Intelligent Exploration Method to Adapt Exploration Rate in XCS, Based on Adaptive Fuzzy Genetic Algorithm

Ali Hamzeh, Adel Rahmani, Nahid Parsa
Computer Engineering Department
Iran University of Science and Technology
Narmak, Tehran, Iran
{hamzeh, rahmani}@iust.ac.ir, nahid.parsa@dtasoft.com

Abstract—**In this paper, we propose an extension to the Intelligent Exploration Method which is introduced in our previous work. IEM is an intelligent exploration method that is used to tune the exploration rate in XCS. In this paper we improve the IEM's performance using a learning fuzzy controller instead of the static one in IEM. The new system is called IEMII (IEM 2) and is compared with the IEM and the traditional XCS in some benchmark problems.**

Keywords—**Learning Classifier Systems, XCS, Exploration/ Exploitation Dilemma**

I. INTRODUCTION

Learning Classifier System is a Machine Learning technology that was introduced by John Holland in the paper "Cognitive Systems based on Adaptive Algorithms" [1] for the first time. In this system, an agent learns its environment by applying some actions and obtaining the relevant rewards or punishments as a guideline for its internal environmental model that is designed as a rule based system. One of the major weaknesses in LCS is its credit assignment component and the fitness calculation method. The most important extension to LCS, that is proposed to overcome these problems, is developed by S.W. Wilson: "Accuracy based Classifier System" (XCS) [2]. Intelligent Exploration Method (IEM) [3] is an intelligent exploration method that is used to tune the exploration rate in XCS. This system is based on a fuzzy controller that is used to determine the exploration rate using some predefined input variables. As mentioned in [3], the main weakness of IEM is the lack of learning ability. It means that IEM uses some predefined rules, which are derived experimentally, and could not adapt its rule base with the environmental state and the problem criteria. In this research, we propose a new extension to IEM that is able to learn its rule base using an evolvable fuzzy controller which is introduced in [4]. This new extension is called IEMII.

The rest of this paper is organized as follows: at first XCS is described in brief, then the Exploration/Exploitation dilemma is explained and then some other relevant works on controlling the exploration rate are described. After that, we describe the IEMII's internal structure and its inputs and output parameters. Then we describe our intelligent fuzzy controller, and then the designed experiments and benchmark problems are described. At last, the experimental results of IEMII, IEM and the traditional XCS are described and discussed.

II. XCS IN BRIEF

In this section, we briefly describe XCS. This description is mainly drawn form [5]. XCS is designed for both single and multiple-step tasks. In these environments, an input is presented, the system makes decision, and the environment provides some reward.

Structurally, each classifier *Cj* in XCS's population [P] has a condition, an action, and a set of associated parameters. The condition is a string from {0, 1, #}; the action is an integer. The three principal parameters are: (1) payoff prediction *Pj*, which estimates the payoff of the system will receive if *Cj* matches and its action is chosen by the system; (2) prediction error εj, which estimates the error in *Pj* with respect to actual payoff received; and (3) fitness *Fj*, computed as later explained. It is convenient to divide the description of a single operating cycle or time-step into the traditional performance, update (reinforcement), and discovery components. In the performance component, the system selects the winner classifier to apply its action to the environment. In the update phase, the system receives the environmental reward and divides it between involved classifiers and updates their relevant parameters. Also, in the discovery phase, the system creates a new population of classifiers using GA with respect to the updated parameters in the previous phase. This cycle continues till the system reaches the desired performance.

III. EXPLORATION/EXPLOITATION DILEMMA

The decision to learn is fundamentally a choice between acting based on the best information currently possessed versus acting other than according to what is apparently best. The rational behind this latter approach is to gain new information that may permit higher levels of performance later. Learning risks a short-term cost—the "opportunity cost" of not doing the apparent best—in order to achieve higher returns in the longer run. Not learning risks those potentially higher returns in order to get known benefits now. The tension between learning and performance is often described as the "Explore/Exploit Dilemma" (EED). Holland was one of the first to discuss the dilemma in connection with adaptive systems [6]. He summarizes: "[obtaining] more information means a

1-4244-0023-6/06/$20.00 ©2006 IEEE

performance loss, while exploitation of the observed best runs the risk of error perpetuated".

The Explore/Exploit dilemma will be examined within the basic framework of reinforcement learning [7]. On each discrete time-step, we assume that the system receives a stimulus vector x from the environment, carries out an action a, and receives from the environment a reward r. The system's objective is two-fold. On one hand, it must learn which actions maximize some measure of the reward received over time, such as the sum or a discounted sum of the rewards. On the other hand, it must act to accomplish the maximization.

IV. RELATED WORKS ON EED

In this section, we summarize some previous works that are related to EED issue to show the state of the EED research.

If a system has a well-defined way of making the Explore/Exploit decision at each time-step, we shall say it has an E/E strategy. In [8], Wilson has described some famous Exploration/Exploitation strategies. Those strategies are categorized into *Global Strategies* and *Local Strategies*. *Global Strategies* are those where the argument of probability function that is used to choose between explore and exploit trial is a quantity independent of the system, such as time, or a statistic of the system's overall behavior, such as a moving average of its rate of reward intake. In this case, the strategies do not change with the system's experience and are in that sense non-adaptive or fixed. Below strategies are laid into this category: static probability, variable probability with respect to time, variable probability with respect to averaged reward, variable probability with respect to prediction error and so on. In *Local Strategies*, the degree of exploration is based on a function of quantities associated with the system's response to the current input. The global adaptive strategies sense a condition that is a property of the whole system-environment interaction, then set the system's general explore probability accordingly. In the local (adaptive) strategies, the system senses a condition that is a property of a niche in its interaction with the environment, and chooses its action accordingly. The idea is that learning may be needed in certain input situations but not in others, or, more generally, that statistics associated with a given niche should determine the degree of exploration that takes place there. We can find many strategies such as roulette wheel selection with respect to predicted reward or predicted error in this category.

In [9], the authors propose an adaptive approach based on meta-rules to adapt the choice between exploration and exploitation. That adaptive approach relies on the variations of the performance of the agents. To validate the approach, they apply it to economic systems and compare it to two adaptive methods: one local and one global. They have adopted these methods to economic systems. Finally, they compare different exploration strategies and focus on their influence on the performance of the agents.

In [10], the authors present a model that integrates both exploration and exploitation in a common framework. They define the concept of degree of exploration from a state as the entropy of the probability distribution on the set of admissible actions in that state. This entropy value allows to control the degree of exploration linked to that state, and should be provided by the user. Then, they restate the exploration/exploitation problem as a global optimization problem: define the best exploration strategy that minimizes the expected cumulated cost, while maintaining fixed degrees of exploration. This formulation leads to a set of nonlinear updating rules reminiscent from the "value iteration" algorithm. Interestingly enough, when the degree of exploration is zero for all states (no exploration), these equations reduce to Bellman's equations for finding the shortest path while, when it is maximum, a full "blind" exploration is performed. The authors further show that if the graph of states is directed and acyclic, the nonlinear equations can easily be solved by performing a single backward pass from the destination state. The theoretical results are confirmed by simple simulations showing that the model behaves as expected.

V. EED IN XCS

The EED dilemma also exists in the action selection procedure of XCS. Due to the online performance measurement in XCS, one of the most challenging issues is to create a balance between selecting the winner action with respect to the agent's previous experiments or let the agent to explore its environment to probably find some better rules for further actions. It seems that XCS must create a balance between these two strategies. Hence, the major problem is to create this balance and the minor but important one is the strategy to select the winner action in exploration phases. Should this strategy be pure random selection or must utilize the gathered knowledge of the agent's experiments? In the current implementation of XCS, based on the "Algorithmic Description of XCS" [11] by M. Butz and S.W. Wilson, the balance between Exploration and Exploitation is created using P_{exp} constant. This constant is used to determine the probability of exploring the environment. This probability would be set constant during the agent's life cycle and commonly is equal to 0.5. Therefore, this balance is created using only a constant parameter. Also considering the second question, [11] uses pure random policy to select the winner action in exploration phases and do not use any other parameter such as fitness or strength in the selection procedure. In the next section, we propose an adaptive intelligent technique to create a balance between Exploration and Exploitation.

VI. INTELLIGENT EXPLORATION METHOD II (IEMII)

Our proposed method is based on this hypothesis that the constant rate of exploration could not be optimal in the agent's life cycle. It seems that some adaptive changes in the exploration probability can improve the performance of the XCS [3]. To do so, we designed a system called IEMII. It tries to propose a suitable exploration rate according to its information about the agent's performance and the environmental changes. IEMII tries to distinguish the beginning, middle and the final phases of XCS's lifecycle to

propose an appropriate exploration probability for each phase. Figure 1 shows the overall architecture of XCS with IEMII.

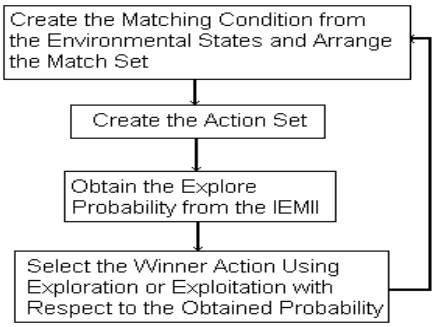

Figure 1. The Overall Architecture of the XCS with IEMII

A. The Internal Architecture of IEMII

In this section, we describe the internal architecture of the IEMII. As shown in Figure 1, the IEMII receives the environmental states via its interface and chooses suitable control parameters with respect to its rule base. The internal architecture of IEMII is divided into four different parts, namely input interface, rule base, inference engine and output interface. These parts are described in the following sections.

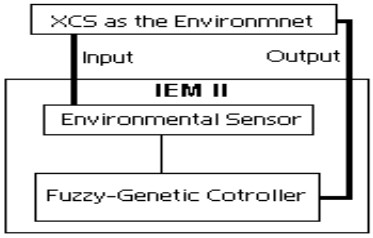

Figure 2. The Overall Architecture of The IEMII

B. The Input Parameters of the IEMII

These input parameters are indicating the current state of XCS with respect to its internal evolutionary process and its online performance. Our chosen input parameters are as follows:

- *PERF*: this factor is calculated using equation 1, where N_{ec} is the number of the exploitation trials with the correct actions from the beginning and N_e is the total number of the exploitation trials from the beginning.

$$PERF = N_{ec} / N_e \quad (1)$$

- N_{exr}: this factor is calculated according to equation 2, where N_r is the total number of the explore trials from the beginning and N_t is the total number of the trials from the beginning.

$$N_{exr} = N_r / N_t \quad (2)$$

- N_{exp}: this factor is calculated using equation 3, where N_t and N_e are as the above.

$$N_{\exp} = N_e / N_t \quad (3)$$

- Age: This parameter is used to distinguish the beginning, middle and final phases of the XCS's lifecycle. *Age* is calculated using formula 4, where N_t is as the above and T is the expected number of the trials that XCS is going to accomplish.

$$Age = N_t / T \quad (4)$$

- F_{best}: The fitness of the best individual in the XCS's population
- F_{mean}: The Mean fitness of the individuals in the XCS's population.
- F_v: The variance of the individuals' fitness in the XCS's population.
- D_{mean}: This parameter and the other remaining one are calculated using the *Hamming Distance* concept. To calculate these parameters, at first we select the best chromosome of the XCS's population with respect to its fitness and then calculate the *Hamming Distance* between each individual of the population and the selected one. Then, we use these values to calculate these two parameters. It is important to note that, these parameters are used to indicate the genotypic distribution of the population.
- D_v[1]: The variance of the calculated *hamming distance* of the individuals in the XCS's population.

The described parameters are chosen due to the following reasons:

- The first parameter determines the overall picture of the XCS's success.
- The second and third parameters show the state of EED balance in the system.
- The forth parameter indicates XCS's age.
- The fifth to ninth parameters are selected to determine the state of the XCS's internal evolutionary process based on the proposed parameters in [4].

[1] All of these values are normalized.

C. The Output Parameters of the IEMII

IEMII has only one output parameter named P_{exp}. It is the *Exploration Probability* in the XCS. It means that XCS chooses its action randomly with the probability of P_{exp} in all epochs.

D. Rule Base of IEMII, Evolutionary Learning System (ELS)

ELS [4] is a XCS-based fuzzy controller. It was designed to extract and tune a fuzzy rule-base with the ability of controlling some predefined output parameters using a set of input parameters.

To use ELS, we must define a set of input parameters with their associated membership functions and the desired output parameters with their corresponding membership functions. Then, ELS must be tuned with respect to a proper learning criterion.

In the current research, ELS is employed as the inference engine component in the IEMII. It must calculate the proper *exploration probability* for XCS with respect to the previously described input parameters. The goal of ELS is to tune a rule-base which is able to improve the reward gathering rate of XCS. Hence, the learning criterion of ELS in this research is defined as follows.

$$R = d_r / d_T \qquad (5)$$

where R is the environmental reward for ELS, r is the obtained reward by XCS and T is the number of elapsed trials from the beginning by XCS.

It is important to note that in this research ELS is tuned separately before the main experiments begin and it continues to learn during the experimental phase. Also, some of the membership functions for input and output parameters in ELS are depicted in Figure 3.

VII. Design of the Experiments

Our used benchmark problems are chosen from some well-known benchmark families of both single and multi step categories; the *multiplexer* family and the *woods2* family. These problems are described here in brief. The interested reader can refer to [2].

Multiplexer Family: Boolean multiplexer functions are defined for binary strings of length $l = k + 2^k$. The function's value may be determined by treating the first k bits as an address that indexes into the remaining 2^k bits, and returning the indexed bit. For example, in the 6-multiplexer (l=6), the value for the input string 100010 is 1, since the "address", 10, indexes bit 2 of the remaining four bits. In disjunctive normal form, the 6-multiplexer is fairly complicated (the primes indicate negation):

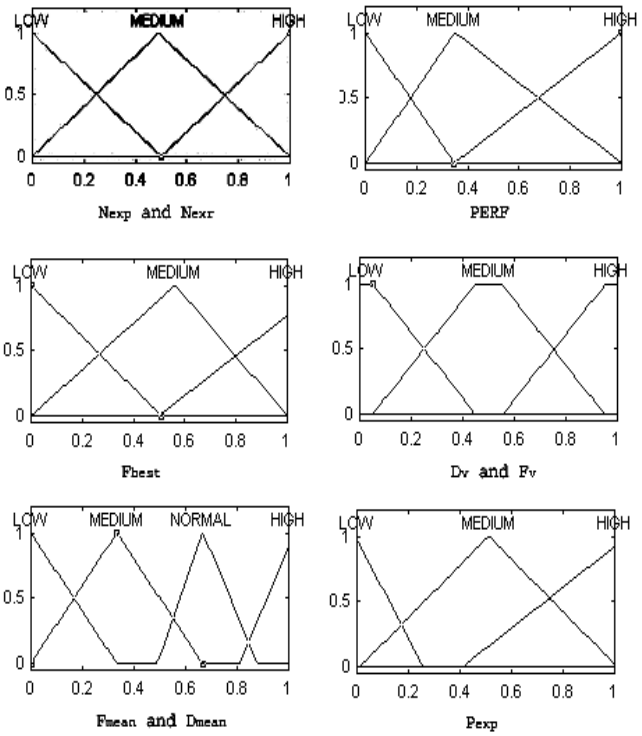

Figure 3. Membership functions of input and output variables of the ELS in IEMII

$$F_6 = x_0'x_1'x_2 + x_0'x_1x_3 + x_0x_1'x_4 + x_0x_1x_5. \qquad (6)$$

To construct our payoff landscape, we associated two payoff values, 300 and 0. Payoff 300 was for the right answer and payoff 0 was for the wrong answer. There are more complicated instances of this problem such as MP11 (*l=11)* or MP20 (*l=20*). In this paper, we use MP11 and MP20 as our benchmark problems.

Woods2 Family: Woods2 has two kinds of "food" and two kinds of "rocks". F and G are the two kinds of food, with sensor codes 110 and 111, respectively. O and Q are the two kinds of rocks, with sensor codes 010 and 011, respectively. Blanks, denoted by ".", have sensor code 000. The system is capable of detecting the sensor codes of objects occupying the eight nearest cells (sensing 000 if the cell is a blank). The system's available actions consist of the eight one-step moves into adjacent cells, with the move directions similarly coded from 0 for north clockwise to 7 for north-west. Our used instance of *Woods2* environment is shown in Figure 4. the goal is to reach the food from a random initial state.

```
.............................
.QQF..QQF..OQF..QQG..OQG..OQF.
.OOO..QOO..OQO..OOQ..QQO..QQQ.
.OOQ..OQQ..OQQ..QQO..OOO..QQO.
.............................
.............................
.QOF..QOG..QOF..OOF..OOG..QOG.
.QQO..QOO..OOO..OQO..QQO..QOO.
.QQQ..OOO..OQO..QQQ..QQQ..OQO.
.............................
.............................
.QOG..QOF..OOG..OQF..OOG..OOF.
.OOQ..OQQ..QQO..OQQ..QQO..OQO.
.QQO..OOO..OQO..OOQ..OQQ..QQQ.
.............................
```

Figure 4. The Used instance of Woods2 Environment

Our experiments are done separately by the XCS with IEM (we call it XCSI), XCS with IEMII (which is called XCSI2) and the original XCS based on [2]. For each problem, the experiments are done 100 times separately and the results are averaged over these runs. These results are shown in Figures 5 to 7. All of the XCS parameters are set as [2]; the population size is set 400 for MP11 and Woods2 and 1000 for MP20.

A. The Experimental Results

In this section the experimental results of applying XCS, XCSI and XCSI2 is shown in Figures 5, 6 and 7 for MP11, MP20 and Woods2 problems. It is notable that XCSI2's learning procedure is continuous. It means that the rules base of IEMII is not reset in each independent run and is passed to the next run. To initiate XCSI2's rule base, it is executed for 15 independent runs before starting of each experiment. Note that the horizontal axis is the number of the iterations and the vertical axis is the percent of the correct answers in last 10 exploit epochs for MP11 and MP20 and the mean steps to food in last 5 exploit epochs for Woods2.

B. Discussion

With respect to the experimental results, it is obvious that XCSI2 works better than traditional XCS and even better than XCSI. To describe this behavior, consider Figures 8a and 8b. These Figures show the exploration rate of XCS, XCSI and XCSI2 in MP20 and Woods2. As shown in these Figures, the main difference between XCSI and XCSI2 is the proposed exploration rate in the first and last steps. In XCSI, our proposed rule emphasized that the exploration rate in the first steps must be low. However, XCSI2's tuned rules propose different behavior. For example, consider below rules from IEMII's rule base:

For MP20 Problem: *"If age is not high then exploration rate must be medium".*

For Woods2 Problem: *"If age is low then exploration rate must be high".*

These rules indicate that our previous assumption in *IEM* is not correct in all situations. These rules cause exploration rate to be high at first steps and due to *XCSI2*'s better performance we can say that these rules are better than previous ones [3]. To quantify these results and to test whether it is statistically significant we apply these experiments 50 times with different random generators independently and then apply one-tailed Wilcoxon signed rank test [12] on resulted values of XCS with IEM, IEMII and the traditional one. The test results are represented in Table 1. Note that in Table 1, we compare the Optimal Reach Point (ORP) of these three algorithms. The ORP is the first point that the mean performance of the algorithm in the previous 50 points for MP11 and MP20 and previous 20 points for Woods2 is equal to the $OptimalValue \pm e_0$ where e_0 is set to .01 for MP11 and MP20 and .1 for Woods2. These resulted values are also depicted in Table 2. With respect to the definition of the P-Values in the Wilcoxon Test, we can confirm that that XCSI2 can improve the performance of XCS and XCSI at a good rate as shown in the figures 5 to 7.

TABLE I. THE P-VALUES OF THE WILCOXON TEST ON THE ORP VALUES

	MP11	**MP20**	**Woods2**
XCSI2 vs. XCS	0.024	0.012	0.002
XCSI2 vs. XCSI	0.123	0.047	0.021

TABLE II. MEAN ORP FOR XCS, XCSI AND XCSI2 (AVERAGED OVER 50 RUNS)

ORP for	**MP11**	**MP20**	**Woods2**
XCS	272	2234	151
XCSI	231	2047	72
XCSI2	197	1925	23

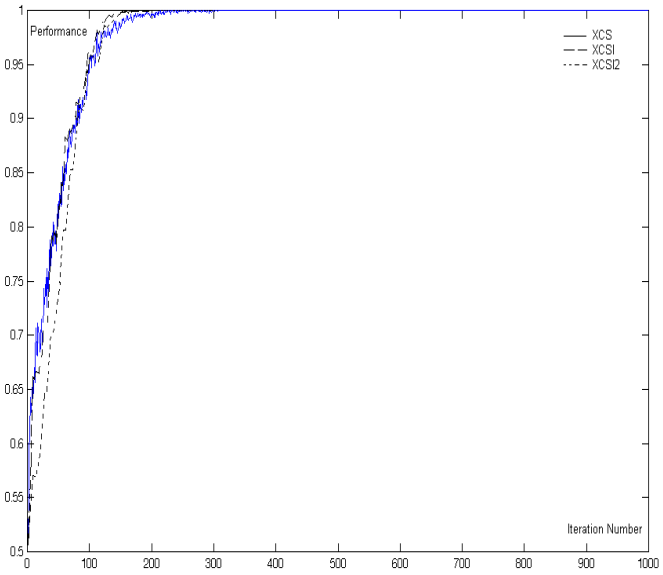

Figure 5. XCSI2 (Dotted Line) , XCSI (Dashed Line) and XCS (Solid line) in MP11 Problem

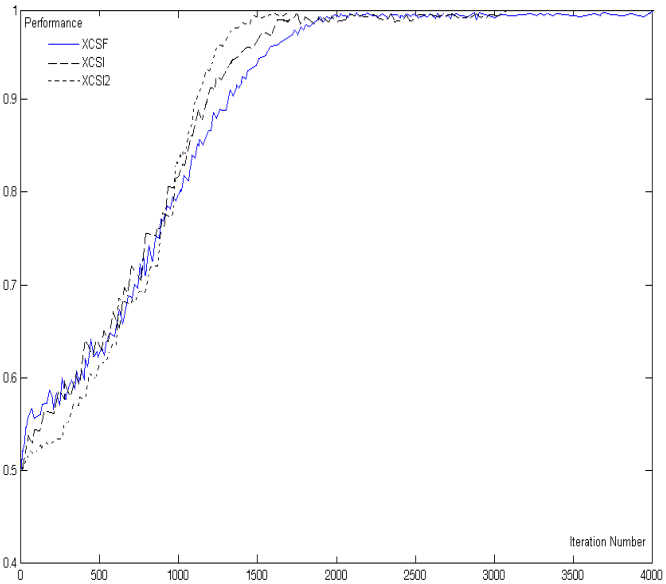

Figure 6. XCSI2 (Dotted Line) , XCSI (Dashed Line) and XCS (Solid line) in MP20 Problem

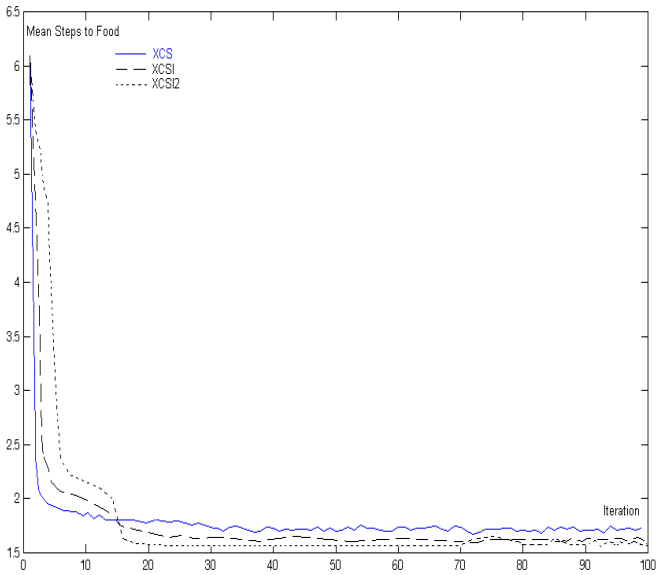

Figure 7. XCSI2 (Dotted Line) , XCSI (Dashed Line) and XCS (Solid line) in Woods2 Problem

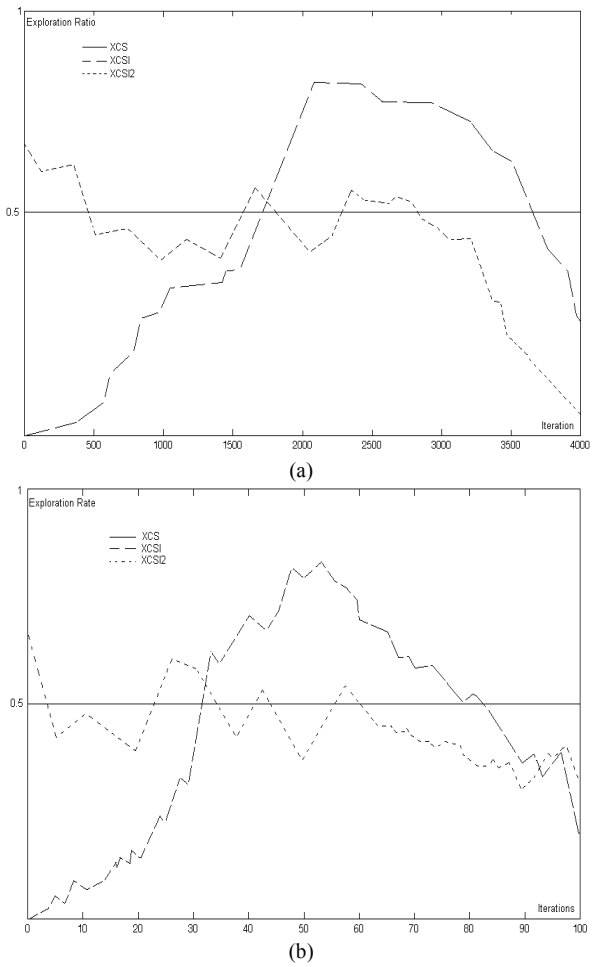

Figure 8. Exploration Rate for XCSI2, XCSI and XCS in MP20 (a) and Woods2(b)-XCS solid Lined, XCSI, Dashed Line, XCSI2, Dotted Line

VIII. SUMMARY

In this paper, we propose a new intelligent method to balance *EED* relation in XCS. This method is based on IEM that is introduced in [3] for the first time. These methods are based on a fuzzy controller that is predefined in IEM and is able to learn in IEMII. In this paper, we described IEMII completely, applied it on some benchmark problems, and compare it with IEM. The experimental results show that IEMII work better than IEM in our benchmark problems.

At last, we can conclude that this intelligent mechanism of extracting rules can tune our previously proposed rules in [3] and also can detect some new relations between inputs and output which cause *XCSI2*'s performance to improve comparing to *XCSI*'s.

REFRENCES

[1] J. H. Holland, J. S. Reitinan, (1998) .*Cognitive Systems Based on Adaptive Algorithms*. In D. A. Waterman and F. Hayes-Roth, editors, Pattern-directed inference systems. New York: Academic Pr 1978. Reprinted in: Evolutionary Computation. The Fossil Record. David B. Fogel (Ed.) IEEE Press, 1998. ISBN: 0-7803-3481-7

[2] S. W. Wilson, (1995). *Classifier Fitness Based on Accuracy*. Evolutionary Computation 3(2):149-175.

[3] A. Hamzeh, A. Rahmani, (2005), *Intelligent Exploration Method for XCS.* In Proceeding of 11'Th International Workshop of Classifier Systems (IWLCS2005), In Proceeding of Genetic and Evolutionary Computation Conference 2005 (GECCO2005).

[4] N. Parsa, (2005). *A Fuzzy Learning System to Adapt Genetic Algorithms*, MSc. Thesis, Iran University of Science and Technology, Computer Engineering Department.

[5] S.W. Wilson, (2002). *Classifiers that approximate functions*. Journal of Natural Commutating 1 (2-3).

[6] J.H. Holland, (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press. Republished by the MIT press, 1992.

[7] R.S. Sutton, (1991). *Reinforcement learning architectures for animats*. In J.-A. Meyer & S. W. Wilson (eds.), From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior (pp. 288-296). Cambridge, MA: The MIT Press/Bradford Books.

[8] S.W. Wilson, (1996). *Explore/Exploit strategies in autonomy*. In From animals to animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior (pp. 325-332), Cambridge MA: The MIT Press/Bradford Books.

[9] L. Rejeb and Z. Guessoum, (2005), *The Exploration-Exploitation Dilemma for Adaptive Agents*, Fifth European Workshop on Adaptive Agents and Multi-Agent Systems (AAMAS'05), to appear in Springer Lecture Note Series.

[10] Y. Chbany, F. Fouss, L. Yen, A. Pirotte and M. Saerens, (2005), *Managing the trade off between exploration and exploitation in reinforcement learning*. Technical report, Information System Unit, Universite cathaolique de louvain, Belgium

[11] M. V. Butz, S. W. Wilson. (2002). *an Algorithmic Description of XCS.* Journal of Soft Computing, 6(3-4):144-153, 2002.

[12] Kanji, G. 1994. *100 Statistical Tests*, SAGE Publications.