Mathware & Soft Computing 8 (2001) 321-335

Improvement to the Cooperative Rules Methodology by Using the Ant Colony System Algorithm[*]

R. Alcalá[1], J. Casillas[2], O. Cordón[2] and F. Herrera[2]
[1]Department of Computer Science.
University of Jaén, E-23071 Jaén, Spain
e-mail: alcala@ujaen.es
[2]Department of Computer Science and Artificial Intelligence
University of Granada, E-18071 Granada, Spain
e-mail: {casillas,ocordon,herrera}@decsai.ugr.es

Abstract

The cooperative rules (COR) methodology [2] is based on a combinatorial search of cooperative rules performed over a set of previously generated candidate rule consequents. It obtains accurate models preserving the highest interpretability of the linguistic fuzzy rule-based systems.

Once the good behavior of the COR methodology has been proven in previous works, this contribution focuses on developing the process with a novel kind of metaheuristic algorithm: the ant colony system one. Thanks to the capability of this algorithm to include heuristic information, the learning process is accelerated without model accuracy losses.

Its behavior is successful compared with other processes based on genetic algorithms and simulated annealing when solving two modeling applications.

Keywords: linguistic fuzzy modeling, learning, cooperative rules, ant colony system

1 Introduction

In [2], a learning methodology that obtains accurate fuzzy models by inducing a better cooperation among the fuzzy rules is proposed: the cooperative rules (COR) methodology. This approach arises as an effort to exploit the accuracy ability of linguistic (Mamdani-type) fuzzy rule-based systems (FRBSs) by exclusively focusing on the rule set design. In this case, the rest of components (membership

[*]This research was supported by CICYT, project PB98-1319

functions, model structure, etc.) remains invariable, thus resulting in the highest interpretability.

The COR methodology [2] is based on a combinatorial search of cooperative rules performed over a set of previously generated candidate rule consequents to find those with the best cooperation. Instead of selecting the consequent with the highest performance in each fuzzy input subspace as ad hoc data-driven learning methods (eg. [16]) usually do, the COR methodology considers the possibility of using another consequent, different from the best one, that allows the FRBS to globally achieve a best accuracy.

The COR methodology was initially applied with a classical metaheuristic, simulated annealing, and good accuracy results were obtained [2]. Although the use of this technique allows COR to perform a quick learning, the process could be accelerated with a more sophisticated metaheuristic that considers additional information to guide the search. Therefore, once a trade-off between the accuracy and interpretability of the models obtained by COR have been proved in previous works [2, 3], this contribution focuses on proposing a novel algorithm to decrease the time required in the learning process, the ant colony system (ACS) algorithm [11].

The time expended in this fuzzy rule learning may take a great importance together with the accuracy and interpretability of the obtained fuzzy model. Indeed, a quick learning has some interesting advantages as the capability of being used as a previous mechanism to understand the nature of the problem being solved [4], being used as a first learning stage to subsequently refine the obtained results with a more complex postprocessing [5, 13], being integrated within a meta-learning process [8, 9], etc.

The paper is structured as follows. Section 2 presents the COR methodology. Section 3 is devoted to introduce all the aspects related to apply the ACS algorithm to the COR methodology. In Section 4, the behavior of the proposed method when solving two different example applications is analyzed and it is compared to other well-known fuzzy rule generation processes. Finally, in Section 5, some concluding remarks and future work are pointed out. An introduction to ant colony optimization (ACO) algorithms is described in Appendix A.

2 COR: A Methodology to Improve the Cooperation Among Rules

Let E be the input-output data set, $e_l = (x_1^l, \ldots, x_n^l, y^l)$ one of its elements (example), and n be the number of input variables. Let $\mathcal{A}_i$ be the set of linguistic terms of the i-th input variable and $\mathcal{B}$ the set of linguistic terms of the output variable. Figure 1 shows the COR methodology structure.

For example, from the subspace $S_s = (high, low)$ and the candidate consequent set in such a subspace $\mathbf{B^s} = \{small, medium, large\}$, we will obtain the fuzzy rule:

$$R_s = \textsf{IF } X_1 \textsf{ is } \textit{high} \textsf{ and } X_2 \textsf{ is } \textit{low} \textbf{ THEN } Y \textsf{ is } B_s,$$

with $B_s \in \mathbf{B^s}$ being the consequent label selected by the combinatorial search to represent to the subspace S_s associated to the rule R_s.

1. Define a set of fuzzy input subspaces, $\{S_s \mid s \in \{1,\ldots,N_S\}\}$, with the antecedent combinations containing at least a positive example, i.e., $S_s = (A_1^s,\ldots,A_i^s,\ldots,A_n^s) \in \mathcal{A}_1 \times \ldots \times \mathcal{A}_n$ such that $E'_s \neq \emptyset$ (with A_i^s being a label of the i-th input variable, E'_s being the set of positive examples of the subspace S_s, and N_S the number of subspaces with positive examples).

   In this contribution, we will define the set of positive examples for the subspace S_s as follows:

   $$E'_s = \{e_l \in E \ \mid \ \forall i \in \{1,\ldots,n\}, \forall A_{ij} \in \mathcal{A}_i,\ \mu_{A_i^s}(x_i^l) \geq \mu_{A_{ij}}(x_i^l)\} \ ,$$

   with A_{ij} being a label of the i-th input variable and μ_T the membership function of the label T.

2. For each subspace S_s, obtain a set of candidate consequents (i.e., linguistic terms of the output variable) $\mathbf{B^s}$ to build the corresponding fuzzy rule.

   The set of candidate consequents for the subspace S_s is defined as follows:

   $$\mathbf{B^s} = \{B_k \in \mathcal{B} \mid \exists e_{l^s} \in E'_s \ where \ \forall B_l \in \mathcal{B},\ \mu_{B_k}(y^{l^s}) \geq \mu_{B_l}(y^{l^s})\} \ ,$$

   with B_k being a label of the output variable.

3. Perform a *combinatorial search* among these sets looking for the combination of consequents (one for each subspace) with the best global accuracy.

Figure 1: COR methodology

Since the search space tackled is usually large, it is necessary to use approximate search techniques. In [2, 3], accurate linguistic models have been obtained using simulated annealing. Nevertheless, these results could be improved incorporating heuristic information to the learning process. This consideration would guide the algorithm in the search, making it quicker on finding good solutions. The ACS algorithm [11] is a good support for such intention thanks to the inherent use of heuristic information.

3 Application of the Ant Colony System Algorithm to the COR Methodology

The following five subsections describe the way of performing the learning process with the ACS algorithm following the COR methodology. A brief introduction to ACO algorithms is presented in Appendix A.

3.1 Problem Representation

For applying the ACS algorithm in the COR methodology, it is convenient to see it as a combinatorial optimization problem with the capability of being represented on a graph. In this way, we can face the problem considering a fixed number of rules and interpreting the learning process as the way of assigning consequents — i.e., labels of the output fuzzy partition — to these rules with respect to an optimality criterion (i.e., following the COR methodology).

Hence, we are in fact dealing with an assignment problem and the problem representation can be similar to the one used to solve the *quadratic assignment problem* (QAP) [1] — briefly explained in Appendix A —, but with some peculiarities. We may draw an analogy between *rules* and locations and between *consequents* and facilities. However, unlike the QAP, the *set of possible consequents for each rule may be different* and *it is possible to assign a consequent to more than one rule* (two rules may have the same consequent). We can deduce from these characteristics that the order of selecting each rule to be assigned a consequent is not determinant since one assignment does not restrict the remaining ones, i.e., the assignment order is irrelevant. The graph is constructed taking the steps described in Figure 2.

1. *Determine the subspaces* — As shown in Figure 1, consider the fuzzy input subspaces where there is located at least one example.

2. *Link the subspaces to consequents* — The subspace S_s will be linked to all the consequents that contain at least a positive example, i.e., $\forall B_k \in \mathbf{B}^{\mathbf{s}}$.

Figure 2: Graph construction process

Figure 3 shows an example of the learning process. In Figure 3(c), the possible consequents for each antecedent combination are shown according to the data set and membership functions considered (Figure 3(a)). To construct a complete solution, an ant iteratively goes over each rule and chooses a consequent with a probability that depends on the pheromone trail τ_{sk} and the heuristic information η_{sk}, as usual (see Figure 3(d)). As said, the order of selecting the rules is irrelevant. In Figure 3(e), we may see the possible paths that an ant can take in this example and Figure 3(g) shows the rule set encoded by a specific solution.

3.2 Heuristic Information

The heuristic information on the potential preference of selecting a specific consequent, B_k, in each antecedent combination (rule) is determined as described in Figure 4.

Since the heuristic information is based on covering criteria, it will be zero for a specific consequent when no examples located in the fuzzy input subspace are covered by it. This means that for a rule, only those links to consequents whose heuristic information is greater than zero will be considered. In Figure 3(d) we

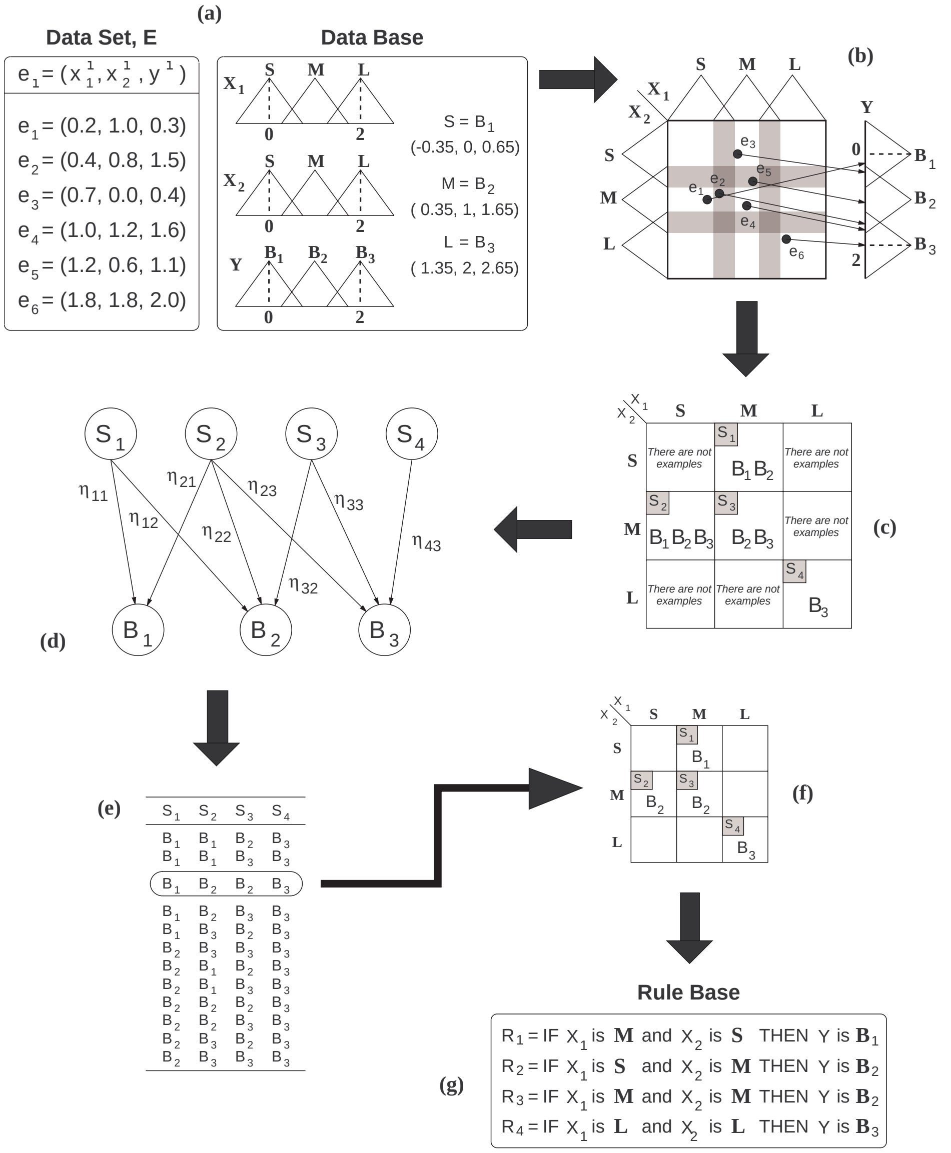

Figure 3: Learning process for a simple problem with two input variables ($n = 2$) and three labels in the output fuzzy partition ($|\mathcal{B}|$=3): (a) Data set (E) and fuzzy partitions previously defined; (b) The six examples are located in four different subspaces that determine the antecedent combinations and candidate consequents of the rules; (c) Set of possible consequents for the four (N_S = 4) possible rules (only the rules where at least one example is located in the corresponding subspace are considered); (d) Graph of paths where $\eta_{sk} \neq 0$ except η_{13}, η_{31}, η_{41}, and η_{42}, which are zero; (e) The search space is composed by twelve different paths (combinations of consequents); (f) Rule decision table for the third combination; (g) rule set generated from the third combination

For each subspace S_s do:

1. Build the set E'_s as shown in Figure 1.

2. Make use of an initialization function based on covering criteria to give a heuristic preference degree to each choice. Many different possibilities may be considered. In this paper, we will work with the following one:

$$\eta_{sk} = \max_{e_l \in E'_s} Min\left(\mu_{A_s}(x^l), \mu_{B_k}(y^l)\right).$$

with $\mu_{A_s}(x^l) = Min\left(\mu_{A_{1s}}(x^l_1), \ldots, \mu_{A_{ns}}(x^l_n)\right)$.

Figure 4: Heuristic assignment process

may observe that the consequent B_3 can not be assigned to the subspace S_1, the consequent B_1 can not be assigned to the subspace S_3, and the consequents B_1 and B_2 can not be assigned to the subspace S_4 because their heuristic informations (covering degrees) are zero.

3.3 Pheromone Initialization

The initial pheromone value of each assignment is obtained as follows:

$$\tau_0 = \frac{\sum_{s=1}^{N_S} \max_{s=1}^{|\mathcal{B}|} \eta_{is}}{N_S}.$$

In this way, the initial pheromone will be the mean value of the path constructed taking the best consequent in each rule according to the heuristic information (a greedy assignment) as the fuzzy rule learning algorithm presented in [6].

3.4 Fitness Function

The fitness function establishes the quality of a solution. The measure considered will be the function called *mean square error* (MSE), which is defined as

$$\mathrm{MSE}(RB_k) = \frac{1}{2 \cdot |E|} \sum_{e_l \in E} (y^l - F_k(x^l_0))^2,$$

with $F_k(x^l_0)$ being the output obtained from the FRBS (inferred using the rule base generated by the ant k, RB_k) when it receives the input x^l_0 (input component of the example e_l), and y^l being the known desired output. The closer to zero the measure, the better the solution.

3.5 Ant Colony System Algorithm

Once the previous components have been defined, an ACO algorithm has to be given to solve the problem. In this contribution, the well-known ACS [11] is considered. Their components adapted to our problem is introduced in the following subsections. We must remark that no local search is used in our proposal.

3.5.1 Solution Construction

The algorithm introduces a transition rule that establishes a balance between biased exploration and exploitation of the available information. The node k (i.e., the consequent B_k) is selected for the subspace S_s as follows:

$$k = \begin{cases} arg \max\limits_{u \in J(s)} \{(\tau_{su})^{\alpha} \cdot (\eta_{su})^{\beta}\}, & \text{if } q < q_0 \\ \\ T, & \text{otherwise} \end{cases} ,$$

with τ_{sk} being the pheromone of the trail (s, k); η_{sk} being the heuristic information; α and β being parameters which determine the relative influence of the pheromone strength and the heuristic information; $J(s) = \{k \ s.t. \ \eta_{sk} \neq 0\}$ being the set of nodes attainable from S_s, i.e., the set of consequents that can be associated to it; q being a random variable uniformly distributed over $[0, 1]$; $q_0 \in [0, 1]$ being a threshold defining the probability of selecting the more hopeful coupling (exploitation); and with T being a random node selected according to the following transition rule (biased exploration):

$$p(s,k) = \begin{cases} \dfrac{(\tau_{sk})^{\alpha} \cdot (\eta_{sk})^{\beta}}{\sum\limits_{u \in J(s)} (\tau_{su})^{\alpha} \cdot (\eta_{su})^{\beta}}, & \text{if } k \in J(s) \\ \\ 0, & \text{otherwise} \end{cases} ,$$

We should note that, as in the QAP, the transition rule becomes an assignment rule but, contrary to that problem, there is not a need for the ant to keep a tabu list with the previous assignments made, since the same consequent can be assigned to different rules.

3.5.2 Pheromone Trail Update Rule

The pheromone trail update rule is performed in two stages, global and local:

- *Global pheromone trail update rule*: Only an ant — the one who generated the best solution (T_{best}) till now — releases pheromone on a coupling. The formula is the following:

$$\tau_{sk} \leftarrow (1 - \rho) \cdot \tau_{sk} + \rho \cdot \Delta\tau_{sk} ,$$

with $\rho \in [0,1]$ being the pheromone evaporation parameter, m being the number of ants, T_a being the solution constructed by the ant a, and with

$$\Delta\tau_{sk} = \begin{cases} \dfrac{1}{\mathrm{MSE}(RB_{best})}, & \text{if } (s,k) \in T_{best} \\ 0, & \text{otherwise} \end{cases} .$$

- *Local pheromone trail update rule*: Each time an ant covers a coupling, a local pheromone update is done as follows:

$$\tau_{sk} \leftarrow (1-\rho) \cdot \tau_{sk} + \rho \cdot \Delta\tau_{sk} \ .$$

In this paper, we will consider $\Delta\tau_{sk} = \tau_0$ [11].

4 Experimental Study

In this section, some examples of application will be shown. With the aim of analyzing the behavior of the proposed process, we will compare it with some fuzzy rule learning methods: the method proposed by Wang and Mendel (WM) [16], the one proposed by Cordón and Herrera (CH) [6], the one proposed by Nozaki, Ishibuchi, and Tanaka (NIT) [14], and the genetic algorithm-based learning method proposed by Thrift (T) [15]. Actually, the CH method is directly based on the heuristic information used in the ACS algorithm by taking the consequent with the highest value for each rule, i.e., it is a greedy algorithm. Another learning method based on the COR methodology using a simulated annealing algorithm (COR-SA) [2] will also be considered. Table 1 summarizes the compared methods.

Table 1: Methods considered in this experimental study

Method	**Algorithm**	**Comments**
WM [16]	AHDD	Well-known learning method
CH [6]	AHDD	Heuristic-information-based greedy algorithm
NIT [14]	AHDD	Uses two weighted rules in each subspace
T [15]	GA	Capability of learning the number of rules
COR-SA [2]	SA	Based on the COR methodology
COR-ACS	ACO	ACS applied to the COR methodology

AHDD = ad hoc data-driven, **GA** = genetic algorithm, **SA** = simulated annealing

The performance of these learning methods will be analyzed when solving two different applications: the modeling of a three-dimensional surface defined by the function $\mathrm{F}(x_1, x_2) = x_1^2 + x_2^2$ [5] and a low-voltage electrical estimation problem [7]. The former one has a training data with 1,681 values and a test data with 168 values. The latter one has two input variables and a data set of 495 examples, randomly divided into a training set of 396 values and a test set of 99 values.

An initial membership function set constituted by a primary fuzzy partition for each variable will be considered in each case. Every partition is formed by 7 (in the three-dimensional surface problem) or 5 (in the electrical problem) labels with triangular-shaped equally distributed fuzzy sets giving meaning to them, and the appropriate scaling factors to translate the generic universe of discourse into the one associated with each problem variable. With respect to the FRBS reasoning method used, we have selected the *minimum t-norm* playing the role of the implication and conjunctive operators, and the FITA (first infer, then aggregate) approach, with the *center of gravity weighted by the matching* strategy acting as the defuzzification operator.

The following subsections shows the obtained results and an analysis of them. The analysis is accomplished from two different angles: *accuracy*, i.e., approximation (MSE_{tra}) and generalization (MSE_{tst}) degrees of the obtained models and *efficiency*, i.e., time required by the learning methods to find their best solutions (determined by EBS).

4.1 Obtained Results and Values of Parameters Used

Table 2 collects the results obtained by the analyzed methods solving the two considered applications. The best results are shown in boldface. In that table, #R stands for the number of rules, MSE_{tra} and MSE_{tst} for the values obtained over the training and test data sets respectively, and EBS for the number of evaluations needed to obtain the best solution. The values of the parameters used by each probabilistic method in each problem to obtain these results are collected in Table 3.

Table 2: Results obtained by the analyzed methods

	Three-dimensional surface				*Electrical application problem*			
Method	**#R**	**MSE$_{tra}$**	**MSE$_{tst}$**	**EBS**	**#R**	**MSE$_{tra}$**	**MSE$_{tst}$**	**EBS**
WM	**49**	2.048137	2.255928	—	**13**	298,450	282,029	—
CH	**49**	2.048137	2.255928	—	20	310,319	286,750	—
NIT	98	2.465487	1.768125	—	40	229,115	206,648	—
T	**49**	**1.609890**	**1.193721**	27,301	25	**218,551**	215,665	18,037
COR-SA	**49**	1.609891	1.213388	6,924	20	220,850	247,733	191
COR-ACS	**49**	1.610508	1.234023	**2,437**	20	221,003	**196,341**	**152**

Table 3: Values of the parameters considered in the probabilistic methods

Method	*Three-dimensional surface*	*Electrical application problem*
T	PS: 61, #G: 1000, P_c: 0.6, P_m: 0.2	PS: 61, #G: 1000, P_c: 0.6, P_m: 0.4
COR-SA	T_0: 40, #Neig.: 98, #Accept.: 98	T_0: 40, #Neig.: 40, #Accept.: 40
COR-ACS	$\rho = 0.4$, $\alpha = 1$, $\beta = 1$, $q_0 = 0.8$	$\rho = 0.6$, $\alpha = 1$, $\beta = 2$, $q_0 = 0.6$

The NIT method makes use of a linguistic modifier to improve the accuracy but it has not been considered ($\alpha = 1$) since this paper aims at analyzing the fuzzy rule generation process without modifying the membership function definitions. On the other hand, the parameters PS, #G, P_c, and P_m of the T method, respectively stand for the population size, the number of generations, and the probabilities of crossover and mutation. In the COR-SA method, T_0 stands for the initial temperature, #Neig. for the maximum number of neighbors generated at each temperature, and #Accept. for the maximum number of acceptances before decreasing the temperature.

Concerning the parameters used in the ACS method, the number of ants (m) will be the number of rules in each case, the number of iterations will be 50, and for the rest of parameters (ρ, α, β, q_0), the considered values are showed in each case.

4.2 Accuracy Analysis

The obtained results lead us to highlight the good behavior of the ACO-based learning method. It significantly improves the accuracy of the models generated by the WM, CH, and NIT methods, and it obtains similar accuracy degrees to the remaining methods. Moreover, the best generalization degree in the electrical problem is obtained by the ACS algorithm.

We may graphically analyze this behavior. Figure 5 collects the rule bases generated by the CH, the COR-SA, and the COR-ACS methods in the electrical application problem. In the two COR-based methods, the consequents differing from the ones generated by the greedy assignment CH method (Figure 5(a)) are shown in boldface and italics. The COR-SA method generates a rule base (Figure 5(b)) with a great number of consequents (10) different from the ones obtained by the CH method (greedy approach). On the contrary, the rule base generated by the COR-ACS method (Figure 5(c)) differs only in three consequents with respect to the CH method, presenting a similar approximation degree to the COR-SA method but a better generalization one. This behavior is related to the consideration of heuristic information to discriminate among the different candidate consequents.

In view of the linguistic fuzzy models generated in the electrical problem, we may conclude that the consideration of heuristic information — as the proposed ACO-based learning method does — involves generating models slightly different to the obtained with the greedy approach but with a good accuracy degree and quicker than the other COR-based approach.

4.3 Efficiency Analysis

The ACS algorithm stands out in speed terms finding good solutions quicker than the genetic algorithm (T method) and simulated annealing (COR-SA method) approaches. This fact is due to the use of heuristic information that guides the ACS algorithm in the search process.

In Figure 6, the evolution chart (*evaluations* vs. *fitness* of the best solution) during the first seven thousand evaluations of the T, COR-SA, and COR-ACS

				X1		
		VS	S	M	L	VL
	VS	VS	VS	S		
	S	S	S	S	M	
X2	M	S	S	L	M	M
	L	S	M	M	L	S
	VL	M	M	M		

(a) CH's rule base, MSE$_{tra/tst}$ = 310,319/286,750

				X1		
		VS	S	M	L	VL
	VS	VS	VS	S		
	S	VS	M	VS	S	
X2	M	S	M	VL	L	M
	L	M	M	M	L	S
	VL	M	L	L		

(b) COR-SA's rule base, MSE$_{tra/tst}$ = 220,850/247,733

				X1		
		VS	S	M	L	VL
	VS	VS	VS	S		
	S	S	S	S	M	
X2	M	S	M	VL	M	M
	L	M	M	M	L	S
	VL	M	M	M		

(c) COR-ACS's rule base, MSE$_{tra/tst}$ = 221,003/196,341

Figure 5: Some RBs generated in the electrical application problem

methods is represented when solving the three-dimensional surface problem. This chart clearly shows the different convergence speeds and number of evaluations (i.e., time) needed to find good solutions.

As we can see, the three learning methods start from solutions with significantly different qualities. The differences between the T method and the COR-based methods are clearly related to the two different search spaces tackled, significantly larger in the T method case ($1.78e44$ and $2.84e19$ solutions for the three-dimensional surface and electrical problems, respectively). Focusing on the COR-SA method and the COR-ACS method (both with the same search space), the latter begins with better solutions thanks to the use of heuristic information. During the first iterations, the COR-ACS method performs a quick convergence. The other two methods, the T method and the COR-SA method, will respectively need eleven and three times more evaluations to obtain a model close to the one obtained by the ACO-based method.

Figure 6: Evolution chart of the T, COR-SA, and COR-ACS methods in the three-dimensional surface problem

5 Concluding Remarks

This contribution has presented a novel and interesting improvement of the COR methodology by applying the ACS algorithm to it. Opposite to other learning methods based on different kinds of optimization techniques as simulated annealing and genetic algorithms, the ACS algorithm quickly obtains good solutions performing an appropriate convergence thanks to the use of heuristic information to guide the global search.

As further work, we propose to improve the behavior of the proposed ACS learning method with two different mechanisms: firstly, by adding a local search process (which is a usual consideration in ACO algorithms) and secondly, by allowing the COR methodology to have the capability of removing fuzzy rules with bad cooperation.

A Ant Colony Optimization Algorithms

A new family of bio-inspired algorithms has recently appeared, ACO algorithms [1, 10]. Since the first proposal, the Ant System algorithm [12] — applied to the Traveling Salesman Problem —, numerous models has been developed to solve a wide set of optimization problems (refer to [1, 10] for a review of models and applications).

ACO algorithms model the behavior of real ant colonies. Particularly, they draw inspiration from the social behavior of these insects to provide food to the colony. In the food search process, consisting of the food find and the return to the nest, the ants deposit a substance called *pheromone*. The ants have the ability of sniffing the pheromone and the colony is guided by it during the search. When an ant is located in an branch, it decides to take the path according to the probability defined by the pheromone existing in each trail.

In this way, the depositions of pheromone terminate in constructing a track between the nest and the food that can be followed by new ants. The continued action of the colony members involves the length of the track is progressively reduced. The shortest paths are finally the more frequently visited ones and, therefore, the pheromone concentration is higher. On the contrary, the longest paths are less visited and the associated pheromone trail is evaporated.

Therefore, ACO algorithms are based on the cooperative action of multiple agents, ants, each of them generating a possible solution to the problem at each iteration of the algorithm. To do so, each ant travels a graph that represents the problem and it makes use of two kinds of information, shared by the colony, that indicate the preference among the different edges between nodes:

- *Heuristic information*: It depends on the problem and it is obtained before running the algorithm, keeping inalterable during the process. The heuristic value for the edge (s, k) is noted as η_{sk}.
- *Pheromone trail information*: It is modified through the algorithm running depending on the paths taken by the ants and the goodness of the generated solutions. This information is represented by the pheromone trail. The value for the edge (s, k) is noted as τ_{sk}.

The basic operation mode is as follows [12]: at each iteration, a population of a specific number of ants progressively construct different tracks on the graph (i.e., solutions to the problem) according to a *probabilistic transition rule* that depends on the available information. After that, the pheromone trails are updated. This is done by first decreasing them by some constant factor (corresponding to the evaporation of the pheromone) and then reinforcing the solution attributes of the constructed solutions considering their quality. This task is developed by the *global pheromone trail update rule*.

Several extensions to this basic operation mode have been proposed. Their improvements mainly consist of using different transition and update rules, introducing new components, or adding a local search phase.

ACO algorithms have been applied to numerous problems with the QAP being one of the best known. It is an NP-hard optimization problem that involves assigning a set of facilities to a set of locations with given flows between the facilities and given distances between the locations. The goal is to associate facilities to locations in such a way that the sum of the product between flows and distances is minimal. Our proposal to apply ACO algorithms to the fuzzy rule learning problem presents certain similarities to the QAP as mentioned in Section 3.1.

To apply ACO algorithms to a specific problem, the five steps shown in Figure 7 have to be performed. In this contribution, these aspects particularized to the COR methodology are described in Section 3.

1. *Problem representation*: Interpret the problem to be solved as a graph or a similar structure easily traveled by ants.
2. *Heuristic information*: Define the way of assigning a heuristic preference to each choice that the ant has to take in each step to generate the solution.
3. *Pheromone initialization*: Establish an appropriate way of initializing the pheromone.
4. *Fitness function*: Define a fitness function to be optimized.
5. *ACO algorithm*: Select an ACO algorithm and apply it to the problem.

Figure 7: Steps followed to apply ACO algorithms to a specific problem

References

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm intelligence. From natural to artificial systems.* Oxford University Press, Oxford, UK, 1999.

[2] J. Casillas, O. Cordón, and F. Herrera. COR: A methodology to improve ad hoc data-driven linguistic rule learning methods by inducing cooperation among rules. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics.* To appear.

[3] J. Casillas, O. Cordón, and F. Herrera. Improving the Wang and Mendel's fuzzy rule learning method by inducing cooperation among rules. In *Proceedings of the 8th Information Processing and Management of Uncertainty in Knowledge-Based Systems Conference*, pages 1682–1688, Madrid, Spain, 2000.

[4] J.L. Castro, M. Delgado, and C.J. Mantas. A hybrid system for approximate reasoning. Internal report. Submmited to *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans.*

[5] O. Cordón and F. Herrera. A three-stage evolutionary process for learning descriptive and approximate fuzzy logic controller knowledge bases from examples. *International Journal of Approximate Reasoning*, 17(4):369–407, 1997.

[6] O. Cordón and F. Herrera. A proposal for improving the accuracy of linguistic modeling. *IEEE Transactions on Fuzzy Systems*, 8(3):335–344, 2000.

[7] O. Cordón, F. Herrera, and L. Sánchez. Solving electrical distribution problems using hybrid evolutionary data analysis techniques. *Applied Intelligence*, 10(1):5–24, 1999.

[8] O. Cordón, F. Herrera, and P. Villar. Analysis and guidelines to obtain a good fuzzy partition granularity for fuzzy rule-based systems using simulated

annealing. *International Journal of Approximate Reasoning*, 25(3):187–215, 2000.

[9] O. Cordón, F. Herrera, and P. Villar. Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base. *IEEE Transactions on Fuzzy Systems*, 9(4):667–674, 2001.

[10] M. Dorigo and G. Di Caro. The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New ideas in optimization*, pages 11–32. McGraw-Hill, New York, NY, USA, 1999.

[11] M. Dorigo and L.M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

[12] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, 26(1):29–41, 1996.

[13] A. González and R. Pérez. A fuzzy theory refinement algorithm. *International Journal of Approximate Reasoning*, 19(3-4):193–220, 1998.

[14] K. Nozaki, H. Ishibuchi, and H. Tanaka. A simple but powerful heuristic method for generating fuzzy rules from numerical data. *Fuzzy Sets and Systems*, 86(3):251–270, 1997.

[15] P. Thrift. Fuzzy logic synthesis with genetic algorithms. In R.K. Belew and L.B. Booker, editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 509–513, San Mateo, CA, USA, 1991. Morgan Kaufmann Publishers.

[16] L.-X. Wang and J.M. Mendel. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(6):1414–1427, 1992.