UDC 004.896
DOI: 10.20535/SRIT.2308-8893.2026.2.11

TRAINING ARTIFICIAL NEURAL NETWORKS FOR THE IMPLEMENTATION OF THE PETRI NETS SYNTHESIS PROCESS

A.A. GURSKIY, A.V. DENISENKO, A.E. GONCHARENKO

Abstract. The important task has been solved in this scientific research related to specific development and verification of the fundamental suitability of the method for automatic synthesis of Petri nets based on the functioning of an artificial neural network. This allows to automate the generating process of logical control algorithms. It has been proposed to divide the artificial neural network into separate local networks for the subsequent stepwise training. Such stepwise training is necessary to form the logical control algorithm and to synthesize the appropriate Petri net. The proposed principles can be applied in the early stages of training for an artificial neural network while synthesizing a Petri net. It has been conducted the experiments linked to the training of local neural networks and to implement the formation of an algorithm for logical control of the laboratory machine for processing chemical solutions. As a result of the experiment, it has been determined the fundamental suitability for the stepwise training method of neural networks for generating Petri nets and the automated formation of logical control algorithms.

Keywords: artificial neural networks, logical control algorithms, training method, hybrid system, automatic synthesis of Petri nets.

INTRODUCTION

The synthesis of Petri nets and the development of logical control algorithms for systems with a logical and dynamic (hybrid) nature of operation are provided using of appropriate software tools. The evolution of such software is primarily tied to up-to-date trends in scientific and technological development and in the development of information and intellectual technologies. The use of artificial neural networks and the creation of intelligent systems is permitted to eliminate partially or completely the human participation in some cases, e.g., in a particular case, to eliminate partially or completely the human participation in the synthesis of Petri nets based on the appropriate software tools.

PROBLEM STATEMENT

The use of artificial neural networks for the automatic generation of Petri nets been considered in the scientific article [1], in which has been presented the certain training methods for artificial neural networks. The disadvantages of these methods are that they can be described as an imitation of learning processes through trial and error. It is quite difficult to synthesize Petri nets or to form logical control algorithms using such methods, also applying the well-known reinforcement learning principles. These methods may not be effective in the initial stages of training

[2, 3], but they are acceptable when the Petri net has already been synthesized and needs to be corrected.

Therefore, it needs to research the principles of training for an artificial neural network using known methods for the purpose of implementing automatic synthesis of Petri nets. It is necessary to consider such methods that can be employed during the early stages of training. For instance, in a particular case it is able to be the well-known backpropagation algorithm.

The purpose of the scientific work is to automate process in synthesis of the control algorithms by implementing automatic synthesis of Petri networks during the operation of an artificial neural network.

To achieve this purpose, it needs to determine principles for the application of known methods of training artificial neural networks that can be used to implement automatic synthesis of Petri nets. In this case, after the training process the artificial neural network should represent a sequence of transitions $\sigma = t_1, t_2 ... t_n$ ensures:

$$M_0 \xrightarrow{t_1} M_h \xrightarrow{t_i} ... \xrightarrow{t_n} M_J .$$

In such a case, the marking M_J of the formed Petri net is directly achievable from the marking M_0 in the same way as for a Petri net synthesized without the use of automatic generation methods based on the artificial neural networks.

REVIEW OF THE LITERATURE

One of the first mentioned ideas about the automatic synthesis of Petri nets was in the work of J. Peterson [4]. It was noted in this work that "the use of Petri net languages would be in the specification and automatic synthesis of Petri nets. If the behavior which is desired can be specified as a language, than it may be possible to automatically synthesize a Petri net whose language is the specified language. This Petri net can be used as a controller, guaranteeing that all and only the sequences specified are possible".

Since that time, it has been appeared the small number of scientific papers tied to such a concept as the automatic synthesis of Petri nets. The development of the relevant field of synthesis can be traced based on these scientific works [5–8]. Thus, it has been presented the principles of synthesis for Petri nets given the composition rules of individual subnets. However, further development for certain constructing principles of Petri nets have been shown that their formation is based on an artificial neural network, representing an intelligent technology for the formation of certain algorithms. The basic principles of training an artificial neural network for the constructing of Petri nets are explained in the work [1, 9].

This scientific work is a continuation of the development of this principle for synthesis of Petri nets and the appropriate principles of training artificial neural networks using known methods.

MAIN PART

The software application in the form of a virtual stand for the automatic synthesis of Petri nets based on the Labview environment is being developed in this work.

The formed Petri nets are able to represent algorithms for logical control of complex technological systems. Automatic synthesis of Petri nets is implemented on the basis of a state graph or a reachability tree [10]. The front panel of the developed virtual stand, which visualizes the synthesis of Petri nets, is shown in Fig. 1. The front panel shows a hybrid system with a logical and dynamic nature of operation. Such a system consists of two parts, such as a continuous-event part and a discrete-event part. The various modes of operation have been formed from the interaction of these two parts. A model of such a system can be presented on the basis of DC-nets [11, 12].

The continuous-event part in continuous space $X(t_{,}|t_k|)$ can be described by equations of state

$$\dot{X}(t_{,}|t_k|) = A_o \cdot \Xi_1({}^d u_o(t_k)) \cdot X(t_{,}|t_k|) + B_o \cdot \Xi_2({}^d u_o(t_k)) \cdot u(t_{,}|t_k|) \qquad (1)$$

and exit

$$Y(t_{,}|t_k|) = C_o \cdot \Xi_3({}^d u_o(t_k)) \cdot X(t_{,}|t_k|) \qquad (2)$$

in matrix-differential form, where $X(t_{,}|t_k|)$ is continuous-event state vector of the continuous part; $Y(t_{,}|t_k|)$ is continuous-event output vector; $u(t_{,}|t_k|)$ is continuous action vector; $\Xi({}^d u_o(t_k))$ is vector function for control the operating modes of a hybrid system [12].

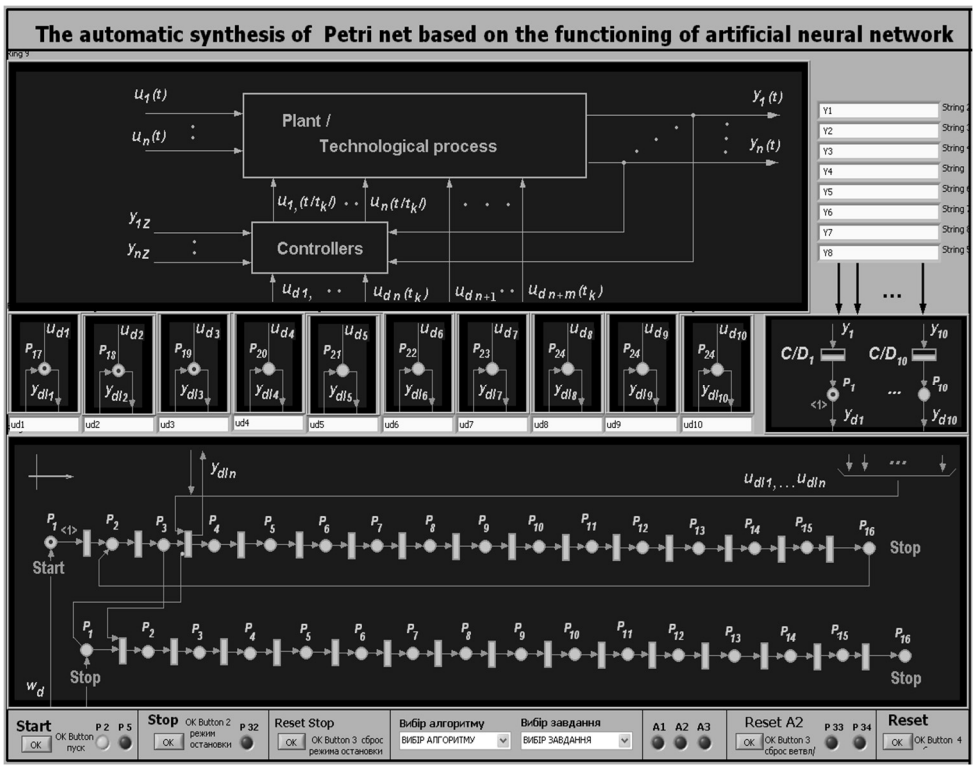

Fig. 1. Front panel of the virtual stand demonstrating the automatic formation of a Petri nets. These nets reflect the logical control algorithms

The discrete-event part (or the logical control device) can also be described by a state equation

$$^{d}X_{L}(t_k) = {}^{d}X_{L}(t_{k-1}) + \left|A_L\right| \cdot {}^{d}v_{L}(t_k) + {}^{d}u_{L}(t_k) + {}^{d}w_{L}(t_k) \qquad (3)$$

and the output equation of the logical part

$$^{d}Y_{L}(t_k) = \Lambda \cdot {}^{d}X_{L}(t_k) \qquad (4)$$

where, $^{d}w_{L}(t_k)$ is external control action; $^{d}X_{L}(t_k)$, $^{d}X_{L}(t_{k-1})$ are discrete states at step t_k, t_{k-1} appropriate to the markings of the Petri net, which represents the logical part; $\left|A_L\right|$ is incident matrix representing the relationship of the elements of the discrete-event part; $^{d}v_{L}(t_k)$ is control vector; Λ is transition matrix.

Thus, the functioning of the logical part of the system can be represented using an automatically generated Petri net.

In the software, application it has been carried out the implementation for automatic synthesis of Petri nets on the basis of the functioning of an artificial neural network. As illustrated in Fig. 2 in this case the artificial neural network interacts with synchronously functioning Petri nets. This interaction is carried out on the principles of feedback. From these synchronously functioning networks it is possible to form a Petri net composition showing the required logical control algorithm. It is shown that each position of the generated Petri net matches a specific position of the Petri net *PNi* connected to the neural network *NN*. In order to compose a Petri net we need to combine transitions fired simultaneously in networks *PN1…PNm*. In this case, according to the diagram shown in Fig. 2, the neural network will be able to form signals appropriate to the columns of the incidence matrix $\left|A_L\right|$ in the equation of state. It has been determined that the coefficients of interneuronal connections in the output layer of the neural network correspond to the incidence matrix of the formed Petri net. The input or hidden layer of a neural network is linked to the conditions for firing of the Petri net transitions.

It has been determined that training of an artificial neural network for the implementation the automatic synthesis of Petri nets must be realized by stages using the known training methods. One of the known methods may be the backpropagation algorithm. In this case, the artificial neural network without an output layer must be divided into separate parts or sectors.

Each such part (sector) may constitute a local neural network performing a specific functional load, tied to conditions for firing of the certain transition t_i where i=1…n. Thus, the number of such parts (sectors) must correspond to the number of transitions of the formed Petri net. In Fig. 3, these parts are designated by input neural t_1, t_2, t_3 of the 1st input layer of the artificial neural network.

The coefficients of interneuron connections of the 1st (input) part of the artificial neural network are determined in the learning process based on the training sample. In this case, the learning process should occur in examples of the data conditions for transition from one state of the system to another or the conditions for firing transitions of the Petri net.

The matrix of coefficients for interneuronal connections of the output layer for the artificial neural network (the 2nd output part) has been determined quite simply by subtraction: $\overline{m}_{k+1} - \overline{m}_k = \overline{W}_j$, where $\overline{m}_{k+1}$ is the marking of Petri net at $k+1$ step; $\overline{m}_k$ is the marking of Petri net at k step; $\overline{W}_j$ is a vector of coefficients for interneuron connections of neurons for the network output layer which corresponds to the j-th column of the incidence matrix of the Petri net.

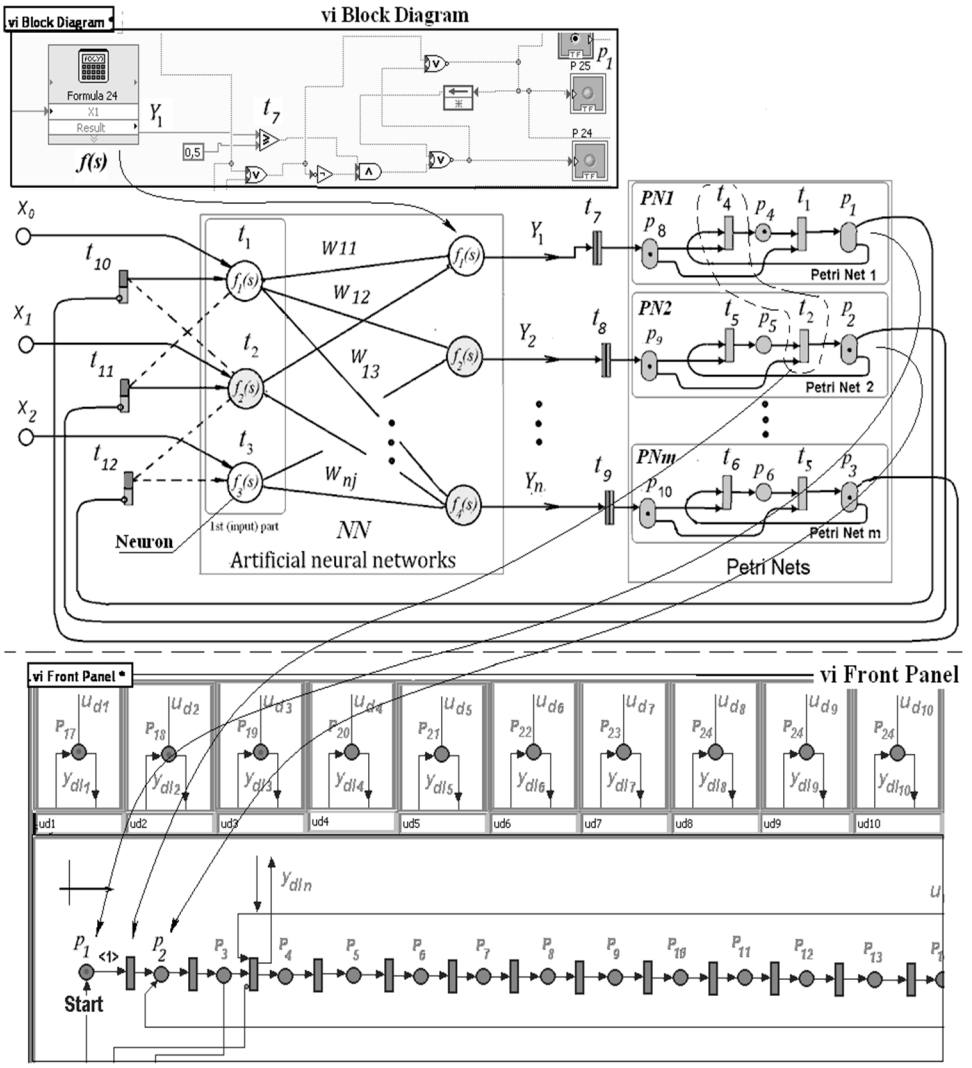

Fig. 2. Diagram which shows the interaction of the artificial neural network and Petri nets during the formation of a logical control algorithm based on a virtual stand

Thus, training or determination of the coefficients of interneuronal connections in an artificial neural network can be implemented on the basis of a state graph or a reachability tree of a Petri net.

It should be conducted the appropriate experiments to determine the fundamental suitability of the presented principle for automatic synthesis of Petri nets. These experiments are related to the processes of training artificial neural networks.

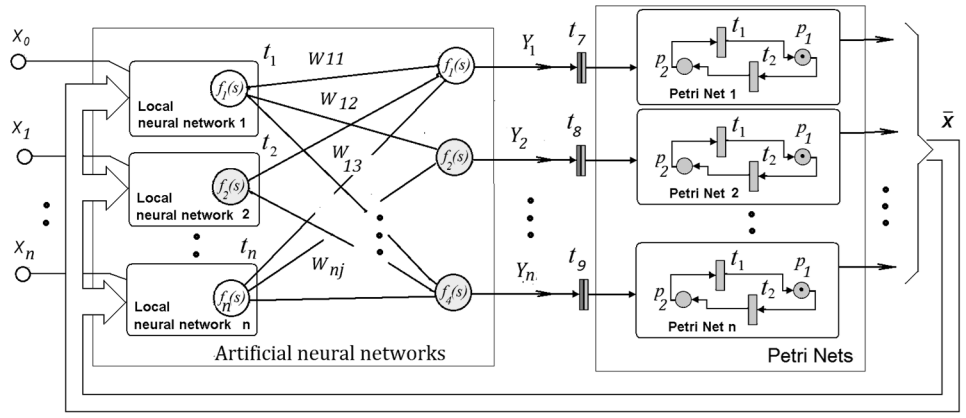

Fig. 3. Diagram representing the interaction of the local artificial neural networks and Petri nets in the formation of a logical control algorithm

EXPERIMENTS

The necessary experiments have been performed in the MATLAB\Simulink 5.2 software environment. The MATLAB software environment possesses the necessary tools for formation and training of the artificial neural networks. These tools can be used to determine the fundamental suitability of a certain method of Petri net synthesis based on the functioning of artificial neural networks. The laboratory machine for processing chemical solutions has been considered (Fig. 4). The modeling of a chemical solution processing machine in the DC-Net environment as a hybrid system has been presented in the paper [11]. The fundamental suitability of the presented principle of automatic synthesis of Petri nets will be established by examining only a fragment of the logical part in this paper.

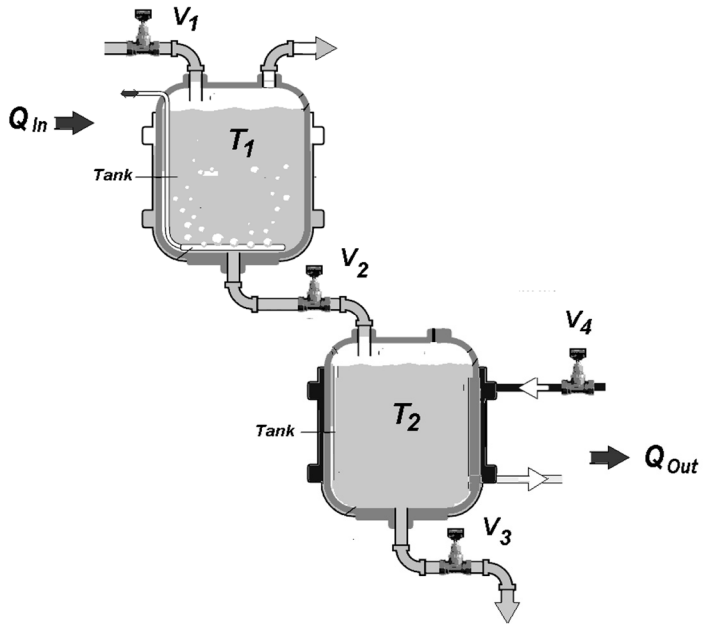

Fig. 4. Technological scheme of the plant for processing chemical solutions

Thus, the logical control algorithm of such a laboratory machine has been presented in the form of a Petri net implemented in the DC-Net software environment (Fig. 5). Each marker in the P_s position of the Petri net defines the state of the system. Accordingly, in the Petri net shown in Fig. 5, if position P_1 is marked $\mu(p_1)=1$, then solution processing mode in tank (container) T_1 takes place; If $\mu(p_2)=1$, then the solution overflows from T_1 to T_2, while valve V_2 is open and $\mu(p_{10})=0$ respectively. When $\mu(p_3)=1$, the post-processing mode is in progress, in this case the valve V_2 is closed, $\mu(p_{10})=1$ respectively. Next, when $\mu(p_6)=0$ the filling mode of the tank T_1 begins, while the valve V_1 is open. When $\mu(p_6)=1$, the transition to stop process, $\mu(p_{10})=0$ accordingly, valve V_3 is open and the solution comes out of the tank T_2. The reachability tree shown in Fig. 6 can be used to trace this marking dynamics.

According to marking dynamics it is able to imagine a training sample for stepwise sequential training of an artificial neural network. It is necessary to show the conditions or states where a certain transition or is fired and those conditions or states where it is not fired.

Let us consider transition t_2. The transition p firing condition is as follows: $\forall p_i \in I(t_2): \mu(p_i)=1 \cup m_1 < g$, where $\mu(p_i)$ is marking of input positions of transition t_2, respectively $\mu(p_2)=1$; g is maximum limit value of the mass of the solution in the tank T_2.

The following executable code must be written in the command window of MATLAB to generate a local neural network:

```
» P=[0  0   1   1; 50  60  50  60];
» T=[0 0 0 1];
» net = newff([0 1; 0 70], [2 1],{'tansig','logsig'});
» net.trainParam.epochs=3000;
» net=train(net, P, T);
TRAINLM, Epoch 0/3000, MSE 0.145251/0, Gradient 14.3558/1e-010
TRAINLM, Epoch 18/3000, MSE 2.63998e-013/0, Gradient 5.00017e-011/1e-010
TRAINLM, Minimum gradient reached, performance goal was not met.
» a = sim(net,P)
a =
   0.0000    0.0000    0.0000    1.0000
» gensim(net)
```

In the present executive code that has been presented, P is the input values of the neural network; t is the corresponding output values of the neural network. This P is the value of the position marking p_2 and the value of the mass for the solution in the tank T_2.

If the output value t=1 then the transition has fired. In the third line of the executive code the number of neurons, the number of layers and activation functions has been set. The last command that is entering to form a neural network in the Simulink environment is the *gensim(net)* command. As a result of executing this command, a local neural network appears, indicated in Fig. 7 as Neural Network 2.

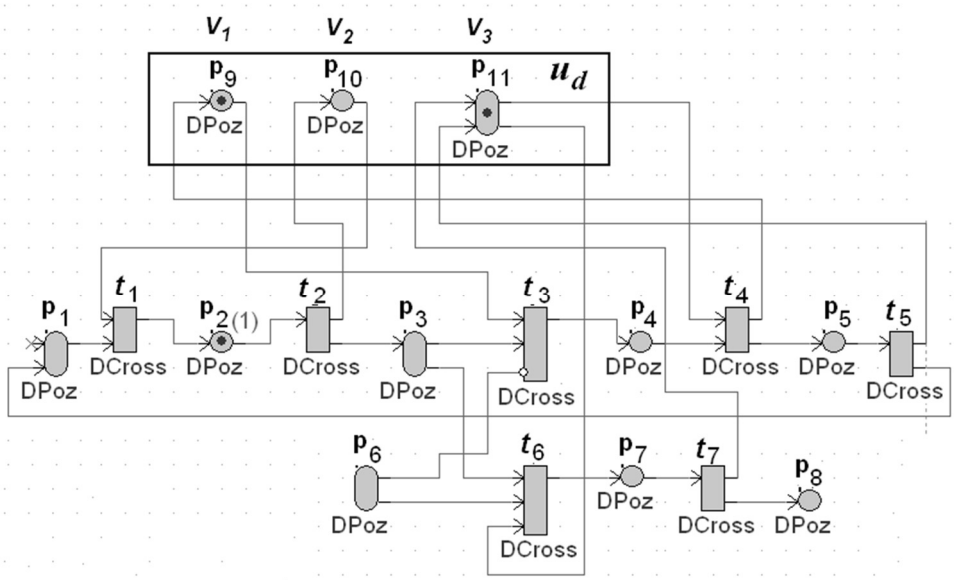

Fig. 5. Fragment of a Petri net presented in the modeling environment of DC-Net hybrid (logical-dynamic) systems

Similarly, we have the ability to write executable code to generate a neural network that is relative to transition t_3.

» P=[0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1; 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1; 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1; 303 303 303 303 303 303 303 303 313 313 313 313 270 270 270 270];

» T=[0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0];

» net = newff([0 1; 0 1; 0 1; 270 400], [4 1],{'tansig','logsig'});

» net.trainParam.epochs=3000;

» net=train(net, P, T);

TRAINLM, Epoch 0/3000, MSE 0.169168/0, Gradient 472.338/1e-010

TRAINLM, Epoch 25/3000, MSE 8.47449e-008/0, Gradient 0.00443561/1e-010

TRAINLM, Epoch 34/3000, MSE 9.84151e-016/0, Gradient 6.63958e-011/1e-010

TRAINLM, Minimum gradient reached, performance goal was not met.

» a = sim(net,P)

a =

Columns 1 through 7

0.0000 0.0000 0.0000 1.0000 0.0000 0.0000 0.0000

Columns 8 through 14

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

Columns 15 through 16

0.0000 0.0000

»

» gensim(net)

According to the set of input values P and output ones t, transition t_3 is fired at $\mu(p_3)=1$, $\mu(p_9)=1$, and at the solution temperature in tank T_2 below 303^0K. In order to eliminate erroneous reactions from the artificial neural network, various input values P are presented to prevent the transition from being fired.

For transition t_6 the execution code is as follows:

» P=[0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 ; 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1; 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1; 303 303 303 303 303 303 303 303 313 313 313 313 313 313 313 313];

» T=[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0];

» net = newff([0 1; 0 1; 0 1; 260 400], [4 1],{'tansig','logsig'});

» net.trainParam.epochs=3000;

» net=train(net, P, T);

TRAINLM, Epoch 0/3000, MSE 0.498623/0, Gradient 231.701/1e-010

TRAINLM, Epoch 25/3000, MSE 0.0449965/0, Gradient 17.2122/1e-010

TRAINLM, Epoch 47/3000, MSE 6.36122e-015/0, Gradient 6.95981e-011/1e-010

TRAINLM, Minimum gradient reached, performance goal was not met.

» a = sim(net,P)

a =

Columns 1 through 7

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

Columns 8 through 14

1.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

Columns 15 through 16

0.0000 0.0000

» gensim(net)

As a result of executing the commands, a local neural network appears, designated in Fig. 7 as Neural Network 6.

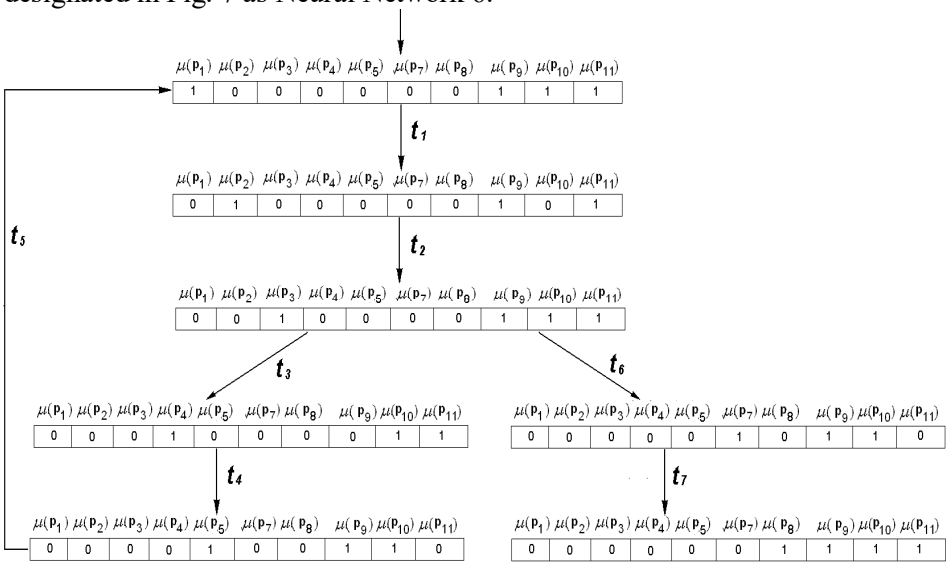

Fig. 6. Fragment of a reachability tree of a Petri net representing the logic control algorithm of the laboratory machine for processing chemical solutions

Next, it can be determined as a result of the subtraction $\bar{m}_{k+1} - \bar{m}_k = \bar{W}_j$ the coefficients of interneuronal connections for the output layer of the artificial neural network as follows:

$$\bar{W}_2 = \bar{m}_{k+1} - \bar{m}_k = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}^T -$$

$$-\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}^T = \begin{bmatrix} 0 & w_{2,2} & w_{3,2} & 0 & 0 & 0 & 0 & 0 & w_{10,2} & 0 \end{bmatrix}^T;$$

$$\bar{W}_3 = \bar{m}_{k+1} - \bar{m}_k = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}^T -$$

$$-\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & w_{3,3} & w_{4,3} & 0 & 0 & 0 & w_{9,3} & 0 & 0 \end{bmatrix}^T;$$

$$\bar{W}_6 = \bar{m}_{k+1} - \bar{m}_k = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}^T -$$

$$-\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & w_{3,6} & 0 & 0 & w_{7,6} & 0 & 0 & 0 & w_{11,6} \end{bmatrix}^T.$$

The appropriate artificial neural network has been implemented in the MATLAB\Simulink software environment. This artificial neural network interacts with Petri nets on the principles of feedback (Fig. 7). Such interaction is also shown in Fig. 3. As a result of such interaction at the outputs of the neural network $y_2 \ldots . y_9$ it is formed the sequence of signals appropriate to the rows of the incidence matrix W of the Petri net.

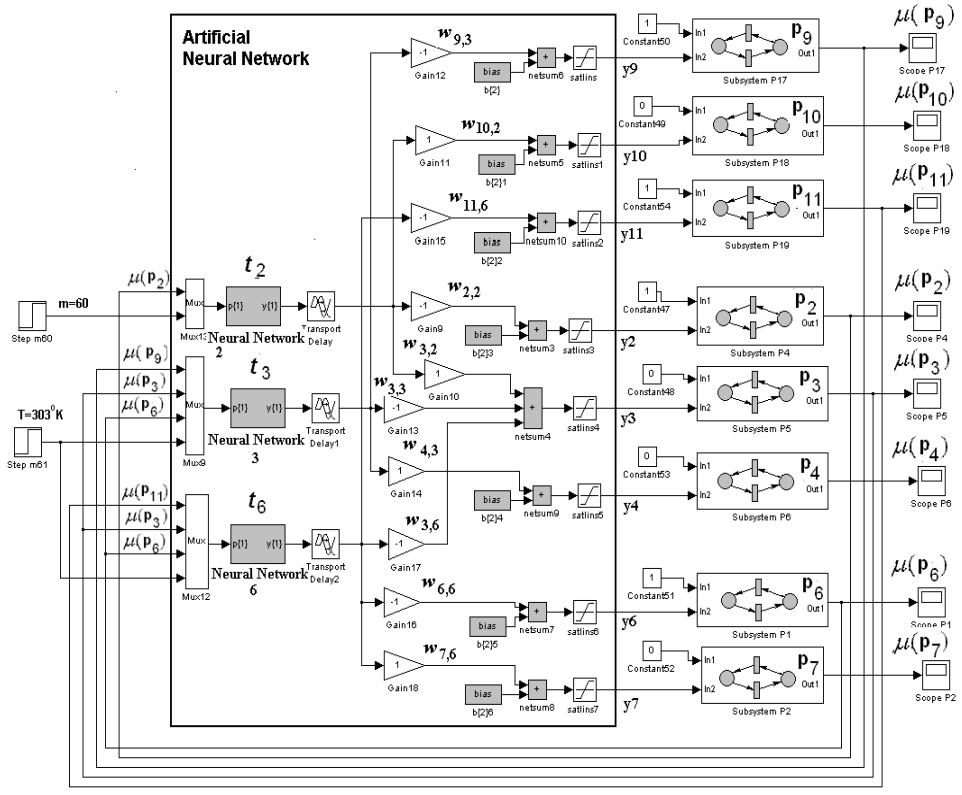

Fig. 7. Scheme of artificial neural network presented by means of MATLAB\Simulink environment for synthesis of Petri nets

Also, as a result of the functioning of the artificial neural network, it is formed the dynamics of the Petri net marking appropriate to the dynamics of the network marking presented in Fig. 5.

It should be noted that in the MATLAB\Simulink environment it has been represented the operation of Petri nets on the basis of RS-triggers due to the lack of an element base for constructing Petri nets.

RESEARCH RESULTS

As a result of modeling the interaction system for an artificial neural network with Petri nets, it has been obtained the dynamics of Petri net marking presented in Fig. 8. In Fig. 8 (left) it is shows the change in time of the Petri net marking in the absence of a stop signal, at which $\mu(p_6) = 0$. And in Fig. 8 (right) it is shows a change in time of the Petri net marking in the presence of a stop signal $\mu(p_6) = 1$. At $\mu(p_6) = 1$, the marker does not move into position p_4 since p_6 is connected by an inhibitory arc with transition t_3. Consequently, when $\mu(p_6) = 1$, transition t_3 is closed. As can be seen, the change in marking during the operation of the circuit with an artificial neural network corresponds to the change in marking of the Petri net shown in Fig. 5.

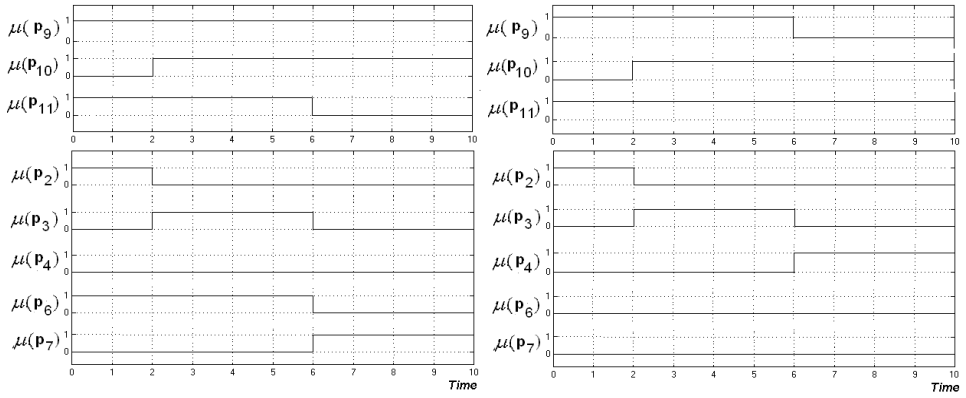

Fig. 8. Diagrams of changes in the marking of Petri net over time obtained as a result of the interaction between the artificial neural network with separate Petri nets

Thus, via training and functioning of the neural network, it has been formed and presented the work of this fragment of the Petri net shown in Fig. 8. Also, during the functioning of the artificial neural network, it is provided the sequence of transitions $\sigma = t_1, t_2 ... t_n$ at which:

$$M_i \xrightarrow{t_1} M_{i+1} \xrightarrow{t_2} ... \xrightarrow{t_4} M_J .$$

In this case, the transition $M_i \longrightarrow M_J$ corresponds to the transition appropriate to the Petri net synthesized in the DC-Net environment.

The presented results confirm the fundamental suitability of the Petri net synthesis method on based the functioning of artificial neural networks. Such method involves the implementation of certain stages.

1. Implementation of the scheme for an artificial neural network interaction with local Petri nets.

2. Division of the neural network into sectors for subsequent stepwise training.

3. Definition of training data (training sample) based on the state graph or reachability tree Petri net and implementing of stepwise training for an artificial neural network.

4. Determination of the incidence matrix which forms Petri net based on the neural network and the launching of its operation.

DISCUSSION

If it is compared the presented method for automatic synthesis of Petri nets with method for the synthesis of similar nets in the DC-Net environment, it is able to determine the appropriate advantages and disadvantages of the presented method. The automatic synthesis method is advantageous because it eliminates the need for a person to construct the Petri net by themselves, as with the DC-Net system [13, 14]. Based on the functioning of the artificial neural network, it is possible to visualize the operation of the logical control algorithm represented in the form of a Petri net. It is only necessary to specify the conditions for the transition from one state of the system to another to train the artificial neural network. Another advantage of the presented method is that it can be implemented on the basis of the MATLAB\Simulink or Labview environment, which in turn allows the generated control algorithm to be linked to the automation system's hardware.

However, it is important to note a few appropriate shortcomings. One of the disadvantages is the need to provide a sufficiently amount of data for training the artificial neural network, both on the conditions for the transition from one state of the system to another and, otherwise, the conditions under which the appropriate transitions are prohibited.

If training data is insufficient, then the artificial neural network can be wrong and present a logical control algorithm which leads to an emergency. This is what makes the system intelligent. The intelligent system is able to represent what had not been considered in the training process with a positive or negative result.

It should be noted that the DC-Net environment provides the ability to visually edit and explore the Petri net, unlike a software application associated with automatic synthesis.

CONCLUSIONS

It has been solved the problem linked to determination of the principles for the training of artificial neural networks in the automatic synthesis of Petri nets in the present work. It is presented the diagram of interaction between a neural network with Petri nets for their compositions. It has been completed the necessary experiments linked to the stepwise training of an artificial neural network using known methods in the MATLAB environment. The performed experiments allow determining the fundamental suitability of the method for automatic synthesis of Petri nets in the formation of control algorithms. Thus, performing the assigned task the further develop the method for automatic synthesis of Petri nets and control algorithms based on the use of artificial neural networks is received.

The development of the method for automatic synthesis of Petri nets allows approaching the solution of a practical problem. Such a task is related to the development of an intelligent system and the appropriate software application. This software must automatically form some logical control algorithms and Petri nets.

Further development of this scientific direction should be directly related to the formation of methods for self-learning of neural networks in the automatic synthesis of Petri nets.

REFERENCES

1. A.A. Gurskiy, A.V. Denisenko, S.M. Dubna, "The automatic synthesis of Petri net based on the functioning of artificial neural network," *Radio electronics, computer science, control*, no. 2 (2021), pp. 84–92. doi: https://doi.org/10.15588/1607-3274-2021-2-9
2. B. Baker, O. Gupta, N. Naik, R. Raskar, "Designing neural network architectures using reinforcement learning," *arXiv preprint*, 18 p., 2016. doi: https://doi.org/10.48550/arXiv.1611.02167
3. Charu C. Aggarwal, *Neural networks and deep learning*. Cham: Springer, 2018. doi: https://doi.org/10.1007/978-3-031-29642-0
4. J.L. Peterson, *Petri net theory and the modeling of systems*. Prentice Hall PTR, 1981, 290 p.
5. D.W. He, B. Strege, H. Tolle, A. Kusiak, "Decomposition in automatic generation of Petri nets for manufacturing system control and scheduling," *International Journal of Production Research*, vol. 38, issue 6, pp. 1437–1457, 2000. doi: https://doi.org/10.1080/002075400188942
6. M.S. Durmuş, U. Yıldırım, M.T. Söylemez, "Automatic generation of Petri Net supervisors for railway interlocking design," *2012 2nd Australian Control Conference, Sydney, NSW, Australia, 2012*, pp. 180–185.
7. M.A. Ndiaye, J.F. Petin, J. Camerini, J.P. Georges, "Performance assessment of industrial control system during pre-sales uncertain context using automatic Colored Petri Nets model generation," *2016 International Conference on Control, Decision and Information Technologies (CoDIT), IEEE, 2016*, pp. 671–676. doi: https://doi.org/10.1109/CoDIT.2016.7593643
8. V. Rätzel, B. Werthmann, M. Haas, J. Strube, W. Marwa, "Disentangling a complex response in cell reprogramming and probing the Waddington landscape by automatic construction of Petri nets," *Biosystems*, vol. 189, 104092, 2020. doi: https://doi.org/10.1016/j.biosystems.2019.104092
9. A.A. Gurskiy, S.M. Dubna, "Nastroıka neıronnoı setı prı avtomatıcheskom sınteze seteı Petrı [Tuning a neural network for automatic synthesis of Petri nets]," *Automation of technological and business processes*, no. 1, pp. 22–32, 2018. doi: https://doi.org/10.15673/atbp.v10i1.877
10. G. Liu, *Petri Nets: Theoretical Models and Analysis Methods for Concurrent Systems*. Springer Nature, 2020, 278 p. doi: https://doi.org/10.1007/978-981-19-6309-4
11. A.A. Gurskiy, A.V. Denisenko, A.G. Nesteryuk, "Diskretno-nepreryvnaya set kak sredstvo modelirovaniya slozhnyh tehnologicheskih processov [Discrete-continuous network as a means of modeling complex technological processes]," *Refrigeration Engineering and Technology*, no. 4, pp. 54–58, 2004.
12. M.Z. Zgurovsky, V.A. Denisenko, *Diskretno neprerivnie sistemi s upravlyaemoi strukturoi [Discrete-continuous systems with controlled structure]*. Kyiv: Naukova dumka, 1998, 350 p.
13. O.G. Nesteryuk, O.R. Sharichev, O.V. Shlemko, "Approach to the development of the visualization module through the process of reduction-decomposition for discrete-continuous net," *VI International scientific and practical conference "The aspects of contemporary scientific research that encompass both theoretical and practical components", January 10–12, 2024, Venice, Italy*, pp. 97–99. Available: https://isu-conference.com/wp-content/uploads/2024/01/The-aspects-of-contemporary-scientific-research-Jan-10-12-2024-Venice-Italy.pdf
14. O.G. Nesteryuk, *Informacijna tekhnologiya modelyuvannya i analizu diskretno-neperervnih avtomatizovanih sistem upravlinnya [Information technology modelling and analysis of discrete-continuous automated control systems]*. Extended abstract of candidate's thesis, Odessa [in Ukraine], 2016.

Received 24.04.2025

INFORMATION ON THE ARTICLE

Alexander A. Gurskiy, ORCID: 0000-0001-5158-2125, Odesa National University of Technology, Ukraine, e-mail: gurskiya2017@gmail.com

Andrey V. Denisenko, ORCID: 0000-0002-8610-0082, National University "Odesa Polytechnics", Ukraine, e-mail: denisenko.a.v@op.edu.ua

Alexander E. Goncharenko, ORCID: 0000-0003-4959-6469, Odesa National University of Technology, Ukraine, e-mail: kholod.automatic@gmail.com

НАВЧАННЯ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ ЗА РЕАЛІЗАЦІЇ ПРОЦЕСУ СИНТЕЗУ МЕРЕЖ ПЕТРІ / О.О. Гурський, А.В. Денисенко, О.Є. Гончаренко

Анотація. Вирішено актуальне завдання, що пов'язане з розробленням і перевіркою на принципову придатність методу автоматичного синтезу мереж Петрі на основі функціонування штучної нейронної мережі. Це дає змогу автоматизувати процес формування визначених алгоритмів логічного керування. Таке завдання також є актуальним, оскільки вперше розглянуто принципи навчання штучних нейронних мереж відомими методами за реалізації задачі синтезу мереж Петрі. Подано розподілення штучних нейронних мереж на окремі локальні мережі для поетапного навчання. Таке поетапне навчання потрібне для синтезу алгоритмів логічного керування та відповідних мереж Петрі. Запропоновано принципи, які можна застосувати на ранніх стадіях навчання штучних нейронних мереж за синтезу мереж Петрі. Виконано експерименти, пов'язані з навчанням локальних нейронних мереж та з реалізацією формування алгоритму логічного керування. У результаті проведених експериментів установлено принципову придатність метода поетапного навчання штучних нейронних мереж для генерації мереж Петрі та автоматизованого формування алгоритмів логічного керування.

Ключові слова: штучні нейронні мережі, мережа Петрі, алгоритм керування, методи навчання, логіко-динамічна система, автоматичний синтез мереж Петрі.