The Parallel Genetic Algorithm for Construction of Technological Objects Neural Network Models

Tynchenko Vadim Sergeevich
Dept. of Production Machinery and Equipment for Petroleum and Natural Gas Engineering
Siberian Federal University
Krasnoyarsk, Russia
vadimond@mail.ru

Petrovsky Eduard Arkadievich
Dept. of Production Machinery and Equipment for Petroleum and Natural Gas Engineering
Siberian Federal University
Krasnoyarsk, Russia

Tynchenko Valeriya Valerievna
Dept. of Informatics
Siberian Federal University
Krasnoyarsk, Russia

***Abstract*— The parallel genetic algorithms implementation for neural networks models construction is discussed. The modification of this global optimization algorithm is proposed. The artificial neural networks are effective instrument to solve most problems of technological objectives and processes modelling. The article describes the aspects of genetic algorithms implementation for neural networks structure-parametric synthesis. It is offered to use different parallelization technique of genetic algorithm to increase computing performance. It is proposed to modify the standard multipopular parallel genetic algorithm adding its base topology dynamic adaptation. This approach allows to make effective algorithm with minimal computational difficulty. The algorithm modification shows best results, when implemented in computer network.**

Keywords— neural networks; optimization; genetic algorithm; parallelization; modelling

I. INTRODUCTION

Artificial neural networks (ANN) represent an effective tool for solving a wide range of complex scientific and engineering problems, in particular, modeling of technological objects and processes. However, the impossibility of formalization of the neural network model structure-parametric synthesis significantly limits the practical application of this approach.

Automating the process of building arbitrary architecture neural networks involves solving complex multiparameter optimization problems, such as choosing the efficient structure of neural network and its training [1, 2].

To solve such optimization problems can be applied only adaptive direct search algorithms, which allows a global search in the solution space and to avoid local optima. Also it does not require information about the optimized function properties. Among these algorithms the most common are evolutionary algorithms (EA).

II. GENETIC ALGORITHMS FOR ARTIFICIAL NEURAL NETWORK CONSTRUCTION

In the genetic algorithm (GA), adapted to the training of neural networks, each individual of the population represents a complete set of neural network weights [3]. Evaluation of fitness of individuals is determined by a fitness function defined as the sum of square errors, i.e. differences between the expected and actual values at the output of the network for different input data. Individuals are encoded as binary sequences (chromosome) [4, 5, 6].

Genetic algorithm is applied to a population of individuals (chromosomes containing the encoded set of weights of a neural network) with the implementation of the model cycle of evolution, consisting of the following four steps. [7, 8, 9]

1. The decoding of each individual (chromosome) from the current generation to restore a plurality of scales and the design of the corresponding neural network with a priori given architecture.

2. The calculation of the total square error between the actual and setpoint values for all outputs of the network. This error defines the fitness of individuals.

3. The reproduction of individuals according to the chosen method of selection.

4. The application of genetic operators of crossing and mutation to obtain a new generation.

On the first step the general conceptual structure of data representation is to be chosen [10, 11]. Information about the structure can be directly encoded as a binary sequence, i.e. each link and each node (neuron) specified a certain number of bits. This way of representation scheme is called direct encoding (strong specification schema).

The second stage of the evolutionary design of neural network structure consists of the following steps [12, 13]:

1. The decoding of each individual of the current population to describe the architecture of the neural network.

2. The training of each neural network with the architecture obtained in the first step, using the genetic algorithm.

3. The evaluation of each individual (encoded architecture) fitness according to the achieved learning outcomes, i.e. the smallest total training square error.

4. The reproduction of individuals according to the chosen method of selection.

5. The application of genetic operators of crossing and mutation to obtain a new generation.

III. Parallel Genetic Algorithms

Performing calculations on evolutionary algorithms require serious computational resources. Application of parallel genetic algorithms (PGA) for the distributed solution of neural network modeling problem allows to significantly improving performance through parallel execution of computations.

The choice of method to parallelize the genetic algorithm depends on the following:

- how is the fitness estimated and the mutation applied;

- a single population is used or several subpopulations are made;

- how occurs the exchange of individuals between different subpopulations;

- how is applied breeding (globally or locally).

Depending on the implementation of each of these points, there are different ways of genetic algorithm parallelization. Massively parallel PGA, PGA with dynamic subpopulations, stationary, nonstationary, and hybrid methods of parallelization of GA [8, 14, 15] require either the presence of shared memory, or a significant information exchange between the computational nodes, which makes them inadequate for implementation in computer networks.

The most common and significant are the following two methods [16]:

- master-slave PGA with distributed evaluation of the fitness (MSPGA);

- multipopulation PGA (MPPGA).

MSPGA uses a single population. The performance growth is achieved by the fitness parallel evaluation. In this case productivity losses are associated with the cost of the information transfer.

In the MPPGA is used a small number of large subpopulations evolving in isolation on different compute nodes. For communication between subpopulations there is added the migration operator. The nature of migration depends on the topology of connections between subpopulations (isolated populations, complete graph, ring, etc.) and the rate of migration (number of individuals moved), migration patterns (some individuals move to another population and be replaced by it – "best", "worst", randomly selected), migration interval (frequency of movement).

MPPGA was selected for detailed study because of the following advantages:

- the covering of a large search space,

- low probability of premature convergence,

- reducing the calculation time is greater than the number of computational nodes, due to the possibility of more efficient implementation of the algorithm.

IV. The Study of Different Parallel Genetic Algorithms Efficiency

As part of the ongoing research has been realized the client-server software system that allows to automate the process of selecting effective patterns of ANN. The program system provides the following features:

- to set the parameters of genetic algorithm for tuning weights of the ANN connections (the amount of population, genetic operators, the number of iterations, etc.);

- to change the parameters of the genetic algorithm for the selection of the ANN effective patterns (amount of population, genetic operators, the number of iterations, etc.);

- to set the parameters of the parallel genetic algorithm (topology of connectivity between subpopulations, speed and schema migration as well as migration interval);

- to monitor the ANN design process.

The computational experiments have been carried out in the network of similar personal computers with Intel Pentium G2020 processor cores. The problem was to build an effective ANN of the ore-thermal melting technological process (OTM) [17].

Let's describe the problem statement. There were chosen the parameters that characterizing the efficiency of the furnace ore-thermal melting. These parameters assess technological, energetic and economic aspects and include:

- loss of non-ferrous metals from slag;

- performance;

- specific consumption of electricity;

- the temperature of the slag;

- emissions of harmful substances into the atmosphere;

- other parameters.

Due to the high temperature and aggressive environment in the furnace it is difficult to obtain continuous, reliable and detailed information from object. The main control actions use:

- the amount of loaded silica;

- the amount of coke fed;

- the amount of converter slag;

- the input electrical power;

- the electrodes penetration;

- voltage;

- other parameters.

The dimension of the input actions vector for a process is equal to 10, the dimension of the output vector parameters is also equal to 10.

For experiments was used the sample of 47 points.

There were made series of the computational experiments on the selection of genetic algorithm parameters to solve the stated problem.

The best values were:

- weights maximum value – 5.0;

- weights minimum value – 5.0;

- weights sampling increment – 0.0001;

- population size – 100 for training, 20 for selecting the ANN structure;

- number of generations – 100;

- selection – tournament (10 individuals in the tournament for training, 5 – to select the ANN structure);

- a crossbreeding – uniform;

- mutation – average;

- the maximum number of hidden neurons – 10;

- number of activation function types – 8.

The characteristic feature of MPPGA is the migration operator, which influence was studied in this paper.

The averaged results of numerical experiments over 50 runs, are presented in table 1 (MSPGA) and table 2 (MPPGA).

TABLE I. THE PROBLEM SOLVING TIME WITH MSPGA (MINUTES).

Number of computational nodes	Solving time, minutes
1	71.3
2	39.2
3	27.5
4	20.8
5	16.3
6	14.2
7	12.4
8	10.8

The tables show the dependence of the problem solving time (minutes) from the PGA type and the subpopulations connections topology.

TABLE II. THE PROBLEM SOLVING TIME WITH MPPGA (MINUTES).

Connections topology	Migration interval	Number of computational nodes						
		2	*3*	*4*	*5*	*6*	*7*	*8*
The ring	1	32.3	19.8	14.8	12.1	9.2	7.6	**6.18**
	3	32.8	19.3	14.1	11.7	8.9	7.2	**6.03**
	5	34	20.1	15.3	12.6	10.1	8.8	7.72
The full graph	1	32.3	20.6	16.1	13	10.6	9.1	8.16
	3	32.8	20.4	15.9	12.8	10.1	8.9	7.93
	5	34	20.6	16.4	13.9	11.8	9.5	8.41

The best results (with a fixed schema migration "the best replaces the worst" [18, 19, 20] and a given ANN model square error 3.5%) were obtained for the topology "ring". The complication of the topology (implementing PGA in a network) leads to a substantial increase of information exchange.

V. THE PARALLEL GENETIC ALGORITHM WITH RESTRUCTURATION

In this paper we propose to modify the migration operator of MPPGA by supplementation the restructurization, which consists of adding the temporary connections between isolated populations that are not well-functioning at the moment.

Such populations could obtain additional migrants from the best individuals of those populations, which show quite good results.

Scheme of modified MPPGA with restructuration.

The base migration scheme is "the best replaces the worst".

Let's introduce notations:

- N – number of populations,

- m_i – the migration speed of the population i ($i = 1, \ldots, N$);

- k_i – the migration speed of the population i ($i = 1, \ldots, N$);

- Q_i – the "quality" of the population i ($i = 1, \ldots, N$);

- f_{ij} – the fitness of individual j in population i.

Let's describe the algorithm on the example of population l.

1. To make k_l cycles of standard GA.

2. To evaluate the quality criteria (1):

$$Q_i = \sum_{j=1}^{m_l} f_{ij}/m_l \qquad (1)$$

for all populations. There f_{ij} – the fitness of individual j in population i from the set of its best m_l individuals.

3. If the inequality (2) is executed

$$Q_l < \sum_{i=1}^{N} Q_i/N \,, \qquad (2)$$

then to add the temporary connection from the population *j* (where $Q_j > \sum_{i=1}^{N} Q_i/N$) to the population *l* with probability (3):

$$P_j = Q_j / \sum_k Q_k , \qquad (3)$$

where $Q_k > \sum_{i=1}^{N} Q_i/N$.

4. To make a migration according the new topology.

5. To restore the previous topology.

There were made series of the computational experiments on the MPPGA with restructuration to solve the OTM problem The averaged results of numerical experiments over 50 runs, are presented in table 2. The table shows the dependence of the problem solving time (minutes).

TABLE III. THE PROBLEM SOLVING TIME WITH MPPGA WITH RESTRUCTURATION (MINUTES).

Base connection topology	**Migration interval**	**Number of computational nodes**					
		3	***4***	***5***	***6***	***7***	***8***
The ring	1	17,55	13,18	10,76	8,20	6,68	**5,48**
	3	17,31	12,68	10,49	8,02	6,50	**5,38**
	5	18,84	15,30	11,54	9,31	8,14	7,07
The average perfomance increase comparing to standart MPPGA (percent)		9,31	7,00	9,94	9,52	9,77	10,17

VI. CONCLUSION

Thus, it can be argued that the proposed modification of multipopulation parallel genetic algorithm can improve the performance of the standard algorithm on average 9,29%. It indicates the efficiency and effectiveness of this approach to solve complex problems of neural network modeling in computer networks.

REFERENCES

[1] Rutkovskaya, D. Neural networks, genetic algorithms and fuzzy systems. M. : Hot Line - Telecom, 2004. – 452 p.

[2] Zayencev, I.V. Neural networks: main models. – Voronezh : VFTI, 1999. – 76 p.

[3] Wasserman, F. Neurocomputing technic. Theory and practice – M. : Mir, 1984. – 256 p.

[4] De Jong, K. A Genetic Algorithms: A 10 Year Perspective / K. De Jong // The First Int. Conf. on Genetic Algorithms: Proceedings. – 1985. – P.167–177.

[5] Goldberg, D. Genetic algorithms in search, optimization and machine learning / D. Goldberg. – Reading, MA: Addison-Wesley, 1989. – P. 230–241.

[6] Akopyan, A.M. Genetic algorithms for glaobal optimization problems solving. URL: http://www.cp.niif.spb.su/inpe/4/gaover/gaover.htm

[7] Batischev, D.I. Genetic algorithms for extremal problems solving. – Voronezh : VFTI, 1995. – 210 p.

[8] Baluja, S. A massively distributed parallel genetic algorithm (mdpga) / S. Baluja // Technical Report CMU-CS-92-196R. – Pittsburg, PA : Carnagie Mellon University, 1992. – P. 134–158.

[9] Haupt, R.L. Practical Genetic Algorithms / R.L. Haupt, S.E. Haupt. – 2ed. – Wiley, 2004. – 261p.

[10] Isaev, S.A. The genetic algorithms in popular. URL: http://saisa.chat.ru/ga/ga-pop.html#top

[11] Janikow, C. Z. Genetic algorithms simulating nature's methods of evolving the best design solution / C. Z. Janikow, D. St. Clair // IEEE Potentials. – 1995, October. – Vol. 39, No.14. – P. 31–35.

[12] Goldberg, D. A comparative analysis of selection schemes used in genetic algorithms / D. Goldberg, K. Deb // In Foundations of Genetic Algorithms. – San Mateo, CA : Morgan Kaufmann, 1991. – P. 69–93.

[13] Starikov, A. Genetic algorithms – the mathematical apparatus. URL: http://www.basegroup.ru/genetic/math.htm

[14] Baluja, S. The evolution of genetic algorithms: Towards massive parallelism / S. Baluja // the Tenth International Conference on Machine Learning: Proceedings. – San Mateo, CA : Morgan Kaufmann, 1993. – P. 1–8.

[15] Nowostawski, M. Review and taxonomy of parallel genetic algorithms /M. Nowostawski // School of Computer Science, The University of Birmingham, UK, May 1999.

[16] Cantu-Paz, E. Designing efficient master-slave parallel genetic algorithms / E. Cantu-Paz // IllGAL Report 97004 / The University of Illinois, 1997.

[17] Gonebnaya, O. E. The expert system for ore-thermal melting: candidate of technical science disertation. – Krasnoyarsk: GUCMiZ, 2004. – 136 p.

[18] Cantu-Paz, E. Migration policies and takeover times in parallel genetic algorithms / E. Cantu-Paz // IllGAL Report 98009. – The University of Illinois, 1998. – P. 56 –81.

[19] Cantu-Paz, E. Migration policies and takeover times in parallel genetic algorithms / E. Cantu-Paz // IllGAL Report 98009. – The University of Illinois, 1998. – P. 56 –81.

[20] Booker, L. Improving search in genetic algorithms / L. Booker // L. Genetic algorithms and Simulated Annealing. – London : Pitman, 1987. – P. 61–73.