ATLANTIS PRESS
Advances in Computer Science Research (ACSR), volume 72
IV International Research Conference "Information Technologies in Science, Management, Social Sphere and Medicine" (ITSMSSM 2017)

Benchmarking of neuroevolutionary approach for controlling task of trolley balance

Sergey Rodzin, Lada Rodzina
Southern Federal University
Rostov-on-Don, Russia
srodzin@sfedu.ru

***Abstract*—The article analyzes the neuroevolution of the problematic issues – a promising approach for solving complex problems of machine learning neural networks, adaptive management, and multi-agent systems, evolutionary robotics, search game strategies, computer art. The authors propose a neuroevolutionary algorithm that allows to "grow" a neural network for solving the problems of machine learning with reinforcement. The crossover operator is not used in the algorithm. The evolution of the network is performed due to slight mutational changes in a limited area. The advantages of the algorithm include its independence from the type of neuron activation functions, the absence of a training sample, and the ability to automatically find the appropriate neural network architecture. The authors demonstrate the results of benchmarking on the benchmark task, namely, the task of balancing a trolley with two flagpoles of different lengths. The simulation results support the hypothesis of the advantages of generation of neurostructures by small mutational changes in a limited area.**

Keywords —Neuroevolution, reinforcement machine learning, optimization, evolutionary computation, fitness function

I. Introduction

Neuroevolution is a promising approach for solving complex such problems of machine learning neural networks as adaptive management and multi-agent systems, evolutionary robotics, gaming strategies, computer art, and others. Implementation of supervised learning is almost impossible for these tasks.

Reinforcement learning is an intermediate type between such tasks as learning with the teacher on the precedents (regression, classification) and learning without a teacher, where you need to find patterns in the data (clustering, search for association rules). Reinforcement learning is a type of learning when it takes place in interaction with the environment, in other words, the algorithm makes some actions in the environment and sometimes gets the feedback (like games, negotiations, the scientific research process, etc.). Reinforcement learning is a compromise between research of unexplored areas and application of existing knowledge.

Neuroevolution algorithms are divided into the following three categories depending on the task: search of weights values in the neural network with a fixed structure; setting the neural network structure; setting activation functions of neurons; and various combinations of the above tasks.

The article is devoted to the problems of simultaneous connections and setting the weights of neural network structure.

II. Issues of neuroevolutionary algorithms

There are several fundamental questions related to the development of neuroevolutionary algorithms:

- evolutionary algorithms manipulate a variety of genotypes. In neuroevolution the term “genotype” means a representation (coding) of the neural network. How can we encode efficiently a network structure in the form of a genotype? Currently, each researcher uses its own encryption method. For example, in the direct coding, the genotype is equivalent to the phenotype, and neurons and connections are directly specified in the genotype. In contrast, in implicit coding we either specify the network settings (the number of layers and neurons) or use specialized grammar (grammar evolution, the evolution of the rules of rules usage);
- crossover problem is that the crossover operator can not be applied to the network if its genetic information is different in length. Besides, descendants generated by crossing over often have much worse value of fitness function than their parents;
- how to create a suitable initial population of neural networks? Generation of the initial population of neural networks with random topology is not the best approach, taking into account the cross-over problems. In other words, the effect of the crossover operator on the neuroevolution is destructive.

In the literature, a neural network which is produced as the evolution of network nodes and connections are called *TWEANNs (Topology & Weight Evolving Artificial Neural Networks)* [1]. Using *TWEANNs* was able to solve one of the most difficult test problems of reinforcement learning: the neural network manage the trolley without having information about speed; the trolley is connected via hinges flagpole with two different lengths and in a state of unstable balance [2]. The advantages of the algorithm are the following properties: independence on the type of activation function for neurons; no

need for the training set; the ability to automatically search for the neural network structure. We must point out some shortcomings of the algorithm: the difficulties of assessing the neural network structure; it requires more memory than using gradient algorithms; the complexity of the organization structure of neural network research.

We should note the following research groups in the field of development neuroevolutionary algorithms: a group of neural network research (head R. Miikkulainen, http://www.cs.utexas.edu/~risto/), the group of evolutionary complexity research (head K. Stanley, http://www.cs.ucf.edu/?kstanley), Intelligent Systems Laboratory (head D. Floreano), Laboratory of autonomous robots and artificial life (head S. Nolfi), Center of Computational Intelligence Research and Applications (head *X. Yao*, http://www.cs.bham.ac.uk/~xin), the group of researchers of adaptive systems optimization (head *Ch. Igel*), Swiss Institute of Artificial Intelligence (head *J. Schmidhuber*, http://www.idsia.ch/~juergen/). In Russia, neuroevolutionary algorithms research are conducted in the following research centers: Optoneuron Technology Center together with Keldysh Institute of Applied Mathematics (adaptive behavior), Siberian Federal University (classification and medical diagnostics), Southern Federal University (evolutionary neurocomputing), in Ulyanovsk State Technical University (Intelligent Decision Support System), Tomsk Polytechnic University (classification, image processing) [3-7]. The paper proposes a new algorithm for neuroevolution *TWEANNs* where the crossover operator is not used at all, and as the main operator, we use the mutation.

III. Evolutionary Synthesis Algorithm for TWEANNs Topology

Neurodevelopmental algorithm starts with a neural network without hidden neurons and goes in the direction of topology complication. Coding of the neural network should allow performing meaningful crossing over different topologies. The following two lists represent the genotype: a list of adjacent vertices (neurons), and a list of synaptic weights. The top is encoded by the following two components: the first one identifies it to the generations of the population, while the second one indicates the type of layer (input, hidden, output). List of synaptic weights includes pointers to adjacent vertices, the value of the weight on the flag (if the compound is used) and the "historical label" which serves to identify the connection for all generations in a population. This allows the operator to apply the crossover without duplication or deletion of genetic information from the parents.

Nevertheless, new offspring generated by crossover operator usually has a fitness function worse than the parental chromosomes. In [1] proposed to apply the crossover to individuals with small structural differences. However, this quickly leads to a topological monotony of obtained neural networks.

The authors propose an approach where the crossover is not used. In this approach, the population is regarded as the central object of neuroevolutional algorithm. The authors proceed from the assumption that evolution is primarily a process of adaptation at the behavioral level. This level of abstraction does not provide recombination. Therefore, there is no crossover operator. A mutation in the proposed neuroevolutional algorithm is the only alternative solutions operator.

Let us consider the encoding of neural networks as a way to develop a reliable method for generating neurostructures with a few changes and only in a limited area. We use the biological concept of the operon. Operon is a group of functionally linked genes whose activity is ordered, and depends on external conditions and on the activities of other genes. Based on this concept, as an operon, we understand a certain subnetwork consisting of a subset of neurons (nodes) and a subset of connections between neurons.

Then genotype, as a string encoding consisting of characters, represented as a subset of the operons that resembles the following:

$$string = \{operon_1, operon_2, ..., operon_N\} \quad (1)$$

$$operon_i = \{\{node_j | \, j \in [1: J]\}, \{edge_k | \, k \in [1: K]\}\} \quad (2)$$

where N is a maximum number of operons($i \in [1: N]$); $node_j$ is the identification number of the j-th neuron included in the set $operon_i$; J is the number of neurons included in $operon_i$ with their own identification numbers; $edge_k$ is the k-th connection in $operon_i$; K is the number of connections in $operon_i$.

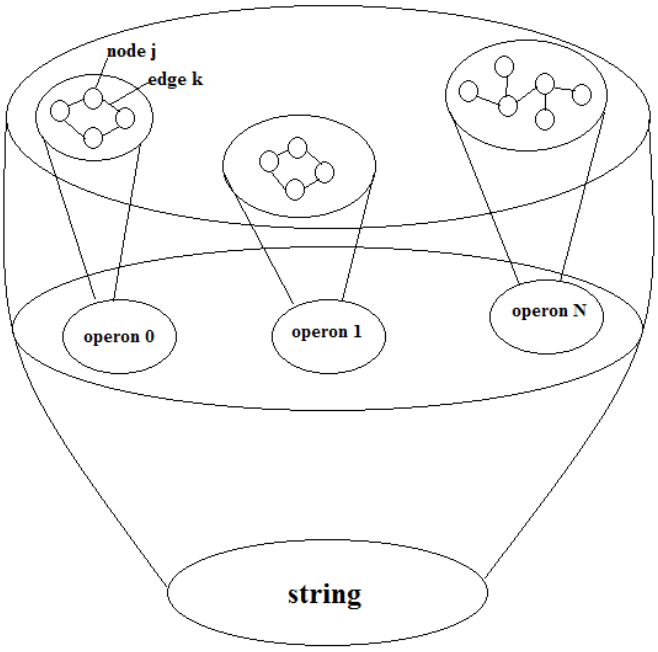

Fig. 1. Coding of neural network: string-operon-(node+edge).

The initial population consists of individuals having only one $operon_0$, which contains only the input and output nodes and connections between them. Here nodes are absent in the hidden layer.

Let us consider two kinds of mutation operator, which are used in the proposed neuroevolution algorithm for adding the nodes and connections in the neural network.

Mutation operator for adding nodes is applied to each operon with constant probability p_m. One of the synaptic

connections is randomly selected for removing. Next, the neuron of the hidden layer and synaptic connections associated with it are added instead of removed synaptic connections. If one of the ends of the removed synaptic connection is connected to the node of $operon_0$, then, the added neuron of the hidden layer and its connections will become a new operon. For example, we have $operon_0$ with two inputs x_1, x_2 with weights w_1, w_2 accordingly, as well as with output neuron y_1 that has a sigmoid as an activation function.

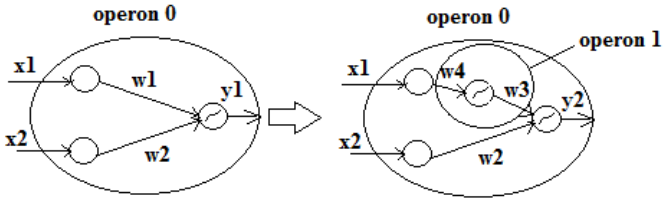

Fig. 2. Mutation for adding nodes.

After removal of the randomly selected synaptic connection with weight w_1 we add the neuron of the hidden layer and two connections with weights w_3 and w_4 accordingly, where w_4 is the weight value of the input x_1, w_3 is the weight of the connection between the input of the new neuron of the hidden layer and the output neuron y_2 operon0. Added neuron and its connections become a new $operon_1$.

To change the output $operon_0$ after the mutation, we must be sure that the values of y_1 and $y_2$are equal. Therefore, the following condition on the assumption that S is a sigmoid function should be carried out:

$$S[w_1x_1 + w_2x_2] = S[w_3S[w_4x_1] + w_2x_2] \quad (3)$$

Hence we find that:

$$w_1x_1 - w_3S[w_4x_1] = 0 \quad (4)$$

Furthermore, under the condition that $w_1 = w_3$ and $S[x] = 1/(1 + exp^{\beta(\alpha - x)})$, we obtain the value of the fitness function $f(x_1)$ after the mutation operator:

$$f(x_1) = x_1 - 1/(1 + exp^{\beta(\alpha - w4x1)}), \quad (5)$$

where in order to simplify the results of experiments the following values were established: $w_4 = 1$, $\alpha = 0{,}5$, $\beta = 5$.

Mutation operator for adding connections is also applied to each operon with a constant probability

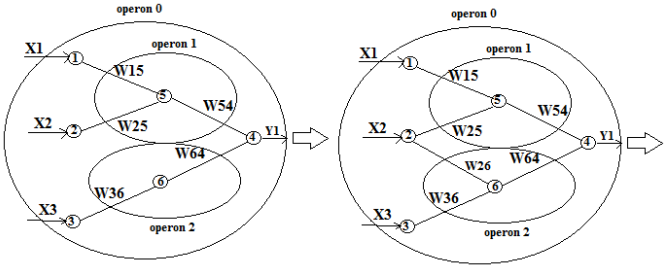

Fig. 3. Mutation for adding connections.

Some node is randomly selected and added to the connection to the node in $operon_i$ or $operon_0$. Weight of the new connection is set to 0, so it does not alter the transmitted signal.

IV. THE TASK OF BALANCING TROLLEY WITH TWO FLAGPOLES

The authors have evaluated the effectiveness of the proposed neuroevolutional algorithm on such standard task as balancing trolley with two flagpoles [8]. Two flagpoles are connected to the moving trolley by hinges (Fig. 4).

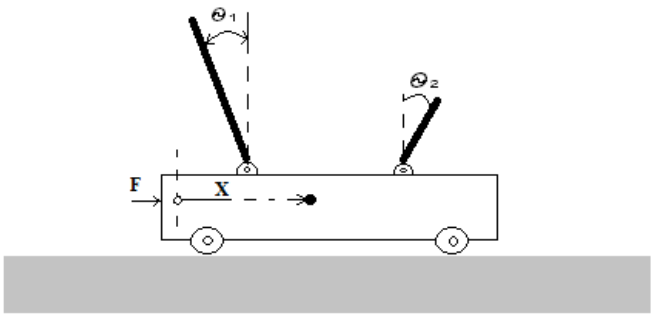

Fig. 4. Trolley with two flagpoles.

A neural network has to manage the movement of the trolley if we want to balance both the flagpole as long as possible. System status is determined by the position x and the speed v_x of the trolley, as well as by the angle γ_1 of deviation from the vertical of the first flagpole and its angular velocity ω_1, and by the angle γ_2 of deviation from the vertical of the second flagpole and its angular velocity ω_2. However, in order to simplify the information about v_x, ω_1 and ω_2 are not used as input data.

Flagpoles have different lengths. Computer modeling was performed with four different ratios of the low and high length flagpole 1:10, 1:5, 1:3 and 1: 2,5. It is well known that the task of balancing flagpoles becomes more difficult when the ratio of the lengths of flagpoles is coming to 1. All other details were the same as in [1] for the computer simulation. The authors used for comparison the data given in [2] for neuroevolutional algorithm *NEAT* (*Neuro-Evolution of Augmenting Topologies*). The reason for choosing the *NEAT* algorithm for comparison is the following: it solves the problem of balancing the flagpole on the cart is 25 times better than the algorithm using cellular encoding [9, 10].

The parameters of the proposed neuroevolutional algorithm and the *NEAT* in the experiments had the following values: the size of the population is 1000; the size of the tournament selection is 20; the weight of connections by using a mutation operator for adding nodes is 1; sigmoidal function parameters $\alpha = 0{,}5$, $\beta = 5$.

The authors have conducted a series of 10 experiments and obtained the following results.

The problem with a probability close to 1 is successfully solved by compared algorithms with the ratio of the lengths between the low and high flagpoles as 1:10 and 1:5. However, further alignment of flagpoles lengths we see as the result that the *NEAT* algorithm became worse. In particular, the most difficult conditions is when the ratio of the lengths of flagpoles 1:2,5. In this case, the balance of the control task is successfully solved for *NEAT* algorithm only once in ten experiments. The proposed algorithm, the likelihood of a successful solution was 0,8.

However, the *NEAT* algorithm with the ratio of the lengths between the low and high flagpoles like 1:10 and 1:5 have found the solution for about 40 generations. This result was significantly better than for the proposed neuroevolutional algorithm. When the length ratio between the low and high flagpoles as 1:3 and 1:2,5, *NEAT* resolve it within approximately 200 generations, the proposed algorithm - during approximately 350-400 generations.

The most successful case are as follows: the ratio of the lengths of the low and high flagpole 1:10. In this case, the algorithm *NEAT* added 6 units and 11 compounds. With the same ratio between the lengths the proposed algorithm gives a comparable result. However, when the ratio between lengths as 1:2,5, the network complexity obtained by the *NEAT* algorithm grows rapidly and reaches a size of more than 50 nodes and over 300 connections. At the same time, the complexity of the neural network topology generated by the proposed algorithm for all four ratios between the lengths of the low and high flagpoles remains virtually unchanged: it is about 10 nodes and 20 connections. Thus, simulation results confirm the hypothesis about the benefits of generating neurostructures by small mutational changes in a limited area.

V. Conclusion

The authors believe that the novel result of the study is the neuroevolutional algorithm that allows you to "grow" the neural network for solving tasks of machine learning with reinforcements. The key feature of the proposed algorithm is a unique opportunity to change the architecture of the neural network by simultaneously adding nodes as well as compounds. The proposed algorithm does not use the crossover operator. Evolution of the network occurs as small mutational changes in a limited area. The advantages of the algorithm are the follows: independence on the type of activation of neuronal functions; no need for the training set; the ability to automatically find a suitable neural network architecture.

The researchers have conducted experiments on the difficult task: the trolley management for balancing of two flagpoles with different lengths fortified on it. The results of the developed algorithm had been compared with the similar algorithm – the NEAT. The developed algorithm is superior to analogue on the following parameters: in the most difficult conditions in the ratio of the lengths of flagpoles 1:2,5 balance control problem is successfully solved by the NEAT algorithm with probability 0,1 and the proposed algorithm with a probability 0.8. Similarly, the complexity of the networks produced by the algorithm HEAT is growing rapidly bit in the proposed algorithm it is almost the same. In addition, the algorithm does not use as an input the trolley speed information and the angular velocity of the flagpoles. The authors consider that the algorithm holds promise for use in evolutionary robotics, searching for game strategies, computer art, and other areas where you want to perform actions in the environment, only occasionally getting feedback.

Acknowledgment

The study was performed as part of the Government assignment of Ministry of Education and Science of Russia (Project № 8.823.2014) and partly supported in by the grant from the Russian Science Foundation (project # 16-07-00335) in the Southern Federal University

References

[1] Stanley K.O., Miikkulainen, R. Evolving Neural Networks Through Augmenting Topologies // Evolutionary Computation. 2002. № 10(2). pp. 99–127.

[2] Stanley K.O.: http://www.cs.ucf.edu/~kstanley/

[3] Kureychik V.V., Kureychik V.M., Rodzin S.I. The theory of evolutionary computation. Moscow: Fizmatlit, 2012. 260 pages.

[4] Tsoy YU.R., Spitsyn V.G. The evolutionary approach to the setting up and training of artificial neural networks // Neuroinformatics. 2006. Vol 1. № 1. pp. 34-61.

[5] L.A. Zinchenko, V.M. Kureychika, V.G. Red'ko. Bionic information systems and their practical application. Moscow: Fizmatlit, 2011. 288 pages.

[6] Rodzin S., Rodzina O.N. Proc. of the 6th IEEE Int. Conf. on Cloud System and Big Data Engineering (Confluence'2016, India). AI and Soft Computing. 2016. pp. 1-5.

[7] Rodzin S., Rodzina L. Theory of bionic optimization and its application to evolutionary synthesis of digital devices // Proc. of the 14th IEEE east-west design & test symposium (EWDTS'14), 2014. pp. 147–152.

[8] Gomez F., Miikkulainen R. Solving non-Markovian control tasks with neuroevolution // Proc. 7th Int. Joint Conf. on Artificial Intelligence, Morgan Kaufmann, San Francisco, 1999. pp. 1356–1361.

[9] Gruau F. Genetic synthesis of Boolean neural networks with a cell rewriting development al process // Proc. of the IEEE Workshop on Combinations of Genetic Algorithms and Neural Networks, 1992. pp. 55-74.

10. Moriguchi H., Honiden S. CMA-TWEANN: Efficient Optimization of Neural Networks via Self-Adaptation and Seamless Augmentation // Proc. of the 14th Annual Conf. on Genetic and Evolutionary Computation, 2012. pp. 903-910.