Communications and Technology, Scholastic Publications1995.

[2] E.Wenger, "Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge" ,Los Altos, CA: Morgan Kaufmann Publishers, Inc1987.

[3] E.Fischetti, A.Gisolfi., "From Computer-Aided Instruction to Intelligent Tutoring Systems", Educ. Technol, 1990, vol, 30, no. 8, pp. 7–17.

[4] R.Winkels , J.Breuker., " What's in an ITS? a functional Decomposition, in New Directions for Intelligent Tutoring Systems", E. Costa, Ed.Berlin, Springer-Verlag, Germany, 1990.

[5] M. K. Hashem; "*Learning automata based intelligent tutorial-like systems*", Ph.D. dissertation, School Comput. Sci., Carleton Univ., Ottawa, ON, Canada, 2007.

[6] J.Self, "The Defining Characteristics of Intelligent Tutoring Systems Research: ITSs Care, Precisel", Int. J. Artif. Intell. Educ,1990, vol. 10, pp. 350–364.

[7] T. A. Atolage , V. Hlupic, "A multimedia intelligent tutoring system for simulation modeling. In Andraddttir" S, Healy KJ, Withers DH, Nelson BL, editors, proceeding of the 29th conference on winter simulation, Atlanta, Georgina. 1997, 504-509.

[8] R. S. Sutton, A. G. Batro, " Reinforcement learning: an intruduction". MIT Press, Cambridge, MA, 1998.

[9] J. E. beck, B. P. woolf, C. R. beal, "A Machine Learning Architecture for Intelligent Tutor Construction", AAAI-00 Proceedings. Copyright © 2000,

[10] R. Sisin, M. Shirnura, "Student modeling and machine learning". International journal of AI in education, 9(1-2), 1998.

[11] C. Frasson, T. Mengelle, E. Aimeur, G. Gouarderes G. "An actor-based architecture for intelligent tutoring system: in intelligent tutoring system". In internatinal conference, ITS, 96: lecture note in computer science, Springer-verlag, 1996, Berlin,57-65.

[12] R.S.Legaspi, R.C.Sison, "Modeling the tutor using reinforcement learning". in Proc. PCSC , 2000, pp. 194–196.

[13] P.Baffes, R.Mooney., "Refinement-based student modeling and automated bug library construction". J.Artif. Intell. educ, 1996, vol. 7, no. 1, pp. 75–116.

[14] R.Lelouche., "A Collection of pedagogical agents for intelligent educational systems in intelligent tutoring systems". 6th international Conference, ITS, Lecture Notes in computer Science, Springer-verlag, Berlin, 2000, pp. 143-152.

[15] Hashem. M. K , B. J. Oommen. "Modeling a Student-Classroom Interaction in a Tutorial-Like System Using Learning Automata," IEEE Trans.Syst., Man, Cybern. B, Cybern. Feb. 2010, vol.40, no.1.

[16] M.K.Hashem, "Modeling a Student's Behavior in a Tutorial-*Like* System Using Learning Automata", Ph.D. dissertation, School Comput. Sci., Carleton Univ., Ottawa, ON, Canada, 2010.

[17] Hashem. M. K , B. J. Oommen. "Modeling the Learning process of the teacher in a Tutorial-Like System Using Learning Automata," IEEE Trans.Syst., Man, Cybern. B, Cybern, Dec. 2013, vol.43, no.6, pp. 2010-2031.

[18] Y. Wang, W. Jiang. "Learning automata based cooperative student-team in tutorial-like system". 10th International conference, ICIC, Taiyuan,china, 2014, 154- 161.

[19] Ge. Hao, Y. Wang, Y. Gua, C.L.P. Chen. S. Li. "Acooperative framework of learning automata and its application in tutorial-like system". ScienceDirect, Elsevier B.V, 2015.

[20] M.L. Testlin. "On the behavior of finite automata in randim media, automation and remote control". 22:1210-1962.

[21] K.S.Narendra., M.A.L.Thathachar, " Learning Automata: An Introduction". Englewood Cliffs, NJ: Prentice-Hall, 1989.

[22] S.Lakshmivarahan, "Learning Algorithms: Theory and Applications". Springer-Verlag: Berlin, Germany, 1981.

[23] K.Najim, A.S.Poznyak, "Learning Automata: Theory and Applications". Oxford, U.K 1994: Pergamon.

[24] B.J.Oommen., M.Agache, "Continuous and Discretized Pursuit Learning Schemes: Various Algorithms and Their Comparison", IEEE Trans. Syst., Man. Cyrern. Jun 2001 B, vol. 31, no. 3, pp. 277-278.

[25] M.A.L.Thathachar, P.S.Sastry, "Varieties of Learning Automata: An Overview", IEEE Trans. Syst., Man, Cybern. B, Cybern, vol. 32, no. 6, pp. 711–722, 2007.

[26] J.V. Neumann, "The theory of self-reproducing automata" Burks AW (ed), Univ. of Illinois Press, Urbana and London 1996.

[27] M.R.Meybodi., H.Beigy, " A Mathematical Framework for Cellular Learning Automata" Advance on Complex Syst. Sep./Dec 2004 Nos. 3-4.

[28] M.R.Meybodi., H.Beigy, M.Taherkhani, "Cellular Learning Automata and its Applications", Sharif Journal of Science and Technology, 2003,vol. 19, no.25, pp.54-77.

[29] J.L. Schiff, "Cellular Learning Automata: A Discrete View of the World, A John Wily & Sons, 2007,Inc., Publication1807.

**Hajar Hajary** received her B.Sc. in Software Computer Engineering from Mobarakeh Azad University, Esfahan, Iran and M.Sc in Software Computer Engineering from Science and Research Branch Islamic Azad university, Tehran, Iran. Her research interests include cloud computing, intelligent systems. Her research interests include distributed intelligent system, machine learning and multi agent system.

**Ali Ahmadi** received his B.Sc. in Electrical Engineering from Amirkabir University, Tehran, Iran in 1991 and M.Sc. and Ph.D. in Artificial Intelligence and Soft Computing from Osaka Prefecture University, Japan in 2001 and 2004, respectively. He worked as a researcher in Research Center for Nanodevices and Systems in Hiroshima University, Japan during 2004–2007. He has been with K.N. Toosi University of Technology, Tehran, Iran as assistant professor from 2007. His research interests include distributed intelligent systems, human-computer interaction, computational intelligence, adaptation and interactive learning models, virtual reality and artificial life.

**Maryam Khani** Received her B.Sc. in Software Engineering from Mobarakeh Azad University of Esfahan, Iran and her M.Sc. in Mechatronic Engineering from Islamic Azad university, South Tehran Branch, Tehran, Iran. Her research interests include intelligent systems, human-agent interaction, multi-agent system, learning machine and trust in social Internet of things as well.