ISSN 2664-9969

UDC 004.032.26 (004.89)
DOI: 10.15587/2706-5448.2020.214769

Leoshchenko S.,
Oliinyk A.,
Subbotin S.,
Zaiko T.

USAGE OF SWARM INTELLIGENCE STRATEGIES DURING PROJECTION OF PARALLEL NEUROEVOLUTION METHODS FOR NEUROMODEL SYNTHESIS

The paper proposes the ways to apply swarm intelligence strategies to parallelize neuroevolution methods for synthesizing artificial neural networks. The proposed approaches will solve a number of problems that usually arise during designing high-performance computing related to the synthesis of neural networks. The object of research is the process of developing a parallel approach for the neuroevolution synthesis of artificial neural networks, namely, the use of swarm intelligence strategies to solve a number of problems in designing a method that would use the resources of a parallel computer system.

One of the most problematic areas is the highly adaptive nature and significant operating time of neuroevolution methods. One way to solve these problems is to use parallel computer systems and distributed computing. However, a number of questions arise when designing a parallel neuroevolution method.

During research a number of tasks were solved, which included the analysis and study of neuroevolution methods for synthesizing artificial neural networks and problems of their parallelization. Attention is also paid to swarm intelligence methods, which have gained popularity recently and show good results.

The new method developed during the work was based on strategies for organizing work with swarm particles. Thus, sub-populations distributed between threads and individuals were analyzed as individual particles that interact with each other and depend on the local environment. Classical genetic operators were modified by criterion mechanisms to improve adaptability.

During the experiments, the developed method was compared with classical methods. During the work, special attention was paid not only to the characteristics of the resulting neuromodels, but also to the load on the processor during Operation. The developed method showed acceptable results for all comparisons. The new approach has significantly improved the quality level of the parallel neuroevolution synthesis method, allowing to evenly use the capabilities of computing nodes in a parallel system.

Keywords: *neuroevolution method, genetic algorithm, swarm intelligence, parallel system, high-performance computing.*

Received date: 29.05.2020
Accepted date: 30.06.2020
Published date: 31.10.2020

1. Introduction

The usage of machine learning techniques and artificial neural network (ANN) tools facilitates, automates and improves accuracy during processing many types of data. ANN-based models are used for diagnostics and forecasting in engineering, manufacturing, medicine, economics, and so on. Benefits are especially noticeable when processing large amounts of data, when, for example, there is a complex technical system and many sensors installed at different stages. During system testing, each sensor measures a certain number of values every second. Sometimes such measurements can be measured in gigabytes of information per second. Of course, to accurately process such a volume, it is necessary to use models with high accuracy that will allow to track the relationship of independent parameters, their impact on categorical (dependent) variables, and, for example, build a forecast.

However, the synthesis of the optimal model is an important issue. The synthesis of a model based on the ANN consists of two main stages: initial determination of the topology (architecture or structure of the ANN) and training (determination of the weights of connections, neurons, etc.). For the training stage, the error propagation method has become widespread and popular. However, this method has a number of disadvantages [1]:

– indefinitely long learning process. Usually, it is difficult to predict how long the training itself will take, because much depends on the input data, a certain network architecture, and so on;
– the problem of network retraining. There are certain risks associated with local extremes;
– excessive dependence on the expert. Quite a lot of free parameters are set up by the expert before starting training, so big risks are associated with the role of an expert.

All in all, the error propagation method is used only for the training stage of the already developed ANN topology. An alternative approach has always been evolutionary methods [1], which in the context of ANN synthesis are referred to as neuroevolution methods.

2. The object of research and its technological audit

The object of research is the process of designing a parallel method of neuroevolution synthesis of ANN, namely, the using of swarm intelligence strategies to solve a number of problems in designing a method that would use the resources of a parallel computer system.

Most neuroevolution methods have a number of advantages over, for example, gradient methods [1–3]. However, one of the main disadvantages of such methods is their highly adaptive nature and significant operating time [4, 5]. These problems are usually explained by the stochastic nature of methods. One way to solve these problems is to use parallel computer systems and distributed computing.

A number of previous works [6–8] identify typical problems that arise in the design of parallel ANN synthesis based on evolutionary strategies. Usually, such problems are associated either with architectural features of the system, or with certain difficulties in determining free parameters of methods.

In the first case, problems can arise due to rather resource-intensive operations of information exchange between threads, which significantly increase overhead and can slow down the overall performance of work. Another common problem is the insufficiently calculated distribution of heavy calculations between lightweight threads of systems based on graphics processors (GPUs) [6, 8].

In the second case, problems are usually associated with determining the amount of information that is distributed between the system threads, and then goes to the main thread. So, sufficiently large amounts of information can significantly reduce the speed of work. Therefore, for example, it is possible to distribute certain initialization-related steps directly between threads [6, 7].

As parallel computer systems become more common today, the task of making meaningful usage of the potential of such systems is becoming particularly important.

3. The aim and objectives of research

The aim of research is to analyze the features of swarm intelligence strategies and procedures for their further use in the design of parallel neuroevolution methods. This should ensure rational use of the potential of parallel computer systems at the stage of ANN synthesis. To achieve this aim, it is necessary to complete the following objectives:

1. Explore strategies and procedures for swarm intelligence methods.

2. Using the results of the study, design a parallel neuroevolution method for the synthesis of ANN.

3. Based on the analysis of experimental results of the developed method, develop further recommendations and instructions.

4. Research of existing solutions of the problem

4.1. Neuroevolution synthesis of artificial neural networks. Neuroevolution is a form of machine learning that uses evolutionary algorithms to train a neural network [2]. This approach is used in many areas: medicine, development of intelligent systems for civil and military purposes, games, training and management of automated agents and robots. In these cases, it is quite simple to measure the performance of the neural network, while it is very difficult or almost impossible to implement training with a teacher. This training method belongs to the category of reinforcement learning methods.

There are a large number of neuroevolution algorithms that are divided into two groups. The first category includes algorithms that produce the evolution of weights for a given network topology, and the other category includes algorithms that, in addition to the evolution of weights, also produce the evolution of the network topology. Although there are no generally accepted conditions for making distinctions, it is accepted that adding or removing connections in a network in the course of evolution is called complication or simplification, respectively [2, 3].

The genetic algorithm (GA), as one of the most common and popular evolutionary methods, has become widely used for the synthesis of ANN [4, 5]. Its application makes it possible to simultaneously synthesize a population of neural networks and then produce point changes (mutation operator). After that, the population is evaluated (the best individuals of the population are selected) and crossing is started. As a rule, simpler types of crossing are used, which guarantee relatively good results with economical consumption of computing resources: single-point or two-point crossover. However, for more fine-tuning of the new generation or the implementation of multi-parent crossing, the uniform crossing operator is used. The new generation goes through all the previous steps – the method is performed until the most acceptable ANN is obtained.

As is clear, the use of such methods in the future provides a number of advantages [1, 3]:

– a wide variety of the obtained topologies allows to choose the most acceptable solution. So, for example, for simple tasks, it is possible to set strict limits for obtaining simple ANNs, and more complex structures (recurrent, deep, convolutional ANN, etc.) will be obtained for more complex tasks;

– adaptability of the obtained ANNs. This means that during synthesis, the method can help synthesize a more accurate and structurally acceptable ANN for a specific task and set of input data;

– versatility. Such methods can work with different input data sets or different ANN topologies, without requiring accurate data on structural features;

– possibility of using a parallel approach. This is especially important in the context of the rapid development of modern parallel computing systems using multiple processor cores (CPUs) or frivolous GPU threads.

4.2. Problems of parallelization of genetic algorithms. However, the using of a parallel approach for GAs is associated with a number of problematic issues. Let's consider the most typical problems [6, 7]:

– selection or development of a strategy for interaction of the components of the method;

– selection of migration frequency between sub-populations;

– determination of migrating individuals and their number;

– determination of the structure of evolution of individual sub-populations.

Let's take a closer look at the problems. The structure of a parallel system is an important factor in the performance of a parallel algorithm, since it determines how fast (or how slow) the best solution propagates between

ISSN 2664-9969

populations [7]. If the system is strongly connected, then the best solutions will spread quickly to all streams and can quickly saturate the population. On the other hand, if the network is poorly connected, solutions will propagate more slowly and threads will be more isolated from each other. Further parallel development and crossover of different solutions can occur to obtain potentially better solutions.

A common trend in parallel genetic algorithms (PGA) is the use of static system structures that are defined before the method is started and remain unchanged [6, 8].

The frequency of migrations also has a big impact on the final decision. As it is known, too frequent migrations lead to population degeneration, and rare ones, on the contrary, lead to a decrease in convergence. Various methods are used to adjust the migration frequency, which can be divided into two types: adaptive and event-based. In the first case, adaptation methods are used to adjust the migration frequency during the algorithm's operation. In the second case, methods are used that determine the need for migration, that is, migration is carried out only when an event occurs [8].

Selection mechanisms are used to select individuals for Migration. It is known that individual chromosomes can contain important fragments of the genetic code, but these parts can be found in chromosomes that are poorly adapted. But at the same time, excluding such solutions can lead to premature convergence, or skipping the global optimum.

The usage of different strategies imposes the main limitation – the need to form the same type of chromosome structure. However, the effect that is possible with successful formation can be much greater than with the use of a single PGA structure in all sub-populations [6, 7].

It is also worth noting that a large number of migrations and even dynamic exchange of intermediate information between threads requires additional overhead, which sometimes significantly negates the achievement of parallel execution of calculations.

4.3. Swarm intelligence. Strategies and procedures. It is noteworthy that a number of problems are related to vertical connections within a parallel system and operations for sending intermediate data between threads. The most popular evolutionary bioinspired algorithms today are the so-called particle swarm optimization (PSO) [9]. Algorithms of swarm behavior are borrowed from nature, where groups of animals (flocks, swarms) show unexpectedly good results despite the fact that each of the individuals has a rather primitive mind [9, 10]. In the last few decades, various fields of science have made unsuccessful attempts to recreate swarm mechanisms of behavior in artificial models. At the same time, many models strive to reproduce the property inherent in natural self-organizing systems. The system in the process of interaction of its elements as a whole acquires features that were not inherent in the elements separately.

The main idea of swarm methods can be presented as follows. Artificial particles moving in an n-dimensional search space can behave like a flock or swarm [10]. Representatives of the swarm are looking for something like feed it is the extremum of a given objective function. By analogy with genetic calculations, it can be argued that a swarm is similar to a population, and particles (individuals) correspond to chromosomes. The particles move in the search space, changing direction according to their own experience and that of their neighbors. Each particle has a velocity vector and a position vector. When optimizing an N-dimensional function, such vectors have dimension N. As in biological life, it is assumed that each particle adjusts its position and velocity vectors according to its own experience (cognitive component), as well as information received from other swarm members. Here, the cognitive experience of a particle is understood as its knowledge of the best position in which it itself was, and the social knowledge of a particle is understood as knowledge of the best position that one of the particles in the same group with it passed through. In the optimization problem, the best position of a particle is understood as the position at which the value of the function is minimized.

Comparing the methods of GA and PSO, it is possible to note a number of similarities:

– both algorithms are stochastic;
– solution is sought based on the population of individuals;
– initial population is most often randomly generated;
– for the method to work, it is necessary to calculate the fitness function of each individual;
– areas of possible use of methods coincide.

But there are and some disadvantages:

– the number of free parameters specified at the beginning in PSO methods is less than in GA;
– classical evolutionary operators are absent, and the particles themselves do not «die» – they are not removed after work.

5. Methods of research

5.1. Using swarm intelligence principles for parallelize a genetic algorithm. Let's look at what approaches of PSO methods can help improve PGA. In the classical theory of artificial intelligence, one intelligent system is created for a certain problem, which has all the necessary resources to solve it. In the theory of multi-agent systems, the opposite principle is used. It is believed that one agent has an incomplete idea of the global problem, so they create a certain set of agents and provide effective interaction between them [9, 10]. Within the framework of «collective» intelligence, the global behavior of the entire system is considered as the result of interactions of a number of simple agents. Thus, the following principles can be drawn from swarm methods:

– a multi-agent system is a population of simple and dependent agents;
– each agent independently determines its own reactions to events in the local environment and interactions with other agents;
– connections between agents are horizontal, i. e. there is no supervisor agent who manages the interaction of other agents;
– there are no exact rules for determining the global behavior of agents;
– behavior, properties and structure at the collective level are generated only by local interactions of agents.

5.2. Design and construction of a dynamic parallel structure. Thus, it is possible to come to the design and construction of a dynamic parallel system. In this case, the flow is not limited to links to a certain fixed number of streams; instead, migrants are directed to streams that meet certain criteria. As such a criterion, the diversity degree of a population or a measure of genotype is taken, that is, the distance between two populations (or the distance from a characteristic individual of the population, for example, a favorite). With

this structure, mechanisms for tracking events in neighboring populations are necessary, and if an event has occurred in one of the neighboring populations, then events should be expected in the second population as well.

6. Research results

Fig. 1 presents a general scheme of the developed parallel neuroevolution method.

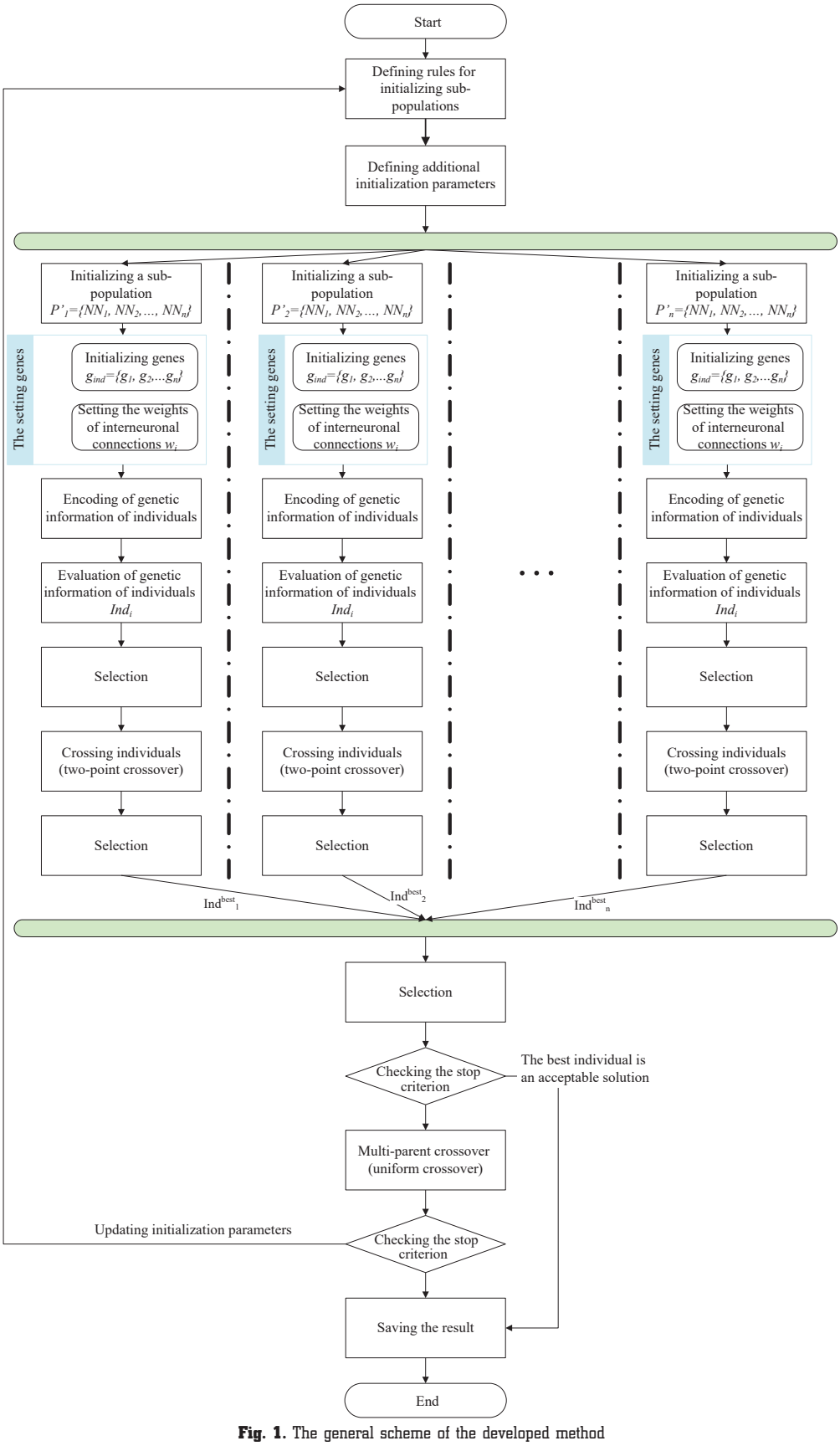

Fig. 1. The general scheme of the developed method

ISSN 2664-9969

The diagram shows that as such, the main flow in the system is not expected. At the beginning, only the main and additional parameters of future sub-populations are determined. These parameters include: the size of each population, topological features, and so on. This information is distributed across parallel system threads. Already through threads sub-populations are independently formed and encoded for operation. The main genetic operators remain, but as long as the process is performed on individual threads, the simplest operators should be used, for example, dot crossover or rank selection. When the best individuals were obtained separately on the threads, they are synchronized. Selection and verification of the best individual from the obtained ones is performed. If such an individual is not yet acceptable, the mechanism of multi-parent crossover is triggered, with the determination of the fate of genetic information that will pass from each of the individuals according to the rank principle. The introduction of rank selection at this stage, with the addition of additional criteria for evaluating not only accuracy, but also topological features, and uniform crossing, is acceptable. After all, a single thread is used, so even if the system uses the GPU for parallelization, this stage will be performed on the CPU, so there will be no braking due to the complexity of operations.

If the best individual from the new generation does not meet the quality criteria, the new generation will not be deleted, but will be used to update the data to form new sub-populations. Thus the designed parallel structure of the method provides:

– simple sub-populations of ANN distributed between threads and independent of each other;
– each agent stream independently determines its own reactions (types of mutations, crosses, etc.) to events in the local environment;
– the main thread is only used to start work first and then synchronize results, and not to control the interaction of other threads;
– determining the best solutions at the general level depends only on local thread scenarios.

For the experimental study, a data set was selected from the work [11] devoted to the research of the pneumonia index to determine the disease among patients.

Table 1 shows the general characteristics of the data sample.

Table 2 compares the performance of ANN synthesis of the developed method with the usual PGA [12] and the parallel implementation of the NeuroEvolution of Augmenting Topologies (NEAT) Method [13].

Table 1

General characteristics of the data set

Total number of values	77490	Number of features	54
Nature of the data	Numeric values (after processing)	Number of instances	1435

Table 2

Comparison of the synthesis process using different methods

Method	Synthesis time, s	Average error during synthesis, %	Average error during testing, %
PSO+PGA	3198.86	0.0000	0.0019
PGA	4017.92	0.1402	0.0847
Parallel NEAT	3269.30	0.0136	0.0214

Graph in Fig. 2 shows CPU usage during iterations.

As can be seen from the graph (Fig. 2) during the execution of a simple PGA, the gap between the loads is significant – this is due to frequent data exchange between cores, which significantly slows down the operation process and overloads the system. Parallel NEAT is more optimized, so the gaps are not so significant, but the load still reaches 98 %. The peak load during the execution of the proposed method did not exceed 85 %, moreover, due to the insignificant share of data transfers, such significant gaps in peak values are not observed.

7. SWOT analysis of research results

Strengths. The proposed approach is based on a special class of evolutionary methods. After the study, strategies were borrowed that could help in the development and design of a new method, significantly improving the quality level of the parallel neuroevolution method of ANN synthesis.

Weaknesses. Certain weaknesses should still include a certain inability to completely abandon genetic operators, which to a certain extent limits the work of the proposed method.

Opportunities. New opportunities include a significantly expanded scope for applying parallel approaches to ANN synthesis. The new method can be implemented for various types of parallel systems without significant losses in the quality of work.

Threats. Certain risks are present when modifying the genetic operators of the method. It is difficult to predict how the quality of work may be affected, for example, by the complication of mutations by introducing a criterion for choosing the type of mutation.

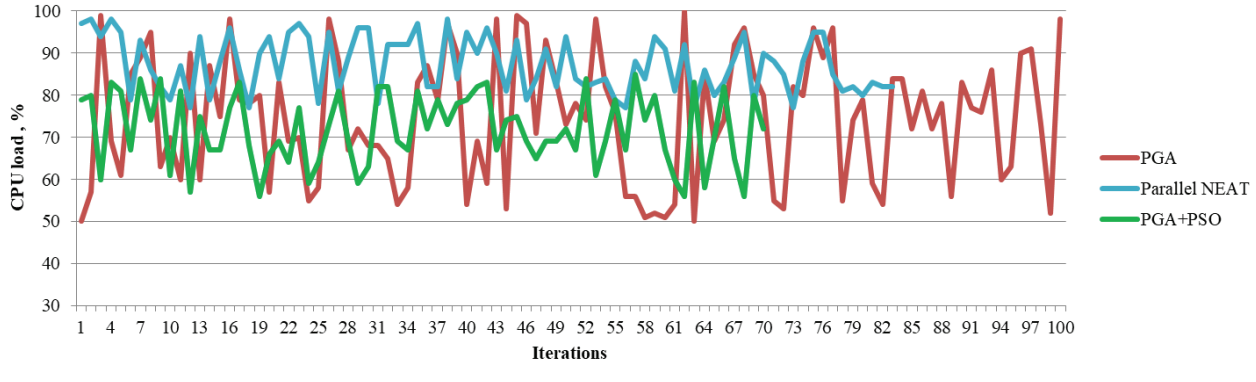

Fig. 2. CPU load during iterations

8. Conclusions

1. Strategies and procedures of swarm intelligence methods are investigated. During the study, certain features were identified that ensure high productivity of swarm methods. Attention was focused on the tools that can be used in the design of parallel approaches to neuroevolution.

2. Using the results of the study, a parallel neuroevolution method for ANN synthesis was developed. The new method made maximum use of strategies borrowed from Swarm methods, namely the structure of the method, which provides:

– simple sub-populations of ANN distributed between streams and independent of each other;

– each agent thread independently determines its own reactions (types of mutations, crossover, etc.) to events in the local environment;

– the main thread is only used to start work first and then synchronize results, and not to control the interaction of other threads;

– determining the best solutions at the general level depends only on local thread scenarios.

3. During the study of experimental results, the developed method was compared with the nearest alternative methods and the results were analyzed. The comparison showed improved characteristics of the developed method in terms of accuracy and resource usage. Further modifications and instructions are based on the introduction of additional criteria mechanisms for adaptive configuration of the method.

Acknowledgements

The work is supported by the state budget scientific research project of Zaporizhzhia Polytechnic National University «Intelligent methods and software for diagnostics and non-destructive quality control of military and civilian applications» (state registration number 0119U100360).

References

1. Albrigtsen, S. I., Imenes, A., Goodwin, M., Jiao, L., Nunavath, V.; Pimenidis, E., Jayne, C. (Eds.) (2018). Neuroevolution of Actively Controlled Virtual Characters – An Experiment for an Eight-Legged Character. *Engineering Applications of Neural Networks. EANN 2018. Communications in Computer and Information Science. Vol. 893*. Cham: Springer, 94–105. doi: http://doi.org/10.1007/978-3-319-98204-5_8
2. Bohrer, J., Grisci, B., Dorn, M. (2020). *Neuroevolution of Neural Network Architectures Using CoDeepNEAT and Keras Computer Science*. Arxiv. Available at: https://arxiv.org/abs/2002.04634
3. Bergel, A. (2020). Neuroevolution. *Agile Artificial Intelligence in Pharo*. Berkeley: Apress, 283–294. doi: http://doi.org/10.1007/978-1-4842-5384-7_14
4. Mason, K., Duggan, J., Howley, E. (2018). Watershed management using neuroevolution. *Modeling Earth Systems and Environment, 4 (4),* 1445–1448. doi: https://doi.org/10.1007/s40808-018-0508-z
5. Arellano, W. R., Silva, P. A., Molina, M. F., Ronquillo, S., Ortega-Zamorano, F.; Rojas, I., Joya, G., Catala, A. (Eds.). (2019). Red-Black Tree Based NeuroEvolution of Augmenting Topologies. *Advances in Computational Intelligence. IWANN 2019. Lecture Notes in Computer Science. Vol. 11507*. Cham: Springer, 678–686. doi: http://doi.org/10.1007/978-3-030-20518-8_56
6. Gonçalves, I., Seca, M., Castelli, M.; Banzhaf, W., Goodman, E., Sheneman, L., Trujillo, L., Worzel, B. (Eds.) (2020). Explorations of the Semantic Learning Machine Neuroevolution Algorithm: Dynamic Training Data Use, Ensemble Construction Methods, and Deep Learning Perspectives. *Genetic Programming Theory and Practice XVII. Genetic and Evolutionary Computation*. Cham: Springer, 39–62. doi: http://doi.org/10.1007/978-3-030-39958-0_3
7. Hoseini Alinodehi, S. P., Moshfe, S., Saber Zaeimian, M., Khoei, A., Hadidi, K. (2016). High-Speed General Purpose Genetic Algorithm Processor. *IEEE Transactions on Cybernetics, 46 (7),* 1551–1565. doi: http://doi.org/10.1109/tcyb.2015.2451595
8. Hou, N., He, F., Zhou, Y., Chen, Y., Yan, X. (2018). A Parallel Genetic Algorithm With Dispersion Correction for HW/SW Partitioning on Multi-Core CPU and Many-Core GPU. *IEEE Access, 6,* 883–898. doi: http://doi.org/10.1109/access.2017.2776295
9. Cleghorn, C. W., Engelbrecht, A. P. (2017). Particle swarm stability: a theoretical extension using the non-stagnate distribution assumption. *Swarm Intelligence, 12 (1),* 1–22. doi: http://doi.org/10.1007/s11721-017-0141-x
10. Nobile, M. S., Cazzaniga, P., Besozzi, D., Colombo, R., Mauri, G., Pasi, G. (2018). Fuzzy Self-Tuning PSO: A settings-free algorithm for global optimization. *Swarm and Evolutionary Computation, 39,* 70–85. doi: http://doi.org/10.1016/j.swevo.2017.09.001
11. Kim, M.-A., Park, J. S., Lee, C. W., Choi, W.-I. (2019). Pneumonia severity index in viral community acquired pneumonia in adults. *PLOS ONE, 14 (3),* e0210102. doi: http://doi.org/10.1371/journal.pone.0210102
12. Nugroho, E. D., Wibowo, M. E., Pulungan, R. (2017). Parallel implementation of genetic algorithm for searching optimal parameters of artificial neural networks. *3rd International Conference on Science and Technology – Computer (ICST)*. Yogyakarta, 136–141. doi: http://doi.org/10.1109/icstc.2017.8011867
13. *Parallel NEAT*. Python Encyclopedia. Available at: https://neat-python.readthedocs.io/en/latest/_modules/parallel.html

Leoshchenko Serhii, *PhD student, Department of Software Tools, Zaporizhzhia Polytechnic National University, Ukraine, e-mail: sergleo.zntu@gmail.com, ORCID: http://orcid.org/0000-0001-5099-5518*

Oliinyk Andrii, *PhD, Associate Professor, Department of Software Tools, Zaporizhzhia Polytechnic National University, Ukraine, e-mail: olejnikaa@gmail.com, ORCID: http://orcid.org/0000-0002-6740-6078*

Subbotin Sergey, *Doctor of Technical Sciences, Professor, Head of Department of Software Tools, Zaporizhzhia Polytechnic National University, Ukraine, e-mail: subbotin@zntu.edu.ua, ORCID: http://orcid.org/0000-0001-5814-8268*

Zaiko Tetiana, *PhD, Associate Professor, Department of Software Tools, Zaporizhzhia Polytechnic National University, Ukraine, e-mail: olejnikaa@gmail.com, ORCID: http://orcid.org/0000-0003-1800-8388*