UDC 519.87

# DEVELOPMENT OF ADAPTIVE GENETIC ALGORITHMS FOR NEURAL NETWORK MODELS MULTICRITERIA DESIGN[1]

C. Brester[2], E. Semenkin[3]

Siberian State Aerospace University named after academician M. F. Reshetnev
31 «Krasnoyarskiy Rabochiy» pr., Krasnoyarsk, 660014, Russia.
E-mail: abahachy@mail.ru[2]
Siberian State Aerospace University named after academician M. F. Reshetnev
31 «Krasnoyarskiy Rabochiy» pr., Krasnoyarsk, 660014, Russia.
E-mail: eugenesemenkin@yandex.ru[3]

*In this paper modifications of single- and multi-objective genetic algorithms are described and testing results of these approaches are presented. The gist of the algorithms is the use of the self-adaptation idea leading to reducing of the expert significance for the algorithm setting and expanding of GAs' application capabilities. On the basis of offered methods the program system realizing the technique for neural network models design was developed. The effectiveness of all algorithms was investigated on a set of test problems.*

*Keywords: genetic algorithms, multicriteria optimization, self-adaptation, neural networks, classification.*

## 1. Introduction

Nowadays genetic algorithms (GA) [1] are widely used for the intelligent information technologies design (specifically for the artificial neural networks (ANN) models design). This development became possible due to the self-adaptation technique allowing automatic choosing of a genetic operator type for the current problem during the algorithm execution. It led to reducing of the expert significance for the algorithm setting and expanding of GAs' application capabilities. The adaptive single-objective GA used in this study is described in Section 1.

Conventional GA is a procedure for solving one-criterion unconstrained optimization problems but most of real problems are multi-criterion ones. There are many multi-criteria GAs but they also need careful choice of their settings for the performance improvement. That is why many approaches to self-adaptation were offered here. Our approach to multi-criterion GAs self-adaptation will be described in Section 2.

ANN-models are exploited in different applications: data analysis systems, speech or images recognition and so on. In this connection it is interesting not only to determine network weight coefficients for solving any problem with sufficient accuracy but also to find a compact network structure which would allow saving end user's computing resources. In this case a mathematical model looks like a multi-criterion optimization problem (MOP) with two objective functions. Therefore it is reasonable to use a combination of GAs: a MOP-algorithm is applied for ANN-structure design and a conventional GA is applied for the search of ANN weighting coefficients, certainly both methods should be self-adaptive. In addition local search algorithms can be used at any stage to determine the found result more exactly. An approach to the design of ANN-based models and results of its application is presented in Section 3. In Conclusion section we shortly describe presented results and discuss the directions of future research.

[1] The study was supported by The Ministry of education and science of Russian Federation, project 14.B37.21.1521.

## 2. Adaptive single-objective GA

The structure of a conventional GA involves well-known genetic operators: selection, crossover and mutation. There are three variants of selection: proportional, tournament and rank; three variants of crossover: one-point, two-point and uniform; three variants of mutation: weak, average and strong. The realized technique of the operator choice is based on the evaluation of application probabilities for all variants of operators.

In the first generation all variants of every genetic operator have equal probabilities, i.e. $q_i^k = 1/n^k$, where $q_i^k$ is the application probability for the *i*-th variant of the *k*-th operator, $n^k$ is the number of different variants of a certain genetic operator, $i,k = \overline{1,3}$. Before the operator is applied it is necessary to raffle the application probabilities.

After each generation the probabilities are recalculated taking into account fitness of individuals which were generated by the given operator. The main idea was borrowed from Banzhaf's article [2] with the only difference: the variable named «$ratio_i^k$» is a fitness sum of individuals generated with the *i*-th variant of the *k*-th operator.

Below there is a rule for probabilities $q_i^k$ calculation:

$$q_i^k = \frac{0.2}{n^k} + 0.8 \cdot \frac{ratio_i^k}{scale^k},$$

where $scale^k = \sum_i ratio_i^k$. The first summand doesn't allow any probability to be equal to zero (what makes all variants of operators available throughout all generations).

The effectiveness of this approach was investigated on a set of one-criterion unconstrained optimization problems and was compared with the effectiveness of a conventional GA. Equal amount of resources was allocated for all algorithms: 100 individuals and 100 generations. Results were averaged over 500 runs. On the basis of reliability values the number of cases in which the adaptive GA lost to the standard GA was fixed. The number of all genetic operators' combinations was 27. Table 1 contains testing results:

*Table 1*

**Testing results of an adaptive single-objective GA**

| № | Test function | The number of cases when an adaptive GA lost to a conventional GA |
|---|---|---|
| 1 | Rastrigin function | 9 |
| 2 | Rosenbrock function | 4 |
| 3 | Katkovnik function | 3 |
| 4 | Griewank function | 4 |
| 5 | Multiplicative potential function | 0 |
| 6 | Additive potential function | 7 |
| 7 | Rastrigin function (ravine function with rotated axis) | 1 |
| 8 | «Foxholes» | 8 |

Thus results of the testing showed that the efficiency of an adaptive genetic algorithm is not lower than the efficiency of an «average» GA and is comparable with «best» GA in many cases. It means than our self-adaptive GA can be recommended to be used instead of conventional GA.

## 3. Multi-objective GA

It should be mentioned that in general a mathematical model of any problem includes a set of quality criteria and all of them should be taken into account.

In 1999 the method named Strength Pareto Evolutionary Algorithm (SPEA) was discovered by Zitzler and Thietle [3]. It is based on the Pareto's dominance idea. The solution of the multi-criterion optimization problem belongs to Pareto's set (PS), and any of these points cannot be preferred to any other point. Their representation in the criteria space is Pareto's front (PF). In SPEA non-dominated individuals are stored in the archive of a limited size named "an outer set". The outer set is upgraded during the algorithm execution and as a result we have an approximation of PS.

This paper is devoted to SPEA-modification based on self-adaptation idea. The effectiveness of SPEA and its adaptive analog is compared below.

In SPEA tournament selection is applied: individuals can be selected both from the current population and from the outer set. Therefore only crossover and mutation operators require adjustment (tuning or control) [4].

Mutation probability can be determined according to one of the rules suggested by Daridi [5]. In the realized program system the following rule was used:

$$p_m = \frac{1}{240} + \frac{0.11375}{2^t},$$

where $t$ is the current generation number.

The self-configurable crossover operator is based on the co-evolution idea [6]: the population is divided into parts and each part is generated with a certain type of crossover. The size of the subpopulation depends on the «fitness» of the corresponding recombination operator. «Fitness» is proportional to the number of non-dominated individuals generated with a certain type of crossover and stored in the outer set. The more non-dominated individuals are generated with the given type of operator, the more resources it gets. In this context «resources» means the number of individuals in the subpopulation corresponding to a certain crossover type. In each $T$-th generation one-point, two-point and uniform recombinations are compared according to their «fitness» in pairs. «Penalty» is a parameter which denotes the amount of resources which the genetic operator with lower «fitness» gives the genetic operator with higher «fitness». Also decreasing of resources must be limited with a parameter named «*social card*». It is necessary to maintain genetic operators' diversity. Initially all types of genetic operators have equal amount of resources.

The fitness value of the $i$-th crossover operator is defined according to the following rule:

$$q_i = \sum_{l=0}^{T-1} \frac{T-l}{l+1} \cdot b_i \ ,$$

where $T$ is the adaptation interval, $k = 0$ corresponds to the latest generation in the adaptation interval, $k = 1$ corresponds to the previous generation, etc. $b_i$ is defined as following:

$$b_i = \frac{\frac{p_i}{|\overline{P}|}}{\frac{n_i}{N}} \ ,$$

where $p_i$ is the number of individuals in the current «outer» set generated with the $i$-th type of crossover operator, $|\overline{P}|$ is the outer set size, $n_i$ is the number of individuals in the current population generated with the $i$-th type of crossover, $N$ is the population size.

The effectiveness of operators is compared in pairs in every *T*-th generation to redistribute resources on the basis of «fitness» values:

$$s_i = \begin{cases} 0, if\ n_i \le social\_card \\ int(\frac{n_i - social\_card}{n_i}), if\ (n_i - h_i \cdot penalty) \le social\_card, \\ penalty, otherwise, \end{cases}$$

where $s_i$ is the size of a resource given by the *i*-th algorithm to those which won, $h_i$ is the number of losses of the *i*-th algorithm in paired comparisons, a *social_card* is the minimum allowable size of the population, a *penalty* is a fee size for defeated algorithms. The parameter *social_card* is introduced to maintain the diversity of operators; *penalty* – for the redistribution of resources.

Test problems [7] developed by the scientific community for the comparison of evolutionary algorithms were used to explore the efficiency of a «standard» SPEA and its adaptive analog suggested in this study. Below are presented some results of testing carried out on a set of multi-criterion problems with two objective functions.

The results of algorithms work were estimated with the *IGD*-metrics:

$$IGD(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|},$$

where $P^*$ is a set of points uniformly distributed along the PF, v is the point of $P^*$, *d(v, A)* is the minimum Euclidean distance between v and *A*, *A* is the approximate set of the PF.

In SPEA we have nine combinations of genetic operators' types and the testing was conducted for all of them. Results were averaged over thirty runs and estimated with the special IGD-metrics. The maximal number of function calculations was set to be 300 000 for every problem.

The analysis of testing results showed that the effectiveness of the adaptive SPEA was not lower than the effectiveness of the «average» SPEA. It means that on this set of test problems the maximal number of SPEA-modification defeats was equal to three (versus nine variants of settings).

*Table 2*

**The adaptive SPEA's testing results**

| Number of a test function | A value of IGD-metrics | The number of cases when the adaptive SPEA's lost to a conventional SPEA |
|---|---|---|
| 1 | 0,10579 | 2 |
| 2 | 0,04160 | 3 |
| 3 | 0,0049 | 1 |
| 4 | 0,04493 | 2 |
| 5 | 0,34105 | 2 |
| 6 | 0,0088 | 2 |
| 7 | 0,11738 | 1 |

Self-adaptation is an alternative to random choice of genetic operators or multiple runs of GA for each variant of settings.

Below there is Fig. 1 reflecting the self-configuration process of crossover for one of test problems which were discussed above:

$$\begin{cases} f_1 = x_1 + \frac{2}{|J_1|} \sum_{j \in J_1} h(y_j) \to min, \\ f_2 = 1 - x_1^2 + \frac{2}{|J_2|} \sum_{j \in J_2} h(y_j) \to min, \end{cases}$$

where $J_1 = \{j \mid j - odd, \quad 2 \le j \le n\}$ и $J_2 = \{j \mid j - even, \quad 2 \le j \le n\}$, $y_j = x_j - sin(6\pi x_1 + \frac{j\pi}{n})$, $n=30$, $j = 2,...,n$, $h(t) = \frac{|t|}{1+e^{2|t|}}$, $\bar{x} \in [0,1] \cdot [-2,2]^{n-1}$.

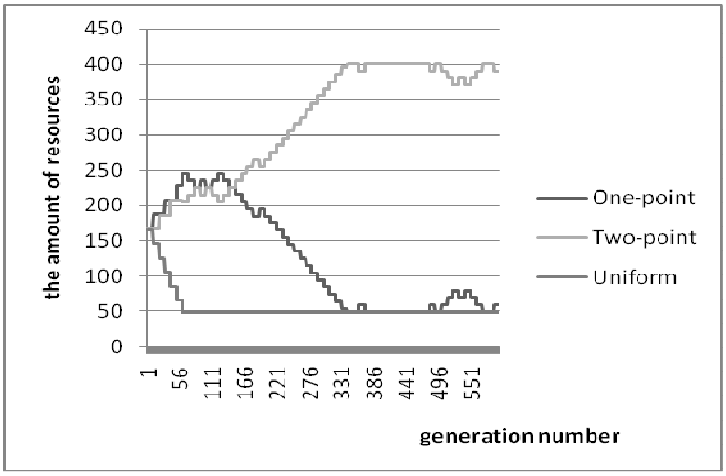


Fig. 1. Self-configurable crossover operator's behavior for one of test problems.

The following parameters of GA were set: the size of population was equal to 500; the number of generations was equal to 600. The «penalty» was 10 and the «social card» was 50.

The study of «standard» SPEA showed that the best variant of crossover for the given problem is a two-point one.

Investigation of a self-configurable crossover operator demonstrated that after the competition in the first several generations the best type of recombination obtains more resources than others.

But not always there is an undisputed leader among variants of genetic operators. One-point, two-point and uniform crossover or only some of them often compete during the algorithm execution. In some generations one type of crossover has the highest «fitness» but in other generations – another one. A specific feature of self-configuration is the opportunity to apply more effective settings in the current generation.

## 4. Application of adaptive GAs for the ANN-design

So realized adaptive single- and multi-objective genetic algorithms were integrated into the program system used for neural networks design. As it was mentioned above an adaptive multi-objective GA was applied to find ANN-models according to the theory of Pareto's optimality. On the one hand it is important to determine the model with the required accuracy and in this case a root mean square or relative error may be used. And on the other hand for saving computational resources it is necessary to find compact structures for ANNs (e.g., the number of neurons and connections between them). Therefore these two criteria were taken into account for ANN-modeling. An adaptive single-objective GA was used to adjust weights for ANNs.

In SPEA structures of ANNs are represented with binary code according to the following rule. Previously the maximal number of hidden layers and the maximal number of neurons in the each layer are set. Activation functions of neurons are chosen from the finite set of functions (e.g., sinus, sigmoid, Heaviside function, linear function, hyperbolic tangent, triangular function and so on) which have index numbers. These index numbers are coded with the sequence of genes which compile a chromosome.

Weights for every ANN-structure also have a binary representation: it is a sequence of binary codes which correspond to certain weights.

The effectiveness of the realized approach was investigated on test problems called «German credits» and «Australian credits» (all samples were borrowed from the machine learning repository [8]). These credit data include the information about creditors: age, gender, marital status, credit history records, job, etc. on the basis of which the decision about credit approvals is made. The data include continuous and categorical variables.

Involved samples contain 1000 and 690 examples respectively. These data sets were randomly divided into learning and test samples with proportion 80% to 20%. The program system was launched with the following values of parameters: an adaptive SPEA had 20 generations and 50 individuals, an adaptive single-objective GA had 20 generations and 20 individuals, in the last generation of SPEA execution a single-objective GA had 100 generations and 100 individuals. After that the local search algorithm (Hooke-Jeeves search) was applied. The maximal number of hidden layers was 2 and the maximal number of neurons in the each layer was 10.

As a result approximations of Pareto's set and front were defined for each problem. Table 3 contains found points of Pareto's front:

*Table 3*

**Approximations of the Pareto's front for benchmark problems**

| Australian Credits | | | German Credits | | |
|---|---|---|---|---|---|
| The number of neurons | | Relative error | The number of neurons | | Relative error |
| Layer 1 | Layer 2 | | Layer 1 | Layer 2 | |
| 8 | 7 | 0,093525 | 8 | 4 | 0,2200 |
| 7 | 6 | 0,107914 | 7 | 2 | 0,2250 |
| 6 | 6 | 0,115108 | 5 | 3 | 0,2350 |
| 4 | 6 | 0,129496 | 6 | 1 | 0,2550 |
| | | | 5 | 1 | 0,3000 |

To estimate the found results we present the results of alternative approaches [9, 10]: two-stage genetic programming algorithm (2SGP), conventional genetic programming (GP), multilayered perceptron (MLP), classification and regression tree (CART), C4.5 decision trees, k nearest neighbors (k-NN), linear regression (LR), Bayesian approach, boosting, bagging, random subspace method (RSM), cooperative co evolution ensemble learning (CCEL).

*Table 4*

**Results of alternative approaches**

| Method | Australian Credits (relative error) | German Credits (relative error) | Method | Australian Credits (relative error) | German Credits (relative error) |
|---|---|---|---|---|---|
| SCGP | 0,0978 | 0,205 | Bayesian approach | 0,153 | 0,321 |
| MGP | 0,1015 | 0,2125 | Boosting | 0,24 | 0,3 |
| 2SGP | 0,0973 | 0,1985 | Bagging | 0,153 | 0,316 |
| GP | 0,1111 | 0,2166 | RSM | 0,148 | 0,323 |
| Fuzzy classifier | 0,109 | 0,206 | CCEL | 0,134 | 0,254 |
| C4.5 | 0,1014 | 0,2227 | CART | 0,1256 | 0,2435 |
| LR | 0,1304 | 0,2163 | MLP | 0,1014 | 0,2382 |

But it should be understood that this comparison is approximate because of the absence of information about computational costs for the compared methods. Also the investigated approach allows finding several solutions taking into account some quality criteria and testing results of alternative approaches contain the information about only one solution (the best or average).

## 5. Conclusion

In this study we have presented two self-adaptive GAs, one for the one-criterion optimization and another one for the multi-criterion optimization, as well as their cooperation in automated design of ANN-based classifiers. Classifiers designed in this way are enough accurate and also have simple structures.

Directions of the future research can be divided into three groups. The first of them is the improvement of GAs self-adaptation abilities. The second direction is the modification of ANN-based models automated design through including different ANNs types (RBF, Hopfield-Tank, etc.). And the third is an expansion of adaptive GAs application areas, e.g., automated design of fuzzy systems, decision trees, etc.

## References


1. Holland J.H. Adaptation in natural and artificial systems / J.H. Holland – Ann Arbor. MI: University of Michigan Press, 1975.
2. Niehaus J. Adaption of Operator Probabilities in Genetic Programming / J. Niehaus, W. Banzhaf // Proceedings of the 4th European Conference on Genetic Programming, Lecture Notes In Computer Science, vol. 2038. – Springer-Verlag, Berlin, Heidelberg. – 2001. – pp 325–336.
3. Zitzler E., Thietle L. Multi-objective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on evolutionary computation*, Vol. 3, No. 4, November 1999.
4. Eiben A.E., Hinterding R., and Michalewicz Z. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, 1999.
5. Daridi F., Kharma N., Salik J.F.N. Parameterless Genetic Algorithms: Review and Innovation. *IEEE Canadian Review*, No. 47, Summer 2004.
6. Sergienko R., Semenkin E. Competitive Cooperation for Strategy Adaptation in Coevolutionary Genetic Algorithm for Constrained Optimization *Proc. of 2010 IEEE Congress on Evolutionary Computation*, pp. 1626-1631, 2010
7. Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari. Multi-objective optimization test instances for the CEC 2009 special session and competition. University of Essex and Nanyang Technological University, Tech. Rep. CES-487, 2008.
8. *UCI Machine Learning Repositiry: Data Sets.* Available at: http://archive.ics.uci.edu/ml/datasets.html.
9. Huang J.-J., Tzeng G.-H., Ong Ch.-Sh. Two-stage genetic programming (2SGP) for the credit scoring model // Applied Mathematics and Computation, 174 (2006): 1039–1053.
10. Sergienko R., Semenkin E., Bukhtoyarov V. Michigan and Pittsburgh Methods Combining for Fuzzy Classifier Generating with Coevolutionary Algorithm for Strategy Adaptation. In: 2011 IEEE Congress on Evolutionary Computation (CEC'2011), June 5-8, 2011, New Orleans, LA.