TECHNICAL TRANSACTIONS | CZASOPISMO TECHNICZNE
AUTOMATIC CONTROL | AUTOMATYKA
4-AC/2013

LÁSZLÓ GÁL*, RITA LOVASSY**, LÁSZLÓ T. KÓCZY***

BACTERIAL TYPE ALGORITHMS USED FOR FUZZY RULE BASE EXTRACTION

WYKORZYSTANIE ALGORYTMÓW BAKTERYJNYCH DO WYDOBYCIA BAZY REGUŁ ROZMYTYCH

Abstract

The paper gives an overview of various bacterial type evolutionary algorithms used for fuzzy rule based identification. In order to find an optimal rule base from the input-output training data set, several improved algorithms have been developed in recent years. The task is to increase the models' accuracy and convergence speeds by modifying a part of the Mamdani-type inference system.

Keywords: *FRBI, pseudo-bacterial genetic algorithm, bacterial evolutionary algorithm, bacterial memetic algorithm with memetic mutation, progressive bacterial algorithm*

Streszczenie

W artykule zawarto przegląd ewolucyjnych algorytmów bakteryjnych wykorzystywanych do identyfikacji bazy reguł rozmytych. W celu znalezienia optymalnej bazy reguł ze zbioru danych testowych wejściowych i wyjściowych, w ostatnich latach opracowano kilka ulepszonych algorytmów. Zamysłem przedstawionych tu badań jest uzyskanie wzrostu dokładności modeli oraz szybkości ich zbieżności poprzez modyfikację systemów wnioskowania typu Mamdaniego.

Słowa kluczowe: *FRBI, pseudo-bakteryjny algorytm genetyczny, bakteryjny algorytm ewolucyjny, bakteryjny algorytm memetyczny z mutacją memetyczną, progresywny algorytm bakteryjny*

* Ph.D. László Gál, e-mail: laci.gal@gmail.com, Department of Technology and Applied Informatics, University of West Hungary, Szombathely.

** Ph.D. Rita Lovassy, Institute of Microelectronics and Technology, Kandó Kálmán Faculty of Electrical Engineering, Óbuda University Budapest.

*** Prof. D.Sc. Ph.D. László T. Kóczy, Department of Automation, Faculty of Engineering Sciences, Széchenyi István University Györ; Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics.

1. Introduction

Special employment of fuzzy systems, the fuzzy controllers are present in every day applications. The design of fuzzy controllers is concerned with the calculus of fuzzy rules [17]. The construction of fuzzy rules, mathematically, sets of fuzzy relations, is one of the key problems of fuzzy reasoning and control. An important task in fuzzy rule extraction is how to obtain a set of appropriate fuzzy rules for a given system. The application of bacterial type algorithms (Pseudo-Bacterial Genetic Algorithm – PBGA and Bacterial Evolutionary Algorithm – BEA) for fuzzy rule base identification (FRBI) was proposed in [16, 17]. A modified, memetic version of the bacterial evolutionary algorithm called the Bacterial Memetic Algorithm – BMA was also proposed in [3]. The combination of evolutionary and gradient based algorithms was used rather successfully in global optimization approaches. In order to improve the system's convergence speed and their function approximation capabilities a series of new bacterial algorithms has been proposed by us: the Improved Bacterial Memetic Algorithm (IBMA) [4], the Bacterial Memetic Algorithm with Memetic Mutation (BMAM) [5], the Modified Bacterial Memetic Algorithm (MBMA) [6], the 3Step BMA (3BMA) [8], and the Progressive Bacterial Algorithm (PBA) [10]. In our previous papers [7, 9], we had examined how using different t-norms instead of the conventional 'min' fuzzy operator affected the system's learning capability and the convergence speed of the Mamdani-type inference system [12]. We had studied how accurately input-output data samples could be reproduced by using fuzzy rule bases obtained via an automatic rule identification process. The extensive investigations showed that the IBMA or MBMA training algorithms with non-parametric *t*-norms like algebraic, trigonometric [7], Hamacher product and a parametric operator like the Hamacher *t*-norm (with optimized parameter value) definitely improved the system learning capability.

After the Introduction, in Section 2 we will briefly review the PBGA, BEA, BMA, IBMA, BMAM, MBMA and PBA algorithms. This section provides some typical simulation results of fuzzy rule based identification processes using various algorithms. Finally, some conclusions are taken and references listed.

2. Bacterial Type Evolutionary and Memetic Algorithms

2.1. Pseudo-Bacterial genetic Algorithm (PBGA)

Nawa et al. [16] proposed a novel type of evolutionary algorithm called the Pseudo--Bacterial Genetic Algorithm (PBGA) for fuzzy rule based extraction. This is a special kind of genetic algorithm with a core that contains a new genetic operation called bacterial mutation. This method mimics the microbial evolution phenomenon. Its basic idea is to improve the parts of chromosomes contained in each bacterium. Bacteria can transfer genes to other bacteria. This mechanism is used in bacterial mutation. For the bacterial algorithm, the first step is to determine how the problem can be encoded in a bacterium (chromosome). The 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 general, 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. The inference system used for model calculations can be any of the various types of fuzzy inference systems (e.g. FRI [11]).

The main steps of the PBGA are as follows:

– Create the initial population: N_{Ind} individuals are randomly created and evaluated. (N_{Ind} is the number of individuals in the population.) Each individual contains N_{Fuzzy_rules} fuzzy rules encoded in the chromosome (N_{Fuzzy_rules} is the number of fuzzy rules of the desired model).
– Apply the bacterial mutation to each individual:
  – Each individual is selected one by one.
  – N_{Clones} copies of the selected individual are created ('clones').
  – Choose the same part or parts randomly from the clones and mutate it (except one single clone that remains unchanged during this mutation cycle).
  – Select the best clone and transfer its mutated part or parts to the other clones.
  – Repeat the part choosing-mutation-selection-transfer cycle until all the parts are mutated and tested exactly once.
  – The best individual is to remain in the population, all other clones are deleted.
  – This process is repeated until all the individuals have gone through the bacterial mutation.
– Apply conventional genetic operations (selection, reproduction and crossover).
– Repeat the procedure above from the bacterial mutation step until a certain termination criterion is satisfied (e.g. maximum number of generations).

The algorithm works efficiently where weak relationships between the parameters encoded in the chromosome exist.

2.2. Bacterial Evolutionary Algorithm (BEA)

Bacterial Evolutionary Algorithm (BEA) is based on the PBGA supported by a new genetic operation called the gene transfer operation [17]. This new operation establishes relationships among the individuals of the population.

The main steps of the gene transfer operation are:

– Sort the population according to the fitness values and divide it into two halves. The half that contains the better individuals is called 'superior half' while the other half is the 'inferior half'.
– Choose one individual (the 'source chromosome') from the superior half and another one (the 'destination chromosome') from the inferior half.

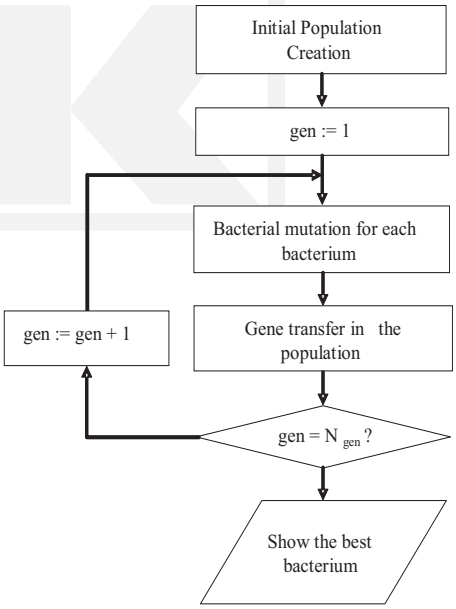

Fig. 1. Flowchart of the BEA

– Transfer a part of the source chromosome to the destination chromosome (select the part randomly or by a predefined criterion).
– Repeat the steps above N_{Inf} times (N_{Inf} is the number of 'infections' to occur in one generation).

The gene transfer operation can be used in place of selection, reproduction, or the crossover in the algorithm described by the PGBA. The BEA flowchart can be seen in Fig. 1.

2.3. Bacterial Memetic Algorithm (BMA)

Bacterial Memetic Algorithm [3] combines evolutionary and local search algorithms [15], in particular the BEA and Levenberg-Marquardt (LM) methods [14]. The algorithm main steps are (after the creation of the initial population):
– the bacterial mutation to each individual,
– a few iterations of the LM method,
– gene transfer operation applied per generation a number of infection times.

The above steps are repeated from the bacterial mutation until a certain stopping criterion is satisfied. When applying this method in the case of the trapezoidal shaped fuzzy membership functions it often happens that the trapezoid breakpoints do not satisfy a certain relationship, namely, the membership function defined by the four breakpoints cannot be interpreted as a fuzzy membership function. In this case (knot order violation, KOV) an update vector reduction factor is applied in the LM method [2].

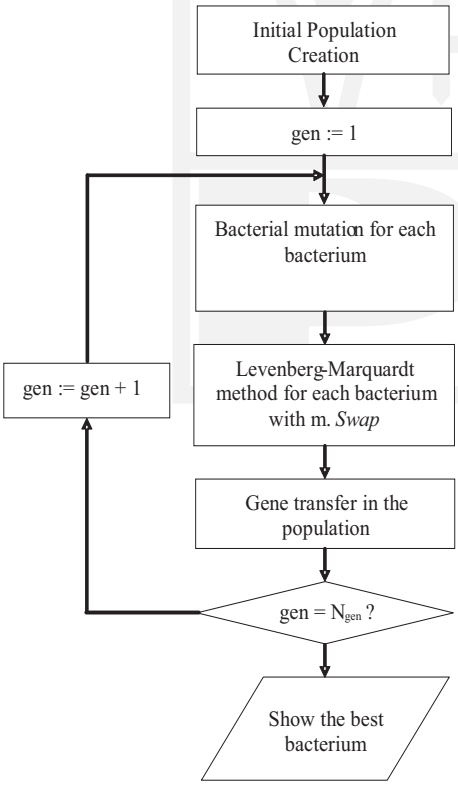

Fig. 2. Flowchart of the IBMA

2.4. Improved Bacterial Memetic Algorithm (IBMA)

Gál et al. proposed in [4] the so called *Improved Bacterial Memetic Algorithm* for a more efficient handling the knot order violations occurred in the bacterial memetic algorithm used for fuzzy rule based extraction. This method performs slightly better than the method used before. The 'merge of the violating knots into a single knot', and 'swapping of the knots that are in the wrong order' methods are introduced, which are easy to implement and to integrate info the BMA. The IBMA flowchart can be seen in Fig. 2.

2.5. Bacterial Memetic Algorithm with Memetic Mutation (BMAM) and Modified Bacterial Memetic Algorithm (MBMA)

Although BMA provides a very good speed of convergence towards the optimal

model parameters, there are some points of the algorithm where the performance could be increased. The *Bacterial Memetic Algorithm with Memetic Mutation* [6] exploits the *Levenberg-Marquardt method* more efficiently. Instead of applying the LM cycle *after* the bacterial mutation as a separate step, the modified algorithm executes several LM cycles during the bacterial mutation after each mutational step.

The bacterial mutation operation changes one or more parameters of the modeled system randomly, then it checks whether the model obtained in this way performs better than the previous models or the models that have been changed concurrently this way in the other clones. The mutation test cycle is repeated until all the parameters of the model have gone through the bacterial mutation.

In the mutational cycle, it is possible to gain a temporary model that has an instantaneous fitness value that is worse than the one in the previous or the concurrent models. However, it is potentially better than those, because it is located in a region of the search space that has a better local optimum than the other models do. In accordance with this, if some *Levenberg-Marquardt* iterations are executed after each bacterial mutational step, the test step is able to choose some potentially valued clones that could be lost otherwise.

In the *Bacterial Memetic Algorithm with Memetic Mutation*, after *each mutational step* of *every single bacterial mutation iteration* several LM iterations are done (*memetic mutation*). Several tests have shown it is enough to run just 3 to 5 of LM iterations per mutation in order to improve the performance of the whole algorithm. The usual test phase of the bacterial mutation operation follows after the LM iterations. After the complete *modified* (*memetic*) *bacterial mutation* follows the LM method that is used in the original BMA, where more, e.g. 10 iterational steps, are done with all the individuals of the population towards reaching the local optimum. After all this, the gene transfer operation is executed if needed.

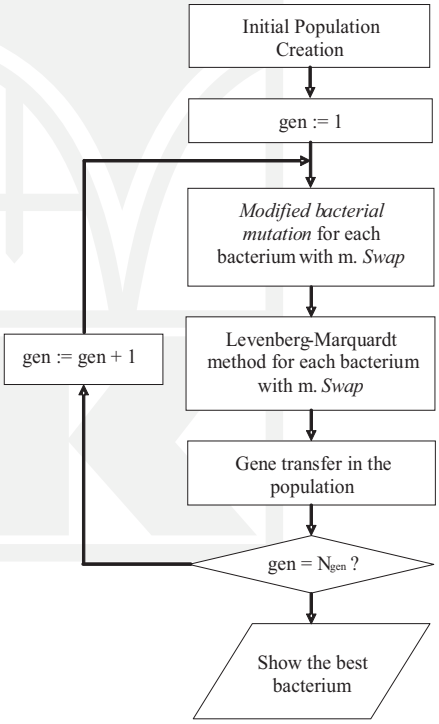

Fig. 3. Flowchart of the MBMA

The advantages of the IBMA and BMAM algorithms are combined in the *Modified Bacterial Memetic Algorithm* (MBMA) [6]. The original bacterial memetic algorithm is modified in the knot order violation handling (affecting the LM method incorporated in the BMA), and in the revised operator execution order (*memetic mutation*). The simulation results [6] proved that this method is superior to the IBMA and BMA algorithms. The MBMA flowchart can be seen in Fig. 3.

We observed that the model convergence speed depends not only on the complexity of the fuzzy rule base, but varies in different phases of the optimization process. For example, we measured less convergence speed in the first 10% of the optimization process, case

of a 2 dimensional test function, and in the 20–30% of the optimization process, case of a 6 dimensional test function. In the various algorithms, the time measured between two generations (one iteration length) is very different. The BEA is the fastest (the cycle includes bacterial mutation and gene transfer), followed by the IBMA (applying the bacterial mutation, LM cycles and gene transfer). A single iteration of the MBMA requires the longest time, including in each cycle modified bacterial mutation (with LM iterations), LM method and gene transfer. Comparing the BEA – IBMA (BMA) resp. the IBMA – MBMA methods we concluded that BEA shows the highest convergence speed in the initial phase of the optimization process.

2.6. Progressive Bacterial Algorithm (PBA)

In [10] we proposed a novel algorithm, the *Progressive Bacterial Algorithm* (PBA), which combines the improvements of BEA, IBMA and MBMA algorithms. In case of unknown applications, the main questions are: What should be the proper sequence of these methods? How can they be combined to obtain an overall high quality model of the optimization process with a very good convergence speed? The main goal is to obtain low model error values during the whole optimization process [1]. We proposed to use BEA in the early stage of the optimization, followed by IBMA, and using MBMA as the last step. In order to determine when to change from one algorithm to another, we concurrently apply two algorithms for different individuals in the population. The individuals were monitored in terms of time. For all those, whom the simplest algorithm provides no more better model errors we switch the individuals training algorithm to the other one. In this way, in each stage of the optimization process, favorable overall MSE values and training characteristics can be obtained.

The PBA flowchart can be seen in Fig. 4. The different algorithms we denoted by: M1 = BEA, M2 = IBMA and M3 = MBMA. The first step is to create the initial population, after that, we apply a different learning algorithm for each individual, for example: at the beginning of the optimization process for the main part of the population, we choose the method M1, while for the rest (for example: 1 + int(NPopulation / 5)), we choose M2. Our proposed approach starts in parallel with two versions, and we chose the best one by calculating and comparing the models MSE value. In order to have comparable MSE data, we have to ensure the same optimization time for each individual. Their learning times are measured and stored for each bacterium along with the current training method. In this novel approach, regarding to one PBA generation, any version (individual's) generation iteration time corresponds to the more complicated method's generation iteration time. The following step of the PBA is to create the individual's next generation. First, we try for a number of individuals the 'complicated' IBMA (M2 iterations without gene transfer). Then the BEA method (M1 iterations without gene transfer) is applied for a few times for the corresponding remainder individual's, while each individual's total optimization time (TM1) is less than a medium total time per individual (TM2) reached with the M2 algorithm.

Then the gene transfer operation within method groups is made (and if a certain termination criterion is satisfied the optimization process is over). During the gene transfer operation a model parameter sequence is transferred from the better individuals into individuals with less fitness values. For them, in many cases, the infections result in temporary high MSE values. That is the reason why the gene transfer operation, in case of sufficient number of individuals, is applied just among individuals which belongs to the same current training method.

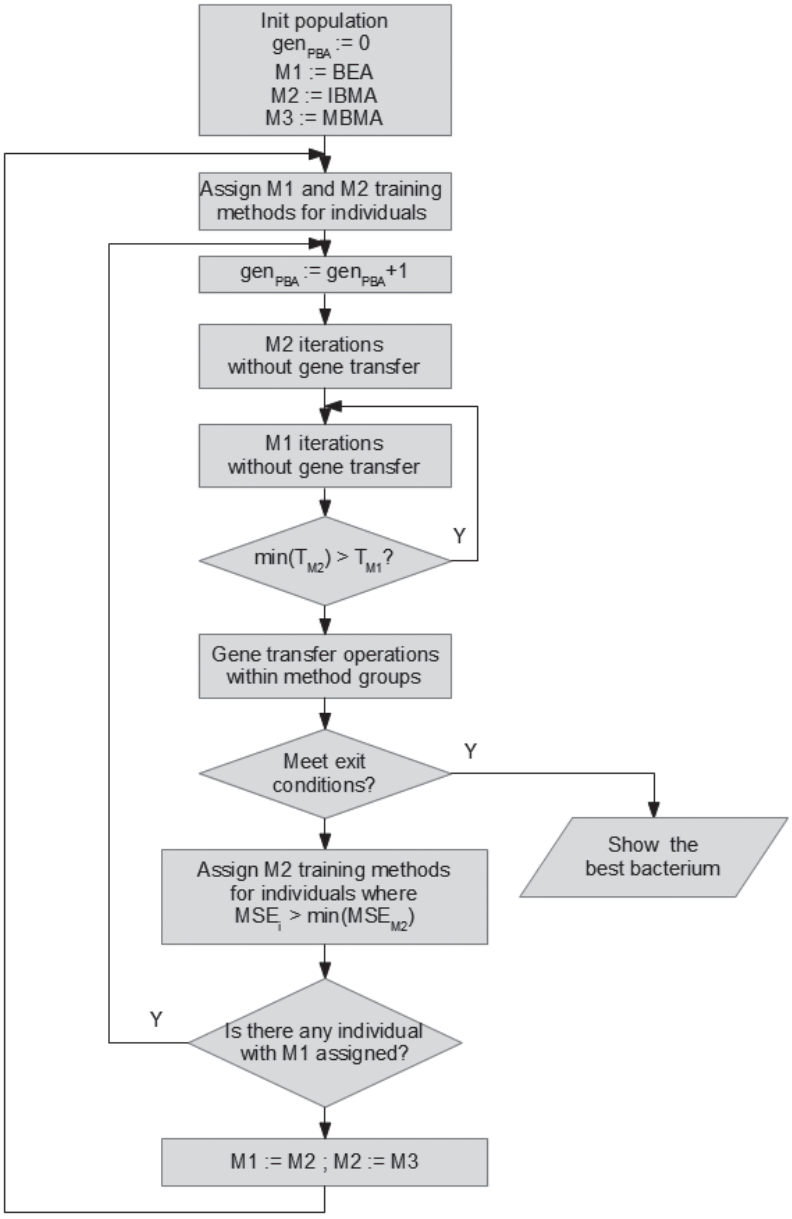

Fig. 4. Flowchart of the PBA

If we have in the population, for example four individuals trained with BEA and one trained with IBMA: application of the gene transfer operation to individuals trained with various methods, it probably happens that the gene transfer (which causes repeated temporary deterioration) will always affect that one trained with IBMA method, thus reducing the model efficiency.

When the minimum MSE value of individuals trained by M2 is higher than the corresponding maximum MSE value of individuals trained by M1, the procedure is repeated from the M2 – M1 training steps until a stopping criterion is satisfied (e.g. maximum number of generations). For individuals where the inequality MSEi < min(MSEM2) is not satisfied, the M2 method is assigned. After all, the population's individuals are switched to M2 algorithm, we assign the method M2 to M1, furthermore, method M3 to M2, and then we start from the beginning, with reassigning methods M1 and M2 for individuals (this method can be extended for more than three suitable training algorithms). In contrast to the former algorithms (e.g. BEA, IBMA, MBMA), in this new algorithm (PBA), the course of the simulation is not replicable due to the granularity in the processing time measurement. Not even for the same initial conditions and pseudo-random number sequences.

2.7. Experimental results for PBA performance evaluation

We investigated the performance properties of the PBA and predecessor algorithms, involving the model accuracy and convergence speed. During the simulations, we examined the relationship between simulation time and model accuracy while training the model parameters of a (fuzzy rule based) Mamdani-type inference system [13] with various bacterial type training algorithms. Two test functions were used, as described below:

a) Test function with 2 input variables (2iv), 200 samples:

$$f_1(\underline{x}) = \sin^5(0.5 \cdot x_1) \cdot \cos(0.7 \cdot x_2) \quad \text{where} \quad x_1 \in [0...3\pi],\ x_2 \in [0...3\pi] \tag{1}$$

b) Test function with 6 input variables (6iv) [17]; 500 samples:

$$f_2(\underline{x}) = x_1 + x_2^{0.5} + x_3 \cdot x_4 + 2 \cdot e^{2 \cdot (x_5 - x_6)} \quad \text{where} \quad x_1 \in [1...5],\ x_2 \in [1...5],\ x_3 \in [0...4],\ x_4 \in [0...0.6],\ x_5 \in [0...1],\ x_6 \in [0...1.2] \tag{2}$$

The algorithms parameters are: fuzzy rules: 5; population size: 7; clones: 7; gene transfers per generation: 3; LM iterations per memetic bacterial mutation: 5; Mamdani inference system aggregation operator: min.

Fig. 5 and 6 present some graphs of typical simulation results with a less complex test function (2 dimensional 2 iv) and a more complex test function (6 dimensional 6 iv) comparing the characteristics for BEA, IBMA, MBMA and PBA algorithms [10]. The graphs show the relationship between the simulation time (horizontal axis) and the model MSE values (vertical axis). Comparing the simulation results, in the case of both test functions, it is shown that the PBA method provides favorable behavior: high model accuracy in the final stage of the optimization and low model error compared to the other algorithms from the first stage. However, in the initial phase of the optimization, the BEA still provides higher convergence speed. The BEA is fast because the main steps of the algorithm are the bacterial mutation and the gene transfer operation. In the PBA, gene transfer operations are not applied until every single (M1) individual's processing time exceeds the method M2 (IBMA) iteration time. This could be the reason for the higher initial convergence speed of BEA. During the optimization process, the BEA is followed by the PBA, IBMA and finally by the MBMA algorithms. In the middle and in the final phase of the process, when the BEA efficiency decreased, the PBA algorithm MSE values are the lowest, highlighting the method's main advantage.

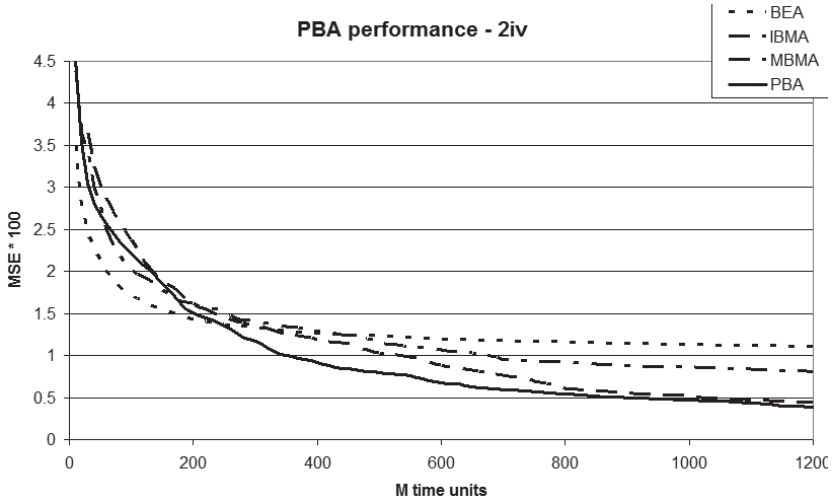

Fig. 5. Typical simulation results – less complex (two dimensional) test function; BEA, IBMA, MBMA, PBA

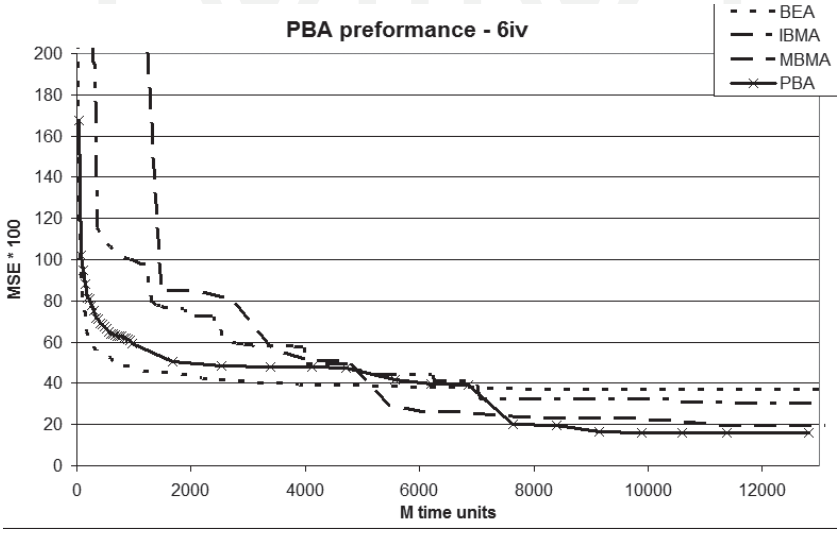

Fig. 6. Typical simulation results – complex (six dimensional) test function; BEA, IBMA, MBMA, PBA

3. Conclusions

In this paper, we revisited various methods to improve the bacterial algorithms performance used for fuzzy rule base extraction. A few evolutionary approaches have been proposed for fuzzy rule base extraction from input-output data such as the *Pseudo-Bacterial Genetic*

Algorithm, the *Bacterial Evolutionary Algorithm*, and various bacterial memetic algorithms developed by us. All these methods have turned out to be helpful with the construction of fuzzy rule based models. Finally, we proposed a special combination of bacterial evolutionary and memetic algorithms. The novel *Progressive Bacterial Algorithm's* (PBA) speed of convergence is comparable with the listed approaches; nevertheless, it can be used as a useful tool by finding a good compromise between the model accuracy and the complexity.

This work is supported by Hungarian Scientific Research Fund (OTKA) K105529, K108405, TÁMOP-4.2.2.A-11/1/KONV-2012-0012, TÁMOP-4.2.2.C-11/1/KONV-2012-0012.

References

[1] Balázs K., Kóczy L.T., *A Remark on Adaptive Scheduling of Optimization Algorithms*, IPMU 2, Vol. 81, 2010, 719-728.

[2] Botzheim J., Kóczy L.T., Ruano A.E., *Extension of the Levenberg-Marquardt algorithm for the extraction of trapezoidal and general piecewise linear fuzzy rules*, IEEE World Congress on Computational Intelligence, Honolulu 2002, 815-819.

[3] Botzheim J., Cabrita C., Kóczy L.T., Ruano A.E., *Fuzzy Rule Extraction by Bacterial Memtic Algorithm*, IFSA, Beijing 2005, 1563-1568.

[4] Gál L., Botzheim J., Kóczy L.T., Ruano A.E., *Fuzzy Rule Base Extraction by the Improved Bacterial Memetic Algorithm*, 6th international Symposium on Applied Machine Intelligence and Informatics, Herlany 2008, 49-53.

[5] 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, Istanbul 2008, 38-43.

[6] 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, Paris 2008, 425-431.

[7] 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, Barcelona 2010, 1514-1521.

[8] 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 2010, 31-36.

[9] Gál L., Kóczy L.T., *Fuzzy Rule Base Extraction by Bacterial Type Algorithms using selected T-norms*, Acta Technica Jaurinensis 4, Series Intelligentia Computatorica, Vol. 3, Györ 2011, 157-175.

[10] Gál L., Lovassy R., Kóczy L.T., *Progressive Bacterial Algorithm*, IEEE 13th International Symposium on Computational Intelligence and Informatics, CINTI, Budapest 2012, 317-322.

[11] Johanyák Zs.Cs., Tikk D., Kovács Sz., Wong K.W., *Fuzzy Rule Interpolation Matlab Toolbox – FRI Toolbox*, Proc. of the IEEE World Congress on Computational

Intelligence (WCCI'06), 15th Int. Conf. on Fuzzy Systems (FUZZ-IEEE'06), Vancouver, BC, Canada, Omnipress, 2006, 1427-1433.

[12] Mamdani E.H., *Application of fuzzy algorithms of simple dynamic plant*, Proceedings of the IEEE, 121, No. 12, 1974, 1585-1588.

[13] Mamdani E.H., Assilian S., *An experiment in linguistic synthesis with a fuzzy logic controller*, International Journal of Man-Machine Studies, Vol. 7, No. 1, 1975, 1-13.

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

[15] Moscato P., *On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithm*, Technical Report Caltech Cocurrent Computational Program, Report.826, California Institute of Technology, Pasadena 1989.

[16] Nawa N.E., Hashiyama T., Furuhashi T., Uchikawa Y., *A Study on Fuzzy Rules Discovery Using Pseudo-Bacterial Genetic Algorithm with Adaptive Operator*, Proc. of IEEE Int. Conf. on Evolutionary Computation, ICEC'97, 1997, 589-593.

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

PK