Reinforcement Learning by an Accuracy-Based Fuzzy Classifier System with Real-valued Output

Jorge Casillas
Dept. Computer Science & Artificial Intelligence
University of Granada
Granada 18071, SPAIN
Email: casillas@decsai.ugr.es

Brian Carse and Larry Bull
Fact. Computing, Engineering & Mathematical Sciences
University of the West of England
Bristol BS16 1QY, UK
Emails: Brian.Carse, Larry.Bull@uwe.ac.uk

***Abstract*— The issue of finding fuzzy models with an interpretability as good as possible without decreasing the accuracy is one of the main research topics on genetic fuzzy systems. When they are used to perform on-line reinforcement learning by means of Michigan-style fuzzy classifier systems, this issue becomes even more difficult.**

Indeed, rule generalization (description of state-action relationships with rules as compact as possible) has received a great deal of attention in the discrete-valued learning classifier system field (e.g., XCS is the subject of extensive ongoing research). However, the same issue does not appear to have received a similar level of attention in the case of Michigan-style fuzzy classifier system. This may be due to the difficulty in extending the discrete-valued system operation to the continuous case. The intention of this contribution is to propose an approach to properly develop a fuzzy XCS system for immediate-reward problems.

I. Introduction

The Michigan-style fuzzy classifier system is a machine learning system which employs linguistic rules and fuzzy sets in its representation and an evolutionary algorithm (EA) for rule discovery. It therefore combines an easily understood representation (as opposed to, for example, neural network approaches) with a general purpose search method. These systems are very useful to perform on-line learning, i.e., to automatically learn fuzzy rules at the same time that data or stimulus (or reward or payoff) from the environment are received. This fact makes Michigan-style fuzzy classifier systems ideal for adaptive systems, control, simulation of animal behavior and, in addition, data mining and knowledge discovery applications.

In order to exploit the fuzzy represention to the full, i.e. to achieve high interpretability, the ability to learn generalization is of great importance. With *generalization* we understand capability to express the state-action (antecedent-consequent) relationships as compact as possible. Generalized rules allow more compact rule bases, scalability to higher dimensional spaces, faster inference, and better linguistic interpretability. The issue of rule generalization, and the interplay between general and specific rules in the same evolving population, has received a great deal attention in the learning classifier systems (i.e., discrete-valued ones) research community (e.g. [1]). The same issue does not appear to have received a similar level of attention in the case of Michigan-style fuzzy classifier systems [2]–[8].

Traditional (non-fuzzy) classifier systems have been "strength-based" in the sense that a classifier accrues strength during interaction with the environment (through rewards and/or penalties). This strength can then be used for two purposes: resolving conflicts between simultaneously matched classifiers during learning episodes; and as the basis of fitness for the EA. A completely different approach can be taken in which a classifier's fitness, from the point of view of the EA, is based on its "accuracy," i.e. how well a classifier predicts payoff whenever it fires. Note that the concept of accuracy used here is different from the traditionally used in fuzzy modeling (i.e., capability of the fuzzy model to faithfully represent the modeled system). This accuracy-based approach offers a number of advantages such as avoiding over-general classifiers, obtaining optimally general classifiers, and learning of a complete "covering map." This accuracy-based classifier system, called XCS, was proposed in [1] and it is currently of major interest to the research community in this field.

This work aims at proposing a new approach to achieve accuracy-based Michigan-style fuzzy classifier systems. The proposal, Fuzzy-XCS, is based on XCS but properly adapted to fuzzy systems. The fuzzy inference and the reinforcement component are changed to consider a different, competitive interaction among rules that allow an accuracy-based operation. An accuracy-based fuzzy classifier system is proposed in [9]. However, it presents a number of important limitations (mainly the lack of the generalization capability and the use of integer-valued output) that are solved by our method.

The paper is organized as follows. Section II introduces some difficulties in developing an accuracy-based fuzzy classifier system. Section III describes the competitive inference approach. Section IV introduces the Fuzzy-XCS system. Section V shows some experimental results. Section VI concludes.

II. Difficulties in Real-valued Output Accuracy-Based Fuzzy Classifier Systems

Most of the difficulty in accuracy-based fuzzy classifier systems is the fuzzy inference process; therefore, this issue is briefly discussed in the next subsection. Following this discussion, the problems when using strength-based fuzzy classifier systems, the advantages of considering accuracy-based fitness, and the difficulties in doing that are introduced.

A. Cooperation-Based Fuzzy Inference

Linguistic (or Mamdani-type) fuzzy rule-based systems are formed by linguistic fuzzy rules with the following structure:

...
also
R^{r-1}: **IF** X_1 is A_1^{r-1} and ... and X_n is A_n^{r-1}
THEN Y_1 is B_1^{r-1} and ... and Y_m is B_m^{r-1} ,
also
R^r: **IF** X_1 is A_1^r and ... and X_n is A_n^r
THEN Y_1 is B_1^r and ... and Y_m is B_m^r ,
also
...

with X_i and Y_j being input and output linguistic variables respectively, and with A_i and B_j being linguistic labels with associated fuzzy sets defining their meaning. These linguistic labels use a global semantic defining the set of possible fuzzy sets used for each variable. This allows this kind of fuzzy system to potentially demonstrate a high level of interpretability.

The most widely used inference scheme in these types of systems is that proposed by Mamdani and Assilian in the first fuzzy controller of 1975. It is called Max-Min and involves the combination of *minimum* for implication (***then***) and *maximum* for aggregation (***also***).

The use of Max-Min inference (broadly, t-conorm/t-norm) leads fuzzy rules to "cooperate" to generate the output, in the sense that an *interpolative reasoning* is performed to define the output in the zones where several rules work to a medium matching degree. Indeed, the fact of using a t-conorm to aggregate the information involves a *union* and, therefore, the effect of each rule is added to the final consensual output.

B. Problems with "Strength-Based" Michigan Fuzzy Classifier Systems

As mentioned in the introduction, the strength-based fuzzy classifier system is characterized by using the same parameter (i.e. strength) to resolve conflicts between matched classifiers and to compute their fitness. A number of problems arise from this dual use of classifier strength. These include:

1) The cooperation/competition problem. High-strength, potentially cooperative classifiers go on to compete under the action of the EA.
2) Over-general rules with relatively high (but inconsistent) payoff can come to dominate the population.
3) In some environmental states, the maximum payoff achievable (by performing the best possible action for that state) may be relatively low. Although a classifier might be the best that can exist for that state, it can be eradicated from the population by other classifiers that achieve higher rewards in other states. This results in gaps in the system's "covering map."

C. Advantages of Using "Accuracy-Based" Fitness

A completely different approach can be taken in which a classifier's fitness, from the point of view of the EA, is based on its "accuracy," i.e. how well a classifier predicts payoff whenever it fires. Such an accuracy-based approach offers a number of advantages. Firstly, it can distinguish between accurate and over-general classifiers: an over-general classifier will have relatively low accuracy since payoff will vary according to the input states covered by the classifier. Indeed, it has been shown (in the discrete valued case) that the accuracy-based approach can lead to evolution of optimally general classifiers. Additionally it can maintain both consistently correct and consistently incorrect classifiers which allows learning of a complete "covering map." A potential drawback of the accuracy-based approach is that it is likely to require larger populations of classifiers.

D. Difficulties in Moving to "Accuracy-Based" Fuzzy Classifier Systems

Firstly, in a traditional fuzzy classifier system several rules fire in parallel (this is how the system achieves interpolation); credit assignment is much more difficult in the fuzzy case and it may well be that apportioning credit in proportion to a fuzzy classifier's activation level is not appropriate. A further difficulty is measuring the accuracy of a rule's predicted payoff since (particularly early in the search) a fuzzy rule will fire with many different other fuzzy rules at different time-steps, giving very different payoffs. Yet another difficulty is that the payoff a fuzzy rule receives depends on the input vector — an active fuzzy rule will receive different payoffs for different inputs. This further complicates payoff predictions used as the basis for accuracy-based fitness.

III. Competitive Fuzzy Inference

The problems explained in the previous section are mainly due to the fact that, usually, all the matched rules *cooperate* to define the final solution in a interpolative behavior as explained in Section II-A. If it is the problem, why not to change our approach and look for competitive interaction? This section introduces such an approach.

A. Why Use Competitive Inference?

Michigan-style learning classifier systems in general, and XCS in particular, do not consider the interaction of the different rules as a cooperative action, instead they consider that each rule competes with the rest to be the best one for a particular input vector. This involves that the action is only due to a set of rules that were the winners among the rules composing the match set. Extending this approach to fuzzy modeling, it makes sense to deal with competitive fuzzy rules, in which case, we need to know how to work with such rules.

In off-line fuzzy system learning, it seems that cooperative (i.e., Max-Min-style) inference has some clear advantages that make it be more suitable for this problem. In this case, we know the whole data set *a priori* and our objective is to build a landscape of the output data. To do that, it seems a good approach to fill the gaps between data with interpolation, and this is the reason why cooperative inference is successful.

However, in on-line fuzzy system learning, why is it necessary to interpolate the actions of the rules? The objective in this problem should not be to define a landscape of the output, but a landscape of the reward. This means taking the best action in each state. For this purpose, a competitive action makes more sense since our objective is to optimize the rules individually to have the best reward.

With competitive fuzzy inference we propose a decision making process where the matched rules compete among themselves, and only one is the winner. The output finally obtained is mainly due to the winner rule. That is, loser rules do not have an important influence on the output. Nevertheless, competition does not involve a loss of interaction. There is an interaction, but with a selfish objective. To perform competitive inference we only need to change the roles of the inference operators.

B. Fuzzy Operators for Competitive Inference

First of all, since each rule competes with the rest, we should use an intersection (t-norm) as *also* operator, instead of the union approach followed by the cooperative inference. We use the *minimum* in this paper. Then, the implication should follow the classical Boolean view ($p \Rightarrow q \equiv \neg p \vee q$) assuming that each rule holds all the information about the relationship between fact and consequence. Thus, the belief develops separately for each rule. Therefore, we can use S-implications—defined from the t-conorm S as $I^S(x,y) = S(N(x), y)$—like Łukasiewicz ($I^S_{\Lambda^*} = min\{1, 1 - \mu_A(x) + \mu_B(y)\}$) or Kleene-Dienes ($I^S_{Max} = max\{1 - \mu_A(x), \mu_B(y)\}$).

To understand better the different behavior between cooperative (or Max-Min-based) and competitive (or S-implication-based) inference, Figure 1 shows the output generated by two fuzzy systems that only differ in the inference mechanism. In the cooperative inference, minimum, minimum, and maximum are used as conjunction, implication, and aggregation operators, respectively. In the competitive inference, minimum, Łukasiewicz, and minimum are used as conjunction, implication, and aggregation operators, respectively. The center of gravity is used as defuzzification process in both cases. Both systems consist of two input variables and one output variable, with 5 triangular fuzzy sets uniformly distributed in the universe of discourse [0,1] for each variable. Among the 25 fuzzy rules considered, only 7 have a consequent different from the medium linguistic term (with vertex at 0.5). As we can see, both approaches generate a smooth output but, while cooperative inference fills the gaps by interpolating, competitive inference isolates the effect of each rule.

On the other hand, Figure 2 illustrates an example of competitive inference using the Łukasiewicz's S-implication. The *minimum* aggregation operator is used. We can see how the action strength of the winner rule R_1 (the one with the highest matching degree) not only depends on its matching degree but also on the difference in the matching degree with respect to fuzzy rules with different actions (consequents), i.e., rival fuzzy rules. The more different these matching degrees, the higher the action strength of the selected rule. This involves

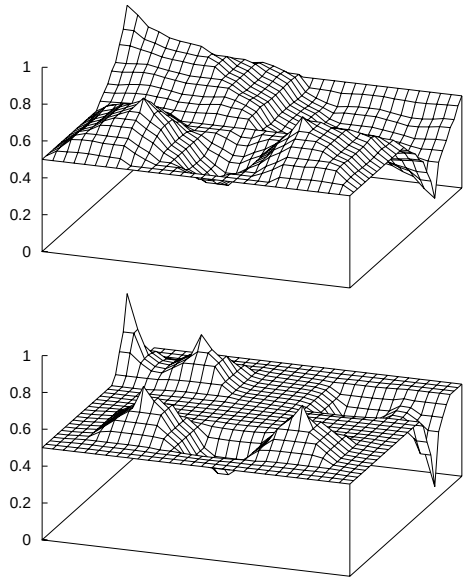

Fig. 1. Output generated by *cooperative* (top) and *competitive* (bottom) inferences

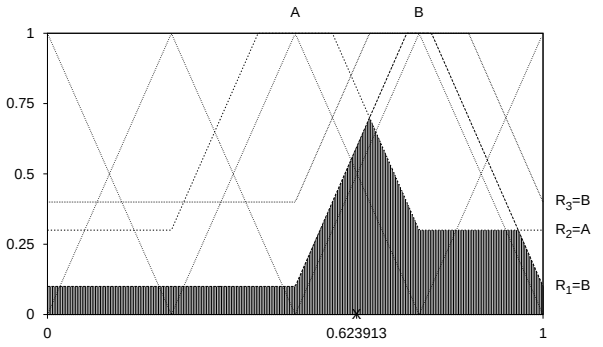

Fig. 2. Results of the Łukasiewicz's fuzzy implication using minimum as *also* operator. Matching degrees: $\mu_{A_{R_1}} = 0.9$, $\mu_{A_{R_2}} = 0.7$, and $\mu_{A_{R_3}} = 0.6$

that actually, the competition is performed among different linguistic actions, i.e., consequents. Therefore, we should say that competitive inference leads to competitive actions, but not necessarily competitive rules.

The competition among rules with the same action is only performed when using the minimum operator as aggregation, since in this case only the best rule of that action is considered in the inference. The use of other aggregation operators, such as the product, will lead the inference to a cooperation among rules with the same actions. According to XCS, this approach is not desired since it aims at obtaining the $X \times A \Rightarrow P$ map with the best classifier for each state-action combination [1].

Finally, we should say that the consideration of competitive actions seems to fit properly with the XCS aim. Other fuzzy learning classifier systems, like Bonarini *et al.*'s works [2], [3], focus on the interaction among rules in the antecedent (state) instead of the consequent (action). They propose a competition among rules with the same antecedent but a cooperation among rules with different ones. Besides, the implication and

aggregation considered in their proposal leads implicitly to a cooperation between the consequents. Therefore, these authors propose a competition/cooperation scheme for state/action, while we come up with a competition/competition scheme.

IV. FUZZY-XCS

This section shows the Fuzzy-XCS's operation mode with a competitive inference based on S-implications.

A. Generalization Representation

First of all, a representation of fuzzy classifiers to allow proper generalization must be done. We propose the use of *disjunctive normal form* (DNF) fuzzy rules with the following structure:

$$\textbf{IF } X_1 \text{ is } \widetilde{A_1} \text{ and } \ldots \text{ and } X_n \text{ is } \widetilde{A_n} \textbf{ THEN } Y \text{ is } B$$

where each input variable X_i takes as a value a set of linguistic terms $\widetilde{A_i} = \{A_{i1} \vee \ldots \vee A_{il_i}\}$, whose members are joined by a disjunctive (t-conorm) operator, whilst the output variable remains a usual linguistic variable with a single label associated. We use the *bounded sum* ($min\{1, a + b\}$) as t-conorm.

This structure uses a more compact description that allows rules with different generalization degrees. Moreover, the structure naturally supports the absence of some input variables in each rule (simply making $\widetilde{A_i}$ be the whole set of linguistic terms).

In order to use this representation in Fuzzy-XCS, we propose a binary coding scheme for the antecedent of the fuzzy rule with size equal to the sum of the number of linguistic terms used in each input variable. The allele '1' means that the corresponding linguistic term is used in the corresponding variable. For the consequent of the rule, an integer coding scheme is used where each gene contains the index of the linguistic terms used for the corresponding output variable. For example, assuming we have three linguistic terms (S, M, and L) for each input/output variable, the fuzzy rule [IF X_1 is S and X_2 is {M or L} THEN Y_1 is M and Y_2 is L] is encoded as [100|011||23].

B. Performance Component

In XCS [1], the performance component consists of three stages: match set construction, prediction array computation, and action set selection. This process has the final objective of inferring a specific action from the set of classifiers that matches the current state. In Fuzzy-XCS the process is different, as described as follows:

- *Match set* ([M]): To avoid an excessive number of matched rules, only those rules with a matching degree greater or equal to a specific value (θ_M) are included in the match set.
- *Computation of candidate subsets*: This stage could be equivalent to the prediction array computation. XCS partitions [M] into a number of mutually exclusive sets according to the action of each rule. However, in real-valued output systems like Fuzzy-XCS, several "linguistic actions" (consequents) could/should be considered together. Thus, Fuzzy-XCS redefines the concept of prediction array computation. We can assume that what should not be accepted in our case is to have an action set with inconsistent or redundant rules, i.e. rules with the same antecedent and different (inconsistent) or equal (redundant) consequent. In DNF-type fuzzy rules, to have the same antecedent also involves rules where the antecedent of some of them are contained in others.
  Therefore, different groups of consistent and non-redundant fuzzy rules (S_i) with the maximum number of rules in each group are formed. To do that we consider a simple greedy algorithm (though an implicit enumeration algorithm could be used) as follows:

```
Let R = {R_i / R_i ∈ [M]}
Shuffle R {to avoid order-biasing}
for i = 1 to |R| do
  S_i ← {R_i}
  for j = 1 to |R| (j ≠ i) do
    if (S_i ∪ {R_j} is consistent and
    non-redundant) then
      S_i ← S_i ∪ {R_j}
Remove repeated S_i
```

  We perform an *exploration/exploitation* scheme with probability 0.5. On each exploitation step, only those fuzzy classifiers sufficiently experienced (they have been updated—see Sect. IV-C.2—a threshold $\theta_{exploit}$ number of times) are considered. The most experienced classifier is considered when any one holds this condition. On exploration step, the whole match set is considered.
  Using only experienced rules during exploitation steps allows the system to give the best action according to the available knowledge, while on exploration steps the system is able to learn or corroborate classifiers. Notice that though this is a new interpretation of Wilson's XCS scheme, we are addressing the fuzzy context where we should not decide between one action or another—all fuzzy rules should act together in order to give a sound real-valued output. So, in Fuzzy-XCS the exploration/exploitation concept is moved from the action selection framework to the rule selection one.
- *Action set selection*: The action set selection chooses the consistent and non-redundant classifier subset (S_i) with the highest mean prediction.

C. Reinforcement Component

The p_j (prediction), ϵ_j (prediction error), and F_j (fitness) values are adjusted by the reinforcement learning standard techniques used in XCS for each fuzzy classifier C_j. (We are applying Fuzzy-XCS to a single-step problem where immediate reward is returned by the environment; therefore, any form of Q-learning is used.)

However, an important difference is considered in Fuzzy-XCS: the distribution among the classifiers must be made proportionally to the degree of contribution of each classifier

to the obtained output. This is a crucial issue because the interaction among the classifiers (with competition actions in our case) is developed here.

The reinforcement performed in Fuzzy-XCS acts on the action set [A] when an exploration step is performed. The following subsections detail the reinforcement distribution process and the adjustment of the parameters.

1) Distribution of the reinforcement: The reinforcement distribution among the classifiers of the action set [A] is made by analyzing the contribution of each classifier to generate the aggregated output fuzzy set. Let B'_j be the scaled output fuzzy set generated by the fuzzy rule R_j:

$$B'_j = I^S(\mu_{A_{R_j}}(x), B_j), \tag{1}$$

where $\mu_{A_{R_j}}(x)$ is the matching degree of the rule R_j, I^S is the used S-implication, and B_j the fuzzy set of the consequent of the rule R_j. Let $R_1 \in$ [A] be the winner rule, i.e., $\mu_{A_{R_1}}(x) \geq \mu_{A_{R_j}}(x), \forall R_j \in$ [A] $- \{R_1\}$.

The process involves analyzing the area that the rival fuzzy rules (rules with lower matching degrees than R_1) "bite" into the area generated by the winner rule.

Thus, the weight of the winner rule is:

$$w_1 = \frac{\int \bigwedge_{j=1}^{|[A]|} \mu_{B'_j}(y)\ dy}{\int \mu_{B'_1}(y)\ dy}, \tag{2}$$

with |[A]| being the action set size and $\wedge$ the t-norm used as aggregation operator (the minimum in our case), while the weights of the rival rules are computed as follows:

$$w_j = \frac{(1-w_1)\cdot\left(\int \mu_{B'_1}(y)\ dy - \int \mu_{B'_1}(y) \wedge \mu_{B'_j}(y)\ dy\right)}{\sum_{i=2}^{|[A]|}\left(\int \mu_{B'_1}(y)\ dy - \int \mu_{B'_1}(y) \wedge \mu_{B'_i}(y)\ dy\right)} \tag{3}$$

This distribution is designed to take into account that competitive inference is being considered. To illustrate this behavior, we can see that, from the example shown in Figure 2, the competitive inference generates the weights $w_1 = 0.711$, $w_2 = 0.289$, and $w_3 = 0$, while a cooperative-based distribution proportional to the matching degrees generates the weights $w_1 = 0.409$, $w_2 = 0.318$, and $w_3 = 0.273$. Indeed, the selection pressure in the competitive inference is higher, thus allowing to discriminate between good and bad rules.

2) Parameter updates: To adjust the parameters of each classifier, firstly the P (payoff) value is computed directly from the immediate reward r received from the environment, $P \leftarrow r$, since we are considering single-step problems.

Then, the following adjustment process is performed for each fuzzy classifier belonging to the action set:

1) Increase its experience, $exp_j \leftarrow exp_j + 1$.
2) Adjust the error values ϵ_j using the standard Widrow-Hoff delta rule with learning rate parameter β ($0 < \beta \leq 1$) toward $|P - p_j|$ considering the weights w_j computed in eqs. (2) and (3) to distribute the adjustment, i.e.

$$\epsilon_j \leftarrow \epsilon_j + \beta \cdot w_j \cdot (|P - p_j| - \epsilon_j). \tag{4}$$

The MAM (modified adaptive method) technique is used by adjusting ϵ_j to the average of the $|P - p_j|$ values, instead of the above equation, during the first $1/\beta$ times that the corresponding classifier is adjusted, i.e., if $exp_j < 1/\beta$.

3) Then, adjust prediction values

$$p_j \leftarrow p_j + \beta \cdot w_j \cdot (P - p_j). \tag{5}$$

Again, weights are considered to distribute the reinforcement. MAM technique is also used here during the $1/\beta$ first adjustments.

4) Finally, recalculate the fitness values F_j from the updated values of ϵ_j, i.e.

$$F_j \leftarrow F_j + \beta \cdot (k'_j - F_j), \tag{6}$$

with

$$k'_j = \frac{k_j}{\sum_{R_i \in [A]} k_i}, \qquad k_j = \begin{cases} (\epsilon_j/\epsilon_0)^{-\nu} & \epsilon_j > \epsilon_0 \\ 1 & \text{otherwise.} \end{cases} \tag{7}$$

No weights w_j are considered to update the fitness since it depends on the prediction error instead of the received payoff. Again, MAM technique is used.

D. Discovery Component

The EA for Fuzzy-XCS acts only on the action set [A] when an exploration step is performed. In order to apply an EA, the average time period since the last EA application in the action set must be greater than the threshold θ_{GA}. When applied, it selects two classifiers by roulette-wheel selection based on fitness, applies crossover and mutation operators with probabilities χ and μ_{chom} (per chromosome), respectively, and inserts the offspring in the population. If the population contains the maximum number of fuzzy classifiers allowed, two individuals are deleted to make room. They are randomly selected proportionally to the prediction error.

A simple two-point crossover operator that only acts on the antecedent part of the chromosomes (binary coding scheme) is considered. Prediction, prediction error, and fitness values of offspring are initialized to the mean values of the parents.

The mutation randomly selects an input/output variable of the rule. If an input variable is selected, one of the three following possibilities is applied: *expansion*, which flips to 1 a gene of the selected variable; *contraction*, which flips to 0 a gene of the selected variable; or *shift*, which flips to 0 a gene of the variable and flips to 1 the gene immediately before or after it. The selection of one of these mechanisms is made randomly among the available choices (e.g., contraction can not be applied if only a gene of the selected variable has the allele 1). If an output variable is selected, the mutation operator simply increases or decreases the integer value. Prediction, prediction error, and fitness values of mutated classifiers are not changed.

An EA subsumption is performed. Thus, if the offspring is logically contained by either of its parents and this parent is sufficiently experienced (it has been updated a threshold θ_{GA}

number of times), the offspring is not added to the pool but the parent's numerosity is incremented.

When no fuzzy rules cover the state with the highest matching degree, a covering mechanism is used to include a fuzzy classifier with the input linguistic term set that best matches the state and a random action. The population is left empty at the beginning of the algorithm.

V. Experimental Results

Experiments have been performed to test the behavior of Fuzzy-XCS. We have developed a laboratory problem to play a similar role as the multiplexer problem for discrete-valued classifier systems. Thus, we have generated an example data set from a previously defined rule base with different degrees of generalization. Two input variables and one output variable are considered. A total of 576 examples uniformly distributed in the input space (24×24) were generated. Five linguistic terms are considered for each variable. Uniformly distributed triangular-shaped membership functions are used. The fuzzy rule base considered to generate the data set is shown in Table I. The reward depends inversely on the difference between the inferred and the desired output in a non-linear way, i.e, $r = (1-|y-\hat{y}|)/(1+|y-\hat{y}|/\delta)$, with $\delta = 0.3$, y being the output obtained by the fuzzy classifier system, and $\hat{y}$ the expected output according to the data set for the current state. The objective is to obtain the set of rules that best approximate the data with the highest degree of generalization, i.e., a rule base as accurate and compact as possible.

TABLE I

Rule base used to generate the data set

	X_1					X_2					Y				
	VS	S	M	L	VL	VS	S	M	L	VL	VS	S	M	L	VL
R_1	x	x				x	x				x				
R_2			x	x	x	x	x					x			
R_3	x							x	x	x			x		
R_4		x						x	x	x				x	
R_5			x	x	x			x	x	x					x

The used parameter values are the following: maximum number of classifiers = 200, $\beta = 0.2$, $\nu = 3$, $\epsilon_0 = 0.05$, $\alpha = 0.1$, $\theta_{exploit} = \theta_{del} = \theta_{GA} = 30$, $\chi = 0.7$, $\mu_{chrom} = 0.1$, $\theta_M = 0.3$, *also* operator = min, and *then* operator = Łukasiewicz. Notice that standard parameter values are used.

Figure 3 shows the average behavior of 10 runs of Fuzzy-XCS with competitive inference and reinforcement during 30,000 exploit steps (60,000 trials in total). The upper figure depicts the relative numerosity (number of copies of the classifier divided by the population size) of the five optimal fuzzy classifiers. It shows the capability of the algorithm to find and keep the optimal solution. The bottom figure depicts the mean approximation error ($|y - \hat{y}|$) of the last 50 exploit steps. It shows the capability of the algorithm to provide the appropriate action (output) to the corresponding state (input). Note that a pseudo-optimal solution is found.

On the other hand, Fig. 4 shows the average behavior of an algorithm following a cooperative inference and reinforcement. To do that, the only changes made to the algorithm

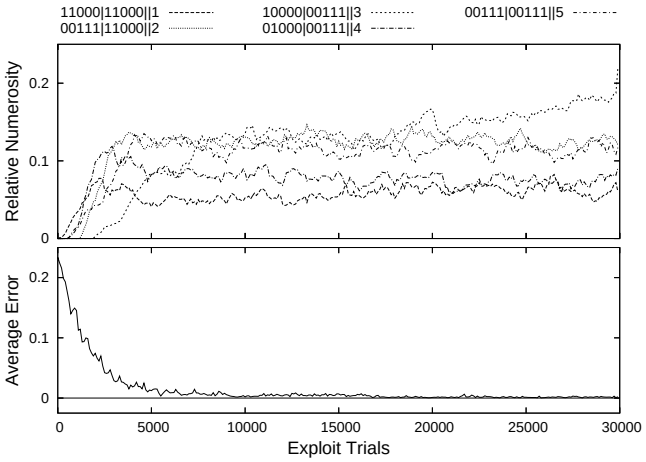

Fig. 3. Results of Fuzzy-XCS with *competitive* inference and reinforcement approach

proposed in this paper are the use of *minimum* as implication and *maximum* as aggregation in the inference engine, and a distribution of the reward proportional to the matching degrees in the reinforcement component. This algorithm is tested with a data set generated with the same cooperative inference (Max-Min) to avoid biasing the experiment. Notice that the cooperative approach is unable to find a good solution.

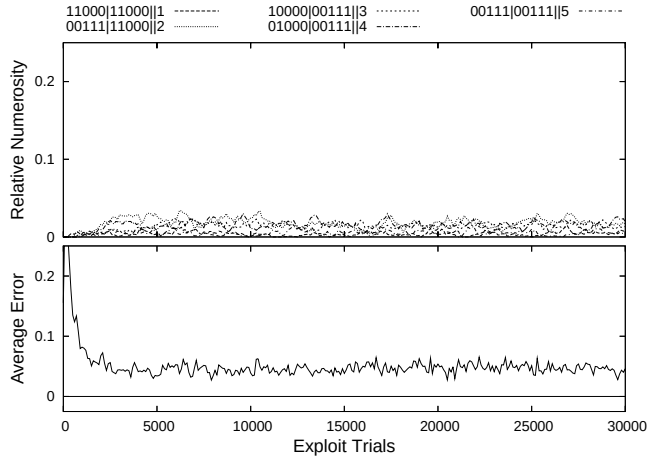

Fig. 4. Results of Fuzzy-XCS with *cooperative* inference and reinforcement approach

In order to test the performance of our on-line learning process by a Michigan-style approach, we have also applied off-line learning using a simple Pittsburgh-style genetic fuzzy system to the same problem. Briefly, this algorithm consists of the following components. A *generational approach* with direct replacement (offspring replace the corresponding parents) is considered. An *elitism* component that ensures the best chromosome survival of the previous generation is applied. The *fitness* is the mean square error (MSE), $f(S) = \sum_{i=1}^{N} (S(x^i) - y^i)^2/N$, with S being the evaluated fuzzy system, N the data set size (576 in our experiment), and

(x^i, y^i) the ith input-output pair of the data set. The same *coding scheme* than the proposed Fuzzy-XCS (Sect. IV-A) for each fuzzy rule is used, but taking into account that now a chromosome is composed of a set of rules. *Variable-length chromosome size* is considered. The population is randomly initialized. *Binary tournament selection* is used. The *crossover* operator randomly chooses a cross point between two fuzzy rules at each chromosome, and exchanges the right string of them. Therefore, the crossover only exchanges complete rules, but it does not create new ones since it respects rule boundaries on chromosomes representing the individual rule base. In the case that inconsistent rules appear after crossover, the ones whose antecedent is logically subsumed by the antecedent of a more general rule are removed. Redundant rules are also removed. The *mutation* operator works in a similar way to the one proposed for our Fuzzy-XCS algorithm in Sect. IV-D. In the same way, specific rules appeared after mutation are subsumed by the most general ones and redundant rules are removed. The parameter values used in the experiments were as follows: population size = 50, number of generations = 200, $\chi = 0.7$, and $\mu_{chrom} = 0.1$. We have considered Max-Min fuzzy inference and the example data set is generated with the same inference to avoid biasing the experiment.

Table II summarizes the results obtained by the analyzed method. The final fuzzy rule set returned by Fuzzy-XCS is composed of such classifiers of the final pool with experience and prediction error values holding the following condition: $(exp_j \geq \theta_{del}) \wedge (\epsilon_j \leq \epsilon_0)$; with $\theta_{del} = 30$ and $\epsilon_0 = 0.05$ in our experiment. In the Pittsburgh-style genetic fuzzy system, the returned fuzzy rule set is the one with the best fitness in the last population that, due to the considered elitism, coincides with the best solution found during all the process.

That table shows the mean number of times (over the 10 runs performed for each algorithm) that each of the five optimal rules (according to Table I) appear in the returned fuzzy rule set. It also shows the mean number of other suboptimal rules (i.e., those whose antecedent is subsumed by and the consequent is equal to an optimal rule) and non-suboptimal rules included in the returned fuzzy rule set. We have also included the mean MSE values of the fuzzy rule sets returned by both Fuzzy-XCS (with competitive approach) and Pittsburgh-style algorithms.

We can see that our competitive-based Fuzzy-XCS algorithm always find the five optimal rules (excepting one run where only four of them are found). On the contrary, the cooperative approach is unable to properly find neither optimal nor suboptimal rules. Finally, regarding Pittsburgh-style genetic fuzzy system, it has serious difficulties to find optimal solutions and, moreover, the knowledge resource used to do that (respecting the number of analyzed examples) is tremendously higher than the Michigan-style approach.

In this respect, it is interesting to highlight that the proposed Fuzzy-XCS finds a pseudo-optimal solution after around 20,000 analyzed examples (10,000 exploit ones according to Fig. 3), which is equivalent to 35 evaluated chromosomes, i.e, less than the number of evaluations needed for the initial population of the Pittsburgh-style genetic fuzzy system.

TABLE II
RESULTS OBTAINED BY THE ANALYZED METHODS

	Fuzzy-XCS (competitive)	Fuzzy-XCS (cooperative)	Pittsburgh-style genetic fuzzy system
R_1	0.9	0.0	0.1
R_2	1.0	0.1	0.2
R_3	1.0	0.0	0.0
R_4	1.0	0.0	0.2
R_5	1.0	0.0	0.1
suboptimal rules	0.8	1.1	7.6
non-suboptimal rules	1.1	0.0	2.0
MSE	0.000302	—	0.001892
analyzed examples	60,000	60,000	4,212,864

VI. CONCLUDING REMARKS

The paper has presented a proposal to properly develop an accuracy-based Michigan-style fuzzy classifier system for real-valued input and output. It is mainly based on a different inference approach that considers the interaction among fuzzy classifiers (rules) from a competitive point of view. The reinforcement component, based on XCS, is adapted to allow this behavior. The approach has the advantage of performing a higher selection pressure that results in a proper discrimination between good and bad fuzzy classifiers.

Promising results of the proposal have been obtained in a simple laboratory problem. Current and future work involves investigating the behavior of the proposal in multi-step (with delayed-reinforcement) and real-world problems.

ACKNOWLEDGMENT

Research partially supported by a stay grant of *Secretaría General de Universidades e Investigación de la Junta de Andalucía* (Spain), the Spanish Ministry of Science and Technology under grant no. TIC2002-04036-C05-01, and ERDF.

REFERENCES

[1] S. Wilson, "Classifier fitness based on accuracy," *Evolutionary Computation*, vol. 3, no. 2, pp. 149–175, 1995.
[2] A. Bonarini, "Evolutionary learning of fuzzy rules: competition and cooperation," in *Fuzzy modelling: paradigms and practice*, W. Pedrycz, Ed. Norwell, MA, USA: Kluwer Academic, 1996, pp. 265–284.
[3] A. Bonarini and V. Trianni, "Learning fuzzy classifier systems for multi-agent coordination," *Information Sciences*, vol. 136, pp. 215–239, 2001.
[4] H. Ishibuchi, T. Nakashima, and T. Murata, "Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems," *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, vol. 29, pp. 601–608, 1999.
[5] A. Parodi and P. Bonelli, "A new approach to fuzzy classifier systems," in *Proceedings of the 5th International Conference on Genetic Algorithms*. San Mateo, CA, USA: Morgan Kaufmann Publishers, 1993, pp. 223–230.
[6] M. Valenzuela-Rendón, "The fuzzy classifier system: a classifier system for continuously varying variables," in *Proceedings of the 4th International Conference on Genetic Algorithms*. San Mateo, CA, USA: Morgan Kaufmann Publishers, 1991, pp. 346–353.
[7] ——, "Reinforcement learning in the fuzzy classifier system," *Expert Systems with Applications*, vol. 14, pp. 237–247, 1998.
[8] J. Velasco, "Genetic-based on-line learning for fuzzy process control," *International Journal of Intelligent Systems*, vol. 13, pp. 891–903, 1998.
[9] D. Gu and H. Hu, "Accuracy based fuzzy Q-learning for robot behaviours," in *Proceedings of the 12th IEEE International Conference on Fuzzy Systems*, 2004, pp. 1126–1131.