Learning Concurrently Granularity, Membership Function Parameters and Rules of Mamdani Fuzzy Rule-based Systems

Michela Antonelli, Pietro Ducange, Beatrice Lazzerini, Francesco Marcelloni

Dipartimento di Ingegneria dell'Informazione: Elettronica, Informatica, Telecomunicazioni
University of Pisa
Via Diotisalvi 2, 56122 Pisa
Email: {michela.antonelli, p.ducange, b.lazzerini, f.marcelloni}@iet.unipi.it

Abstract—*In this paper we tackle the issue of generating Mamdani fuzzy rule-based systems with optimal trade-offs between complexity and accuracy by using a multi-objective genetic algorithm, which concurrently learns rule base, granularity of the input and output partitions and membership function parameters. To this aim, we exploit a chromosome composed of three parts, which codify, respectively, the rule base, and, for each variable, the number of fuzzy sets and the parameters of a piecewise linear transformation of the membership functions. We show the encouraging results obtained on a real world regression problem.*

Keywords— Accuracy-Interpretability Trade-off, Granularity Learning, Mamdani Fuzzy-Rule-Based Systems, Multi-objective Evolutionary Algorithms, Piecewise Linear Transformation.

1 Introduction

In the last years, the problem of finding the right trade-off between interpretability and accuracy of Mamdani fuzzy rule-based systems (MFRBSs) [1] has arisen a growing interest in the fuzzy community [2]. To this aim, Multi-Objective Evolutionary Algorithms (MOEAs) have been extensively used for tuning or learning the data base (DB) and the rule base (RB) of the MFRBSs [3][4]. All these approaches generate approximated Pareto fronts with non-dominated solutions in the interpretability-accuracy space. For example, in [5], authors use a predefined DB and evolve only the RB, while in [6] the membership function (MF) parameters are adapted to a specific context exploiting a predefined RB. Furthermore, in [7] and [8] the tuning of the DB is performed together with a rule selection.

The ideal approach would be to learn concurrently DB and RB. So far, only approximations of this ideal approach have been proposed such as to learn simultaneously the overall RB and, for the DB, only the MF parameters [9][10] or the granularities of the uniform partitions defined on the input and output variables [11][12].

On the other hand, both granularity and MF parameters are a critical factor in MFRBS generation [6][13], since both affect accuracy and interpretability. Indeed, the former fixes an upper bound to the number of rules in the MFRBS and the latter adapts the meaning of the linguistic values to the specific application context.

In this paper, we propose a multi-objective evolutionary approach to generate MFRBSs with different trade-offs between complexity and accuracy. The main novelty of our approach is that the RB, the granularity of the input and output partitions and the MF parameters of each fuzzy set are learnt concurrently during the evolutionary process. To deal with RBs defined on different granularities, we exploit the concept of virtual RBs introduced in [12]. Further, the learning of the MF parameters is performed by using a piecewise linear transformation [14][15], which allows us to obtain a high modelling capability with a limited number of parameters.

Chromosomes are composed of three parts, which codify the RB, and, for each linguistic variable, the number of fuzzy sets and the parameters of the piecewise linear transformation, respectively. In the evolutionary process, we adopt the mating operators proposed in [5] for the fist part of the chromosome, and standard crossover and mutation operators for the second and third parts, respectively.

Our approach has been tested on a real world regression problem, with nine input variables, and has provided Pareto fronts with solutions characterized by better trade-offs between accuracy and complexity than solutions belonging to Pareto fronts generated by learning only the rule base with a fixed DB, and concurrently the rule base and the MF parameters with fixed granularities, respectively.

2 Mamdani Fuzzy Systems

Let $\mathbf{X} = \{X_1, ..., X_f, ..., X_F\}$ be the set of input variables and X_{F+1} be the output variable. Let $U_f\ (f = 1, ..., F+1)$ be the universe of the f^{th} variable. Let $P_f = \left\{A_{f,1}, ..., A_{f,T_f}\right\}$ be a fuzzy partition of T_f fuzzy sets on variable X_f. An MFRBS is composed of M rules expressed as:

$$R_m: \text{IF } X_1 \textbf{ is } A_{1,j_{m,1}} \text{ AND } \ldots \text{ AND } X_F \textbf{ is } A_{F,j_{m,F}} \text{ THEN } X_{F+1} \textbf{ is } A_{F+1,j_{m,F+1}} \quad (m = 1, ..., M) \qquad (1)$$

where $j_{m,f} \in [1, T_f]$ identifies the index of the fuzzy set (among the T_f fuzzy sets of partition P_f), which has been selected for X_f in rule R_m.

To take the "don't care" condition into account [16] a new fuzzy set $A_{f,0}\ (f = 1, ..., F)$ is added to all the F input partitions P_f. This fuzzy set is characterized by a membership function equal to 1 on the overall universe.

The terms $A_{f,0}$ allow generating rules which contain only a

ISBN: 978-989-95079-6-8

subset of the input variables. It follows that $j_{m,f} \in [0,T_f]$, $f = 1,...,F$, and $j_{m,F+1} \in [1,T_{F+1}]$.

An MFRBS can be completely described by the following matrix $J \in \mathbb{N}^{M\times(F+1)}$

$$J = \begin{bmatrix} j_{1,1} & \cdots & j_{1,F} & j_{1,F+1} \\ \cdots & \cdots & \cdots & \cdots \\ j_{m,1} & \cdots & j_{m,F} & j_{m,F+1} \\ \cdots & \cdots & \cdots & \cdots \\ j_{M,1} & \cdots & j_{M,F} & j_{M,F+1} \end{bmatrix}$$

where the generic element (m,f) indicates that fuzzy set $A_{f,j_{m,f}}$ has been selected for variable X_f in rule R_m.

We adopt the product and the weighted average method as AND logical operator and defuzzification method, respectively.

Given a set of *N* input observations $\mathbf{x}_n = [x_{n,1},...,x_{n,F}]$, with $x_{n,f} \in \Re$, and the set of the corresponding outputs $x_{n,F+1} \in \Re$, $n = 1,...,N$, we apply a multi-objective evolutionary algorithm which produces a set of MFRBSs with different trade-offs between accuracy and complexity by learning simultaneously the RB, the granularity of the partitions of each variable and the MF parameters. The choice of the appropriate trade-off between accuracy and complexity depends on the particular application.

3 Granularity and MF Parameter Learning

3.1 Granularity Learning

To determine simultaneously the granularity of partitions and the RB in the evolutionary process is not an easy task since the RB depends on the number of fuzzy sets used to partition the variables. In [12] we have introduced the concept of *virtual RB*. A virtual RB is composed of rules defined by considering the variables partitioned with a user-defined maximum number $T_{\max}$ of fuzzy sets (*virtual partition*). All the mating operators are applied to virtual RBs: the actual granularity is used only in the computation of the fitness. In practice, we generate virtual RBs and assess their quality using each time different "lens" depending on the actual number of fuzzy sets used to partition the single variables. Thus, we do not worry about the actual granularity in applying crossover and mutation operators.

To map the virtual RB defined on variables uniformly partitioned with $T_{\max}$ fuzzy sets into a concrete RB defined on variables uniformly partitioned with T_f fuzzy sets, we adopt a mapping strategy. Let X_f **is** $\hat{A}_{f,h}$, $h \in [0,T_{\max}]$, be a generic fuzzy proposition defined in a rule of the virtual RB. Then, the proposition will be mapped to X_f **is** $\tilde{A}_{f,s}$, with $s \in [0,T_f]$, where $\tilde{A}_{f,s}$ is the fuzzy set more similar to $\hat{A}_{f,h}$ among the T_f fuzzy sets $\hat{A}_{f,j}$ defined on X_f. For the sake of simplicity, we have trivially considered as similarity measure the distance between the centroids of the two fuzzy sets. If there are two fuzzy sets in $\tilde{P}_f = \left\{\tilde{A}_{f,1},...,\tilde{A}_{f,T_f}\right\}$ with centroids at the same distance from the centroid of $\hat{A}_{f,h}$, we choose randomly one of the two fuzzy sets.

Note that different rules of the virtual RB can be mapped to equal rules in the concrete RB. This occurs because distinct fuzzy sets defined on the partitions used in the virtual RB can be mapped to the same fuzzy set defined on the partitions used in the concrete RB. In the case of equal rules, only one of these rules is considered in the concrete RB. The original different rules are, however, maintained in the virtual RB. Indeed, when the virtual RB will be interpreted by using different "lens", all these rules can again be meaningful and contribute to increase the accuracy of the MFRBS. Thus, the concept of virtual RB allows us to explore the search space and concurrently exploiting the optimal solutions achieved during the evolutionary process.

3.2 MF Parameter Learning

We approach the problem of learning the MF parameters by using a piecewise linear transformation [14][15]. The transformation is described in Fig. 1 for a generic variable X_f. In the following, we assume that the interval ranges of the original and transformed variables are identical. Further, we consider triangular fuzzy sets $A_{f,j}$ defined by the tuple ($a_{f,j}, b_{f,j}, c_{f,j}$), where $a_{f,j}$ and $c_{f,j}$ correspond to the left and right extremes of the support of $A_{f,j}$, and $b_{f,j}$ to the core. Finally, given a generic partition $P_f = \left\{A_{f,1},...,A_{f,T_f}\right\}$, we assume that, for $j = 2\ldots T_f - 1$, $b_{f,j} = c_{f,j-1}$ and $b_{f,j} = a_{f,j+1}$, and $a_{f,1} = b_{f,1}$ and $b_{f,T_f} = c_{f,T_f}$.

Before passing the input value x_f to the MFRBS, we apply the transformation $t(x_f)$. Thus, we have that:

$$A_{f,j}\left(x_f\right) = \tilde{A}_{f,j}\left(t\left(x_f\right)\right) = \tilde{A}_{f,j}\left(\tilde{x}_f\right)$$

where $\tilde{A}_{f,j}$ and $A_{f,j}$ are two generic fuzzy sets from the uniform and non-uniform fuzzy partitions, respectively. In those regions where *t* has a high value of the derivative (high slope of the lines), the fuzzy sets $A_{f,j}$ are narrower; otherwise, the fuzzy sets $A_{f,j}$ are wider.

As observed in [14], the transformation *t* must be non-decreasing. Further, the fuzzy sets $A_{f,j}$ of the non-uniform fuzzy partition induced by the transformation *t* are not necessarily of triangular shape. This could bring to fuzzy partitions difficultly interpretable. To preserve the shape of the MFs, we force the change of slopes in *t* to coincide with the cores of the fuzzy sets in the partitions.

Let $b_{f,1},...,b_{f,T_f}$ and $\tilde{b}_{f,1},...,\tilde{b}_{f,T_f}$ be the cores of $A_{f,1},...,A_{f,T_f}$ and $\tilde{A}_{f,1},...,\tilde{A}_{f,T_f}$, respectively. Transformation *t* can be defined as:

$$t\left(x_f\right) = \frac{\tilde{b}_{f,j} - \tilde{b}_{f,j-1}}{b_{f,j} - b_{f,j-1}}\left(x_f - b_{f,j-1}\right) + \tilde{b}_{f,j-1}, \quad b_{f,j-1} \le x_f < b_{f,j},$$

with j=2…T_f.

Once fixed the granularity of the partitions, $\tilde{b}_{f,1},...,\tilde{b}_{f,T_f}$ are fixed and therefore known. Further, $b_{f,1}$ and b_{f,T_f} coincide with the extremes of the universe U_f of X_f. Thus, $t\left(x_f\right)$ depends on T_f –2 parameters, that is, $t\left(x_f;b_{f,2},...,b_{f,T_f-1}\right)$. Once fixed $b_{f,2},...,b_{f,T_f-1}$, the partition $P_f=\left\{A_{f,1},...,A_{f,T_f}\right\}$ can be obtained simply by transforming the three points ($\tilde{a}_{f,j},\tilde{b}_{f,j},\tilde{c}_{f,j}$), which describe the generic fuzzy set $\tilde{A}_{f,j}$, into ($a_{f,j},b_{f,j},c_{f,j}$) applying $t^{-1}\left(\tilde{x}_f\right)$.

We define the piecewise linear transformation on the maximum granularity. When we reduce the granularity, to maintain the original shape of the MFs, we do not apply the piecewise linear transformation to all the points of the universe, but only to the three points, which define the generic triangular MF. In practice, we transform ($\tilde{a}_{f,j},\tilde{b}_{f,j},\tilde{c}_{f,j}$) into ($a_{f,j},b_{f,j},c_{f,j}$) by applying $t^{-1}\left(\tilde{x}_f\right)$. Fig. 2 shows an example of this transformation for granularity 5 by using the piecewise linear transformation in Fig. 1, defined with granularity 7.

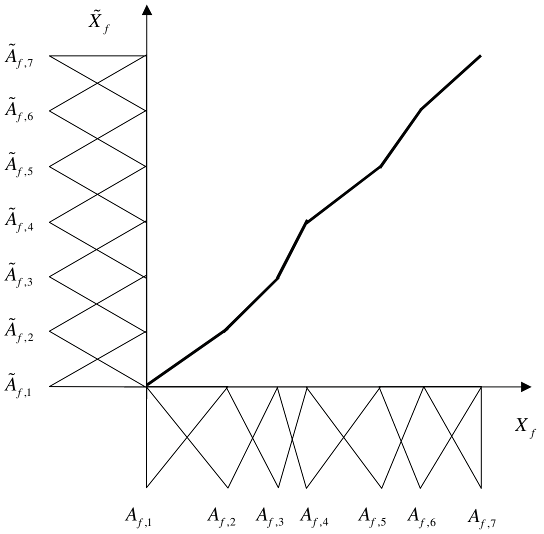

Figure 1: An example of piecewise linear transformation.

4 The Multi-Objective Evolutionary Approach

4.1 Chromosome coding

Each solution is codified by a chromosome C composed of three parts (C_1,C_2,C_3), which define the virtual RB, and the granularities and the piecewise linear transformations of all the variables, respectively. In particular, C_1 encodes the virtual RB by considering that each variable X_f is uniformly partitioned by using $T_{\max}$ fuzzy sets, that is, $P_f=\left\{A_{f,1},...,A_{f,T_{\max}}\right\}$. As described in [5], C_1 is composed of $M\cdot(F+1)$ natural numbers where M is the number of rules currently present in the virtual RB. The RB (defined as *concrete RB*) used to compute the fitness is obtained by means of the mapping strategy using the actual granularities fixed by C_2.

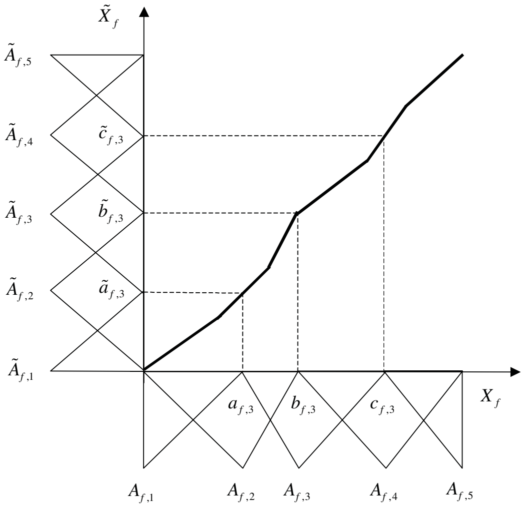

Figure 2: An example of piecewise linear transformation with granularity different from $T_{\max}$.

C_2 is a vector containing $F+1$ natural numbers: the f th element of the vector contains the number $T_f\in[2,T_{\max}]$ of fuzzy sets which partition the linguistic variable X_f. $T_{\max}$ is fixed by the user and is the same for all the linguistic variables.

C_3 is a vector containing $F+1$ vectors of $T_{\max}-2$ real numbers: the f th vector contains the $\left[b_{f,2},...,b_{f,T_{\max}-1}\right]$ points which define the piecewise linear transformation for the linguistic variable X_f. To preclude that the piecewise linear transformation can become decreasing, we force $b_{f,j}$ to vary in $\left[\tilde{b}_{f,j}-\frac{\tilde{b}_{f,j}-\tilde{b}_{f,j-1}}{2},\tilde{b}_{f,j}+\frac{\tilde{b}_{f,j+1}-\tilde{b}_{f,j}}{2}\right]$, $\forall j\in[2,T_{\max}-1]$.

Each chromosome is associated with a bi-dimensional objective vector. The first element of the vector measures the complexity of the MFRBS as the number of propositions used in the antecedents of the rules contained in the concrete RB (the number of antecedents may be different, as explained in the next subsection, between the virtual and concrete RBs). The second element assesses the accuracy as the half of the mean square error (MSE) between the output of the MFRBS and the expected output.

4.2 Genetic operators

In order to generate the offspring populations, we exploit both crossover and mutation. We apply separately the one-point crossover to C_1 and C_2 and the BLX-α crossover, with α = 0.5, to C_3. Let s_1 and s_2 be two selected parent chromosomes. The common gene for C_1 is chosen by extracting randomly a number in [$M_{\min},\rho_{\min}$], where $M_{\min}$ is the minimum number of rules, which must be present in a rule base, and $\rho_{\min}$ is the minimum number of rules in s_1 and s_2. The common gene for C_2 is extracted randomly in $[1,F+1]$.

As regards mutation, we apply two mutation operators for C_1. The first operator adds γ rules to the virtual RB, where γ is randomly chosen in [1, $\gamma_{\max}$]. The upper bound $\gamma_{\max}$ is fixed by the user. If $\gamma + M > M_{\max}$, then $\gamma = M_{\max} - M$. For each rule R_m added to the chromosome, we generate a random number $v \in [1, F]$, which indicates the number of input variables used in the antecedent of the rule. Then, we generate v natural random numbers between 1 and F to determine the input variables which compose the antecedent part of the rule. Finally, for each selected input variable f, we generate a random natural number $j_{m,f}$ between 1 and $T_{\max}$, which determines the fuzzy set $A_{f,j_{m,f}}$ to be used in the antecedent of rule R_m in the virtual RB. To select the consequent fuzzy set $A_{F+1,j_{m,F+1}}$, a random number between 1 and $T_{\max}$ is generated.

The second mutation operator randomly changes δ elements of the matrix J associated with the virtual RB. The number δ is randomly generated in [1, $\delta_{\max}$]. The upper bound $\delta_{\max}$ is fixed by the user. For each element to be modified, a number is randomly generated in [0, $T_{\max}$].

The mutation applied to C_2 randomly chooses a gene $f \in [1, F+1]$ and changes the value of this gene by randomly adding or subtracting 1. If the new value is lower than 2 and larger than $T_{\max}$, then mutation is not applied.

The mutation applied to C_3 first chooses randomly a variable $f \in [1, F+1]$, then extracts a random value $j \in [2, T_{\max} - 1]$ and changes the value of $b_{f,j}$ to a random value in the allowed interval $\left[\tilde{b}_{f,j} - \frac{\tilde{b}_{f,j} - \tilde{b}_{f,j-1}}{2}, \tilde{b}_{f,j} + \frac{\tilde{b}_{f,j+1} - \tilde{b}_{f,j}}{2}\right]$.

We experimentally verified that these mating operators ensure a good balancing between exploration and exploitation, thus allowing the multi-objective evolutionary algorithm described in the next subsection to create good approximations of the Pareto fronts.

4.3 Multi-objective evolutionary algorithm

We adopted the (2+2)M-PAES proposed in [5]. Unlike classical (2+2)PAES, which uses only mutation to generate new candidate solutions, (2+2)M-PAES exploits both crossover and mutation. Further, in (2+2)M-PAES, current solutions are randomly extracted at each iteration rather than maintained until they are not replaced by solutions with particular characteristics.

Fig. 3 shows a pseudo-code which describes the application scheme of the different operators to generate the offspring solutions o_1 and o_2 from the selected parents s_1 and s_2. Note that P_{c1}, P_{c2} and P_{c3} represent the probabilities of applying the crossover operator on C_1, C_2 and C_3, respectively. Similarly, P_{m1}, P_{m2} and P_{m3} represent the probabilities of applying the mutation operator on C_1, C_2 and C_3, respectively, and P_{add} and P_{inc} represent the probabilities of adding rules and of increasing the granularity of the selected variable, respectively. In Fig. 3 we also report the values of the probabilities we used in our experimentations.

At the beginning, we generate two solutions s_1 and s_2. The genes of C_2 in both the solutions are set to the maximum value $T_{\max}$, while the genes of C_1 and C_3 are randomly generated. At each iteration, the application of crossover and mutation operators produces two new candidate solutions from the current solutions s_1 and s_2. These candidate solutions are added to the archive only if they are dominated by no solution contained in the archive; possible solutions in the archive dominated by the candidate solutions are removed. Typically, the size of the archive is fixed at the beginning of the execution of the (2+2)M-PAES. In this case, when the archive is full and a new solution z has to be added to the archive, if z dominates no solution in the archive, then we insert z into the archive and remove the solution (possibly z itself) that belongs to the region with the highest crowding degree [5]. If the region contains more than one solution, then, the solution to be removed is randomly chosen.

```
Pc1 = Pc2 = Pc3 = 0.5;
Pm2 = Pm3 = 0.2;
Padd = 0.55;
Pinc = 0.85;
…
//Generate two new solutions
[s1, s2] = random_selection(archive)
o1 = s1
o2 = s2
if (rand() < Pc1)
   [o1.C1,o2.C1]=crossover_C1 (s1.C1,s2.C1);
   Pm1 = 0.01;
else
   Pm1 = 1;
endif
if (rand() < Pc2)
   [o1.C2,o2.C2]=crossover_C2 (s1.C2,s2.C2);
endif
if (rand() < Pc3)
   [o1.C3,o2.C3]=crossover_C3 (s1.C3,s2.C3);
endif
loop i=1,2
   if (rand() < Pm1)
       if (rand() <Padd)
         oi.C1 = first_mutation_operator();
       else
         oi.C1 =second_mutation_operator();
       endif
   endif
   if (rand() < Pm2)
       if (rand <Pinc)
         oi.C2 = increase_granularity();
       else
         oi.C2 = decrease_granularity();
       endif
   endif
   if (rand() < Pm3)
      oi.C3 = transf_function_mutation();
   endif
endloop
```

Figure 3: Genetic operators application scheme.

5 Experimental results

We tested our approach on a real world regression problem provided by the Bilkent University Function Approximation Repository [17]. The dataset contains the weather data of Ankara, a Turkish town, collected from 01/01/1994 to 28/05/1998. Here, the goal is to predict the mean temperature from the values of nine weather features, namely *max temperature*, *min temperature*, *dewpoint*, *precipitation*, *sea level pressure*, *standard pressure visibility*, *wind speed*, and *max wind speed*. The dataset contains 1609 instances of these features and corresponding mean temperatures. In order to asses the reliability of our approach, we performed a five-fold cross-validation and executed six trials, with different seeds, for each fold. We set $T_{\max} = 5$, $M_{\min} = 5$, $M_{\max} = 30$, $\gamma_{\max} = 5$, $\delta_{\max} = 5$, the maximum number of evaluations to 300000 and the archive size to 64.

To assess the advantages of learning concurrently RB, granularities of partitions and MF parameters, we compared the results achieved by our approach with the ones obtained by applying the (2+2)M-PAES to learn only rules, that is, using only the first part of the chromosome and fixing the number of fuzzy sets in all partitions to $T_{\max}$, and to learn concurrently the RB and the MF parameters, using only the first and third parts of the chromosome with fixed granularities. We denote these two approaches as PAES-RB and PAES-SF, respectively, while the proposed approach is denoted as PAES-SFG.

Figures 4 and 5 show the average Pareto fronts achieved by the three algorithms on the training and test sets, respectively. The average Pareto front consists of the average values of the twenty most accurate solutions of each of the thirty Pareto front approximations (at least twenty solutions were always generated in all the trials for both datasets). In practice, the average Pareto fronts are obtained as follows. First, the solutions in the Pareto front approximations produced in each of the thirty trials are ordered for increasing MSE values. Then, only the twenty solutions with the lowest MSEs are retained for each Pareto front approximation. Finally, the average values, on the thirty Pareto front approximations, of complexity and MSE for these twenty solutions are computed. The choice of considering only the twenty solutions with the lowest MSEs was motivated by the observation that the other solutions are in general characterized by quite high MSEs which make these solutions impractical.

We note that the average Pareto front generated by PAES-SFG outperforms the other two average Pareto fronts both on the training and on the test sets. This proves that learning concurrently the RB, the granularities of the input and output partitions and the membership parameters allows generating FRBSs with better trade-offs between accuracy and complexity than the other two MOEAs used for comparison.

In Table 1 we show the average results corresponding to three representative points of the average Pareto fronts: the first (the most accurate), the median and the last (the least accurate) point. We refer to these average values as First, Median and Last, respectively. It is interesting to observe that the average MSEs of the most accurate solutions generated by PAES-SFG are 47% and 68%, and 33% and 56% lower than the corresponding solutions in the Pareto fronts generated by PAES-SF and PAES-RB with comparable complexity on the training and test sets, respectively.

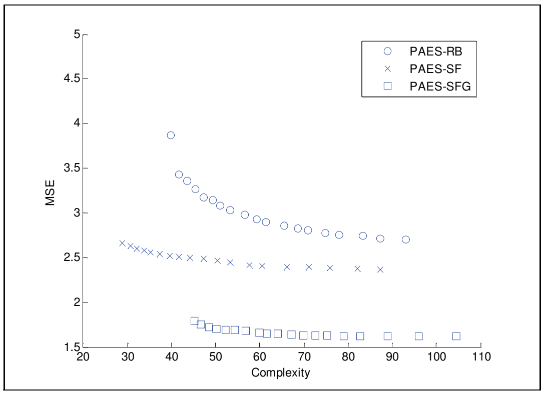

Figure 4. Average Pareto fronts on training set.

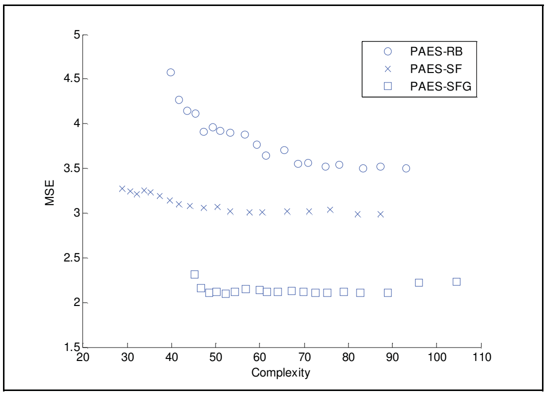

Figure 5. Average Pareto fronts on test set.

Table 1. Results obtained on three representative points of the average Pareto fronts

	MSE_{Tr}	MSE_{Ts}	Complexity	# Rules
	PAES-RB			
First	2.70±1.27	3.50±1.90	93.20±19.99	28.30±2.65
Median	2.93±1.40	3.77±2.17	59.33±17.02	22.80±4.48
Last	3.87±3.08	4.58±3.43	39.80±16.36	17.67±5.00
	PAES-SF			
First	2.36±1.60	2.99±2.18	87.23±17.33	24.20±4.24
Median	2.49±1.71	3.06±2.21	47.30±14.11	17.23±4.08
Last	2.66±1.88	3.28±2.51	28.80±11.10	12.97±3.69
	PAES-SFG			
First	1.61±0.34	2.24±1.11	104.53±27.86	25.77±4.59
Median	1.64±0.35	2.12±0.62	64.13±21.15	20.97±4.81
Last	1.79±0.62	2.31±1.11	45.10±15.76	17.33±4.19

Finally, Figures 6 and 7 show an example of complete Pareto fronts generated by the three algorithms in a single trial on the training and test sets, respectively. We have plotted only the solutions of the final archive with the MSEs lower than 5, since the other solutions in the archive are characterized by quite high MSEs and tend to visually flatten the plot. On the other hand, these solutions, though characterized by very low complexity, are not very interesting due to the high MSEs. We observe that the approximated Pareto fronts obtained by PAES-SFG are wide and quite dense. Further, except for some solutions in the low complexity zone, all the solutions in the Pareto front generated by PAES-SFG dominate the solutions at the same complexity of the other two fronts.

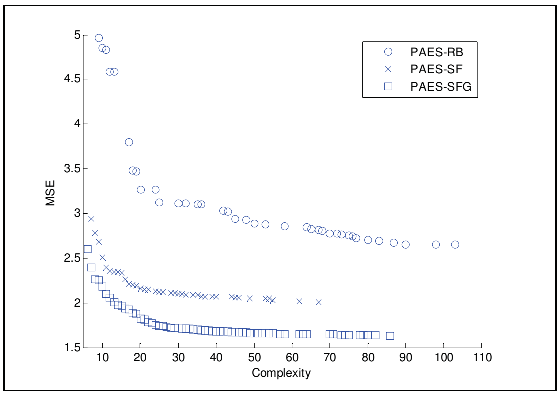

Figure 6. Examples of complete Pareto fronts on training set.

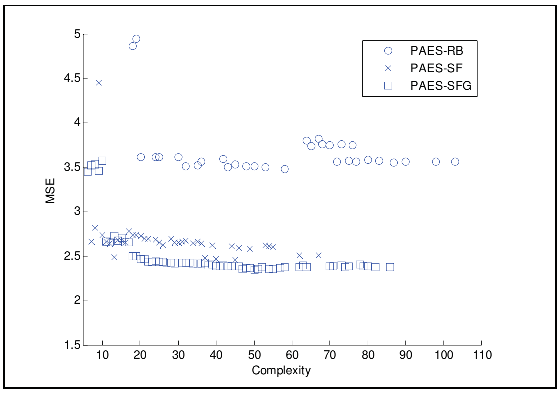

Figure 7. Examples of complete Pareto fronts on test set.

6 Conclusions

In this paper we have proposed a multi-objective evolutionary algorithm to generate a set of Mamdani rule-based fuzzy systems with optimal trade-offs between accuracy and complexity. We have exploited a modified version of the well known (2+2)PAES where the chromosome consists of three parts which codify the RB, and, for each variable, the number of fuzzy sets and the parameters of a piecewise linear transformation of the membership functions, respectively. To manage the dependence between granularity, and rule and membership parameters definition, we have adopted the following solution: the RB coded in the chromosome and the piecewise linear transformation are always defined on linguistic variables partitioned with a fixed maximum number of fuzzy sets. Only when accuracy and complexity have to be evaluated, the RB is actualized by using the real number of fuzzy sets determined by the second part of the chromosome and the DB by computing the corresponding piecewise linear transformation of the fuzzy sets. This approach has proved to be very efficient and effective, allowing both a good exploitation of the solutions and an accurate exploration of the search space.

The algorithm has been tested on a real world regression problem and has provided Pareto fronts with solutions characterized by better trade-offs between accuracy and complexity than solutions belonging to Pareto fronts generated by learning only the RB with a fixed BD, and concurrently the RB and the MF parameters with fixed granularities, respectively.

References

[1] E. H. Mamdani and S. Assilian, An experiment in linguistic synthesis with a fuzzy logic controller, International Journal on Man-Machine Studies, 7: 1-13, 1975.

[2] J. Casillas et al. (eds.), Interpretability issues in fuzzy modeling, Springer, 2003.

[3] F. Herrera, Genetic fuzzy systems: taxonomy, current research trends and prospects, Evolutionary Intelligence, 1(1): 27-46, 2008.

[4] H. Ishibuchi, Multiobjective genetic fuzzy systems: review and future research direction, in Proc. of FUZZ-IEEE 2007 Int. Conf. on Fuzzy Systems, London, 2007.

[5] M. Cococcioni et al., A Pareto-based multi-objective evolutionary approach to the identification of Mamdani fuzzy systems, Soft Computing, 11(11): 1013-1031, 2007.

[6] A. Botta et al., Context adaptation of fuzzy systems through a multi-objective evolutionary approach based on a novel interpretability index, Soft Computing, 13(5): 437-449, 2009.

[7] R. Alcalá et al., A multi-objective genetic algorithm for tuning and rule selection to obtain accurate and compact linguistic fuzzy rule-based systems, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 15(5): 539-557, 2007.

[8] M. J. Gacto, R. Alcalá, and F. Herrera, Adaptation and application of multi-objective evolutionary algorithms for rule reduction and parameter tuning of fuzzy rule-based systems, Soft Computing, 13(5): 419-436, 2009.

[9] P. Ducange et al., Knowledge Base Learning of Linguistic Fuzzy Rule-Based Systems in a Multi-objective Evolutionary Framework, Lecture Notes on Computer Science, Hybrid Intelligent Systems, pp. 747-754, 2008.

[10] P. Pulkkinen and H. Koivisto, Fuzzy classifier identification using decision tree and multiobjective evolutionary algorithms, International Journal of Approximate Reasoning, 48: 526-543, 2008.

[11] H. Ishibuchi and Y. Nojima, Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning, International Journal of Approximate Reasoning, 44(1): 4-31, 2007.

[12] M. Antonelli et al., A multi-objective genetic approach to concurrently learn partition granularity and rule bases of Mamdani fuzzy systems, in Proc. of the 8th International Conference on Hybrid Intelligent Systems (HIS-2008), pp. 278–283, Barcelona, Spain, 2008.

[13] O. Cordón, F. Herrera, and P. Villar, Analysis and guidelines to obtain a good uniform fuzzy partition granularity for fuzzy rule-based systems using simulated annealing, International Journal of Approximate Reasoning 25(3):187-215, 2000.

[14] F. Klawonn, Reducing the number of parameters of a fuzzy system using scaling functions. Soft Computing, 10(9): 749-756, 2006.

[15] W. Pedrycz and F. Gomide, Fuzzy Systems Engineering: Toward Human-Centric Computing, Wiley-IEEE Press, 2007.

[16] H. Ishibuchi et al., Selecting fuzzy if-then rules for classification problems using genetic algorithms, IEEE Transactions on Fuzzy Systems, 3(3): 260-270, 1995.

[17] H. Altay Guvenir and I. Uysal, Bilkent University Function Approximation Repository, http://funapp.cs.bilkent.edu.tr, 2000.