# Embedded Genetic Learning of Highly Interpretable Fuzzy Partitions

Jorge Casillas
CITIC-UGR (Research Center on Communication and Information Technology),
Dept. Computer Science and Artificial Intelligence, University of Granada, E-18071, Granada, Spain
E-mail: casillas@decsai.ugr.es, URL: http://decsai.ugr.es/∼casillas

***Abstract***— *A new algorithm is proposed to learn fuzzy partitions with a high interpretability degree. The set of input variables, the number of linguistic terms per variable, and the type (triangular or trapezoidal) and parameters of the membership functions are learnt by means of a meta-algorithm that uses a simple learning method to generate a fuzzy rule set from the derived fuzzy partitions. Interpretability constrains and powerful genetic operators are considered. A multi-objective optimization approach is used to generate different interpretability-accuracy tradeoffs. The algorithm is tested in a set of real-world regression problems with successful results compared to other methods.*

***Keywords***— Fuzzy rule-based systems, fuzzy partitions, granularity, interpretability, multi-objective genetic algorithms.

## 1 Introduction

In the last few years the interpretability (i.e. the capability of the fuzzy model to express the behavior of the real system in an understandable way) has received more and more attention in the fuzzy community. Indeed, this property of the fuzzy rule-based systems represents one of the main competitive advantages compared to other system modeling techniques.

In the literature we may find many alternatives to improve the interpretability such as reducing the number of fuzzy rules [1], selecting a subset of input variables [2], or using more compact fuzzy rule expressions [3]. Within these options, one of the more important approaches involves to learn the optimal number of linguistic terms per variable [4, 5, 6]. Indeed, to use a reduced number of linguistic terms is crucial to understand the meaning of the variables and directly influences on the fuzzy rule set size.

In this paper we propose a new algorithm to learn the number of input variables, the number of linguistic terms per variable, and the types of membership functions (triangular or trapezoidal) and their parameters with the aim of generating highly interpretable fuzzy partitions. The learning is performed by a wrapper-based embedded process where a meta-algorithm generates different fuzzy partitions and a simple learning method derives fuzzy rule sets from them.

The proposed algorithm, called EGLFP, has some interesting characteristics that make it very competitive: it uses strong fuzzy partition and includes distinguishability constrains for a better interpretability, and it uses variable-length coding schemes, powerful original crossover and mutation operators, and multi-objective optimization for a better search process.

The paper is organized as follows. Section 2 describes the proposed algorithm. Section 3 shows the results obtained in a set of real-world problems compared with other fuzzy rule learning methods. Finally, Sect. 4 concludes and suggests some further works.

## 2 EGLFP Algorithm

EGLFP is a multi-objective genetic algorithm with a generational evolutionary approach. A crossover operator that recombines membership function parameters, a first mutation operator that tunes these parameters, and a latter mutation operator that changes the fuzzy partition granularity (i.e., the number of linguistic terms) per variable are used in EGLFP. The multi-objective approach is based on the well-known NSGA-II algorithm [7]. The following sections detail the different components of EGLFP.

### *2.1 Coding Scheme*

For the sake of a good interpretability and in order to reduce the search space tackled by the genetic algorithm, we propose the use of strong fuzzy partitions. Each gene ($g$) is a 2-tuple that contains the information related to a linguistic term of a specific variable. It consists of two real-valued fields ($g_{left}$ and $g_{right}$) that encode the left and right extremes of the core of the associated fuzzy set (i.e., the semantic rule of the linguistic term) normalized to the interval $[0, 1]$. Figure 1 illustrates an example of gene's coding.

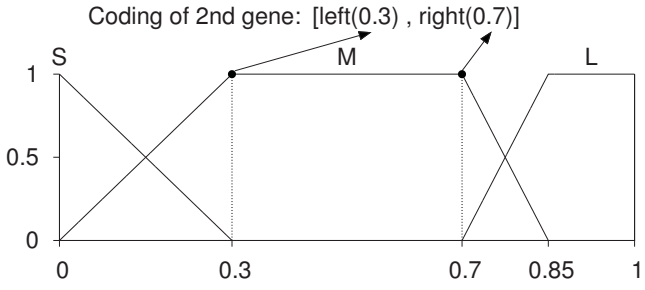


Figure 1: Example of a gene coding. Each gene, which encodes a linguistic term, consists of two fields (*left* and *right*) to encode the extremes of the fuzzy set's core (normalized to $[0, 1]$). Note that, since strong fuzzy partitions are used, the extremes of the core of each fuzzy set coincide with the extremes of the support of the adjacent fuzzy sets

A chromosome is a variable-length string (the length will depend on the number of linguistic terms used in each variable) of these genes that will encode the complete definition of the fuzzy partitions of all the input and output variables:

$$C = \bigcup_{v=1}^{n+m} \bigcup_{i=1}^{l_v} (g^v_{i,left}, g^v_{i,right}) \quad (1)$$

with $n$ being the number of input variables, $m$ the number of output variables, and $l_v$ the number of linguistic terms used in the variable $v$.

ISBN: 978-989-95079-6-8

It is worth mentioning this coding scheme allows the absence of some variables by using only one linguistic term. Thus, if the variable $v$ is not used, there will be only one gene associated to it with the value $(g^v_{1,left}, g^v_{1,right}) = (0, 1)$.

### 2.2 Initialization

The initialization process generates the first pool. Among the generated individuals, one of them will encode uniformly distributed triangular-shaped strong fuzzy partitions with the maximum number of linguistic terms allowed for each variable ($maxLV_v$). The remaining pool is filled up at random with individuals of two types.

One the one hand, some individuals (approximately a half of the pool) will preserve in some degree the above mentioned uniformly distributed fuzzy partitions but with a lesser number of linguistic terms. To do so, a fusion operation (which is described below in Sect. 2.5.1) on neighboring fuzzy sets is applied over the original fuzzy partitions. On the other hand, other individuals (approximately the another half of the pool) will be generated completely at random. To do so, firstly the number of linguistic terms is randomly assigned to each variable, then each fuzzy set adopts a triangular or trapezoidal shape at random and, finally, random values are assigned to the extremes of the fuzzy sets' cores.

### 2.3 Crossover

The crossover generates two individuals that inherit the membership function parameter definition given by two parents, even when these parents hold fuzzy partitions with different number of linguistic terms. This feature, the fact of crossing the parameters of fuzzy partitions with different granularities, provides the proposed algorithm with a powerful search process and is one of the original contributions of the paper.

The proposed crossover is a kind of parent-centric crossover operator [8] where each son is generated from a parent that plays the main role (*dominant parent*) and the another parent playing the secondary role (*recessive parent*). Thus, the son $S_1$ is generated focused on the parent $C_1$, but using the parent $C_2$ to add diversity; analogously with the son $S_2$.

Since the considered chromosomes have a variable length, a process is followed to select a gene of the recessive parent to be crossed with each gene of the dominant parent. This match is based on the distance (see eq. 5 below) where, given a fuzzy set of the dominant parent to be crossed, the closest fuzzy set in the recessive parent is chosen. Furthermore, to maintain the interpretability and well definition of the fuzzy partitions, an interval variation is fixed every time a gene is going to be crossed. An example is depicted in Fig. 2(a).

Once the dominant and recessive genes are fixed and the interval variation is determined, an original real-coding crossover operator is used. We have named this operator as the Constrained Parent-Centric Crossover (CPCX). Contrary to other previously proposed parent-centric crossover operators [8], CPCX is designed to generate the son's gene constrained to a given interval, which is crucial in our EGLFP algorithm to ensure well-defined and distinguishable fuzzy partitions. The operator is described in Algorithm 1 and an example is given in Figure 2(b).

**Algorithm 1**: Constrained Parent-Centric Crossover (CPCX) operator

**Input**: $(g, h, min, max, m, M)$ with $g$ being the dominant value and $h$ the recessive value to be crossed, $[min, max]$ the variation interval of the dominant value, and $[m, M]$ the definition interval of the variable
**Output**: New value $g'$
**begin**
  **if** $g = h$ **then**
    $g' \leftarrow g$
  **else if** $h < g$ **then**
    $\alpha \leftarrow (g - h)/(g - m)$
    $l \leftarrow g - \alpha(g - min)$
    $r \leftarrow g$
    $g' \leftarrow U[l, r]$ */* Random number */*
  **else**
    $\alpha \leftarrow (h - g)/(M - g)$
    $l \leftarrow g$
    $r \leftarrow g + \alpha(max - g)$
    $g' \leftarrow U[l, r]$ */* Random number */*
**end**

### 2.4 Parameter Mutation

The parameter mutation changes the real-valued membership function parameter values of the input and output variables encoded in the chromosome. To do that, an original real-coding mutation operator is proposed in this paper. Firstly, a random process is followed to select the field of the gene to be mutated. When the gene to be mutated encodes a triangular fuzzy set, both *left* and *right* fields are mutated with the same value to preserve the original type.

Then, given a gene's field $g^v_{i,e}$ to be mutated, a variation interval around its value is defined with the aim of avoiding lack of distinguishability—i.e., two fuzzy sets very close—and other kinds of deformities that would decrease the interpretability degree. In this way, the following equations are used—Fig. 3(a) shows an example:

$$min_{g^v_{i,e}} = \begin{cases} min_v & \text{if } i = 1 \\ g^v_{i-1,right} + \delta & \text{if } e = \textit{left} \text{ or } g^v_{i,left} = g^v_{i,right} \\ g^v_{i,left} & \text{if } e = \textit{right} \text{ and } g^v_{i,left} \neq g^v_{i,right} \end{cases} \quad (2)$$

$$max_{g^v_{i,e}} = \begin{cases} max_v & \text{if } i = l_v \\ g^v_{i+1,left} - \delta & \text{if } e = \textit{right} \text{ or } g^v_{i,left} = g^v_{i,right} \\ g^v_{i,right} & \text{if } e = \textit{left} \text{ and } g^v_{i,left} \neq g^v_{i,right} \end{cases} \quad (3)$$

with $\delta = 1/(2 \cdot l_v)$ being the allowed minimum distance between the extremes of the cores of the fuzzy sets for the sake of a good distinguishability degree and $l_v$ the number of linguistic terms used in the variable $v$.

Finally, an original real-coding mutation operator is applied on $g^v_{i,e}$ constrained to the interval $[min_{g^v_{i,e}}, max_{g^v_{i,e}}]$. We have named this operator as the Constrained Asymmetric Mutation (CAM), which is described in Algorithm 2. Figure 3(b) shows an example of the resulting asymmetric probability density function used to mutate a value.

ISBN: 978-989-95079-6-8

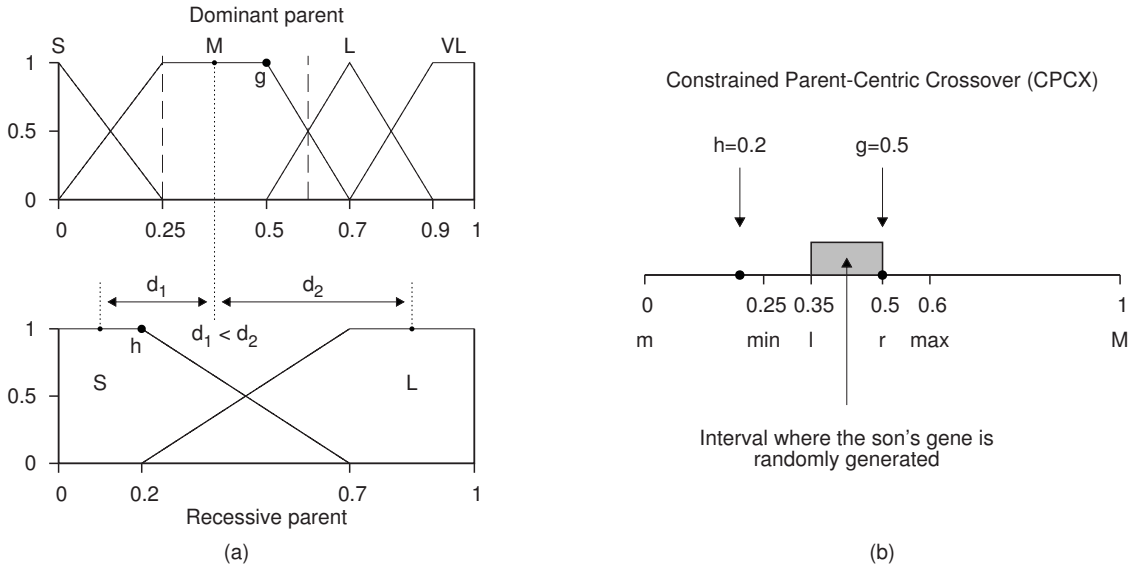


Figure 2: Example of crossover operation: (a) Given the gene $g$ (which encodes the right extreme of the core of the fuzzy set labeled as $S\_M$) of the dominant parent to be crossed, the allowed interval variation $[0.25, 0.6]$ is calculated (dashed lines) and the gene $h$ that encodes the right extreme of the closest fuzzy set in the recessive parent ($VS\_S\_M$) is selected; (b) CPCX operator applied on the dominant gene $g = 0.5$, the recessive gene $h = 0.2$, the allowed interval variation $[min, max] = [0.25, 0.6]$, and the definition interval $[m, M] = [0, 1]$; a random value is generated in the interval $[l, r] = [0.35, 0.5]$ ($\alpha = 0.6$)

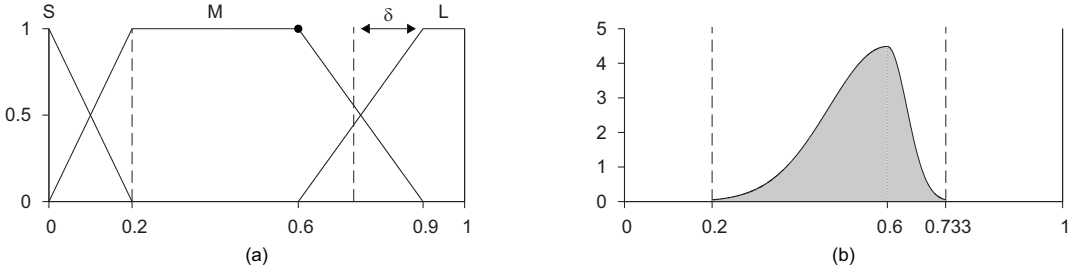


Figure 3: Example of parameter mutation operation: (a) the right extreme of the core of the fuzzy set $M$ is selected to be mutated; then, the interval variation is defined by eq. 2 and 3 with $\delta = 0.167$ since there are three linguistic terms ($l_v = 3$); (b) asymmetric probability density function used in the CAM operator to mutate the right extreme of $M$'s core constrained to the given interval $[0.2, 0.733]$

### *2.5 Granularity Mutation*

The granularity mutation changes the number of linguistic terms in both input and output variables. To do that, it can *fuse* (merge) two neighboring fuzzy sets (thus decreasing in one the number of linguistic terms of the corresponding variable) or *fission* (split) a fuzzy set in two parts (thus increasing in one the number of linguistic terms of the corresponding variable). When fusion is applied on a variable with two linguistic terms, it involves removing the variable from the fuzzy model.

There are some constraints to apply fusion and fission. On the one hand, if fusion is applied on an input variable, this variable must have at least two linguistic terms and there must be more than one input variable with at least two linguistic terms (in order to avoid a fuzzy system without input variables after fusion). When fusion is applied on an output variable, this variable must have at least three linguistic terms since output variable removal is not allowed.

On the other hand, fission on variables with the maximum number of linguistic terms is not allowed. A second constraint is considered to avoid lack of distinguishability after fission. Indeed, if the core width of the fuzzy set associated to the gene to be fissioned is lower than a value inversely proportional to the number of linguistic terms of the corresponding variable, the core of the two fuzzy sets resulting from fission will be too close and, therefore, fission is not allowed. It involves that triangular fuzzy sets can not be fissioned since its core width is zero.

The two following subsections explain in detail how fusion and fission operate.

#### *2.5.1 Fusion (merge) operation*

Given a gene $g_i^v$ to be fused, let $h$ be the closest gene to $g_i^v$:

$$h = \begin{cases} g_{i-1}^v & \text{if } i = l_v \\ g_{i+1}^v & \text{if } i = 1 \\ g_{i-1}^v & \text{if } d(g_i^v, g_{i-1}^v) \leq d(g_i^v, g_{i+1}^v) \\ g_{i+1}^v & \text{otherwise} \end{cases} \quad (4)$$

with

$$d(g, h) = |(g_{left} + g_{right})/2 - (h_{left} + h_{right})/2| \quad (5)$$

To fuse $g_i^v$ and $h$, the following steps are done:

- Firstly, if $g_{i,left}^v < h_{left}$ then $g_{i,right}^v \leftarrow h_{right}$; otherwise, $g_{i,left}^v \leftarrow h_{left}$.

ISBN: 978-989-95079-6-8

**Algorithm 2**: Constrained Asymmetric Mutation (CAM) operator

**Input**: $(x, min, max)$ with $x$ being the value to be mutated and $[min, max]$ its variation interval
**Output**: Mutated value $x'$
**begin**
  $p_{left} \leftarrow (x - min)/(max - min)$
  **if** $U[0,1] < p_{left}$ **then**
    $\sigma \leftarrow (x - min)/3$
    $x' \leftarrow N[x, \sigma]$ */* Normal random */*
    **if** $x' > (2x - min)$ **then** $x' \leftarrow 2x - min$
    **else if** $x' < min$ **then** $x' \leftarrow min$
    **if** $x' > x$ **then** $x' \leftarrow x - (x' - x)$
  **else**
    $\sigma \leftarrow (max - x)/3$
    $x' \leftarrow N[x, \sigma]$ */* Normal random */*
    **if** $x' > max$ **then** $x' \leftarrow max$
    **else if** $x' < (2x - max)$ **then** $x' \leftarrow 2x - max$
    **if** $x' < x$ **then** $x' \leftarrow x + (x - x')$
**end**

- Secondly, gene $h$ is removed from the chromosome and the number of linguistic terms is decreased by one ($l_v \leftarrow l_v - 1$).

#### 2.5.2 Fission (split) operation

The fission operation splits a trapezoidal fuzzy set into two triangular ones. Thus, given a gene $g_i^v$ to be fissioned, the following steps are done:

- A new gene, $h$, is inserted at the right of $g_i^v$ ($h$ will be the new $g_{i+1}^v$) with $h_{left} \leftarrow g_{i,right}^v$ and $h_{right} \leftarrow g_{i,right}^v$. Set $g_{i,right}^v \leftarrow g_{i,left}^v$.
- The number of linguistic terms is increased by one ($l_v \leftarrow l_v + 1$).

### 2.6 Embedded Genetic Learning Approach

Every time a new individual (which encodes the fuzzy partition of each variable) is evaluated, a fuzzy rule set is firstly derived and then the complete fuzzy model (the fuzzy partitions plus the fuzzy rule set) is analyzed. To design the fuzzy rule set an efficient *ad hoc* data-driven method, the well-known Wang-Mendel (WM) algorithm [9] in this paper, is used. Therefore, a wrapper-based embedded genetic learning process is followed [4].

### 2.7 Inference Mechanism

We consider FITA (first inference, then aggregate) approach, the Max-Min inference scheme (i.e., T-conorm of maximum as aggregation and T-norm of minimum as relational operator), the T-norm of minimum as conjunction, and center-of-gravity as defuzzification. Moreover, the mean of the output domain is returned when no rules are fired for the given input data (this fact may only occur with test data since the algorithm ensures complete fuzzy rule set regarding training data).

### 2.8 Objective Functions

The multi-objective optimization performed in EGLFP is based on three objective functions to be minimized: $O_1$ assesses the error of the system, $O_2$ the complexity of the derived fuzzy rule set, and $O_3$ the complexity of the learnt fuzzy partitions. Thus, $O_1$ is focused on improving the accuracy of the fuzzy model while $O_2$ and $O_3$ improve its interpretability.

#### 2.8.1 $O_1$, accuracy objective

The root mean squared error (RMSE) is used to compute the accuracy of the learnt fuzzy model $S$:

$$O_1(S) = RMSE(S) = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(F_S(\mathbf{x}_k) - \mathbf{y}_k)^2} \quad (6)$$

#### 2.8.2 $O_2$, rule set complexity objective

This second objective simply involves the final number of fuzzy rules obtained after applying the WM algorithm on the decoded fuzzy partitions as explained in Sect. 2.6:

$$O_2(S) = r(S) \quad (7)$$

#### 2.8.3 $O_3$, fuzzy partition complexity objective

The third objective is the sum of the linguistic terms used in input and output variables:

$$O_3(S) = \sum_{v=1}^{n+m} L_v \quad \text{with} \quad L_v = \begin{cases} l_v & \text{if } l_v > 1 \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

The use of $L_v$ is to not computing the removed input variables, where the number of linguistic terms is $l_v = 1$.

Since the fuzzy partition complexity has a direct influence on the fuzzy rule set complexity, it is expected that a low value of $O_3$ will imply a low value of $O_2$. Therefore, $O_2$ and $O_3$ are correlated in some degree. To improve the interpretability, $O_2$ seems to be more influent than $O_3$. However, we have decided to also use the objective $O_3$ with the aim of polishing good solutions by reducing the number of linguistic terms as much as possible.

This correlation will involve that, in the practice, the Pareto front shape with three objectives will be similar to the one with the two former ones and therefore, the inclusion of the third objective will not significantly degrade the multi-objective optimization quality.

### 2.9 Multi-objective Approach

A generational approach with the multi-objective elitist replacement strategy of NSGA-II [7] is used. Crowding distance in the objective function space is considered. Binary tournament selection based on the nondomination rank (or the crowding distance when both solutions belong to the same front) is applied. Instead fixing initial intervals for the objectives (which is not easy with the considered objectives) to compute the crowding distance, it is normalized for each objective according to the extreme values of the solutions contained in the analyzed front.

## 3 Experimental Analysis

This section includes the obtained results of the proposed EGLFP algorithm in six real-world regression problems where all the input and output variables are real-valued, and compares them with other fuzzy model learning methods.

ISBN: 978-989-95079-6-8

### 3.1 Problems and Learning Methods

We have considered the following regression problems: *Ele1* (the data set and partitions used in this paper are available at the author's website [1]), *Laser* (available at KEEL website [2]), *Ele2* [1], *DEE* [2], *Concrete* (obtained from the UCI Repository [3]), and *Comp-activ* (obtained from L. Torgo's website[4]).

Table 1 collects the main features of each problem, where *#InputVar* stands for the number of input variables, *#Exam* for the total number of examples, and $maxLV_v$ for the initial number of uniformly distributed triangular-shaped linguistic terms considered for each input variable in the experimental analysis. The output variable is always initially provided with seven uniformly distributed triangular-shaped linguistic terms.

Table 1: Data sets considered in the experimental analysis

| Problem | #InputVar | #Exam | maxLV$_v$ |
|---|---|---|---|
| Ele1 | 2 | 495 | 7 |
| Laser | 4 | 993 | 5 |
| Ele2 | 4 | 1066 | 5 |
| DEE | 6 | 365 | 5 |
| Concrete | 8 | 1030 | 5 |
| Comp-activ | 21 | 8192 | 3 |

The experiments shown in this paper have been performed with a 5-fold cross validation. Thus, the data set is divided into five subsets of (approximately) equal size. The algorithm is then applied five times to each problem, each time leaving out one of the subsets from training, but using only the omitted subset to compute the test error.

We have considered two learning methods for comparison (both of them use the same inference engine described in Sect. 2.7 for our proposal). The *Wang and Mendel (WM)* [9] algorithm is a simple learning method that, in spite of not obtaining accurate results, is a traditional reference in the research community; the algorithm has been implemented by us. The *Cordón, Herrera, and Villar (CHV)* [4] algorithm is a competitive genetic fuzzy system that provides a learning flexibility similar to our EGLFP algorithm since it learns both the number of linguistic terms per variable and the membership function parameters. As in our case, this genetic learning process is also performed by wrapping the WM algorithm. We have used the original algorithm implementation provided by the authors.

Our algorithm has been run with the following parameter values: 1000 iterations, $p = 30$ as population size, $p_c = 0.7$ as crossover probability, and $p_{pm} = p_{gm} = 0.2$ as parameter and granularity mutation probability per chromosome, respectively. We have not performed any previous analysis to fix these values, so better results may probably be obtained by tuning them though we have informally noticed our algorithm is not specially sensitive to any parameter. The same parameter values are also used in the CHV algorithm in addition to $a = 0.35$ (for MMA crossover), $b = 5$ (for non-uniform mutation), and $\alpha = 0.2$ (weight of the number of rules in the fitness function).

### 3.2 Analysis

Table 2 collects the obtained results according to different quality measures such as approximation error, number of rules, number of linguistic terms, number of input variables, and fitness values. Average values of the five data partitions for each problem are reported.

We have included two versions of our algorithm in the comparative. The first one (EGLFP-1) is guided by a single fitness function in the same way as done in CHV [4] for the sake of a fair comparison. This function is a normalized aggregation of the error and the number of rules as follows:

$$f(S) = 0.8 \cdot MSE(S) + 0.2 \cdot \frac{MSE(S_{WM})}{r(S_{WM})} \cdot r(S) \quad (9)$$

with $S_{WM}$ being the fuzzy model generated by the WM algorithm using uniformly-distributed triangular-shaped strong fuzzy partitions with the maximum number of linguistic terms allowed per variable ($maxLV_v$) and $MSE(S) = RMSE(S)^2$.

The second version (EGLFP-3) is guided by the above three-objectives multicriteria approach. Since in this case several solutions are returned in each run, we show five representative solutions from the final Pareto-optimal set. They are computed by sorting in ascending order the solutions according to the training RMSE and getting the 1 (i.e. best error), 25, 50 (i.e. mean), 75, and 100 (i.e. worst error) percentiles.

From the obtained results we can observe that our algorithm EGLFP-1 clearly overcomes, according to the fitness values (column Fit.), the CHV algorithm in all the considered problems. Indeed, our method generates more accurate fuzzy models in five cases. As regards the interpretability, it is not clear between EGLFP-1 and CHV which method obtains simpler models in terms of number of rules and number of linguistic terms since it depends on the problem. However, our method generates fuzzy models with more easily distinguishable fuzzy partitions thanks to the constraints tackled by the proposed genetic operators. Furthermore, CHV needs, in order to achieve the obtained accuracy values, to use badly formed fuzzy partitions where the extreme fuzzy sets of every variable may not hold normality within the corresponding domain.

If we analyze the results obtained by EGLFP-3 we can see that the process is able to generate a wide range of solutions with different accuracy-interpretability tradeoffs. It worths noticing that the most accurate solution of EGLFP-3 overcomes EGLFP-1 in both accuracy and interpretability in several problems. This fact is due to the niche-based search process caused by the use of the multi-objective approach, which leads the algorithm to a better exploration of the search space. It is also interesting to highlight the significant improvements in interpretability compared with the solutions provided by the WM algorithm. Very simple fuzzy models with a low number of variables, linguistic terms, and fuzzy rules are obtained while preserving a good accuracy.

## 4 Conclusion and Further Work

We have proposed a competitive genetic algorithm to simultaneously learn many components of a fuzzy model such as the

[1] J. Casillas. FMLib: fuzzy modeling library. http://decsai.ugr.es/~casillas/FMLib/

[2] KEEL: Knowledge extraction based on evolutionary learning. http://www.keel.es

[3] UC Irvine Machine Learning Repository. http://archive.ics.uci.edu/ml/

[4] L. Torgo. Collection of regression datasets. http://www.liacc.up.pt/~ltorgo/Regression/DataSets.html

ISBN: 978-989-95079-6-8

Table 2: Obtained results with $\mathbf{E}_{tra/tst}$ standing for the RMSE on the training/test data set ($O_1$, eq. 6), **#R** the number of fuzzy rules ($O_2$, eq. 7), **#L** the sum of the number of linguistic terms of each variable ($O_3$, eq. 8), **#I** the number of input variables, and **Fit.** the fitness (eq. 9)

| Method | $\mathbf{E}_{tra}$ | $\mathbf{E}_{tst}$ | #R | #L | #I | Fit. |
|---|---|---|---|---|---|---|
| **Ele1** | | | | | | |
| WM | 650.743 | 674.910 | 22.0 | 21.0 | 2.0 | |
| CHV | 591.268 | 627.132 | 6.8 | 15.8 | 2.0 | 305924 |
| EGLFP-1 | 558.793 | 618.857 | 12.8 | 15.0 | 2.0 | 298660 |
| | 560.346 | 638.527 | 17.0 | 16.0 | 2.0 | 315981 |
| | 577.568 | 632.652 | 11.1 | 12.9 | 2.0 | |
| EGLFP-3 | 588.422 | 618.726 | 7.7 | 10.5 | 2.0 | |
| | 643.637 | 651.702 | 4.2 | 6.5 | 2.0 | |
| | 1017.088 | 988.744 | 1.8 | 4.0 | 1.0 | |
| **Laser** | | | | | | |
| WM | 15.994 | 16.521 | 58.4 | 27.0 | 4.0 | |
| CHV | 8.791 | 10.602 | 58.8 | 25.0 | 4.0 | 97.694 |
| EGLFP-1 | 8.509 | 9.750 | 33.6 | 18.6 | 4.0 | 87.611 |
| | 7.955 | 9.050 | 30.2 | 18.8 | 4.0 | 77.168 |
| | 9.570 | 11.099 | 17.8 | 14.8 | 3.8 | |
| EGLFP-3 | 12.322 | 13.273 | 10.7 | 11.2 | 3.2 | |
| | 22.477 | 24.230 | 5.2 | 8.1 | 2.4 | |
| | 29.411 | 29.536 | 2.0 | 4.0 | 1.0 | |
| **Ele2** | | | | | | |
| WM | 312.446 | 314.944 | 65.0 | 27.0 | 4.0 | |
| CHV | 235.053 | 238.263 | 22.6 | 20.4 | 4.0 | 50745 |
| EGLFP-1 | 185.383 | 187.804 | 24.2 | 18.4 | 3.2 | 35371 |
| | 183.883 | 193.290 | 17.6 | 17.4 | 3.0 | 32462 |
| | 211.290 | 212.999 | 12.7 | 14.4 | 3.0 | |
| EGLFP-3 | 314.150 | 314.470 | 6.9 | 10.6 | 3.0 | |
| | 553.541 | 566.163 | 3.7 | 8.1 | 2.0 | |
| | 903.109 | 917.182 | 2.0 | 4.0 | 1.0 | |
| **DEE** | | | | | | |
| WM | 0.36302 | 0.47091 | 178.4 | 37.0 | 6.0 | |
| CHV | 0.34580 | 0.42109 | 112.0 | 32.8 | 0.0 | 0.11226 |
| EGLFP-1 | 0.31916 | 0.41802 | 142.2 | 33.0 | 6.0 | 0.10254 |
| | 0.31795 | 0.43921 | 154.8 | 33.8 | 6.0 | 0.10376 |
| | 0.34773 | 0.43448 | 105.5 | 28.1 | 6.0 | |
| EGLFP-3 | 0.37248 | 0.43070 | 50.5 | 21.4 | 5.2 | |
| | 0.41404 | 0.43448 | 13.3 | 13.5 | 3.5 | |
| | 0.79297 | 0.79075 | 2.0 | 4.0 | 1.0 | |
| **Concrete** | | | | | | |
| WM | 8.2581 | 9.6227 | 309.8 | 47.0 | 8.0 | |
| CHV | 5.5686 | 7.4688 | 335.8 | 43.6 | 8.0 | 39.058 |
| EGLFP-1 | 5.8652 | 7.1887 | 235.0 | 36.0 | 8.0 | 37.614 |
| | 5.6360 | 7.3113 | 324.8 | 41.2 | 8.0 | 39.286 |
| | 6.3962 | 7.1763 | 210.0 | 30.8 | 7.1 | |
| EGLFP-3 | 7.2349 | 7.8924 | 91.4 | 22.2 | 5.8 | |
| | 8.9515 | 8.8944 | 16.8 | 12.3 | 3.9 | |
| | 14.9830 | 15.0084 | 2.0 | 4.0 | 1.0 | |
| **Comp-activ** | | | | | | |
| WM | 11.9387 | 11.9762 | 425.6 | 70.0 | 21.0 | |
| CHV | 3.6004 | 3.6022 | 103.6 | 51.0 | 21.0 | 17.2945 |
| EGLFP-1 | 3.3237 | 3.3848 | 14.0 | 19.4 | 6.4 | 9.7949 |
| | 3.3506 | 3.4277 | 12.6 | 17.6 | 6.6 | 9.8424 |
| | 3.4751 | 3.5417 | 8.6 | 14.1 | 5.3 | |
| EGLFP-3 | 4.4530 | 4.5192 | 6.3 | 10.5 | 3.7 | |
| | 6.6188 | 6.6736 | 4.0 | 8.4 | 2.8 | |
| | 18.3997 | 18.3876 | 1.8 | 4.0 | 1.0 | |

subset of input variables, the number of linguistic terms per variable, the type of membership functions (trapezoidal or triangular), the membership function parameter values, and the fuzzy rule set. The algorithm is designed to generate highly legible and compact fuzzy partitions thanks to the use of interpretability constraints and powerful genetic operators.

The originality and good performance of the proposal mainly lies in the definition of new real-coding crossover and mutation operators that properly deals with the constraints for distinguishability, orderliness, and non-deformity imposed to the learnt fuzzy partitions; the design of a variable-length coding scheme and a parent-centric crossover capable of handling fuzzy partitions with different number of linguistic terms; and the use of several criteria and a multi-objective optimization scheme that provide with a range of different interpretability-accuracy tradeoffs and endow the algorithm with an effective niche-based search process. Successful results are obtained in six real-world regression problems with up to 21 continuous input variables.

As further work we will investigate the scalability of the algorithm to large-scale problems and the use of more criteria to assess the interpretability quality of the obtained fuzzy models.

## Acknowledgment

This work was supported in part by the Spanish Ministry of Science and Innovation (grant no. TIN2008-06681-C06-01) and the Andalusian Government (grant no. P07-TIC-3185).

### References

[1] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka. Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 3(3):260–270, 1995.

[2] T.-P. Hong and J.-B. Chen. Finding relevant attributes and membership functions. *Fuzzy Sets and Systems*, 103(3):389–404, 1999.

[3] A. González and R. Pérez. Completeness and consistency conditions for learning fuzzy rules. *Fuzzy Sets and Systems*, 96(1):37–51, 1998.

[4] 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.

[5] J. Espinosa and J. Vandewalle. Constructing fuzzy models with linguistic integrity from numerical data-AFRELI algorithm. *IEEE Transactions on Fuzzy Systems*, 8(5):591–600, 2000.

[6] J.-N. Choi, S.-K. Oh, and W. Pedrycz. Identification of fuzzy models using a successive tuning method with a variant identification ratio. *Fuzzy Sets and Systems*, 159(21):2873–2889, 2008.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarevian. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[8] C. García-Martínez, M. Lozano, F. Herrera, D. Molina, and A. M. Sánchez. Global and local real-coded genetic algorithms based on parent-centric crossover operators. *European Journal of Operational Research*, 185:1088–1113, 2008.

[9] 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.

ISBN: 978-989-95079-6-8