ISSN 2347 - 3983
Volume 8, No. 1 January 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter24812020.pdf
https://doi.org/10.30534/ijeter/2020/24812020

Application of Artificial Neural Networks in Genetic Algorithm Control Problems

David Aregovich Petrosov[1], Roman Alexandrovich Vashchenko[2], Alexey Alexandrovich Stepovoi[3], Natalya Vladimirovna Petrosova[4]
[1]Financial University under the Government of the Russian Federation, Russia
[2]Belgorod State Technological University named after V.G. Shukhov, Russia
[3]Belgorod State Technological University named after V.G. Shukhov, Russia
[4]Belgorod state agricultural university named after V. Gorin, Russia

ABSTRACT

In contemporary intelligent decision support systems, there is still a problem associated with increasing the performance speed of the structural-parametric synthesis of large discrete systems with a given behavior based on genetic algorithms. Currently, there are two main research areas that are designed for mathematical or hardware performance speed improvement. One way to improve hardware performance speed is the use of parallel computing, which includes general-purpose computing on graphics processing units (GPGPU). This article deals with the possibility of improving the performance speed of intelligent systems using the mathematical tool of artificial neural networks by introducing a control module of the genetic algorithm directly when performing the synthesis of solutions. Control of the structural-parametric synthesis process is achieved by predicting and evaluating the state of the genetic algorithm (convergence, attenuation, finding the population in local extremes) using artificial neural networks. This allows changing the operating parameters directly in the course of decision synthesis, changing their destructive ability relative to the binary string, which leads to a change in the trajectory of the population in the decision space, and as a result, should help to improve the performance speed of intelligent decision support systems.

Key words: genetic algorithm, intelligent information systems, artificial neural networks, system analysis.

1. INTRODUCTION

Contemporary intelligent decision support systems based on evolutionary procedures in the synthesis of the structural and parametric problem of large discrete systems with a given behavior need to improve performance speed.

In the structural-parametric synthesis of a large discrete system with a given behavior, the number of elements and parameters of their functioning is so large that the solution of this problem by iteration is impossible even with the use of contemporary computing systems. Therefore, it is advisable to use directional search methods. The range of such methods is quite wide and includes the following algorithms:

- genetic algorithms;
- simulated annealing method;
- set of algorithms based on the ant colony method;
- rough random search (Monte Carlo method);
- binary search algorithm;
- algorithm with a return at a failed step;
- algorithm with recalculation at a failed step;
- search with bans.

Genetic algorithms have proven themselves well in the field of structural-parametric synthesis, but when solving synthesis problems with a large number of elements and connections between them, it is often necessary to adjust the functioning parameters of operators.
This is associated with the following:

- getting the population into the local extremum;
- attenuation of the genetic algorithm;
- slow convergence.

Therefore, it is advisable to develop methods that can predict and evaluate the state of the genetic algorithm directly in the course of operation.

2. MATERIALS AND METHODS

The use of artificial neural networks that have well proved themselves in solving problems of forecasting, management, and analysis is proposed in this work as the main tool.

3. RESULTS

Let consider in more detail the use and application of the genetic algorithm [1].

The genetic algorithm can be represented as a tuple of operators:

$GA = <SEL, CROSS, MUT, RED>$,

where *SEL* is the selective operator; *CROSS* is the crossover operator; *MUT* is the mutation operator, and *RED* is the reduction operator.

Each tuple element is represented as a set of possible functioning parameters:

$SEL=\{Psel_i\}_{i=1}M$,

where *Psel* is the i-th parameter of the selective operator functioning (roulette, tournament, etc.). For example, $Psel_1$ is the tournament selection; $Psel_2$ is the roulette selection.

$CROSS = \{Pcross_j\}_{j=1}^{L}$,

where *Pcrossj* is the j-th functioning parameter of the crossover operator (single-point, two-point, etc.). For example, $Pcross_1$ is the single-point crossing; $Pcross_{2\ is}$ the two-point crossing; $Pcross_3$ is the multipoint crossing.

$MUT= \{Pmut_f\}_{f=1}^{K}$,

where *Pmutf* is the f-th functioning parameter of the mutation operator (different probabilities of the operator triggering). For example, $Pmut_1$ is the probability of mutation 0.1; $Pmut_2$ is the probability of mutation 0.3; $Pmut_3$ is the probability of mutation 0.5.

$RED=\{Pred_o\}_o = {}_1^r$,

where $Pred_o$ is the o-th functioning parameter of the reduction operator (depends on the approach to the number and quality of individuals left in the generation by the objective function value). For example, $Pred_1$ is the number of individuals in the generation of 50% according to the best value of the objective function; $Pred_2$ is the number of individuals in the generation of 40% according to the best value of the objective function; $Pred_3$ is the number of individuals in the generation of 60% according to the best value of the objective function.

Thus, when solving the problem of structurally parametric synthesis of large discrete systems, the genetic algorithm with the functioning parameters of operators can be presented in the following form:

$GA = Psel_i, Pcross_j, Pmut_f\ Pred_o$.

In problems of structural and parametric synthesis of large discrete system models it is possible to modify the element base or elements' operating parameters, while, as practice shows, the parameters of the genetic algorithm operators will require adjustment because the decay quickens. The problem arises to control the genetic algorithm directly in the course of finding solutions, i.e. changing the parameters of its operators that will increase the performance speed of the intelligent decision support system and improve the quality of decision synthesis.

In this case, it is proposed to consider the parameters of the operators in terms of their destructiveness, since each operator performs a change in the binary string, destroying it relative to the original state. When showing signs of attenuation, it is advisable to increase the destructive ability of operators, thereby obtaining greater number of individuals with new qualities.
Figure 1 shows an example of the attenuation of the genetic algorithm when solving the problem of structural-parametric synthesis of a large discrete system with the dimension of 18x18 (based on the memory element base: *RS*, *D* and *T* triggers [2]) with the following parameters:

$GA = <Psel_1\ Pcross_1, Pmut_1\ Pred_2>$.

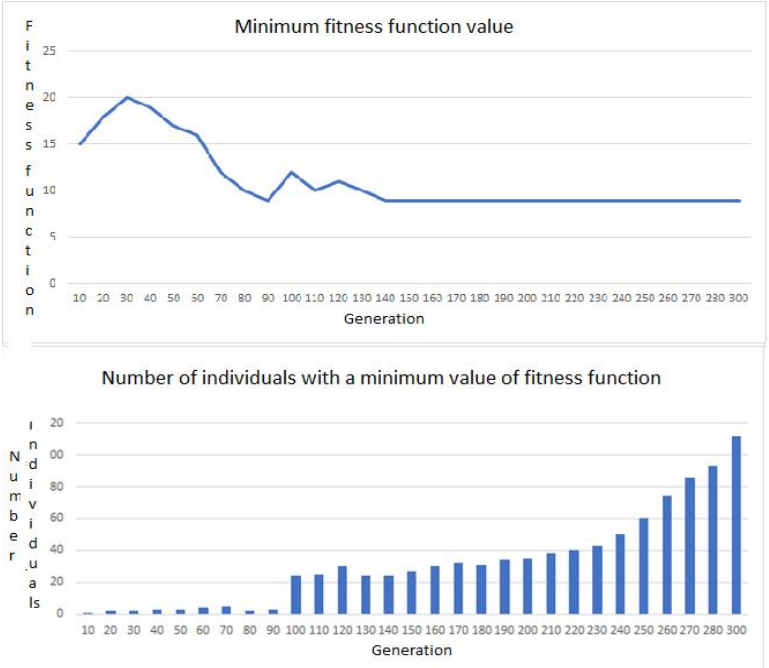

Figure 1: Example of attenuation of the genetic algorithm of discrete systems

When processing 140 generations, damping was formed, at which the minimum value of the objective function did not change for several generations, and the number of individuals with this function value increased. Without using a neural network, the algorithm processed 300 generations and found no solutions.

In this regard, it is proposed to use artificial neural networks to solve the problem of the genetic algorithm control.

The technology of the joint use of genetic algorithms and neural networks is very popular in various scientific studies. Genetic algorithms are used to search for the structure and training of neural networks, as well as to search for neural networks to work inside the genetic algorithm. The first approach is more common. The present article considers just the application of the second approach – the use of neural networks to configure the genetic algorithm [3].

The first step is to choose the neural network structure. There are two classes of the most frequently used neural network structures [4]. These are neural networks without feedback, so-called networks with unidirectional signal propagation, and neural networks that have feedback (recurrent neural networks). For the problem to be solved, it was found experimentally that the use of recurrent neural networks in comparison with direct propagation networks did not lead to a significant increase in its accuracy. To analyze the change in the fitness function, fifty of its last values were selected. Accordingly, the number of inputs to the neural network was also equal to fifty. Therefore, a multilayer direct distribution network with one hidden layer and an output layer was chosen (Fig. 2). The number of neurons in the output layer was the same as the number of output parameters.

The training was performed using the Levenberg–Marquardt algorithm. Each neural network output corresponds to the number of operators of the genetic algorithm [5].

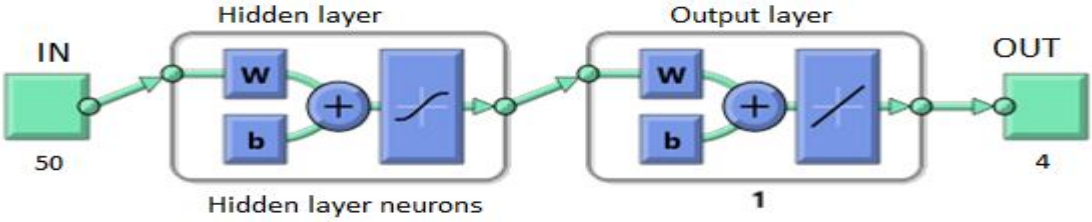

Figure 2: Neural network structure

After determining the overall structure of the network, it is necessary to select the number of neurons in the hidden layer. On the one hand, too few hidden neurons lead to insufficient accuracy of the model. On the other hand, too many neurons can lead to low generalization ability, when the model produces good results on the examples included in the training sample, but almost does not work on other examples. To select a rational number of neurons in the hidden layer, several neural networks with the number of neurons in the hidden layer from 2 to 200 were tested.

Due to the fact that the training results are greatly influenced by the selection of the initial values of the network weights, and random selection of the initial values of the weights is used (due to the lack of a universal method to select weights), each of the neural networks was trained and tested 25 times. The best one was selected among all tested networks, providing the least error. At that, the fitness function value was chosen as an accuracy indicator.

Figure 3 shows the neural network accuracy depending on the number of neurons in the hidden layer obtained using the training data and the supporting set.

As is seen from the graph below (Fig. 3), the error on the sample used for training decreases as the number of neurons in the hidden layer increases, while error on the data not involved in training increases.

The network showing the minimum error calculated as the average between the error on the new and training data was chosen as the best one. This fact indicates the high generalizing ability of the chosen network. Such a result was shown by a neural network with 19 neurons in the hidden layer.

To assess the feasibility of using artificial neural networks, a computational experiment was conducted using genetic algorithm models based on nested Petri nets [6; 7].

Figure 3: Dependence of neural network accuracy on the number of neurons in the hidden layer.

In the course of the experiment, the neural network recognized the attenuation of the genetic algorithm on the 180th generation, and performed changes in the operation parameters:

GA = <Psel1 Pcross2, Pmut2, Pred3>.

This change led to the finding of a solution in the 270th generation and the convergence of the genetic algorithm (Fig. 4).

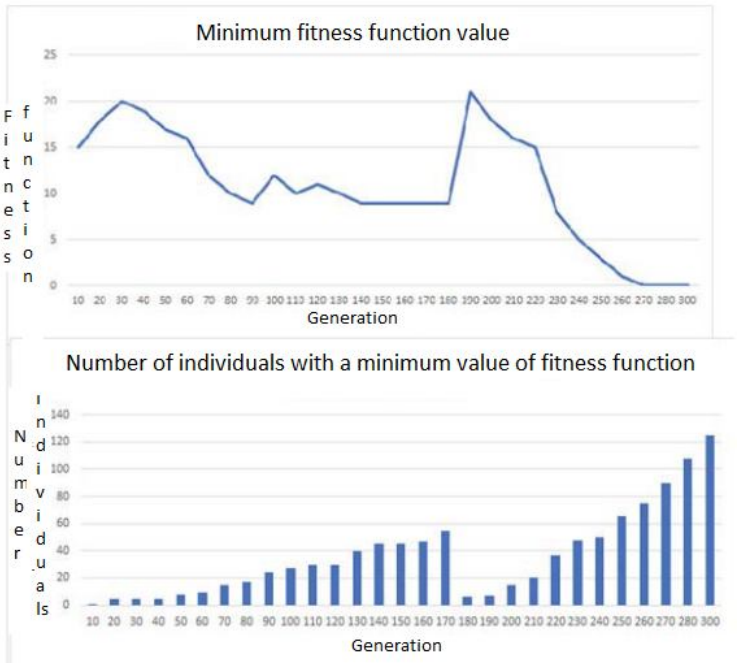

Figure 4: Convergence of the genetic algorithm after controlling the artificial neural network

4. DISCUSSION

Thus, it can be concluded that changing the operating parameters of the genetic algorithm in the course of searching for solutions can lead to an exit from the state of decay, and contributes to its convergence. In general, this approach cannot only recognize the attenuation of the genetic algorithm but also helps to reduce the time of finding solutions. To do this, it is necessary to study all possible conditions of the genetic algorithm in which changing the operating parameters can have a positive effect, as well as determine these parameters, perform their visualization, and train an artificial neural network.

5. CONCLUSION

The neural network approach in the genetic algorithm control problems is a promising research line in the field of artificial intelligence. In the framework of the present research, it is advisable to perform simulation of an artificial neural network using the mathematical apparatus of Petri nets that will allow combining the genetic algorithm model based on nested Petri nets and the artificial neural network model. Due to the property of parallelism, which is peculiar to evolutionary methods and Petri net theory, it is possible to use the GPGPU technology when conducting software implementation of an intelligent information system.

ACKNOWLEDGMENTS

The work was supported by the grant of the Russian Foundation for Basic Research No. 18-07-00634-A.

REFERENCES

1. A.N. Orlov, V.V. Kureychik, A.E. Glushchenko. Kombinirovannyj geneticheskij algoritm resheniya zadachi raskroya [A combined genetic algorithm for solving the cutting problem]. Bulletin of the South Federal University. Technical sciences, 6(179), pp. 5-13, 2016.
2. D.A. Petrosov, V.A. Lomazov, A.I. Dobrunova, S.L. Matorin, V.I. Lomazova. Evolutionary synthesis of large discrete systems with dynamic structure. Biosciences Biotechnology Research Asia, 12(3), pp. 2971-2981, 2015.
https://doi.org/10.13005/bbra/1981
3. V.G. Manzhula, D.S. Fedyashov. Nejronnye seti Kohonena i nechetkie nejronnye seti v intellektual'nom analize dannyh [Kohonen neural networks and fuzzy neural networks in database mining]. Fundamental Research, 4, pp. 108-114, 2011.
4. S.O. Haykin. Neural networks: A complete course, p. 1104, 2006.
5. D.A. Petrosov, V.A. Lomazov, V.I. Lomazova, A.V. Glushak. Applications of parallel computations in the problems of structural-parametric synthesis of discrete systems based on evolution methods. Journal of Advanced Research in Dynamical and Control Systems, 10(10), Special Issue, pp. 1840-1846, 2018.
6. D.A. Petrosov, V.A. Lomazov, A.L. Mironov, S.V. Klyuev, K.A. Muravyov, F.M. Vasilieva. Intellectual

structural-parametric synthesis of large discrete systems with a specified behavior. Journal of Engineering and Applied Sciences, 13(8), pp. 2177-2182, 2018.

7. D.A. Petrosov, V.A. Lomazov, A.I. Dobrunova, S.I. Matorin, V.I. Lomazova. Large discrete systems evolutionary synthesis procedure. Biosciences Biotechnology Research Asia, 12(2), pp. 1767-1775, 2015. https://doi.org/10.13005/bbra/1841