Application of Self-adapting Genetic Algorithms to Generate Fuzzy Systems for a Regression Problem

Tadeusz Lasota[1], Magdalena Smętek[2], Zbigniew Telec[2], Bogdan Trawiński[2], and Grzegorz Trawiński[3]

[1] Wrocław University of Technology, Institute of Informatics, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
[2] Wrocław University of Environmental and Life Sciences, Dept. of Spatial Management ul. Norwida 25/27, 50-375 Wrocław, Poland
[3] Wrocław University of Technology, Faculty of Electronics, Wybrzeże S. Wyspiańskiego 27, 50-370 Wrocław, Poland
{zbigniew.telec,magdalena.smetek,bogdan.trawinski}@pwr.edu.pl, tadeusz.lasota@up.wroc.pl, grzegorz.trawinsky@gmail.com

Abstract. Six variants of self-adapting genetic algorithms with varying mutation, crossover, and selection were developed. To implement self-adaptation the main part of a chromosome which comprised the solution was extended to include mutation rates, crossover rates, and/or tournament size. The solution part comprised the representation of a fuzzy system and was real-coded whereas to implement the proposed self-adapting mechanisms binary coding was employed. The resulting self-adaptive genetic fuzzy systems were evaluated using real-world datasets derived from a cadastral system and included records referring to residential premises transactions. They were also compared in respect of prediction accuracy with genetic fuzzy systems optimized by a classical genetic algorithm, multilayer perceptron and radial basis function neural network. The analysis of the results was performed using statistical methodology including nonparametric tests followed by post-hoc procedures designed especially for multiple $N \times N$ comparisons.

Keywords: self-adaptive GA, mutation, crossover, genetic fuzzy systems.

1 Introduction

The execution time of genetic algorithms constitute a big challenge, especially when they are used in hybrid methods to create and optimize different classification and prediction models such as genetic fuzzy systems and genetic neural networks. Many researchers have developed numerous techniques for speeding up the convergence of Genetic Algorithms (*GA*) or Evolutionary Algorithms (*EA*) for above two decades. The methods for adapting the values of various parameters to optimize processes in evolutionary computation has been extensively studied and the issue of adjusting *GA/EA* to the problem while solving it still seems to be a promising area of research. The probability of mutation and crossover, the size of selection tournament, or the

D. Hwang et al. (Eds.): ICCCI 2014, LNAI 8733, pp. 49–61, 2014.

population size belong to the most commonly set parameters of *GA/EA*. Three taxonomies of parameter setting forms in evolutionary computation have been devised by Angeline [1], Smith and Fogarty [2], and Eiben, Hinterding, and Michalewicz [3]. The first determines three different adaptation levels of *GA/EA* parameters: population-level where parameters that are global to the population are adjusted, individual-level where changes affect each member of the population separately, and component-level where each component of each member may be modified individually. The second classification is based on three division criteria: what is being adapted, the scope of the adaptation, and the basis for change which is further split into two categories: evidence upon which the change is carried out and the rule or algorithm that executes the change.

The third taxonomy [3] is a general one distinguishing two major forms of parameter value setting, i.e. parameter tuning and parameter control. The first consists in determining good values for the parameters before running *GA/EA*, and then tuning the algorithms without changing these values during the run. However, this approach stands in contradiction to the dynamic nature of *GA/EA*. The second form is an alternative and relies in dynamic adjusting the parameter values during the execution. The third can be categorized into three classes deterministic, adapting and self-adapting parameter control. Deterministic parameter control is applied when the values of evolutionary computation parameters are modified according to some deterministic rules without using any feedback from the optimization process. In turn, adaptive parameter control is employed when some form of feedback from the process is used to determine the trend or strength of the change to the *GA* parameter. Self-adaptive parameter control takes place when the parameters to be adapted are encoded into the chromosomes and undergo mutation and recombination.

Numerous parameter control methods have been proposed in the literature [4], [5], [6]. Several mechanisms of mutation and crossover adaptation and self-adaptation have been developed and experimentally tested [7], [8], [9], [10].

For several years we have been developing and evaluating techniques for building regression models to aid in property valuation based on various machine learning algorithms. Our study included genetic fuzzy systems and artificial neural networks as both single models [11], [12], [13] and ensembles built using different resampling techniques [14], [15], [16], [17], [18], [19]. An especially good performance revealed evolving fuzzy models applied to cadastral data [20], [21]. Evolving fuzzy systems are suitable for modelling the real estate market dynamics because they can be regularly updated on demand based on new incoming samples and the data of property sales ordered by the transaction date can be treated as a data stream. We have also explored the methods to predict from a data stream of real estate sales transactions based on ensembles of genetic fuzzy systems [22], [23], [24], [25]. Our former investigations on the use of evolutionary algorithms to optimize fuzzy systems, which included the generation of rule base and tuning the parameters of membership functions, showed it is an arduous and computationally expensive process. For this reason we attempted to incorporate self-adapting techniques into genetic fuzzy systems aimed to generate regression models for property valuation.

The research presented in this paper is also a continuation of our former study on self-adaptive genetic algorithms [26], [27]. We developed genetic algorithms with self-adaptive mutation and crossover based on an idea developed by Maruo et al. [28] and tested them using several selected multimodal benchmark functions. The algorithms employing self-adaptive mutation and crossover revealed better performance than a traditional genetic one. We also applied the self-adapting genetic algorithms to compose heterogeneous bagging ensembles [29].

2 SAGA Techniques Used to Construct Fuzzy Systems

Six variants of self-adapting genetic algorithms (*SAGA*) with varying mutation (*M*), crossover (*C*), and selection (*T*) were developed. They were named in the paper *SAM, SAC, SAMC, SACT, SAMT,* and *SAMCT*, respectively. In all variants of *SAGA* algorithms constant length chromosomes were used and their structures are illustrated in Figure 1. To implement self-adaptation the main part of a chromosome which comprised the solution was extended to include mutation rates, crossover rates, and/or tournament size. The solution part comprised the representation of a fuzzy system and was real-coded whereas to implement the proposed self-adapting mechanisms binary coding was employed. The mutation rate could be set to values from the bracket 0 to 0.3, and crossover rate from the range 0.5 to 1.0. Therefore, to encode the mutation rate 5 genes and crossover rate 7 genes were used. In turn, the tournament size was encoded using 3 binary genes to represent the range from 1 to 7.

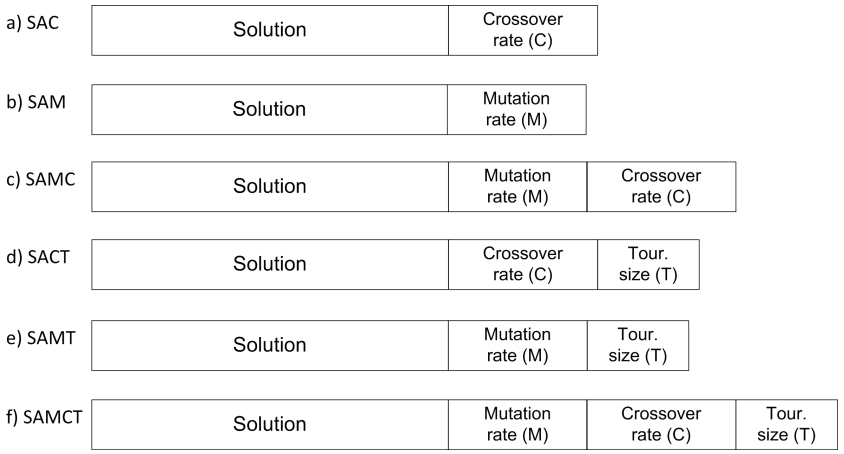

Fig. 1. Chromosome structures of individual self-adaptive genetic algorithms

Solution. For each input variable three triangular and trapezoidal membership functions, and for output - five functions, were automatically determined by the symmetric division of the individual attribute domains. The evolutionary optimization process combined both learning the rule base and tuning the membership functions using real-coded chromosomes. Similar designs are described in [30], [31]. The shapes of the triangular and trapezoidal membership functions after evolutionary tuning are presented in Figure 2.

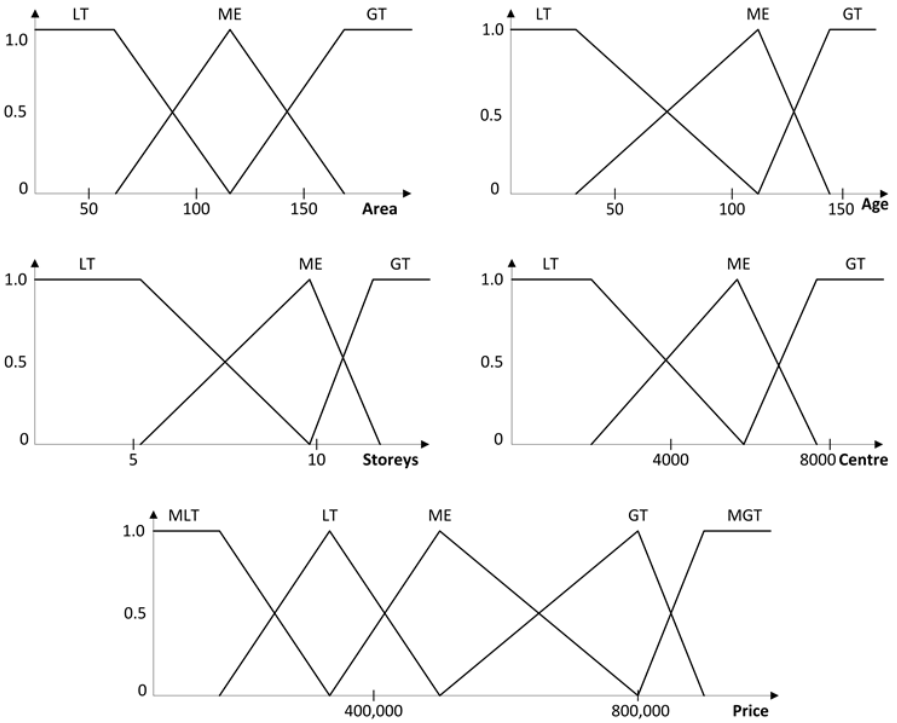

Fig. 2. The shapes of the membership functions after evolutionary tuning

The rule base was real-coded using the Pittsburgh method, where one chromosome comprised whole rule base. Each rule was represented by $n+1$ genes corresponding to n input and one output variables The genes contained natural numbers from 1 to 5 referring to linguistic values of variables, i.e. *MLT* (much less than), *LT* (less than), *ME* (medium), *GT* (greater than), *MGT* (much greater than), respectively. Zero value on the position of a given input meant that this attribute did not occur in the rule. The number of genes to encode membership functions was minimized based on the assumption that the vertices of adjacent functions overlap. Thus, only 3 genes are needed for each input mf and 5 genes suffice to represent the output mf. The real-coded representation of a fuzzy system in the chromosome is depicted in Figure 3.

Rule 1					Rule 2					Rule 3					...	Rule 15				
1	3	1	2	3	2	0	0	0	2	0	1	1	2	1	...	2	1	0	3	2

Input mf 1			Input mf 2			...	Output mf				
18.1	53.9	117.4	31.7	103.3	120.5	...	149.4	336.9	466.8	641.3	757.6

Fig. 3. Encoding rules and membership functions in the solution part of the chromosome

Self-adaptive crossover. The self-adaptive crossover, which is depicted in Figure 4, is different from a traditional *GA* crossover. A special $K \times 1$ table with real, randomly selected values from the brackets 0.5 to 1.0 is created, where K is the number of chromosomes in the population. Each chromosome is connected with one real value in the table. The self-adaption of the crossover goes on in the following way. For each chromosome from population:

- extract the genes representing the crossover rate from the chromosome,
- calculate the value of crossover rate extracted from chromosome,
- if the value from the table is lower than the value of crossover rate from the chromosome, then the chromosome is selected to a classic crossover process,
- the $K \times 1$ table remains unchanged during the execution of the *SAGA* algorithm.

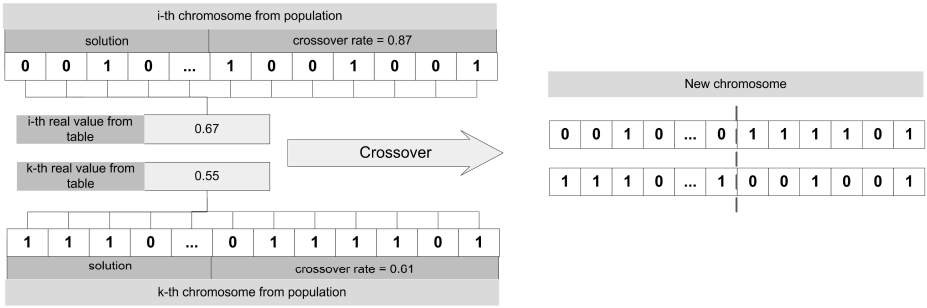

Fig. 4. Self-adaptive crossover in *SAGA* algorithms

Self-adaptive mutation. The self-adaptive mutation applied in *SAGA* algorithms is illustrated in Figure 5. It differs from a standard *GA* mutation which rate remains constant during the run. Each chromosome from the population can be subject to the mutation. A special $K \times L$ matrix with real, randomly selected values from the range 0 to 0.3 is created, where K is the number of chromosomes in the population, and L stands for the number of genes in a chromosome. Each gene in each chromosome is connected with one real value in the matrix. The self-adaptation of the mutation proceeds as follows. For each chromosome from population:

- extract the genes representing the mutation rate from the chromosome,
- calculate the value of mutation rate extracted from chromosome,
- if the value from the matrix is lower than the value of the mutation rate taken from the chromosome, then the chromosome mutates in a traditional way,
- the $K \times L$ matrix remains unchanged during the execution of the *SAGA* algorithm.

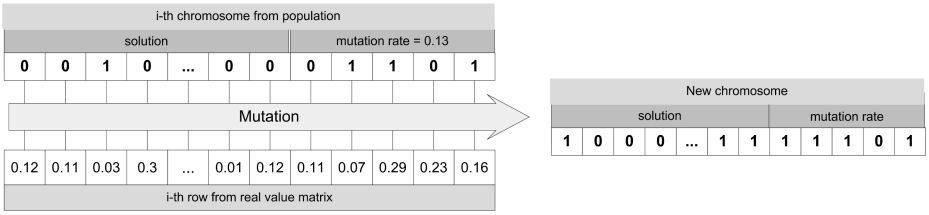

Fig. 5. Self-adaptive mutation in *SAGA* algorithms

Self-adaptive selection. Three genes were added to the chromosome to encode the tournament size which could be set to integer number from 1 to 7. Therefore, before selection average tournament size in the population was calculated and this value was used as the final tournament size in the selection operation.

3 Experimental Setup

The experiments were conducted with our system implemented in Matlab. The system was designed to carry out research into machine learning algorithms using various resampling methods and constructing and evaluating ensemble models for regression problems. We have recently extended our system to include functions for building and tuning fuzzy systems by means of self-adapting genetic algorithms.

Real-world dataset used in experiments was derived from a cadastral system and included records referring to residential premises transactions accomplished in one Polish big city within 14 years from 1998 to 2011. After selection and cleansing the final dataset counted 9795 samples. Four following attributes were pointed out as main price drivers by professional appraisers: usable area of a flat (*Area*), age of a building construction (*Age*), number of storeys in the building (Storeys), the distance of the building from the city centre (*Centre*), in turn, price of premises (*Price*) was the output variable.

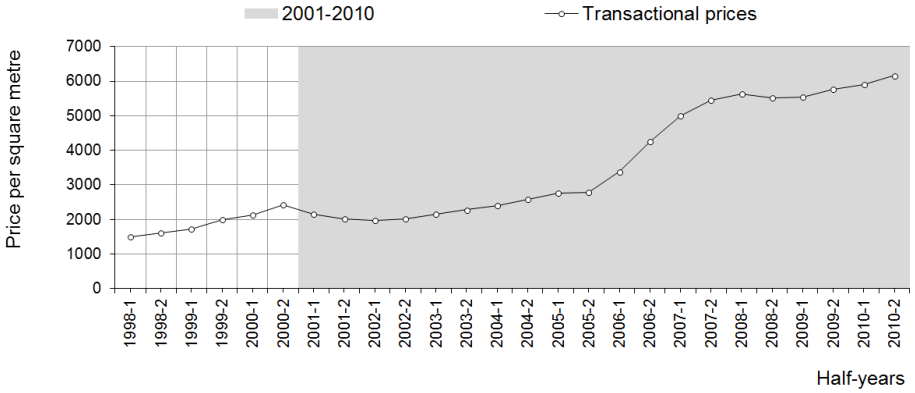

Fig. 6. Change trend of average transactional prices per square metre over time

The evaluating experiments were conducted for 36 points of time from 2002-01-01 to 2010-10-01. Single models were built over training data delineated by the time span of 12 months. In turn, the test datasets were determined by the interval of 3 months, current for a given time point. The selected dataset ensured the variability of individual points of observation due to the dramatic rise of the prices of residential premises during the worldwide real estate bubble (see Fig. 6).

For comparative tests we took the results of our previous investigations into genetic fuzzy systems optimized by a classical genetic algorithm (*GA*) [32] (see parameters in Table 1) and artificial neural networks: multilayer perceptron (*MLP*) and radial basis function neural networks (*RBF*) [33] conducted over the same datasets. To examine the better convergence of *SAGA* algorithms fuzzy systems were built within 50 generations whereas with a classical *GA* 100 generations were executed. The number of epochs to learn each neural network was equal to 100. As the performance measure the root mean square error (*RMSE*) was used.

Table 1. Parameters of GA to optimize fuzzy sets used in experiments

Fuzzy system	Genetic Algorithm
Type of fuzzy system: Mamdani	Chromosome: rule base and mf, real-coded
No. of input variables: 4	Population size: 100
Type of membership functions (mf): triangular	Fitness function: MSE
No. of input mf: 3	Selection function: tournament
No. of output mf: 5	Tournament size: 4
No. of rules: 15	Elite count: 2
AND operator: prod	Crossover fraction: 0.8
Implication operator: prod	Crossover function: two point
Aggregation operator: probor	Mutation function: custom
Defuzzyfication method: centroid	No. of generations: 100

The analysis of the results was performed using statistical methodology including nonparametric tests followed by post-hoc procedures designed especially for multiple $N \times N$ comparisons [34], [35], [36], [37]. The routine starts with the nonparametric Friedman test, which detect the presence of differences among all algorithms compared. After the null-hypotheses have been rejected the post-hoc procedures are applied in order to point out the particular pairs of algorithms which produce differences. For $N \times N$ comparisons nonparametric Nemenyi's, Holm's, Shaffer's, and Bergmann-Hommel's procedures are employed.

4 Analysis of Experimental Results

4.1 Performance of SAGA Fuzzy Systems

The performance of single *SAGA* fuzzy models over 36 points of observation is depicted in Figure 7. The values of *RMSE* are given in thousand PLN. We can see unstable behaviour of *SAMT* models which produce excessive errors for two points.

However, the differences among the models are not visually apparent, therefore one should refer to statistical tests of significance.

Average rank positions of single *SAGA* models determined during Friedman test are shown in Table 2, where the lower rank value the better model. Adjusted p-values for Nemenyi's, Holm's, Shaffer's, and Bergmann-Hommel's post-hoc procedures for *N*×*N* comparisons for all possible pairs of *SAGA* methods are shown in Table 3. For illustration the p-values of the paired Wilcoxon test are also given. The significance level considered for the null hypothesis rejection was 0.05. Significant differences were observed only for one pair of SAGAs: *SAC* ensembles surpassed the *SAMCT* ones. The paired Wilcoxon test can lead to over-optimistic decisions because it allows for rejection of a greater number of null hypotheses.

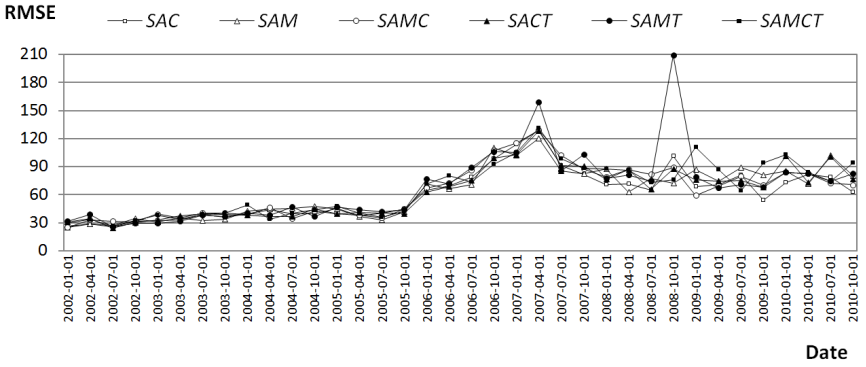

Fig. 7. Performance of *SAGA* models over 36 points o observations

Table 2. Average rank positions of *SAGA* models determined during Friedman test

1st	2nd	3rd	4th	5th	6th
SAC (2.86)	SAM (3.22)	SACT (3.22)	SAMT (3.69)	SAMC (3.78)	SAMCT (4.22)

Table 3. Adjusted p-values for *N*×*N* comparisons of *SAGA* models for all 15 hypotheses

Method vs Method	pWilcox	pNeme	pHolm	pShaf	pBerg
SAC vs SAMCT	*0.0184*	*0.0304*	*0.0304*	*0.0304*	*0.0304*
SAM vs SAMCT	*0.0192*	0.3501	0.3268	0.2334	0.2334
SACT vs SAMCT	*0.0443*	0.3501	0.3268	0.2334	0.2334
SAC vs SAMC	*0.0169*	0.5645	0.4516	0.3764	0.3764
SAC vs SAMT	*0.0095*	0.8817	0.6466	0.5878	0.4115
SAM vs SAMC	0.5094	1.0000	1.0000	1.0000	1.0000
SAMC vs SACT	0.3300	1.0000	1.0000	1.0000	1.0000
SAMT vs SAMCT	0.5297	1.0000	1.0000	1.0000	1.0000
SAM vs SAMT	0.4321	1.0000	1.0000	1.0000	1.0000
SACT vs SAMT	0.1126	1.0000	1.0000	1.0000	1.0000
SAMC vs SAMCT	0.3300	1.0000	1.0000	1.0000	1.0000
SAC vs SACT	0.2784	1.0000	1.0000	1.0000	1.0000
SAC vs SAM	0.3705	1.0000	1.0000	1.0000	1.0000
SAMC vs SAMT	0.7296	1.0000	1.0000	1.0000	1.0000
SAM vs SACT	0.8628	1.0000	1.0000	1.0000	1.0000

4.2 Comparison of *SAGA* Fuzzy Models with Other Approaches

For comparison we took the results of our previous investigations into genetic fuzzy systems optimized by a classical genetic algorithm (*GA*) [32] and artificial neural networks: multilayer perceptron (*MLP*) and radial basis function (*RBF*) [33] conducted over the same datasets. For statistical tests we selected the *SAGA* fuzzy models providing the best performance, namely *SAC, SAM*, and *SACT* ones. The *RMSE* of examined methods was computed for the same 36 observation time points. The Friedman test values showed that there were significant differences among models. Average ranks of compared methods produced by the test are shown in Table 4, where the lower rank value the better model. Adjusted p-values for the paired Wilcoxon test as well as Nemenyi's, Holm's, Shaffer's, and Bergmann-Hommel's post-hoc procedures for *N*×*N* comparisons for all possible pairs of algorithms are shown in Table 5. The p-values indicating the statistically significant differences between given pairs of algorithms are marked with italics. The significance level considered for the null hypothesis rejection was 0.05.

Table 4. Average rank positions of compared models determined during Friedman test

1st	2nd	3rd	4th	5th	6th
GA (2.06)	SAC (2.58)	SAM (2.69)	SACT (2.75)	RBF (5.36)	MLP (5.56)

Table 5. Adjusted p-values for N×N comparisons of SAGA models for all 15 hypotheses

Method vs Method	pWilcox	pNeme	pHolm	pShaf	pBerg
GA vs MLP	*1.68E-07*	*3.10E-14*	*3.10E-14*	*3.10E-14*	*3.10E-14*
GA vs RBF	*1.68E-07*	*9.85E-13*	*9.19E-13*	*6.56E-13*	*6.56E-13*
SAC vs MLP	*1.83E-07*	*2.37E-10*	*2.05E-10*	*1.58E-10*	*1.58E-10*
SAM vs MLP	*1.99E-07*	*1.30E-09*	*1.04E-09*	*8.68E-10*	*6.07E-10*
SACT vs MLP	*1.68E-07*	*2.98E-09*	*2.18E-09*	*1.99E-09*	*1.19E-09*
SAC vs RBF	*1.68E-07*	*4.48E-09*	*2.99E-09*	*2.99E-09*	*1.79E-09*
SAM vs RBF	*2.17E-07*	*2.21E-08*	*1.32E-08*	*1.03E-08*	*5.89E-09*
SACT vs RBF	*1.68E-07*	*4.79E-08*	*2.55E-08*	*2.23E-08*	*1.28E-08*
SACT vs GA	*0.003476*	1.000000	0.807034	0.807034	0.807034
SAM vs GA	*0.014252*	1.000000	0.884254	0.884254	0.807034
SAC vs GA	0.123650	1.000000	1.000000	1.000000	0.807034
MLP vs RBF	0.292521	1.000000	1.000000	1.000000	1.000000
SAC vs SACT	0.278352	1.000000	1.000000	1.000000	1.000000
SAC vs SAM	0.370519	1.000000	1.000000	1.000000	1.000000
SAM vs SACT	0.862796	1.000000	1.000000	1.000000	1.000000

Following main observations could be done: despite *GA* took the first position in the Friedman test rank, the post hoc procedures indicated no significant differences among *GA, SAC, SAM*, and *SACT* models. It should be noted that *SAGA* algorithms were run for 50 generations whereas the classical *GA* for 100 generations. All these four methods outperformed significantly *MLP* and *RBF* neural networks. The paired

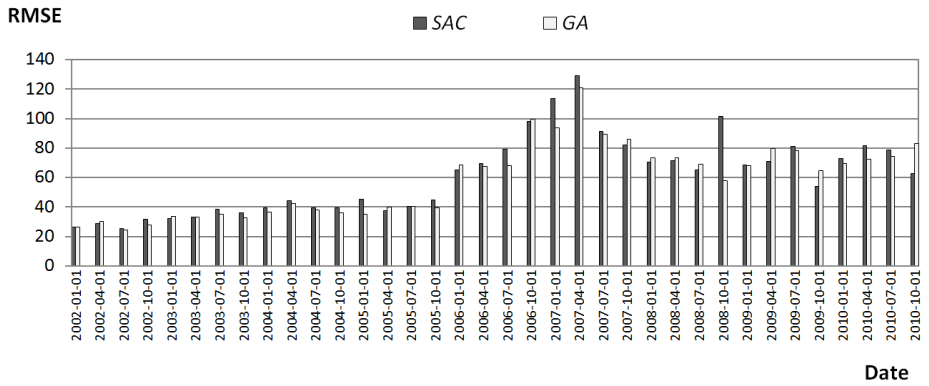

Fig. 8. Illustration of *SAC* and *GA* fuzzy system performance over 36 points of observation

Wilcoxon test again allowed for rejection of a greater number of null hypotheses which could lead to over-optimistic decisions. For illustration the performance of *SAC* and *GA* models over 36 points of observation is shown in Fig. 8. The values of *RMSE* are given in thousand PLN.

5 Conclusions and Future Work

In the paper we proposed to apply self-adapting genetic algorithms to generate rule base and tune membership functions of fuzzy systems build over cadastral data to predict the values of real estates. We devised and implemented six self-adapting genetic algorithms with varying mutation, crossover, and selection. To implement self-adaptation the main part of a chromosome which comprised the solution, i.e. the definition of a fuzzy system, was extended to include mutation rates, crossover rates, and tournament size. The solution part was real-coded whereas to reflect the parameters of self-adaptive mechanisms binary coding was employed. The resulting self-adaptive genetic fuzzy systems were evaluated using records of sale and purchase transactions of residential premises taken from a cadastral system. They were also compared in respect of prediction accuracy with multilayer perceptron and radial basis function neural networks as well as with genetic fuzzy systems optimized by a classical genetic algorithm. The analysis of the results was performed using nonparametric methodology including the Friedman tests followed by the Nemenyi's, Holm's, Shaffer's, and Bergmann-Hommel's post-hoc procedures designed especially for multiple $N \times N$ comparisons.

The preliminary results showed that there were not statistically significant differences in accuracy among the majority of *SAGA* models. We compared three *SAGA* models with three competing methods, namely genetic fuzzy systems optimized by a classical genetic algorithm (*GA*) and two artificial neural networks: multilayer perceptron (*MLP*) and radial basis function (*RBF*). No significant differences among *GA, SAC, SAM*, and *SACT* models were observed It should be noted that *SAGA* algorithms were run for twice less number of generations than the

classical *GA* was. Moreover, all *SAGA* models outperformed significantly the neural networks. It is planned to continue the investigation of *SAGA* algorithms applied to generate and tune fuzzy systems.

We intend continue our study to find optimal parameters of self-adapting mechanisms as well as explore the convergence of *SAGA* algorithms compared to the classical ones. Further experiments will be conducted using greater number of benchmark datasets taken from the UCI repository.

Acknowledgments. This work was partially supported by the National Science Centre under grant no. N N516 483840 and the "Młoda Kadra" funds of Wrocław University of Technology.

References

1. Angeline, P.J.: Adaptive and self-adaptive evolutionary computations. In: Palaniswami, M., Attikiouzel, Y. (eds.) Computational Intelligence: A Dynamic Systems Perspective, pp. 152–163. IEEE Press, New York (1995)
2. Smith, J.E., Fogarty, T.C.: Operator and parameter adaptation in genetic algorithms. Soft Computing 1(2), 81–87 (1997)
3. Eiben, E., Hinterding, R., Michalewicz, Z.: Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation 3(2), 124–141 (1999)
4. Bäck, T., Schwefel, H.-P.: An Overview of Evolutionary Algorithms for Parameter Optimization. Evolutionary Computation 1(1), 1–23 (1993)
5. Meyer-Nieberg, S., Beyer, H.-G.: Self-Adaptation in Evolutionary Algorithms. In: Lobo, F.G., et al. (eds.) Self-Adaptation in Evolutionary Algorithms. SCI, vol. 54, pp. 47–75. Springer, Heidelberg (2007)
6. Hinterding, R., Michalewicz, Z., Eiben, A.E.: Adaptation in Evolutionary Computation: A Survey. In: Proceedings of the Fourth International Conference on Evolutionary Computation (ICEC 1997), pp. 65–69. IEEE Press, New York (1997)
7. Deb, K., Beyer, H.-G.: Self-adaptive genetic algorithms with simulated binary crossover. Evolutionary Computation 9(2), 197–221 (2001)
8. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation 9(2), 159–195 (2001)
9. De Jong, K.: An analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan (1975)
10. Lobo, F.: The parameter-less genetic algorithm: rational and automated parameter selection for simplified genetic algorithm operation. PhD thesis, Nova University of Lisboa (2000)
11. Król, D., Lasota, T., Nalepa, W., Trawiński, B.: Fuzzy system model to assist with real estate appraisals. In: Okuno, H.G., Ali, M. (eds.) IEA/AIE 2007. LNCS (LNAI), vol. 4570, pp. 260–269. Springer, Heidelberg (2007)
12. Król, D., Lasota, T., Trawiński, B., Trawiński, K.: Comparison of Mamdani and TSK Fuzzy Models for Real Estate Appraisal. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part III. LNCS (LNAI), vol. 4694, pp. 1008–1015. Springer, Heidelberg (2007)
13. Graczyk, M., Lasota, T., Trawiński, B.: Comparative Analysis of Premises Valuation Models Using KEEL, RapidMiner, and WEKA. In: Nguyen, N.T., Kowalczyk, R., Chen, S.-M. (eds.) ICCCI 2009. LNCS, vol. 5796, pp. 800–812. Springer, Heidelberg (2009)

14. Lasota, T., Telec, Z., Trawiński, B., Trawiński, K.: Exploration of Bagging Ensembles Comprising Genetic Fuzzy Models to Assist with Real Estate Appraisals. In: Corchado, E., Yin, H. (eds.) IDEAL 2009. LNCS, vol. 5788, pp. 554–561. Springer, Heidelberg (2009)
15. Lasota, T., Telec, Z., Trawiński, B., Trawiński, K.: A Multi-agent System to Assist with Real Estate Appraisals Using Bagging Ensembles. In: Nguyen, N.T., Kowalczyk, R., Chen, S.-M. (eds.) ICCCI 2009. LNCS, vol. 5796, pp. 813–824. Springer, Heidelberg (2009)
16. Graczyk, M., Lasota, T., Trawiński, B., Trawiński, K.: Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal. In: Nguyen, N.T., Le, M.T., Świątek, J. (eds.) Intelligent Information and Database Systems. LNCS, vol. 5991, pp. 340–350. Springer, Heidelberg (2010)
17. Krzystanek, M., Lasota, T., Telec, Z., Trawiński, B.: Analysis of Bagging Ensembles of Fuzzy Models for Premises Valuation. In: Nguyen, N.T., Le, M.T., Świątek, J. (eds.) Intelligent Information and Database Systems. LNCS, vol. 5991, pp. 330–339. Springer, Heidelberg (2010)
18. Kempa, O., Lasota, T., Telec, Z., Trawiński, B.: Investigation of bagging ensembles of genetic neural networks and fuzzy systems for real estate appraisal. In: Nguyen, N.T., Kim, C.-G., Janiak, A. (eds.) ACIIDS 2011, Part II. LNCS, vol. 6592, pp. 323–332. Springer, Heidelberg (2011)
19. Lasota, T., Telec, Z., Trawiński, G., Trawiński, B.: Empirical Comparison of Resampling Methods Using Genetic Fuzzy Systems for a Regression Problem. In: Yin, H., Wang, W., Rayward-Smith, V. (eds.) IDEAL 2011. LNCS, vol. 6936, pp. 17–24. Springer, Heidelberg (2011)
20. Lasota, T., Telec, Z., Trawiński, B., Trawiński, K.: Investigation of the eTS Evolving Fuzzy Systems Applied to Real Estate Appraisal. Journal of Multiple-Valued Logic and Soft Computing 17(2-3), 229–253 (2011)
21. Lughofer, E., Trawiński, B., Trawiński, K., Kempa, O., Lasota, T.: On Employing Fuzzy Modeling Algorithms for the Valuation of Residential Premises. Information Sciences 181, 5123–5142 (2011)
22. Trawiński, B., Lasota, T., Smętek, M., Trawiński, G.: An Attempt to Employ Genetic Fuzzy Systems to Predict from a Data Stream of Premises Transactions. In: Hüllermeier, E., Link, S., Fober, T., Seeger, B. (eds.) SUM 2012. LNCS, vol. 7520, pp. 127–140. Springer, Heidelberg (2012)
23. Trawiński, B., Lasota, T., Smętek, M., Trawiński, G.: An Analysis of Change Trends by Predicting from a Data Stream Using Genetic Fuzzy Systems. In: Nguyen, N.-T., Hoang, K., Jędrzejowicz, P. (eds.) ICCCI 2012, Part I. LNCS, vol. 7653, pp. 220–229. Springer, Heidelberg (2012)
24. Trawiński, B., Lasota, T., Smętek, M., Trawiński, G.: Weighting Component Models by Predicting from Data Streams Using Ensembles of Genetic Fuzzy Systems. In: Larsen, H.L., Martin-Bautista, M.J., Vila, M.A., Andreasen, T., Christiansen, H. (eds.) FQAS 2013. LNCS, vol. 8132, pp. 567–578. Springer, Heidelberg (2013)
25. Trawiński, B.: Evolutionary Fuzzy System Ensemble Approach to Model Real Estate Market based on Data Stream Exploration. Journal of Universal Computer Science 19(4), 539–562 (2013)
26. Smętek, M., Trawiński, B.: Investigation of Genetic Algorithms with Self-adaptive Crossover, Mutation, and Selection. In: Corchado, E., Kurzyński, M., Woźniak, M. (eds.) HAIS 2011, Part I. LNCS, vol. 6678, pp. 116–123. Springer, Heidelberg (2011)

27. Smętek, M., Trawiński, B.: Investigation of Self-adapting Genetic Algorithms using Some Multimodal Benchmark Functions. In: Jędrzejowicz, P., Nguyen, N.T., Hoang, K. (eds.) ICCCI 2011, Part I. LNCS, vol. 6922, pp. 213–223. Springer, Heidelberg (2011)
28. Maruo, M.H., Lopes, H.S., Delgado, M.R.: Self-Adapting Evolutionary Parameters: Encoding Aspects for Combinatorial Optimization Problems. In: Raidl, G.R., Gottlieb, J. (eds.) EvoCOP 2005. LNCS, vol. 3448, pp. 154–165. Springer, Heidelberg (2005)
29. Smętek, M., Trawiński, B.: Selection of Heterogeneous Fuzzy Model Ensembles Using Self-adaptive Genetic Algorithms. New Generation Computing 29(3), 309–327 (2011)
30. Cordón, O., Herrera, F.: A Two-Stage Evolutionary Process for Designing TSK Fuzzy Rule-Based Systems. IEEE Tr. on Sys., Man and Cyber., Part B 29(6), 703–715 (1999)
31. Król, D., Lasota, T., Trawiński, B., Trawiński, K.: Investigation of evolutionary optimization methods of TSK fuzzy model for real estate appraisal. International Journal of Hybrid Intelligent Systems 5(3), 111–128 (2008)
32. Trawiński, B., Smętek, M., Lasota, T., Trawiński, G.: Evaluation of Fuzzy System Ensemble Approach to Predict from a Data Stream. In: Nguyen, N.T., Attachoo, B., Trawiński, B., Somboonviwat, K. (eds.) ACIIDS 2014, Part II. LNCS, vol. 8398, pp. 137–146. Springer, Heidelberg (2014)
33. Telec, Z., Trawiński, B., Lasota, T., Trawiński, K.: Comparison of Evolving Fuzzy Systems with an Ensemble Approach to Predict from a Data Stream. In: Bădică, C., Nguyen, N.T., Brezovan, M. (eds.) ICCCI 2013. LNCS, vol. 8083, pp. 377–387. Springer, Heidelberg (2013)
34. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
35. García, S., Herrera, F.: An Extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons. Journal of Machine Learning Research 9, 2677–2694 (2008)
36. Graczyk, M., Lasota, T., Telec, Z., Trawiński, B.: Nonparametric Statistical Analysis of Machine Learning Algorithms for Regression Problems. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part I. LNCS, vol. 6276, pp. 111–120. Springer, Heidelberg (2010)
37. Trawiński, B., Smętek, M., Telec, Z., Lasota, T.: Nonparametric Statistical Analysis for Multiple Comparison of Machine Learning Regression Algorithms. International Journal of Applied Mathematics and Computer Science 22(4), 867–881 (2012)