Fuzzy Rule Base Extraction by Bacterial Type Algorithms using selected T-norms

L. Gál[1,2], L. T. Kóczy[1,3]

[1] SZE, 9026 Győr, Egyetem tér 1.
Phone: 96 503 400, fax: 96 503 400
[2] NyME-SEK, 9700 Szombathely, Károlyi G. tér 4.
Phone: 94 504-300, fax: 94 504-404
[3] BME, 1111 Budapest, Műegyetem rkp. 3-9.
Phone: 1 463-1111, fax: 1 463-1110
e-mail: laci.gal@gmail.com , koczy@sze.hu

Abstract: In our previous paper we proposed the Trigonometric t-norm and t-conorm and we found that it showed good properties (among various types of t-norms and t-conorms) in Fuzzy Flip-Flop based Neural Networks.
In our previous research we have done a number of efforts to improve the efficiency of the *bacterial memetic type algorithms* in the field of the *fuzzy rule base identification*. The efficiency of the bacterial type algorithms were tested by Mamdani type inference system.
Now, we have examined how the *different t-norms* used instead of the widely used *min* operator affect the learning capabilities of the system applied, the speed of the convergence in case of various training algorithms. We implemented and tested some promising t-norms (*Algebraic*, *Hamacher* (p=0) and *Trigonometric*) for the Mamdani type inference system in order to reduce computational effort and processing time. This research is focused on *non-parametric t-norms*.

Keywords: T-norms, Mamdani inference system, bacterial type algorithms, Bacterial Memetic Algorithm

1. Introduction

Since Zadeh defined the triplet of standard fuzzy complement, union and intersection in his seminal paper [13], a large variety of fuzzy set operators have been proposed. Several classes of functions have been introduced whose members satisfy the axiomatic skeleton for fuzzy sets.

Recently we investigated the function approximation properties of Fuzzy Flip-Flop based Neural Networks built on various types of t-norms and t-conorms. We were looking for a *new t-norm pair* in order to improve the capabilities of these types of neural networks. We proposed the *Trigonometric t-norm and t-conorm* [3]. In that application, among the various types of fuzzy flip-flops [9] this new t-norm pair based J-K fuzzy flip-flop performed best, or at least at the same level as the best ones [3].

In our previous research we did a number of efforts to improve the efficiency of the *bacterial memetic type algorithms* in the field of the *fuzzy rule base identification* (**FRBI**) (IBMA, BMAM, MBMA, 3 Step BMA) [4, 5, 6]. On this area the efficiency of the bacterial type algorithms were tested by *Mamdani type inference system* with *trapezoidal shaped fuzzy membership functions*, *min fuzzy operator* and *COG defuzzification method*, as these are widely used and general enough. The application of the BMA type algorithms for FRBI is possible through the method proposed by Botzheim et al. [1, 2], which determined the derivatives needed for applying the *Levenberg-Marquardt method* (***LM***) [10].

During our past FRBI-related researches we examined that which of the various bacterial algorithms show better results concerning the size of the rule base, the function approximation capabilities of the identified rule base and system, and the speed of convergence of the algorithm applied in case of certain input-output data sets. The input-output data sets we used were derived from more or less complicated test functions and from data samples of real world applications, such as

One variable functions: sine wave section, two sine waves combined with different amplitude and frequency, pH-problem, Nickel-Metal Hydride accumulator charge characteristic [3];

Two variable functions: combination of various trigonometric functions, Inverse Coordinate Transform problem;

Six variable function: a six dimensional non-polynomial function (introduced in [11].

As we mentioned before, the model used can be characterized as a *Mamdani type inference system*, with *trapezoidal shaped membership functions*, *min fuzzy operator* and *COG defuzzification*. The most interesting algorithms from the viewpoint of the examinations made were the *Bacterial Evolutionary Algorithm* (***BEA***) [11], the *Improved Bacterial Memetic Algorithm* (***IBMA***) [4] and the *Modified Bacterial Memetic Algorithm* (***MBMA***) [5].

Now, we examined how the *different t-norms* used instead of the *min fuzzy operator* affect the *learning capabilities* of the system applied and the *speed of the convergence* in case of various training algorithms. We also investigated how accurately the system thus constructed could reproduce the output values from the known inputs. Beside the *min* and the new *Trigonometric t-norms* we used other *non-parametric t-norms* that seemed to be useful in our previous research [3]. This way we applied the following t-norms in our examinations as fuzzy operators: *minimum*, *algebraic*, *Hamacher* (p=0 in the parametric version), *Trigonometric*.

In order to use these in BMA type algorithms we had to determine all the partial *derivatives* needed for the *Jacobian matrix* used by the *Levenberg-Marquardt method.* This work was based on the method (elaborated for the min fuzzy operator) [2], the derivatives were modified for the corresponding t-norms, respectively.

The outline of this paper is as follows. After the Introduction, Section 2 deals with the fuzzy rule base identification process. In Section 3 we present various bacterial type evolutionary algorithms and calculate the derivatives for the Jacobian computation.. Various t-norms and co-norms are given in Section 4. Section 5 shows how we apply other t-norms in FRBI. Finally, Section 6 provides some *simulation results* of *fuzzy rule base identification processes* using *different t-norms* in a *Mamdani type inference system* and *three different bacterial type evolutionary algorithms*, followed by a brief Conclusion and References.

2. Fuzzy rule base identification process

When we use *Mamdani type inference systems* the system utilizes the fuzzy rule base to transform the input into output data. The fuzzy rule base can be built up from rules specified by human experts, or – if no such rules are available – can be formed by automatic fuzzy rule base identification based on input-output data obtained from certain systems.

Methods developed earlier by us [4, 5, 2] are able to generate quasi-optimal rule bases based upon such input-output data. The method we used for identifying fuzzy rule bases can be characterized as follows. The input data can be obtained from a certain system in operation, can be some commonly used benchmark data, or can be produced by some kind of test function. We dealt with 1, 2 and 6-dimensional input data. The *inference system* applied deployed *with trapezoidal shaped fuzzy membership functions*, in case of more than one input variables we utilized the *minimum t-norm* as *aggregation operator*, and the *COG method* was used for *defuzzification*.

In order to obtain *quasi-optimal fuzzy rule bases* the first step is to generate a population of random fuzzy rules, then evaluate the rule bases. The evaluation of the rule bases built upon the examining of the differences between the (input and) output data and the output data produced by the inference system (***MSE*** – *Mean Squared Error*). Then we begin to improve the rule base by some method that modifies the rule base towards having a smaller error. These methods can be global or local search methods, as well as a combination of both of them the so-called memetic algorithms. We applied several bacterial type evolutionary algorithms (***PBGA*** – *Pseudo Bacterial Genetic Algorithm* [12], **BEA**, ***BMA*** – *Bacterial Memetic Algorithm*, **IBMA**, ***BMAM*** – *Bacterial Memetic Algorithm with Modified Operator Execution Order* [4] and **MBMA**; more details in Section 3) for that purpose because they proved to be rather successful in the area of the *fuzzy rule base identification*. In case of bacterial memetic algorithms we applied the *Levenberg-Marquardt method* as the local searcher part as this has a very impressive speed of convergence. During the process of fine tuning of the rule bases (the training) we can analyze the speed of the convergence in case of various training algorithms, and we check how accurately the input-output data samples are reproduced by the system using the identified fuzzy rule base.

In case of Mamdani type inference system the relative importance of the j^{th} fuzzy variable in the i^{th} rule is:

$$\mu_{ij}(x_j) = \frac{x_j - a_{ij}}{b_{ij} - a_{ij}} N_{i,j,1}(x_j) + N_{i,j,2}(x_j) + \frac{d_{ij} - x_j}{d_{ij} - c_{ij}} N_{i,j,3}(x_j) \tag{1}$$

where a_{ij}, b_{ij}, c_{ij} *and* d_{ij} are the breakpoints of the trapezoidal shaped fuzzy membership functions, $a_{ij} \le b_{ij} \le c_{ij} \le d_{ij}$ must hold, and

$$N_{i,j,1}(x_j) = \begin{cases} 1, & if \;\; x_j \in [a_{ij}, b_{ij}] \\ 0, & otherwise \end{cases}, \quad N_{i,j,2}(x_j) = \begin{cases} 1, & if \;\; x_j \in [b_{ij}, c_{ij}] \\ 0, & otherwise \end{cases}, \quad N_{i,j,3}(x_j) = \begin{cases} 1, & if \;\; x_j \in [c_{ij}, d_{ij}] \\ 0, & otherwise \end{cases}. \tag{2}$$

The activation degree of the ith rule (if the t-norm is the minimum):

$$w_i = \min_{j=1}^{n} \mu_{ij}(x_j) \tag{3}$$

where n is the number of input dimensions. Then the *COG method* is applied as the *defuzzification* method [1].

In this paper we discuss the impact of the learning capabilities of such a system, where other t-norms are used instead of the minimum t-norm as aggregation operator.

In the cases of *non-memetic* bacterial algorithms (PBGA, BEA) it is enough to replace the minimum t-norm to another t-norm in the inference system. However, if the algorithm applied is one of the *bacterial memetic algorithms* (e.g. IBMA, MBMA) then the *derivatives* needed by the application of the *Levenberg-Marquardt method* have to be redetermined as well.

3. Bacterial type evolutionary algorithms

The original *genetic algorithm* (***GA***) was developed by Holland [7] and was based on the process of evolution of biological organisms. These processes can be easily applied in optimization problems where one individual corresponds to one solution of the problem.

Nawa, Hashiyama, Furuhashi and Uchikawa proposed a novel type of evolutionary algorithm called *Pseudo-Bacterial Genetic Algorithm* (***PBGA***) for fuzzy rule base extraction in 1997 [12]. The *Pseudo-Bacterial Genetic Algorithm* is essentially a special kind of g*enetic algorithm*. Its core contains a new genetic operation called *bacterial mutation*, which is inspired by the biological bacterial cell model, so this method mimics the microbial evolution phenomenon. Its basic idea is to improve the *parts* of *chromosomes* contained in each *bacterium*.

For the *bacterial algorithm*, the first step is to determine how the problem can be encoded in a *bacterium* (*chromosome*). Our task is to find the optimal *fuzzy rule base* for a pattern set. Thus, the parameters of the *fuzzy rules* must be encoded in the *bacterium*. In the general case the parameters of the rules are the *breakpoints* of the trapezoids, thus, a *bacterium* will contain these *breakpoints*.

The next step is to *optimize the parameters*. Therefore a procedure is working on changing the parameters, testing the model obtained by this way and selecting the best models (*bacterial mutation*). In the course of testing the input-output data used for training are compared to the input and the output of the model (SSE, *MSE*, BIC). The smaller is the error, the better is the performance of the model. The *inference system* used for the model calculations can be any of the various types of *fuzzy inference systems*.

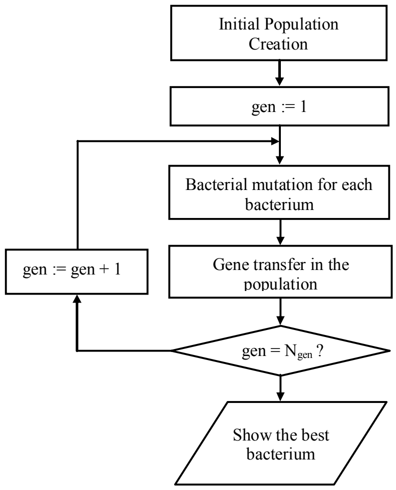
After introducing the *PBGA* further *bacterial type evolutionary algorithms* (based on the *bacterial mutation operation*) were proposed. The *BEA* is an enhancement over the PBGA (by the *gene transfer operation)*. The *BMA* is a *memetic algorithm*, which combines the BEA and the LM method for the global and local search task. The *IBMA* improves the efficiently of the BMA specifically for FRBI applications with trapezoidal shaped fuzzy membership functions. The *BMAM* is the further development of the BMA, where higher convergence speed can be achieved by the efficiently use of the LM method. The *MBMA* combines the steps of the IBMA and the BMAM, taking advantage of their beneficial properties when using *trapezoidal shaped fuzzy membership functions* in FRBI.

For our tests the *BEA*, the *IBMA* and the *MBMA* was selected from the above methods, as these are properly representative of the 3 different groups of *bacterial type evolutionary algorithms*.

3.1. Bacterial Evolutionary Algorithm (BEA)

BEA is based on the PGBA [11]. It introduces a new genetic operation, called *gene transfer operation*. This new operation establishes *relationships* between *bacteria*. BEA is not a memetic algorithm, as it doesn't contains a local search part (in our cases the Levenberg-Marquardt method), therefore, there is no need to determine certain derivatives in order to apply it. This way, it can be easily adapted for various applications. Its main operation is the *bacterial mutation*, and the supplementary operation is the *gene transfer operation*. Its simple to implement, the algorithm's complexity is low, but the speed of convergence and the obtainable accuracy is similarly low. Figure 1 shows the flowchart of the BEA.

In our cases – namely, *using different t-norms in the inference system* and applying BEA for FRBI – it is enough to modify the aggregation operation to another one in the inference system, which is extremely simple to implement.



Flowchart of the BEA

Figure 1. Flowchart of the BEA

3.2. Improved Bacterial Memetic Algorithm (IBMA)

IBMA is practical modification of the BMA, which combines the BEA and the LM method [4]. The *evolutionary part* is able to *find the global optimum region*, but is not suitable to find the local minimum in practice. The *gradient based part* is able to reach the *local optimum*, but is very sensitive to the initial position in the search space and is unable to avoid the local optimum. Combining global and local search is expected to be beneficial.

The main operations of the IBMA are the *bacterial mutation* followed by the *Levenberg-Marquardt method*, and the supplementary operation is the *gene transfer operation*. It is almost identical to the BMA except the *knot order violation handling method* in the Levenberg-Marquardt part (method *swap* for *handling knot order violation*). Figure 2 shows the flowchart of the IBMA.

Applying the BMA or IBMA for FRBI requires to determine certain derivatives (Jacobian matrix). In case of using the minimum t-norm in the Mamdani inference system the derivatives can be computed as these are determined (by J. Botzheim et al.) [1].

In our cases – *utilizing different t-norms* – the derivatives had to be *redetermined*, so the adaptation needs much more effort.

3.2.1. Derivatives for the Jacobian computation

The key in applying the LM method is how to get the LM update vector. To calculate the update vector the Jacobian matrix $\underline{J}$ with respect to the parameters in the rules has to be computed. In case of *minimum t-norm aggregation operation* and trapezoidal shaped fuzzy membership function based Mamdani inference system with COG defuzzification method this will be done as shown below, in a pattern by pattern (*pt*) basis. [2]

$$\underline{\underline{J}}[k] = \left[\frac{\partial y(\underline{x}^{(pt)})[k]}{\partial \underline{p}[k]} \right] \tag{4}$$

where *k* is the iteration variable. This can be written as follows:

$$\underline{\underline{J}} = \left[\frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial a_{11}} \frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial b_{11}} \cdots \frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial a_{12}} \cdots \frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial d_{1}} \cdots \frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial d_{R}} \right], \tag{5}$$

where

$$\begin{aligned}
\frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial a_{ij}} &= \frac{\partial y}{\partial w_i} \frac{\partial w_i}{\partial \mu_{ij}} \frac{\partial \mu_{ij}}{\partial a_{ij}} \\
\frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial b_{ij}} &= \frac{\partial y}{\partial w_i} \frac{\partial w_i}{\partial \mu_{ij}} \frac{\partial \mu_{ij}}{\partial b_{ij}} \\
\frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial c_{ij}} &= \frac{\partial y}{\partial w_i} \frac{\partial w_i}{\partial \mu_{ij}} \frac{\partial \mu_{ij}}{\partial c_{ij}} \\
\frac{\partial y\left(\underline{x}^{(pt)}\right)}{\partial d_{ij}} &= \frac{\partial y}{\partial w_i} \frac{\partial w_i}{\partial \mu_{ij}} \frac{\partial \mu_{ij}}{\partial d_{ij}}
\end{aligned} \tag{6}$$

w_i denotes the activation degree of the i^{th} rule (the t-norm is the minimum):

$$w_i = \min_{j=1}^{n} \mu_{ij}\left(x_j\right) \tag{7}$$

where *n* is the number of the input dimensions and μ_{ij} denotes the relative importance of the j^{th} fuzzy variable in the i^{th} rule.

$$\mu_{ij}(x_j) = \frac{x_j - a_{ij}}{b_{ij} - a_{ij}} N_{i,j,1}(x_j) + N_{i,j,2}(x_j) + \frac{d_{ij} - x_j}{d_{ij} - c_{ij}} N_{i,j,3}(x_j) \tag{8}$$

$$\begin{aligned}
N_{i,j,1}(x_j) &= \begin{cases} 1, & if \;\; x_j \in \left[a_{ij}, b_{ij}\right] \\ 0, & othervise \end{cases} \\
N_{i,j,2}(x_j) &= \begin{cases} 1, & if \;\; x_j \in \left[b_{ij}, c_{ij}\right] \\ 0, & othervise \end{cases} \\
N_{i,j,3}(x_j) &= \begin{cases} 1, & if \;\; x_j \in \left[c_{ij}, d_{ij}\right] \\ 0, & othervise \end{cases}
\end{aligned} \tag{9}$$

The i^{th} output is being cut in the height w_i, and with the *Center of Gravity* (COG) *defuzzification method* the output is calculated:

$$y(\underline{x}) = \frac{\sum_{i=1}^{N_{FuzzRules}} \int_{y \in \operatorname{supp}\mu_i(y)} y\mu_i(y)dy}{\sum_{i=1}^{N_{FuzzRules}} \int_{y \in \operatorname{supp}\mu_i(y)} \mu_i(y)dy} \quad (10)$$

If this defuzzification method is used, the integrals can be easily computed:

$$y(x) = \frac{1}{3} \frac{\sum_{i=1}^{N_{FuzzRules}} (C_i + D_i + E_i)}{\sum_{i=1}^{N_{FuzzRules}} 2w_i(d_i - a_i) + w_i^2(a_i - b_i + c_i - d_i)} \quad (11)$$

$$C_i = 3w_i(d_i^2 - a_i^2)(1 - w_i)$$
$$D_i = 2w_i^2(c_i d_i - a_i b_i)$$
$$E_i = w_i^3(c_i - d_i + a_i - b_i)(c_i - d_i - a_i + b_i)$$

It can be seen from (7) that w_i depends on the membership functions, and each membership function depends only on four parameters (breakpoints). So, the derivatives of w_i will be:

$$\frac{\partial w_i}{\partial \mu_{ij}} = \begin{cases} 1, & if \quad \mu_{ij} = \min_{k=1}^{n} \mu_{ik} \\ 0, & otherwise \end{cases} \quad (12)$$

$$\frac{\partial \mu_{ij}}{\partial a_{ij}} = \frac{x_j^{(p)} - b_{ij}}{\left(b_{ij} - a_{ij}\right)^2} N_{i,j,1}\left(x_j^{(p)}\right) \quad , \quad \frac{\partial \mu_{ij}}{\partial b_{ij}} = \frac{a_{ij} - x_j^{(p)}}{\left(b_{ij} - a_{ij}\right)^2} N_{i,j,1}\left(x_j^{(p)}\right) \quad (13)$$

$$\frac{\partial \mu_{ij}}{\partial c_{ij}} = \frac{d_{ij} - x_j^{(p)}}{\left(d_{ij} - c_{ij}\right)^2} N_{i,j,3}\left(x_j^{(p)}\right) \quad , \quad \frac{\partial \mu_{ij}}{\partial d_{ij}} = \frac{x_j^{(p)} - c_{ij}}{\left(d_{ij} - c_{ij}\right)^2} N_{i,j,3}\left(x_j^{(p)}\right) \quad (14)$$

$\frac{\partial y}{\partial w_i}$ and the derivatives of the output membership functions' parameters can be also computed from (11).

After calculating the Jacobian matrix the LM update vector can be calculated as follows:

$$\underline{s} = -\begin{bmatrix} \underline{\underline{J}} \\ \sqrt{\lambda}\,\underline{\underline{I}} \end{bmatrix}^{+} \begin{bmatrix} \underline{y}(\underline{p}) - \underline{t} \\ \underline{0} \end{bmatrix} \quad (15)$$

where the operator $^+$ denotes the Moore-Penrose pseudoinverse.

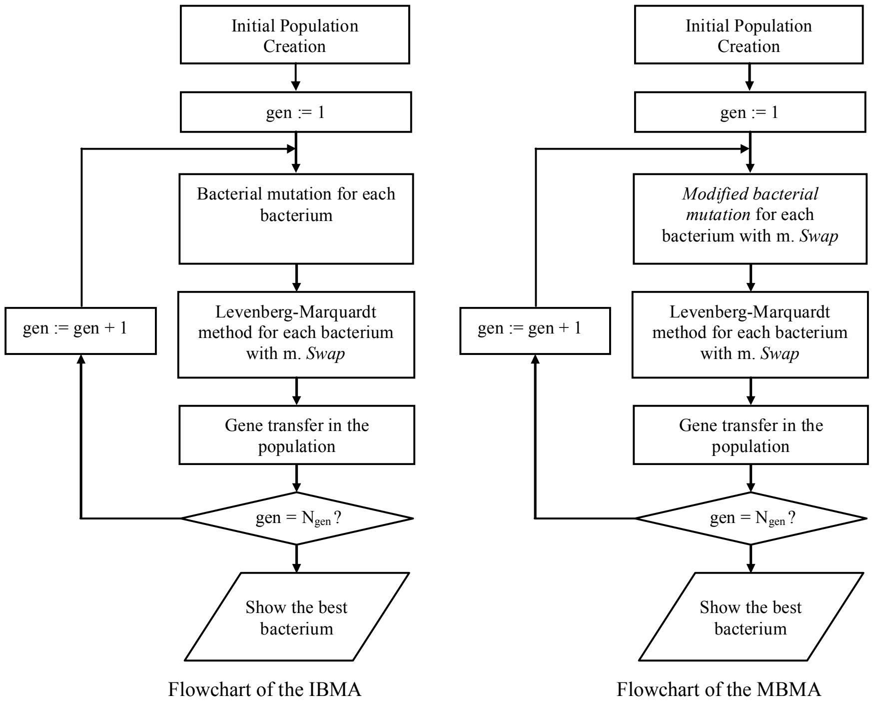


Figure 2. Flowcharts of IBMA and MBMA

3.3. Modified Bacterial Memetic Algorithm (MBMA)

MBMA is a very recent approach (2008) [5]. It differs from the IBMA in the *bacterial mutation* step, as the MBMA utilizes a *modified version of the bacterial mutation operation*. In this *modified bacterial mutation* a number of Levenberg-Marquardt steps are executed *after each bacterial mutational step*. This way higher convergence speed and lower MSE values can be achieved. The main steps are: *bacterial mutational steps + LM iterations*, *LM iterations*, *gene transfer*. Figure 2 shows the flowchart of the MBMA.

As this algorithm is working with the same operation set as the IBMA there is nothing else need to implement for utilizing the other t-norms, if the modifications for the IBMA are done.

4. Various t-norms an co-norms

Klement, Mesiar and Pap enumerate and give the basic definitions and properties of the most general fuzzy operations in [8] including also graphical illustrations and comparisons. Table 1 shows some selected t-norms and t-conorms. Each class contains parameterized as well as non-parameterized operators. All t-norms could be used as aggregation operator in the Mamdani type inference system. We implemented the *min*, *Algebraic*, *Hamacher* (p=0) (= *Dombi* (α=1)) types from the t-norms in Table 1 and a new t-norm (*Trigonometric*) for FRBI.

Parameters w, α, v and s lie within the open interval $(0, \infty)$, $d \in [0,1]$, $s \neq 1$ and $p \neq 0$.

Table 1. T-norms and t-conorms

Fuzzy operation	t-norm; i (a, b)	co-norm; u (a, b)
Standard (min-max)	$\min(a,b)$	$\max(a,b)$
Algebraic	ab	$a+b-ab$
Drastic	a when b=1, b when a=1, 0 otherwise	a when b=0, b when a=0, 1 otherwise
Łukasiewicz	$\max(0, a+b-1)$	$\min(1, a+b)$
Yager	$1-\min\left[1,((1-a)^w+(1-b)^w)^{1/w}\right]$	$\min\left[1,(a^w+b^w)^{1/w}\right]$
Dombi	$\dfrac{1}{1+\left[(1/a-1)^{\alpha}+(1/b-1)^{\alpha}\right]^{1/\alpha}}$	$\dfrac{1}{1+\left[(1/a-1)^{-\alpha}+(1/b-1)^{-\alpha}\right]^{-1/\alpha}}$
Hamacher	$\dfrac{ab}{v+(1-v)(a+b-ab)}$	$\dfrac{a+b-(2-v)ab}{1-(1-v)ab}$
Frank	$\log_s\left[1+\dfrac{(s^a-1)(s^b-1)}{s-1}\right]$	$1-\log_s\left[1+\dfrac{(s^{1-a}-1)(s^{1-b}-1)}{s-1}\right]$
Dubois-Prade	$\dfrac{ab}{\max(a,b,d)}$	$\dfrac{a+b-ab-\min(a,b,1-d)}{\max(1-a,1-b,d)}$
Schweizer-Sklar	$\max(0,a^p+b^p-1)^{1/p}$	$1-\max(0,(1-a)^p+(1-b)^p-1)^{1/p}$

4.1. The Trigonometric t-norm and co-norm

The fuzzy literature offers a large variety of triangular operators; researchers still propose again and again new fuzzy operations to be used in a given field. Obviously, the performance of fuzzy systems depends on the choice of different triangular operators. Despite the variety of available fuzzy set operators, however, the classic triplet of complement, intersection and union still bear particular significance, especially in the practical applications. The big challenge for fuzzy researchers is to fit the fuzzy sets into the context of applications.

In [3] a new triangular t-norm and t-conorm was presented. The novel fuzzy operations combined with the standard negation are applied in a practical problem, namely, they are proposed as suitable triangular norms for defining a fuzzy flip-flop based neuron.

The proposed connectives consist of simple combinations of trigonometric functions. Naturally, the new t-norm and t-conorm satisfy the axiomatic skeleton for fuzzy sets and are continuous. The basic motivation for constructing new norms was to have fuzzy flip-flops with sigmoid transfer characteristics in some particular cases.

For fuzzy intersection and union we have defined the following:

The Trigonometric t-norm is specified by the function

$i_T : [0,1] \times [0,1] \rightarrow [0,1]$ defined as:

$$i_T(x, y) = \frac{2}{\pi} \cdot \arcsin\left(\sin\left(x \frac{\pi}{2}\right) \cdot \sin\left(y \frac{\pi}{2}\right)\right) \tag{16}$$

The Trigonometric t-conorm is specified by the function

$$u_T : [0,1] \times [0,1] \rightarrow [0,1]$$

$$u_T(x, y) = \frac{2}{\pi} \cdot \arccos\left(\cos\left(x \frac{\pi}{2}\right) \cdot \cos\left(y \frac{\pi}{2}\right)\right) \tag{17}$$

The characters of the well known *standard*, *algebraic*, *Hamacher*, *Dombi*, *Yager* and of the novel *Trigonometric t-norms* for selected parameter values (p=0, $w = 2$ and $\alpha = 2$) are illustrated in Figure 3. It can be seen, that the *Yager t-norm* produced a horizontal plate at the beginning of the interval.

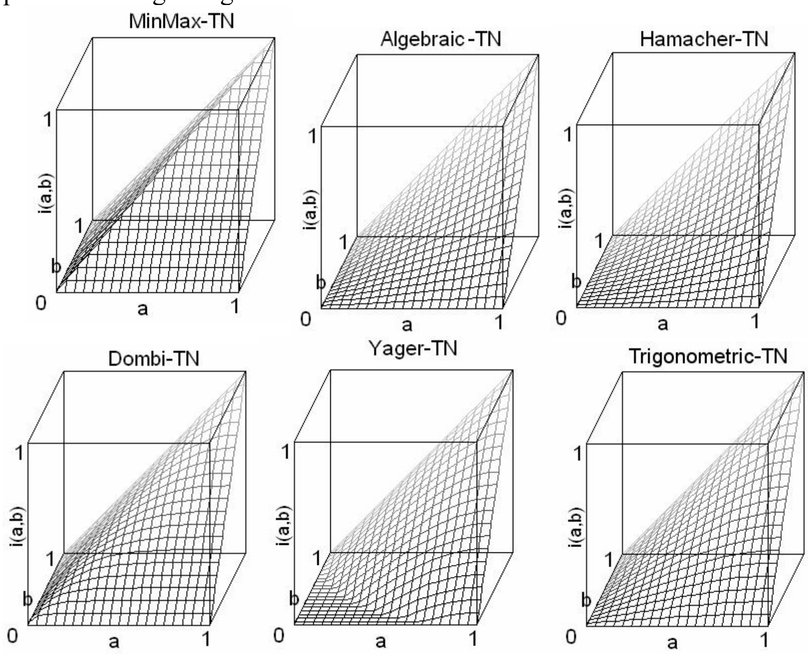


Figure 3. Characteristics of some selected t-norms

5. Applying other t-norms in FRBI

Beside the *minimum t-norm* we decided to implement some promising t-norms for the Mamdani type inference system and for the FRBI process. We avoided using parametric t-norms (much more complicated derivatives and computation complexity) and t-norms with min or max operations (because the horizontal plates in the graph should be avoided as those result 0 valued derivatives).

We have selected the following t-norms for comparison in the area of FRBI:

- Standard $\min(a,b)$ (18)
- Algebraic $a \cdot b$ (19)
- Hamacher (p=0) (Dombi (α=1)) $\dfrac{a \cdot b}{a+b-a \cdot b}$ (20)
- Trigonometric $\dfrac{2}{\pi} \cdot \arcsin\left(\sin\left(x\dfrac{\pi}{2}\right) \cdot \sin\left(y\dfrac{\pi}{2}\right)\right)$ (21)

When using the Levenberg-Marquardt method in FRBI process (BMA, IBMA, BMAM, MBMA), you need the derivatives of the multiple argument aggregation expressions.

5.1. Derivatives for the Jacobian computation

The derivatives can be calculated in the same way as for the minimum t-norm based ones (3.2.1), except the parts concerned by the t-norm used (3, 6, 7, *12*). These parts can be calculated by the following.

5.1.1. Algebraic

$$i_{A\lg}(a,b) = a \cdot b \tag{22}$$

$$w_i = \mu_{11} \cdot \mu_{12} \cdot \mu_{13} = \prod_{j=1}^{n} \mu_{ij} = \mathrm{P}_i \tag{23}$$

$$\frac{\partial w_i}{\partial \mu_{ij}} = \frac{\mathrm{P}_i}{\mu_{ij}} = \frac{1}{\mu_{ij}} \cdot w_i \tag{24}$$

if $w_i > 0$.

5.1.2. Hamacher (p=0) *(=Dombi (α=1))*

The Hamacher (p=0) operator is the Dombi operator if α=0.

$$i_{Ham}(a,b) = \frac{a \cdot b}{a+b-a \cdot b} \tag{25}$$

$$w_i = \frac{\mu_{11} \cdot \mu_{12} \cdot \mu_{13}}{\mu_{11} \cdot \mu_{12} + \mu_{11} \cdot \mu_{13} + \mu_{12} \cdot \mu_{13} - 2\mu_{11} \cdot \mu_{12} \cdot \mu_{13}} \tag{26}$$

$$w_i = \frac{\mu_{11} \cdot \mu_{12} \cdot \mu_{13} \cdot \mu_{14}}{\mu_{11} \cdot \mu_{12} \cdot \mu_{13} + \mu_{11} \cdot \mu_{12} \cdot \mu_{14} + \mu_{11} \cdot \mu_{13} \cdot \mu_{14} + \mu_{12} \cdot \mu_{13} \cdot \mu_{14} - 3 \cdot \mu_{11} \cdot \mu_{12} \cdot \mu_{13} \cdot \mu_{14}} \tag{27}$$

$$w_i = \frac{1}{1 - n + \sum_{j=1}^{n} \frac{1}{\mu_{ij}}} = \frac{1}{S_r} \tag{28}$$

$$\frac{\partial w_i}{\partial \mu_{ij}} = \frac{1}{\mu_{ij}^2 \cdot \left(1 - n + \sum_{j=1}^{n} \frac{1}{\mu_{ij}}\right)^2} = \frac{1}{\mu_{ij}^2 \cdot S_r^2} = \frac{1}{\mu_{ij}^2} \cdot w_i^2 \tag{29}$$

if $w_i > 0$.

5.1.3. Trigonometric

$$i_{Trig}(a,b) = \frac{2}{\pi} \arcsin\left[\sin\left(a \cdot \frac{\pi}{2}\right) \cdot \sin\left(b \cdot \frac{\pi}{2}\right)\right] \tag{30}$$

$$w_i = \frac{2}{\pi} \arcsin\left[\sin\left(\mu_{11} \cdot \frac{\pi}{2}\right) \cdot \sin\left(\mu_{12} \cdot \frac{\pi}{2}\right) \cdot \sin\left(\mu_{13} \cdot \frac{\pi}{2}\right)\right] \tag{31}$$

$$w_i = \frac{2}{\pi} \arcsin\left[\prod_{i=1}^{n} \sin\left(\mu_{ij} \cdot \frac{\pi}{2}\right)\right] = \frac{2}{\pi} \arcsin[\mathrm{P}_i] \tag{32}$$

$$\mathrm{P}_i = \prod_{i=1}^{n} \sin\left(\mu_{ij} \cdot \frac{\pi}{2}\right) \tag{33}$$

$$\frac{\partial w_i}{\partial \mu_{ij}} = \frac{\cos\left(\mu_{ij} \cdot \frac{\pi}{2}\right) \cdot \frac{\mathrm{P}_i}{\sin\left(\mu_{ij} \cdot \frac{\pi}{2}\right)}}{\sqrt{1 - \mathrm{P}_i^2}} \tag{34}$$

$$\frac{\partial w_i}{\partial \mu_{ij}} = \frac{\cos\left(\mu_{ij} \cdot \frac{\pi}{2}\right)}{\sin\left(\mu_{ij} \cdot \frac{\pi}{2}\right)} \cdot \frac{\mathrm{P}_i}{\sqrt{1 - \mathrm{P}_i^2}} = \cot\left(\mu_{ij} \cdot \frac{\pi}{2}\right) \cdot \frac{\mathrm{P}_i}{\sqrt{1 - \mathrm{P}_i^2}} \tag{35}$$

if $0 < w_i < 1$.

6. Simulations

We have run extensive simulations in order to examine how the *different t-norms affected the learning capabilities* of the system applied, the *speed of the convergence* in case of using different training algorithms.

With 1-dimensional input there is no sense to examine the effect on using different t-norms, as these are only required in the Mamdani inference machine for more than one input. Therefore, the 1-dimensional ones from our standard test functions were omitted; the tests were performed with two and six variable test functions. These were as follows:

6.1. Test functions, t-norms tested, training algorithms used

a, Test function with 2 input variables (2iv)

$$\begin{gathered} y = f_2(\underline{x}) = \sin^5(0.5 \cdot x_1) \cdot \cos(0.7 \cdot x_2) \\ x_1 \in [0...3\pi], x_2 \in [0...3\pi] \end{gathered} \tag{36}$$

200 samples
T-norms: Minimum, Algebraic, Hamacher (p=0), Trigonometric
Training algorithms: BEA, IBMA, MBMA

b, Test function with 6 input variables (6iv)

$$\begin{gathered} y = f_3(\underline{x}) = x_1 + x_2^{0.5} + x_3 \cdot x_4 + 2 \cdot e^{2 \cdot (x_5 - x_6)} \\ x_1 \in [1...5], x_2 \in [1...5], x_3 \in [0...4], \\ x_4 \in [0...0.6], x_5 \in [0...1], x_6 \in [0...1.2] \end{gathered} \tag{37}$$

500 samples
T-norms: Minimum, Algebraic, Hamacher (p=0), Trigonometric
Training algorithms: BEA, IBMA, MBMA

Common parameters:

Fuzzy rules: 5
Population size: 7
Clones: 7
Gene transfers per generation: 3
MSE: 20 runs average
BEA: 100.000 evaluations/run
IBMA, MBMA: 10.000 evaluations/run

6.2. Results

Figures 4-6 and Figures 7-9 present the graphs of the simulations in case of the test function with 2 and 6 input variables, respectively. We compared the behavior of the systems trained with BEA, IBMA and MBMA algorithms. The graphs show the relation between the number of model performance evaluations (horizontal axis) and the model mean or median of the train MSE values (vertical axis).

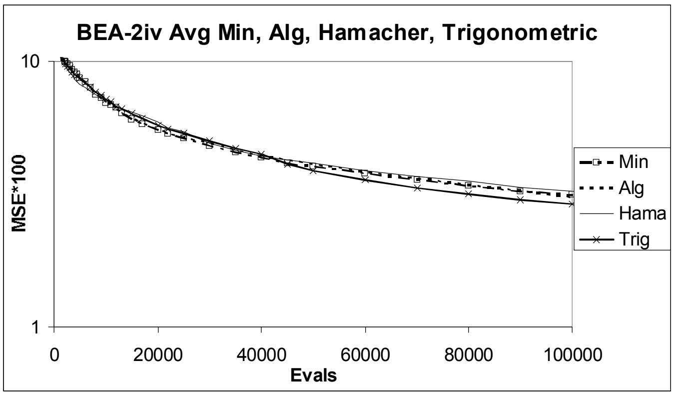


Figure 4. 2iv, BEA

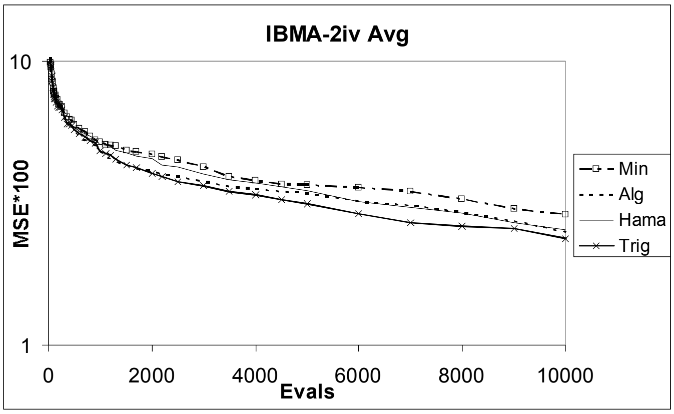


Figure 5. 2iv, IBMA

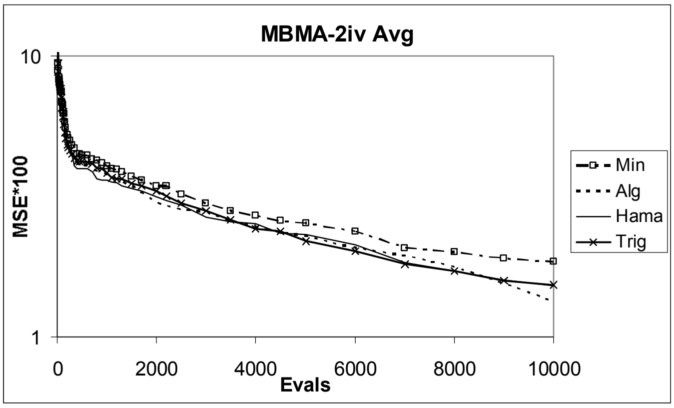


Figure 6. 2iv, MBMA

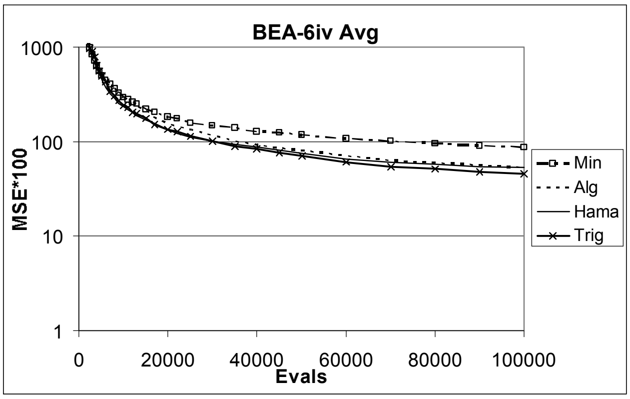


Figure 7. 6iv, BEA

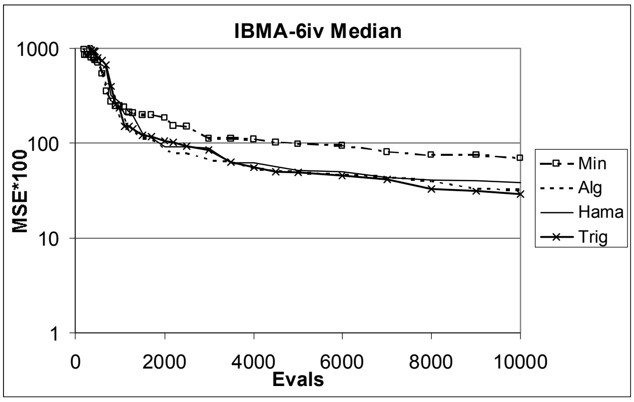


Figure 8. 6iv, IBMA

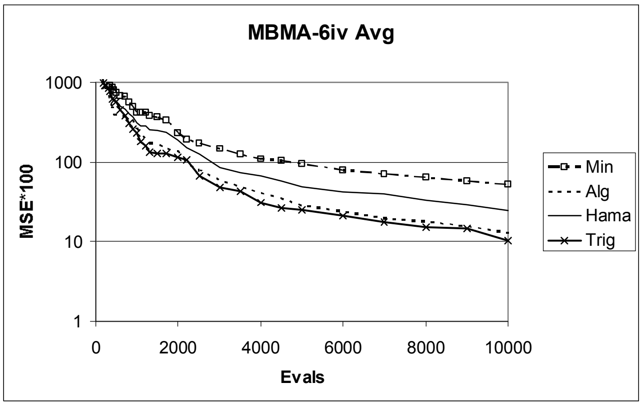


Figure 9. 6vi, MBMA.

Comparing the simulation results it is showed that the system based on Trigonometric and algebraic t-norms provide the best results, respectively. Furthermore, in the last case (Figure 9) it was enough to run about *one fourth* of the 10.000 evaluations (2.500) with the *Trigonometric t-norm* to achieve the same auspicious error values as the final error with the *minimum t-norm*.

Table 2 summarizes the simulation results by indicating the average and median of the train MSE values of the various systems trained with BEA, IBMA and MBMA algorithms, case of test function with 2 and 6 input variables.

In the table the worst MSE results are highlighted in *italics*, while the best MSE values are highlighted by **bold** characters. The *improvements* related to the *minimum aggregation operation* based values are also calculated in the best cases. For example, –80% means five times better result. In the last column we show the total number of *model performance evaluations* (Evals) for one simulation (one run), and additionally below them the approximately number of evaluations needed with the *Trigonometric t-norm* to achieve the same error as with the minimum t-norm (~Evals). For example, 10000 / 2500 means that one fourth of training cycles (and time) is required to achieve the same result.

*Table 2. Simulation results, MSE*100 values*

T-norm	Minimum		Algebraic		Hamacher		Trigonometric		Evals
	Avg.	Med.	avg.	med.	avg.	Med.	avg.	Med.	~Evals
2iv, BEA	*3.148*	*3.01*	3.064	3.07	*3.24*	*3.38*	**2.88**	**2.9**	100000
Improvement							–8%	–4%	81000
2iv, IBMA	*2.886*	*2.661*	2.509	2.527	2.546	2.597	**2.378**	**2.235**	10000
							–18%	–16%	6000
2iv, MBMA	*1.844*	*1.809*	**1.338**	**1.38**	1.519	1.515	1.528	**1.374**	10000
			-27%	-24%			–17%	–24%	7100
6iv, BEA	*86.42*	*86.98*	53.12	50.21	53.34	53.58	**45.32**	**40.09**	100000
							–48%	–54%	36000
6iv, IBMA	*76.32*	*69.68*	33.33	32.66	44.99	38.56	**29.2**	**28.87**	10000
							–62%	–59%	3500
6iv, MBMA	*52.85*	*50.82*	13.02	16.04	24.45	23.77	**10.29**	**7.158**	10000
							–81%	–86%	2500

7. Conclusions

In our previous researches we have investigated the potential enhancements of the *Bacterial Memetic Algorithm used for automatic fuzzy rule base identification*. We have proposed some new algorithms in this field (IBMA, BMAM, MBMA and 3 Step BMA). Furthermore, we proposed a new triangular norm and conorm pair, called *Trigonometric t-norm a t-conorm*, which is proved to be rather successful in fuzzy flip-flop based neural networks.

In this work we have examined how the *different t-norms* used instead of the *min fuzzy operator* affect the *learning capabilities* of the system applied, the *speed of the convergence* in case of different training algorithms, and *how accurately* input-output data samples could be reproduced by using the fuzzy rule base obtained from the *fuzzy rule base identification process*. We have extensively tested the promising *non-parametric t-norms*, included the new *Trigonometric t-norm*. This way we applied the following t-norms in our examinations as fuzzy aggregation operator: *minimum*, *algebraic*, *Hamacher (p=0) (Dombi (α=0))*, *Trigonometric*, and tested them by three different training algorithms, namely BEA, IBMA and MBMA.

In order to use the other t-norms in the inference system trained by a memetic algorithm (IBMA, MBMA) we had to redetermine a number of derivatives in the Jacobian matrix used for the Levenberg-Marquardt method.

We expected that using other t-norms as aggregation operator in the Mamdani type inference system result in improved learning capabilities – higher speed of convergence and lower overall MSE values.

The simulations showed that it was reasonable to investigate the use of the other t-norms in the inference system, because in almost all cases the convergence speed and the accuracy of the fuzzy rule bases had been improved (except in the case of the BEA with 2 input variables test function, where there were no appreciable differences – 8% for the Trigonometric t-norm). The overall errors have been decreased by 18% (simple test function) up to *80%* (more complex test function), when using the Trigonometric t-norm instead of the conventional minimum operation.

While this research was focused on non-parametric t-norms, further effort to investigate the behavior of the system based on various parametric t-norms well known from the literature seems to be necessary.

Acknowledgement:

Research supported by the National Scientific Research Fund Grant OTKA K75711, further by Széchenyi István University Main Research Direction Grant 2010 and Scientific Committee, University of West Hungary, Campus Savaria.

References

[1] Botzheim, J., Cabrita, C., Kóczy, L. T., Ruano, A. E.: *Estimating Fuzzy Membership Functions Parameters by the Levenberg-Marquardt Algorithm*, FUZZ-IEEE 2004, Budapest, Hungary pp. 1667-1672.

[2] Botzheim, J., Cabrita, C., Kóczy, L. T., Ruano, A. E.: *Fuzzy Rule Extraction by Bacterial Memetic Algorithm*, IFSA 2005, Beijing, China, 2005, pp.1563-1568.

[3] Gál, L., Lovassy, R., Kóczy, L. T.: *Function Approximation Performance of Fuzzy Neural Networks Based on Frequently Used Fuzzy Operations and a Pair of New Trigonometric Norms*, WCCI 2010 IEEE World Congress on Computational Intelligence, FUZZ-IEEE, July, 18-23, 2010, Barcelona, Spain, pp. 1514-1521.

[4] Gál, L., Botzheim, J., Kóczy, L. T.: *Improvements to the Bacterial Memetic Algorithm used for Fuzzy Rule Base Extraction*, Computational Intelligence for Measurement Systems and Applications, CIMSA 2008, Istanbul, Turkey, 2008, pp. 38-43.

[5] Gál, L., Botzheim, J., Kóczy, L. T.: *Modified Bacterial Memetic Algorithm used for Fuzzy Rule Base Extraction*, 5th International Conference on Soft Computing as Transdisciplinary Science and Technology, CSTST 2008, Paris, France, 2008.

[6] Gál, L., Kóczy, L. T., Lovassy, R.: *Three Step Bacterial Memetic Algorithm*, Proc. of 14th IEEE International Conference on Intelligent Engineering Systems INES 2010, Las Palmas of Gran Canaria, Spain, pp. 31-36.

[7] Holland, J. H.: *Adaptation in Nature and Artificial Systems: An Introductory Analysis with Applications to Biology*, Control, and Artificial Intelligence, The MIT Press, Cambridge, MA, 1992.

[8] Klement, E. P., Mesiar, R., Pap, E.: *Triangular Norms*, Series: Trends in Logic , vol.8, 2000.

[9] Koshak, E., Noore, A., Lovassy, R.: *Intelligent Reconfigurable Universal Fuzzy Flip-Flop*, IEICE Electronics Express, Vol. 7, No. 15, 2010, pp. 1119-1124.

[10] Marquardt, D. *An Algorithm for Least-Squares Estimation of Nonlinear Parameters*, SIAM J. Appl. Math., 11, 1963, pp. 431-441.

[11] Nawa, N. E., Furuhashi, T.: *Fuzzy Systems Parameters Discovery by Bacterial Evolutionary Algorithms*, IEEE Transactions on Fuzzy Systems 7, 1999, pp. 608-616.

[12] Nawa, N. E., Hashiyama, T., Furuhashi, T., Uchikawa, Y.: *A study on fuzzy rules discovery using pseudo-bacterial genetic algorithm with adaptive operator*, Proceedings of IEEE Int. Conf. on Evolutionary Computation, ICEC'97, 1997.

[13] Zadeh, L. A.: *Fuzzy Sets*, Information and Control 8, 1965, pp. 338-353.