A Unified Design for the Membership Functions in Genetic Fuzzy Systems

Tri Minh Huynh[1], Huy Xuan Nguyen[2], Bac Le[3]

[1] Department of Information Technology, Sai Gon University
Ho Chi Minh City, Vietnam

[2] Institute of Information Technology, Vietnamese Acadamy of Science and Technology, Hanoi city, Vietnam

[3] Department of Computer Science, Ho Chi Minh University of Science
Ho Chi Minh City, Vietnam

Abstract

This paper introduces a unified design for the membership functions in genetic fuzzy systems (GFSs). The membership functions can get one of the symmetric or asymmetric forms of triangles, exponential Gauss, trapezoid and so on in the general unified form. With these unified forms, the meanings of the parameters of the membership functions become more clearly and more understandable. Moreover, these unified forms permit our proposed genetic algorithm (GA) to tune the parameters of the membership functions one by one. This makes our method different because other methods always simultaneously tune the parameters. With this new approach, the better parameters are easier to find out via parameter learning process. In addition, appropriate inferences are defined in the systems. As a result, the obtained genetic fuzzy systems are more effective and more compact.

Keywords: *genetic fuzzy systems, linguistic terms, fuzzy rule set reduction, language hedges, genetic algorithm.*

1. Introduction

Fuzzy Ruled-Based Systems (FRBSs) form the extended systems of classical rule-based systems, because they include ‘‘IF-THEN’’ rules with fuzzy logic antecedents and consequents, instead of classical logic ones. FRBSs provide an efficient tool for describing and processing problems in the real world concerning uncertainty and imprecision. Thus, they have had a wide range of applications in control problems [2, 3], modeling [4], classification, data mining [5, 6].

Genetic algorithms (GAs) have been employed as robust tools to search optimal solutions in complex spaces. They usually give effective and efficient solutions to the complicated real-world problems. So a combination of GAs with fuzzy logic has been led to the birth of Genetic Fuzzy Systems (GFSs). These are hybrid fuzzy systems in which a learning process is based on GAs [7].

The important factor of linguistic FRBSs is the knowledge base (KB). The KB consists of two components, i.e., a data base (DB) and a rule base (RB) [1, 2, 3].

The DB comprises the linguistic term sets and the membership functions defining their meanings.

The RB includes set of fuzzy linguistic IF-THEN rules and joined by “also” operator. That means those rules will be activated simultaneously with the same input data.

Many approaches have been proposed to generate KB automatically using numerical data.

- The first research direction focuses on learning RB from a predefined DB [2, 8, 9]. The FRBS performance is strongly influenced by the predefined DB.
- The second research direction focuses on improving the FRBS performance by tuning the initial DB after RB has been generated [10, 11]. This process only adjusts the parameters of the membership functions but does not vary the number of linguistic terms in each fuzzy partition so the RB still does not change.
- The third research direction focuses on learning to generate the different components of the initial DB [12, 13].

We propose a new approach in which a simple method is used to automatically generate the KB of FRBS from numerical data and a proposed genetic tuning process for the different components of the KB. First, the initial RB and the initial DB of the KB are created from numerical data. Next, we represent new processes to adjust the RB and the DB by using GA. Many other approaches have been proposed to adjust simultaneously the whole DB. Our contribution is to adjust each parameters of the membership functions individually. To achieve this goal, the DB includes the membership functions, whose formulas are in unified forms. In this paper, the parameters of a membership function are m, leftσ, rightσ and midσ. In the graph of a membership function, the parameter m is

the center of the peak. Leftσ and rightσ determine respectively the spread of the shape of the left-most and the right-most curve of the membership functions from the center m (Fig. 1,2,3,4). The shapes of the membership functions are symmetric or asymmetric depending on leftσ = rightσ or leftσ ≠ rightσ. Besides, the parameter midσ is only used exclusively for the trapezoidal formula. The parameter midσ is a half of the length of the small base of the trapezoid (Fig. 3). This is advantage to applying our proposed GA to tune up the DB because these membership functions are in the unified forms. The membership functions can get one of the symmetric or asymmetric forms of triangular, exponential Gaussian, trapezoidal functions and so on in the general unified form.

2. Generating the Initial Kb from Numerical Data

In this paper, we consider a MISO FRBS (Multi-Input, Single Output Fuzzy Rule Based System). Let's assume that the input-output data set used as training data is $T=\{(x^{(k)},y^{(k)}) \mid x^{(k)} \in R^m, y^{(k)} \in R, k=1, 2,.., n\}$, where $x^{(k)} =(x_1^{(k)}, .. , x_m^{(k)})$ is the input vector of the *k-th* input-output pair and $y^{(k)}$ is the corresponding output, m is the dimension of the input vector $x^{(k)}$.

2.1 Step 1: Generating the initial DB

The domain interval $[a_i , b_i]$ of the *i-th* input variable x_i is equally divided into $2N_i+1$ fuzzy sets $A_i^{(1)}, A_i^{(2)},…., A_i^{(2Ni + 1)}$ for i =1, 2, .. , m. Then, let fuzzy sets $A_i^{(1)}, A_i^{(2)},…., A_i^{(2Ni + 1)}$ respectively correspond to linguistic labels L_1 (Low_1), .., L_{Ni} (Low_{Ni}), M (Medium), H_1 ($High_1$), .., H_{Ni} ($High_{Ni}$). Normally, N_i are chosen equal to each other but N_i can be chosen different.

The domain interval [c , d] of the output variable y is equally divided into 2N+1 fuzzy sets $B^{(1)}, B^{(2)},…., B^{(2N + 1)}$. Then, let fuzzy sets $B^{(1)} , B^{(2)},…., B^{(2N + 1)}$ respectively correspond to linguistic labels L_1 (Low_1), .., L_N (Low_N), M (Medium), H_1 ($High_1$), …,H_N ($High_N$).

For each i = 1, 2,..., m, for each j=1,.., $2N_i$ +1, set $\Delta_i=(b_i-a_i)/(2N_i)$, $m_i^{(j)} = a_i +(j-1)\Delta_i$. In addition, we set $left\sigma_i^{(j)} = right\sigma_i^{(j)} = c.\Delta_i$, with c being a real constant prescribed depending on the type of membership functions below. From now on, $left\sigma_i^{(j)}$, $right\sigma_i^{(j)}$ are abbreviated respectively as $\sigma_{\ell i}^{(j)}$, $\sigma_{ri}^{(j)}$ and $m_i^{(j)}$ is the center of the peak of the shape of the membership functions. $\sigma_{\ell i}^{(j)}$, $\sigma_{ri}^{(j)}$ determine respectively the spread of the shape of the left-most and the right-most curve of the membership functions. Moreover, a parameter $mid\sigma_i^{(j)}$ is added in the trapezoidal membership function to describe a half of small bottom length of the trapezoid and $mid\sigma_i^{(j)}$ is abbreviated as $\sigma_{mi}^{(j)}$.

Note that the domain interval $[a_i, b_i]$ or [c, d] can be divided into an even number of fuzzy sets, the process of generating DB is similar as described above.

1) Gaussian membership functions

For each $A_i^{(j)}$, we employ an exponential Gaussian membership function as follows

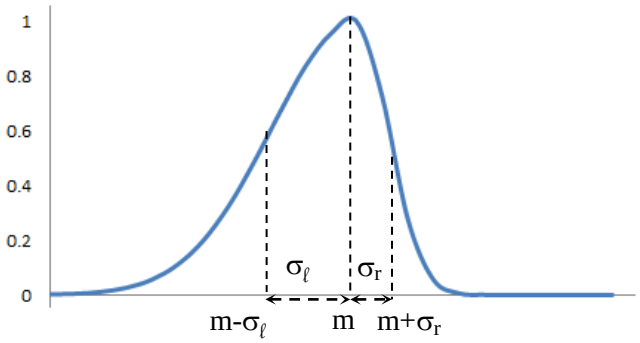

Fig. 1 An illustrated graph of a Gaussian membership function with $\sigma_\ell \neq \sigma_r$

In this case, the real constant c is given by 1/3. At first, let

$\sigma_{\ell i}^{(j)} = \sigma_{ri}^{(j)} = c.\Delta_i = \Delta_i /3.$

2) Triangular membership functions

For each $A_i^{(j)}$, we employ a triangular membership function as follows

$$\mu_{A_i^{(j)}}(x) = \begin{cases} 0 & \text{if } x < m_i^{(j)} - \sigma_{\ell i}^{(j)} \text{ or } x > m_i^{(j)} + \sigma_{ri}^{(j)} \\ \frac{1}{\sigma_{\ell i}^{(j)}}(x - m_i^{(j)} + \sigma_{\ell i}^{(j)}) & \text{if } m_i^{(j)} - \sigma_{\ell i}^{(j)} \le x < m_i^{(j)} \\ -\frac{1}{\sigma_{ri}^{(j)}}(x - m_i^{(j)} - \sigma_{ri}^{(j)}) & \text{if } m_i^{(j)} \le x \le m_i^{(j)} + \sigma_{ri}^{(j)} \end{cases}$$

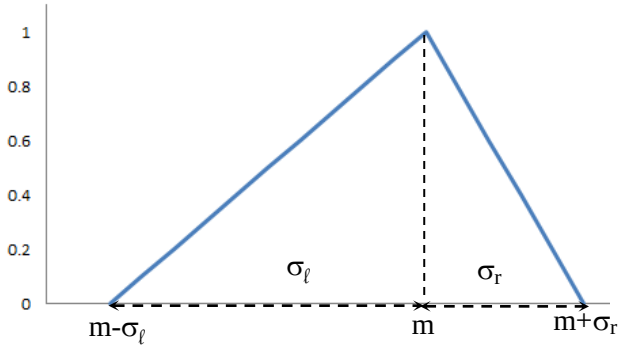

Fig. 2 An illustrated graph of a triangular membership function with $\sigma_l \neq \sigma_r$

In this case, the real constant c is given by 1. At first, let

$\sigma_{\ell i}^{(j)} = \sigma_{ri}^{(j)} = c.\Delta_i = \Delta_i.$

3) Traperzoidal membership functions

For each $A_i^{(j)}$, we employ a traperzoidal membership function as follows

$$\mu_{A_i^{(j)}}(x)=\begin{cases}0 & \text{if } x<m-\sigma_{\ell_i}^{(j)}-\sigma_{m_i}^{(j)} \text{ or } x>m+\sigma_{m_i}^{(j)}+\sigma_{r_i}^{(j)}\\ \frac{1}{\sigma_{\ell_i}^{(j)}}(x-m+\sigma_{m_i}^{(j)}+\sigma_{\ell_i}^{(j)}) & \text{if } m-\sigma_{\ell_i}^{(j)}-\sigma_m\le x<m-\sigma_{m_i}^{(j)}\\ 1 & \text{if } m-\sigma_{m_i}^{(j)}\le x<m+\sigma_{m_i}^{(j)}\\ -\frac{1}{\sigma_{r_i}^{(j)}}(x-m-\sigma_{m_i}^{(j)}-\sigma_{r_i}^{(j)}) & \text{if } m+\sigma_{m_i}^{(j)}\le x\le m+\sigma_{m_i}^{(j)}+\sigma_{r_i}^{(j)}\end{cases}$$

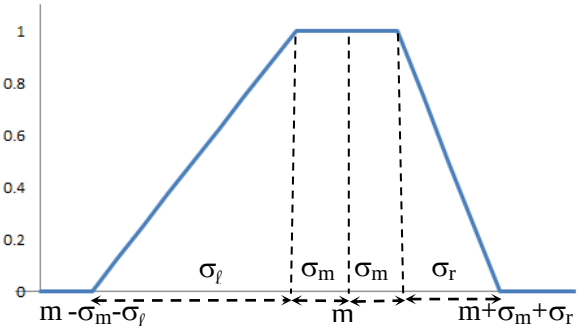

Fig. 3 An illustrated graph of a traperzoidal membership function with $\sigma_l \neq \sigma_r$

In this case, the real constant c is given by 4/5. At first, let $\sigma_{\ell_i}^{(j)}=\sigma_{r_i}^{(j)}=c.\Delta_i=4\Delta_i/5$ and $\sigma_{m_i}^{(j)}=\Delta_i/5$.

4) General bell-shaped membership functions

For each $A_i^{(j)}$, we employ a general bell-shaped membership function as follows

$$\mu_{A_i^{(j)}}(x)=\begin{cases}\frac{1}{1+(\frac{x-m_i^{(j)}}{\sigma_{\ell_i}^{(j)}})^{2b}} & \text{if } x\le m_i^{(j)}\\ \frac{1}{1+(\frac{x-m_i^{(j)}}{\sigma_{r_i}^{(j)}})^{2b}} & \text{if } x>m_i^{(j)}\end{cases}$$

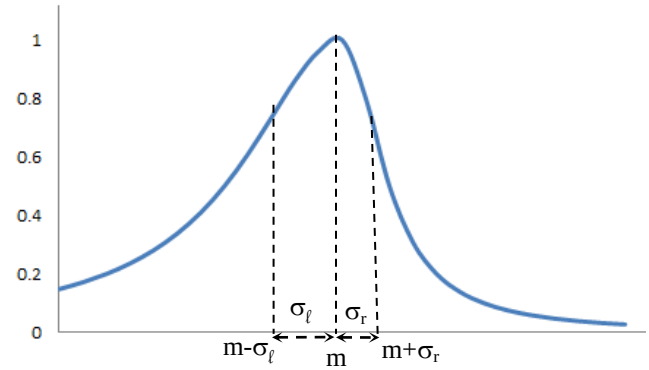

Fig. 4 An illustrated graph of a general bell-shaped membership function with $\sigma_l \neq \sigma_r$

In this case, the real constant c is given by 1/10. At first, let $\sigma_{\ell_i}^{(j)}=\sigma_{r_i}^{(j)}=c.\Delta_i=\Delta_i/10$ and b=1.

2.2 Step 2: Generating the initial RB

Relying on the idea of CH and WM methods [14], in this section, we proposed a method to generate the initial RB from the numeric data as following [1]:

- For i =1,2,…,m, the set of fuzzy sets $A_i^{(1)}$, $A_i^{(2)}$,…., $A_i^{(2Ni+1)}$ is denoted by Ai.
- The set of fuzzy sets $B^{(1)}$, $B^{(2)}$,…, $B^{(2N+1)}$ is denoted by B.
- Each example $e_p=(x^{(p)}, y^{(p)})=(x_1^{(p)}, .. ,x_m^{(p)},y^{(p)}) \in T$ will correspond to a fuzzy rule Rp in the form:

If x_1 is $A_1^{(p_1)}$ and… and x_m is $A_m^{(p_m)}$ then y is $B^{(p)}$ where $A_i^{(p_i)} \in A_i$ is a fuzzy set so that $\mu_{A_i^{(p_i)}}(x_i^{(p)})=\max_j \mu_{A_i^{(j)}}(x_i^{(p)})$ for i=1, 2,…,m and $B^{(p)} \in B$ is a fuzzy set so that $\mu_{B^{(p)}}(y^{(p)})=\max_j \mu_{B^{(j)}}(y^{(p)})$.

Since there are n examples, so we can get maximum of n fuzzy rules. However, in reality there often exist some fuzzy rules having the same antecedent part $A_1^{(p_1)}$,..., $A_m^{(p_m)}$ but the consequence parts are different. These rules form a group, having the same antecedent part. Since there are maximum of n rules, so there will be r distinct groups with $r \le n$. Each group has at least one rule.

- For each fuzzy rule R_p: If x_1 is $A_1^{(p_1)}$ and… and x_m is $A_m^{(p_m)}$ then y is $B^{(p)}$ and each example $e_r=(x^{(r)}, y^{(r)})=(x_1^{(r)}, .. ,x_m^{(r)}, y^{(r)}) \in T$, the covering value CVT of rule R_p bases on data e_r is computed as [14]: $CV_T(R_p,e_r)=T(\mu_{A_1^{(p_1)}}(x_1^{(r)}),..,\mu_{A_m^{(p_m)}}(x_m^{(r)}),\mu_{B^{(p)}}(y^{(r)}))$

with T is t-norm function. In this paper, T is the minimum function.

- For each fuzzy rule R_p, set

$$E_p=\{e_k \in T \mid \mu_{A_1^{(p1)}}(x_1^{(k)})...\mu_{A_m^{(pm)}}(x_m^{(k)})>0\}$$

- The rule value function of the rule R_p, RVF(R_p), is computed as:

$$RVF(R_p)=\max_{e_r \in E_p}\{CV_T(R_p,e_r)\}$$

- We choose one rule that has the greatest rule value function (RVF) from each group to generate the initial RB. Because there are r distinct group, so there are r rules be chosen to set up the initial RB.

As the result, we can obtain a rule base as following:

R_1: If x_1 is $A_1^{(1)}$ and… and x_m is $A_m^{(1)}$ then y is $B^{(1)}$

R_2: If x_1 is $A_1^{(2)}$ and… and x_m is $A_m^{(2)}$ then y is $B^{(2)}$

…

R_r: If x_1 is $A_1^{(r)}$ and… and x_m is $A_m^{(r)}$ then y is $B^{(r)}$

2.3 Step 3: Specifying fuzzification and defuzzification

For each rule Rp and for each x =(x_1,..,x_m), w_p is the firing strength : $w_p(x) = \mu_{A_1^{(p)}}(x_1).\mu_{A_2^{(p)}}(x_2)...\mu_{A_m^{(p)}}(x_m)$. (1)

The output fuzzy set is computed by: $w_p(x).\mu_{B^{(p)}}(y)$.

Defuzzify the output fuzzy sets by using our proposed method similar to the simulated center-of-area method of Lin and Lee [14]:

$$\hat{y} = \frac{\sum_{p=1}^{r} w_p.m^{(p)}.\sigma^{(p)}}{\sum_{p=1}^{r} w_p.\sigma^{(p)}}$$

where ŷ is the output of the system, w_p is given by (1), $m^{(p)}$ are the centers and $\sigma_\ell^{(p)}, \sigma_r^{(p)}$ are respectively the spread of the shape of the left-most and the right-most curve from the center $m^{(p)}$ of membership functions $\mu_{B^{(p)}}(.)$. In this paper, the symmetric membership functions $\mu_{B^{(p)}}(.)$ are employed, ie., $\sigma_\ell^{(p)} = \sigma_r^{(p)}, \forall p = 1,.., r$ and set $\sigma^{(p)} = \sigma_\ell^{(p)} = \sigma_r^{(p)}, \forall p = 1,.., r$.

3. Tuning the Initial KB

With the system presented above (section 2), we can tune the initial KB to improve the accuracy of system by the following approaches:

- The first approach is to tune only the RB by using the language hedges such as very, more or less, more, not, little, etc [15].
- The second approach is to tune only the DB by adjusting the parameters of membership functions [10].
- The third approach is to apply both above approaches [16].

The third approach is the method that we use in this paper. Beside, when the dimensions of data increase in size, the number of rules will grow exponentially and the conditions in the antecedents of rules will grow correlative in number. This makes the KB more complicated and bulky. Overcoming this disadvantage is an important requirement when building FRBS. Thus, in this paper, we also propose a method to resolve these problems by using GA. Our method is to use GA to tune up the DB and also to simplify the RB, reduce the number of rules and simultaneously reduce conditions in the antecedents of the rules. The result gives us a streamlined and efficient system.

The Genetic Algorithm components:

3.1 The objective function

The objective function that needs to be minimized is the following function, the well-known mean square error [18]: $MSE = \frac{1}{2n}\sum_{i=1}^{n}(\hat{y}^{(i)} - y^{(i)})^2$ with n being size of data set, p being the number of system parameters, $\hat{y}^{(i)}$ being the output of the FBRS corresponding with given inputs $x^{(i)}$, and $y^{(i)}$ being the known desired output.

3.2 Coding of KB

Each chromosome is presented as four components P + L + R + C shown in Fig. 5. The P part will encode the basic parameters of the membership functions. The L part will express the language hedges added in different rules of the initial RB. The R part expresses rules chosen from the initial RB. The C part will express the variables chosen or not in the antecedents of different rules of the initial RB.

The P part includes sets of real values $m_i^{(j)}$, $\sigma_{\ell i}^{(j)}$, $\sigma_{ri}^{(j)}$ and $\sigma_{mi}^{(j)}$ being parameters of the membership functions mentioned in section 2. The parameters $m_i^{(j)}$, $\sigma_{\ell i}^{(j)}$, $\sigma_{ri}^{(j)}$ and $\sigma_{mi}^{(j)}$ vary in their variation interval. The variation interval of each parameter is already specified according to the type of membership functions used. The variation interval of each parameter will be described in detail in section 3, 3.5.

Table 1: Linguistic hedges and corresponding functions

$L_{k,i}$	*Linguistic hedges*	*Corresponding functions*
0	"absolutely"	$(\mu_{A_i^{(k)}}(x))^4$
1	No hedge used	$\mu_{A_i^{(k)}}(x)$
2	"extremely"	$(\mu_{A_i^{(k)}}(x))^3$
3	"very"	$(\mu_{A_i^{(k)}}(x))^2$
4	"much more"	$(\mu_{A_i^{(k)}}(x))^{1.75}$
5	"more"	$(\mu_{A_i^{(k)}}(x))^{1.5}$
6	"plus"	$(\mu_{A_i^{(k)}}(x))^{1.25}$
7	"minus"	$(\mu_{A_i^{(k)}}(x))^{0.75}$
8	"more or less"	$(\mu_{A_i^{(k)}}(x))^{0.5}$
9	"slightly"	$(\mu_{A_i^{(k)}}(x))^{0.25}$

The L part is encoded into an integer string with length r(m + 1) where r is rule number, m+1 is the number of

input variables and one output variable. $L_{k,i}$ is the gene corresponding to the linguistic hedge that modifies the membership function associated to the linguistic term of the i-*th* variable in the k-*th* rule. $L_{k,i}$ can take values in {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} corresponding to the linguistic hedges used (Table. 1).

The R part is encoded into an r-bit string in which r is the number of fuzzy rules of the RB. Value 1 at position i in the sequence means that the i-*th* rule is used, otherwise the value 0 then the i-*th* rule is not used.

The C part is encoded into a bit string with length r.m, r is number of rules, m is the number of input variables. $C_{k,i}$ is the gene can take values in the set {0, 1} and $C_{k,i} = 0$ if the variable X_i is not used in the k-*th* rule, whereas $C_{k,i} = 1$ if the variable X_i is used in the k-*th* rule. Note that the output variable Y is always used, so it is so it is not necessary to be included in the C part.

The method tuning the KB by genetic algorithm is described as follows.

3.3 The generation of the initial gene pool consists of two steps

1) A chromosome, representing the initial KB, is included. That means, its genes in the P part receives the values from the initial parameters of membership functions of the system (section 2) and in the L, R and C parts, alleles 1 will be used.

2) The remaining chromosomes of the population are generated with the P part at random within the variation intervals for each gene (section 3, 3.5). Meanwhile, in the L, R and C parts, alleles 1 will be still used. The mutation operators are applied differently on each part of the chromosomes chosen for mutation.

3.4 Crossover operator

In the P part, the max-min-arithmetical operator is used as the crossover operator. If $P_u^t = (c_1,..,c_k,..,c_h)$ and $P_v^t = (c_1',..,c_k',..,c_h')$ are the P parts of the chromosome u and v in the t-*th* generation. If u and v are selected to mate, the four offspring with the P part will be generated:

$$P_1^{t+1} = a\ P_u^t + (1-a)P_v^t$$
$$P_2^{t+1} = a\ P_v^t + (1-a)P_u^t$$
$$P_3^{t+1} \text{ with } \quad c_{3,k}^{t+1} = \min\{c_k\ ,\ c_k^{'}\}$$
$$P_4^{t+1} \text{ with } \quad c_{4,k}^{t+1} = \max\{c_k\ ,\ c_k^{'}\}$$

Where $a \in [0, 0.5]$ is a parameter given by the designer. In this paper, parameter a is randomly chosen in segment [0, 0.5]. In the L, R, C parts, the standard two-point crossover is used. Then all parts are recombined. 32 offspring are generated from the combination of four different P parts, two different L parts, two different R parts and two different C parts (Fig. 6).

We choose the two best offspring among the 32 children to replace their parents.

3.5 Mutation operator

The mutation operators are applied differently on each part of the chromosomes chosen for mutation.

In the P part, the uniform mutation operator is applied. Each selected allele of the genes will be replaced by a randomly generated allele on the variation interval of the gene. As known above, the P part includes sets of real values $m_i^{(j)}$, $\sigma_{\ell_i}^{(j)}$, $\sigma_{\ell_i}^{(j)}$ and $\sigma_{m_i}^{(j)}$ being the parameters of the membership functions. Depending on the type of membership functions, each parameter $m_i^{(j)}$, $\sigma_{\ell_i}^{(j)}$, $\sigma_{r_i}^{(j)}$ or $\sigma_{m_i}^{(j)}$ can get various values in its own variation interval. After mutation, the new values of $m_i^{(j)}$, σ_{ℓ_i} and σ_{r_i} are denoted respectively by $m_i'^{(j)}$, $\sigma'^{(j)}_{\ell_i}$ and $\sigma'^{(j)}_{r_i}$. Particularly in the formula of the traperzoidal membership functions, the new value of $\sigma_{m_i}^{(j)}$ is denoted $\sigma'^{(j)}_{m_i}$.

1) Gaussian membership functions

$$m_i'^{(j)} \in [m_i^{(j)} - \Delta_i/4\ ,\ m_i^{(j)} + \Delta_i/4]$$
$$\sigma'^{(j)}_{\ell_i}\ ,\sigma'^{(j)}_{r_i} \in [5\Delta_i/18\ ,7\Delta_i/18]$$

2) Triangular membership functions

$$m_i'^{(j)} \in [m_i^{(j)} - \Delta_i/4\ ,\ m_i^{(j)} + \Delta_i/4]$$
$$\sigma'^{(j)}_{\ell_i}\ ,\sigma'^{(j)}_{r_i} \in [5\Delta_i/6\ ,7\Delta_i/6]$$

3) Traperzoidal membership functions

$$m_i'^{(j)} \in [m_i^{(j)} - \Delta_i/4\ ,\ m_i^{(j)} + \Delta_i/4]$$
$$\sigma'^{(j)}_{\ell_i}\ ,\sigma'^{(j)}_{r_i} \in [20\Delta_i/30\ ,28\Delta_i/30]$$
$$\sigma'^{(j)}_{m_i} \in [0\ ,7\Delta_i/30]$$

4) General bell-shaped membership functions

$$m_i'^{(j)} \in [m_i^{(j)} - \Delta_i/4\ ,\ m_i^{(j)} + \Delta_i/4]$$
$$\sigma'^{(j)}_{\ell_i}\ ,\sigma'^{(j)}_{r_i} \in [\Delta_i/20\ ,3\Delta_i/20]$$

In the L part, if value of selected gene is 1, it is changed to a random value in the set of {0, 2, 3, 4, 5, 6, 7, 8, 9} otherwise, it is changed to 1.

In the R or C part, if value of selected gene is 1, it is changed to 0 otherwise, it is changed to 1.

If an individual is selected to be mutated, a randomly selected gene from each its part will be applied a corresponding mutation operator.

Baker's stochastic universal sampling (SUS) [19] together with elitism is considered in the paper. Elitism first copies the best chromosome (or a few best chromosomes) to new population so it prevents losing the best found solution.

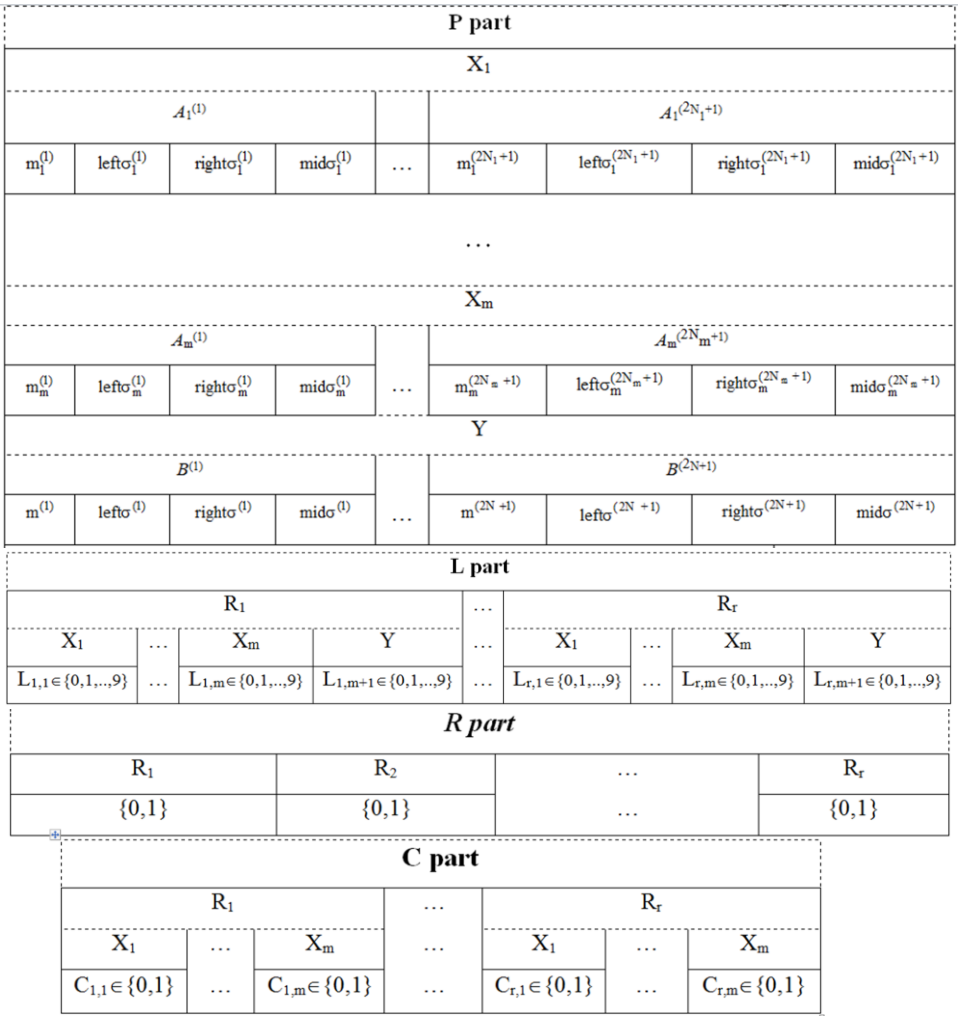

Fig. 5 The components of a chromosome

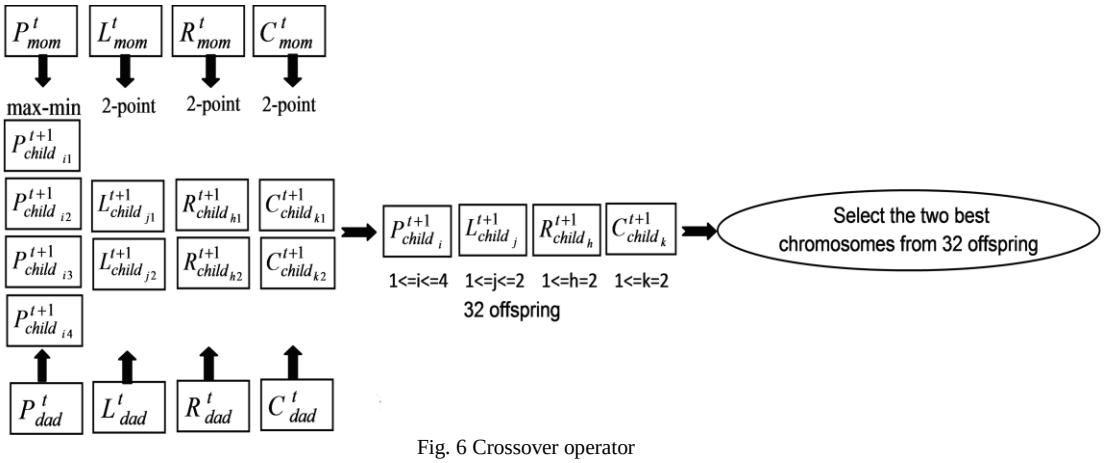

Fig. 6 Crossover operator

4. Experimental Study

Our testing consists of the following two steps:

- Step 1: Using the method mentioned in section 2 are called as TMH (the method has been discussed in [1]) or any other simple methods such as Wang and Medel (WM) in [20] to create the initial KB from numerical data. This process only takes negligible time, much less time than steps 2.
- Step 2: Simplify the RB and tune up the DB in one of six following models.

The operators of GA can be applied on the P part, the L part, the R part and the C part with different ways, it will give different approaches. In this paper, we will mention six approaches with six corresponding models, due to good results the six models offer.

4.1 Model 1:

Implementing Step 1, then the operators of GA are applied simultaneously on the R part and the C part first, then applied only on m, a gene of the P part, finally applied only on σ, a gene of the P part. The two above processes are done sequentially but always go together with the GA operators applied on the L, also known as model TMH+RC+ML+SL.

It means simplifying RB before, then tuning individually m, afterward tuning σ individually. After the phase RC, the set of rules is optimal with smaller size and shorter-length rules. Throughout the two above phase ML and SL processes, the antecedent parts of rules in RB are also simultaneously modified by adding the appropriate linguistic hedges (section 3). In the phase ML, the parameters m of the membership functions have been tuned, learned. In the phase SL, the parameters σ of the membership functions have been tuned, learned. In both of two phases above, the model structure has been extended by using linguistic modifiers.

4.2 Model 2:

This model is similar to model 1 but the phase RC is executed first, then the phase SL is executed before the phase ML is done, also known as model TMH+RC+SL+ML.

4.3 Model 3:

This model is similar to model 1 but the phase SL is executed first, then the phase ML is done and finally the phase RC is done, also known as model TMH+SL+ML+RC.

4.4 Model 4:

This model is similar to model 1 but the phase ML is executed first, then the phase SL is done and finally the phase RC is done, also known as model TMH+ML+SL+RC.

4.5 Model 5:

This model is similar to model 1. First implementing Step 1, then the operators of GA are applied on m, a gene of the P part, applied on σ, a gene of the P part and applied on the L part. Finally, the operators of GA are applied on the R part and the C part simultaneously. The three above phase M, S, L are done simultaneously, also known as model TMH+MSL+RC.

In the phase MSL, the parameters m and σ of membership functions have been tuned, learned and the model structure has been extended by using linguistic modifiers. After the phase RC, the set of rules is optimal with smaller size and shorter-length rules.

4.6 Model 6:

This model is similar to model 1. First implementing Step 1, then the operators of GA are applied on the R part and the C part simultaneously. Finally, the operators of GA are applied on m, a gene of the P part, applied on σ, a gene of the P part and applied on the L. The three above phase M, S, L are done simultaneously, also known as model TMH+RC+MSL.

After the phase RC, the set of rules is optimal with smaller size and shorter-length rules. In the phase MSL, the parameters m and σ of membership functions have been tuned, learned and the model structure has been extended by using linguistic modifiers.

As the result of six models above, the obtained FRBS gets a fuzzy compact and simple rule base with high accuracy and good generalization capacity.

The parameters were used in the tests such as a population size of 50 individuals, 0.6 as crossover probability, 0.2 as mutation probability per chromosome chosen. The results such as the numbers of rules and MSE were calculated by taking average for values of all the tests.

In six proposed models, the phase RC is executed in 50 generations, the phase ML or the phase SL is executed in 75 generations, the phase MSL is executed in 150 generations.

The tests were done on the PC with Pentium Dual core 2.4 GHz processor, 4 GB RAM, Windows 7.0 operating system and development tool C#.Net 2008.

4.7 The Experimental study

The Rice Taste Evaluation Problem

We use the data set presented in [22]. This set is composed of 105 data vectors collecting subjective evaluations of the six variables that relate six factors such as flavor, appearance, taste, stickiness, toughness and the overall evaluation of the kind of rice. The data set was performed by experts on the kinds of rice grown in Japan.

The objective of the problem is to deal with the subjective evaluation of the qualification of rice taste. It is very interesting to represent the existing nonlinear relationships of the problem by means of linguistic fuzzy modeling. The data set has randomly been divided into five different independent partitions.

Each partition is composed of 75 points of data in the training set and 30 in the test one.

Each of six models mentioned above is applied in turn on the above partitions with 10 runs, with different seeds for each run. This ensures the non-biased learning of our proposed models. We use the MSE function [18] to compare our results of six proposed models to the results in the other papers (see Table 2, 3, 4, 5). In Table 2, 3, 4, 5, #R stands for the number of rules, MSEtra, MSEtst for the values of MSE over the training and test data sets respectively.

Table 2: Results of Other Methods

	Rice Problem								
	$\bar{x}$			$\sigma_{\bar{x}_i}$			$\bar{\sigma}_{x_i}$		
Methods	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}
Nozaki [21]	64	27.41	32.01	**0.0**	3.63	5.62	-	-	-
Thrift [23]	18.2	53.10	66.55	1.2	3.68	11.06	1.8	2.74	5.27
Liska [24]	32	30.80	57.52	**0.0**	3.37	16.58	**0.0**	4.23	9.80
WM+PL-tun [22]	15	**8.60**	**17.75**	0.6	**0.70**	**2.74**	-	**0.85**	**2.63**
WM+Reduction+ PL-tun [22]	**5.7**	10.35	24.12	1.1	1.06	7.36	1.1	1.06	6.23
WM+PL-tun& Reduction [22]	12.4	10.18	20.78	1.1	0.98	5.10	1.2	1.09	3.76

*multiplied by 10,000

Table 3: Results of Six Proposed Models when using Triangle Membership Functions

	Rice Problem								
	$\bar{x}$			$\sigma_{\bar{x}_i}$			$\bar{\sigma}_{x_i}$		
Methods	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}
Model 1	10.6	**10.74**	11.98	**0.3**	0.54	3.40	1.4	**0.64**	2.39
Model 2	10.5	11.38	**11.36**	1.2	0.50	**2.30**	1.2	0.98	2.47
Model 3	14.2	11.24	14.61	0.9	**0.49**	5.60	1.2	0.95	2.95
Model 4	14.4	11.65	14.23	0.6	0.56	5.25	**0.9**	0.75	3.28
Model 5	13.8	11.77	14.31	1.1	0.70	4.75	1.4	1.02	3.75
Model 6	**10.0**	11.21	11.88	0.8	0.52	3.67	1.5	0.66	**2.14**

Table 4: Results of Six Proposed Models when using Bell-Shaped Membership Functions

	Rice Problem								
	$\bar{x}$			$\sigma_{\bar{x}_i}$			$\bar{\sigma}_{x_i}$		
Methods	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}
Model 1	10.1	**11.39**	12.23	2.3	**1.25**	**3.73**	**1.5**	1.62	**2.20**
Model 2	**09.9**	12.20	13.46	**0.9**	1.34	6.31	1.6	1.55	3.44
Model 3	11.9	12.98	15.20	**0.9**	1.53	5.45	1.7	1.52	3.86
Model 4	12.0	12.79	14.37	2.6	1.53	9.18	2.4	1.45	3.98
Model 5	11.8	13.03	14.27	1.5	1.99	6.08	1.9	**1.18**	3.82
Model 6	11.4	11.75	**12.14**	2.0	2.07	7.66	1.8	1.44	2.77

Table 5: Results of Six Proposed Models when using Traperzoidal Membership Functions

	Rice Problem								
	$\bar{x}$			$\sigma_{\bar{x}_i}$			$\bar{\sigma}_{x_i}$		
Methods	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}	#R	MSE^*_{tra}	MSE^*_{tst}
Model 1	8.4	**9.14**	13.88	1.1	0.95	4.69	1.2	**0.71**	3.06
Model 2	8.7	9.62	**12.99**	**0.6**	**0.61**	3.93	1.7	0.79	3.04
Model 3	13.1	9.87	15.29	1.3	0.80	3.71	1.0	0.89	3.62
Model 4	12.6	10.04	14.83	1.4	0.86	**2.72**	**0.9**	1.29	3.18
Model 5	12.4	9.69	15.22	0.8	0.71	3.31	1.0	1.15	**2.71**
Model 6	**8.2**	9.99	14.29	**0.6**	0.63	3.41	1.7	1.29	3.85

Because in the rice problem, the values of the out variable are normalized in [0,1], very small errors are obtained. Hence, in Table 2, the results are multiplied by 10,000 to facilitate their reading. The arithmetic mean ($\bar{x}$) over the 30 runs performed; the standard deviation ($\sigma_{\bar{x}_i}$) over the five values, one per data partition; and the arithmetic mean ($\bar{\sigma}_{x_i}$) of the standard deviation values over the five runs for each data partition are included. While ($\sigma_{\bar{x}_i}$) stands for the differences existing among the data partitions,($\bar{\sigma}_{x_i}$) stands for the differences existing among the runs for each data partition. Therefore, the former value shows the robustness of the learning/tuning method to obtain similar results regardless the data partition, while the latter value shows the robustness of the probabilistic algorithm to obtain similar results regardless the followed pseudo-random sequence, for more details to see in [22]. In Table 2, 3, 4, 5, the best results of the methods are presented in boldface.

5. Conclusion

This paper introduces six models to design the membership functions in GFSs, known as the model 1, 2, 3, 4, 5 and 6. All six models include from three to four phases, the first phase is always TMH. After the phase TMH, the initial KB is generated. After the phase RC, RB has been streamlined, that means the number of rules and the conditions in the antecedents of rules have been optimally reduced. After the phase ML+SL or SL+ML, in the model 1, 2, 3 and 4, the KB is tuned by genetic algorithms. These phases are new proposals because the methods of other researchers always simultaneously tune the parameters of the membership functions. Conversely, in our approach, each parameter is tuned one by one. This makes the process for parameter tuning more easy to achieve optimal results. The six proposed models have shown that a good interpretability-accuracy tradeoff is obtained by firstly reducing not only the number but also reducing the lengths of rules. After that, the resulting model is tuned. In this case, the tuning phase can profit from the selected rules adapting them for a good accuracy.

Acknowledgments

Tri.M.H sincerely thanks the colleagues, Faculty of Information Technology, Saigon University and in particular the professors of the Computer Science Division, Faculty of Information Technology, HCMC University of Natural Sciences for comments and supports for our research.

References

[1] Tri. M. H., “A Method to Buid a Fuzzy Streamlined and Effective Rule-based System”, The 2011 3rd International Conference on Machine Learning and Computing, vol. 4, pp 319-326, Feb, 2011.
[2] F. Herrera, Genetic fuzzy systems: taxonomy, current research trends and prospects, Evol. Intel. (2008) 1:27–46, Springer, Berlin.
[3] Palm R, Driankov D, Hellendoorn (1997) Model based fuzzy control. Springer, Berlin.
[4] Pedrycz W (Ed.) (1996) Fuzzy modelling: Paradigms and practice. Kluwer, Dordrecht.
[5] Ishibuchi H, Nakashima T, Nii M (2004) Classification and modeling with linguistic information granules: advanced approaches to linguistic data mining. Springer, Berlin.
[6] Kuncheva L (2000) Fuzzy classifier design. Springer, Berlin.
[7] O. Cordón, F. Herrera, F. Hoffmann, L. Magdalena, Genetic Fuzzy Systems Evolutionary Tuning and Learning of Fuzzy Knowledge Bases.World Scientific, 2001.
[8] J. Casillas, O. Cordón, and F. Herrera, "A Methodology to Improve adhoc Data-Driven Linguistic Rule Learning Methods by Inducing Coop-eration Among Rules, " University of Granada, Spain, Technical Report #DECSAI-000 101, Feb.2000.
[9] A. Gonzalez and R. Pérez, "SLAVE: A genetic learning system based on the iteractive appproach, "IEEE Tran. Fuzzy Syst., vol.7, pp.176-191, 1999.
[10] P. P. Bonissone, P. S. Khedkar, and Y. T. Chen, "Genetic algorithms for automated tuning of fuzzy controllers, a transportation aplication," Proc. Proc. Fifth Int. Fuzzy Syst. Assoc. (FUZZ-IEEE '96), pp. 674-680, 1996 .
[11] O. Cordón and F. Herrera, "A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples," Int. J. Approx. Reason., vol. 17, no. 4, pp. 369-407, 1997.
[12] O. Cordón and F. Herrera, "Hybridizing genetic algorithms with sharing scheme and evolution strategies for designing approximate fuzzy rulebased systems," Fuzzy Sets Syst., vol. 118, no. 2, pp. 235-255,2001.
[13] J. R. Velasco, "Genetic-based on-line learning for fuzzy IKocess control," Int. J. Intell. Syst., vol.13, no.10-11, pp.891-903, 1998.
[14] Jorge 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 Trans, Fuzzy Syst., Man, Cybern.-part B: Cybernetics, vol. 32, no.4, pp 526-536, Aug, 2002.
[15] C. T. Lin and C. S. G. Lee, “Neural-network-based fuzzy logic control and decision ystem,” IEEE Trans. Comput., vol. 40, pp. 1320–1336,Oct. 1991.
[16] Y.Jin, W.von Seelen, and B. Sendhoff, "On generating FC3 fuzzy rule systems from data using evolution strategies”, IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 29, no. 4, pp. 829-845, Aug. 1999.
[17] O.Cordón, M.J. Del Jesus, and F. Herrera, “Genetic learning of fuzzy rule-based classification systems cooperating with fuzzy reasoning methods.” Int. J. Intel. Syst,, vol 13, pp. 1025-1053, 1998.

[18] J.S.R.Jang,"ANFlS: adaptive-network-based fuzzy inference system." IEEE Trans, Syst., Man, Cybern., vol,23.pp.665-685.May1993.
[19] Baker, James E, "Reducing Bias and Inefficiency in the Selection Algorithm". Proceedings of the Second International Conference on Genetic Algorithms and their Application (Hillsdale, New Jersey: L. Erlbaum Associates): 14–21, 1987.
[20] L.-X. Wang and J. M. Mendel, “Generating fuzzy rules by learning from examples,” IEEE Trans. Syst., Man, Cybern., vol. 22, no. 6, pp. 1414–1427, Dec. 1992.
[21] K. Nozaki, H. Ishibuchi, and H. Tanaka, “A simple but powerful heuristic method for generating fuzzy rules from numerical data,” Fuzzy Sets Syst., vol. 86, no. 3, pp. 251–270, 1997.
[22] J.Casillas, O. Cordón, M. J. Del Jesus, and F. Herrera, “Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction.” IEEE Trans, Fuzzy Syst., vol. 13, pp 14-29, Feb, 2005.
[23] P. Thrift, “Fuzzy logic synthesis with genetic algorithms,” in Proc. 4th Int. Conf. Genetic Algorithms, R. K. Belew and L. B. Booker, Eds., San Mateo, CA, 1991, pp. 509–513.
[24] H.Ishibuchi, T. Murata, and I.B.Türks¸en, “Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems,” Fuzzy Sets Syst., vol. 89, no. 2, pp.135-150, 1997.

Tri Minh Huynh - Received the M.Sc. degree in Mathematics in 1997 from the HCMC University of Pedagogy, Viet Nam, the M.Sc. degree in Computer Science in 2000 from the University of Science, Ho Chi Minh City, Viet Nam. He is currently a lecturer in the Department of Information Technology at Sai Gon University, Ho Chi Minh City, Viet Nam. His current research interests include fuzzy systems, genetic fuzzy systems, neural networks, and their applications.

Huy Xuan Nguyen - Received his bachelor of mathematics at State Teachers' College, St. Peterbourg Russia, in 1973. He earned a Ph. D (in 1980) and a Doctor of Sciences (in 1990) in mathematics from Center of Computing, Soviet Union Academy of Sciences. Since 1992 he began to work as associate professor in Institute of Information Technology, Vietnamese Academy of Science and Technology. He published more than 30 scientific papers and is a member of Vietnamese Information Technology Association (since 1982) and member of Vietnamese Math Association (since 1982).

Bac Le - Received the B.S. degree in mathematics from the University of Education and the M.S. degree in computer sciences from the University of Technology, Viet Nam, in 1984 and 1990, respectively. He received his Ph.D. degree in mathematics for computers and computing systems from the University of Sciences, HCM City, Viet Nam, in 2000. He currently is Head of the Computer Science Division and Vice Dean of Postgraduate Training, University of Sciences. His current research interests include artificial intelligence, soft computing, knowledge discovery and data mining, and data hiding.