UDC 004.94; 004.032.26(045)

[1]O. I. Chumachenko,
[2]A. Y. Luzhetskyi

SIMULATION MODELING OF MULTIPROCESSOR SYSTEMS BASED ON NEURAL NETWORKS

Technical Cybernetic Department Nationnal Technical University of Ukraine
"Kyiv polytechnic institute", Kyiv, Ukraine
E-mails: [1]chumachenko@tk.kpi.ua, [2]a.lyzhetskiy@mail.ru

Abstract. *The task simulation modeling of multiprocessor systems based on the use of cellular neural networks. Each processing element is described by a subset of neurons in artificial neural networks. As the algorithmic provision considered locally asynchronous methods specifically targeted for use in cellular neural networks.*

Keywords: multiprocessor systems; cellular neural networks; locally asynchronous algorithm; simulation modeling.

I. INTRODUCTION

At a time when processing information with amazing speed penetrated all new areas of science, production and social life in the field of automation research, has long been marked by problems with the available computing power of electronic computers with traditional architecture. For this reason, remains a large number of scientific problems, among which can be attributed a comprehensive study of diffusion processes in three dimensions, modeling of processes in solid-state physics in the presence of phase transitions with taking into account of quantum effects, the simulation with optimization of large technical applications in planetary scale, complex study of problems in aero hydro dynamics and thermonuclear fusion and many others.[3]

The persistence of these problems in this case is considered only in the degree in which they relate to the use of computers as a tool of self-alternative with respect to classical methods of research. So, this thesis does not mean that in the relevant research areas some good results have not been reached yet. This only highlights the complexity of these problems and the way to solve them is either too complicated or requires higher performance of the computing technique.

The experience of IBM, progress in microelectronics does not allow the traditional ideology of creating a computer significantly to reduce the duration of the cycle nor the reference to memory, nor, especially, the duration of the basic cycle of computer devices. Therefore, improving the performance of computer equipment should be carried out primarily by improving the architecture and computer design techniques.[4]

II. THE PROBLEM STATEMENT

Multiprocessor systems that are considered in this article require special approaches for their development and research. Complexity of the structure of such systems requires a good scientific justification and comprehensive rationality sound technical solutions. Under these conditions significantly increases the role of mathematical models describing the system as formalized by means of abstraction, allowing high exponent adequacy reflect those of its properties that are necessary for the evaluation of the studied characteristics. [4]

Therefore, we believe that simulation modeling is almost inevitable step in the study and evaluation of complex systems with a large number of items, which will certainly include the multiprocessor systems.

Simulation models reproduce the behavior of the system in accordance with certain formal rules, asked by the abstraction device. The dynamic processes specific to the research object, are replaced by processes taking place in an abstract environment, and the degree of adequacy of the simulation model is completely determined by the nature of space-time correlations of these processes.

The structure of the simulation model can be represented as the following functional dependence

$$\overline{\mathbf{P}} = F(\overline{\mathbf{X}}, \overline{\mathbf{Y}}),$$

where $\overline{\mathbf{P}} = (p_1, p_2, \ldots, p_i)$ – is the vector of studied parameters; $\overline{\mathbf{X}} = (x_1, x_2, \ldots, x_j)$ – is the vector of variable arguments; $\overline{\mathbf{Y}} = (y_1, y_2, \ldots, y_k)$ – is the vector of inaccessible arguments; F – is the functional dependence between arguments and parameters studied.

The simulation of such multicomponent structures requires substantial investment of time when using sequential, even quite powerful, computer.

Currently, there is a variety of approaches to the subject, each of them is characterized by its own advantages and disadvantages. The most widely used are the following formal tools for describing multiprocessor systems [1]:

– Petri nets;
– Colored Petri nets;
– E-nets;
– makro-E-nets;
– PRO-nets.

However, in this article we would like to consider simulation modeling of multiprocessor systems based on neural networks, in particular on the basis of cellular neural networks (CNN).

An obvious approach to increase the simulation speed is the use of parallel computation in which each processor element (PE) is determined by a neuron cell or a group of neurons of the neural network.

III. Cellular Neural Networks

Cellular neural network are assigned to one of the most promising directions of development of the theory of artificial neural networks. Interest in them, according to the authors, due primarily to the possibility of easy adaptation to their actual physical structure of the tasks that have natural parallelism. Such structures are well known in contemporary physics as complex systems, self-organizing systems, border chaos systems with collective behavior, etc. In the work [7] they are called cellular nonlinear networks and cellular neural networks (CNN) are considered as their species. According to [7], CNN - is massively parallel computing paradigm defined by the *N*-dimensional discrete space. It consists of an *N*-dimensional array of homogeneous elements (cells). Structure of types of links between CNN cells is shown in Fig. 1.

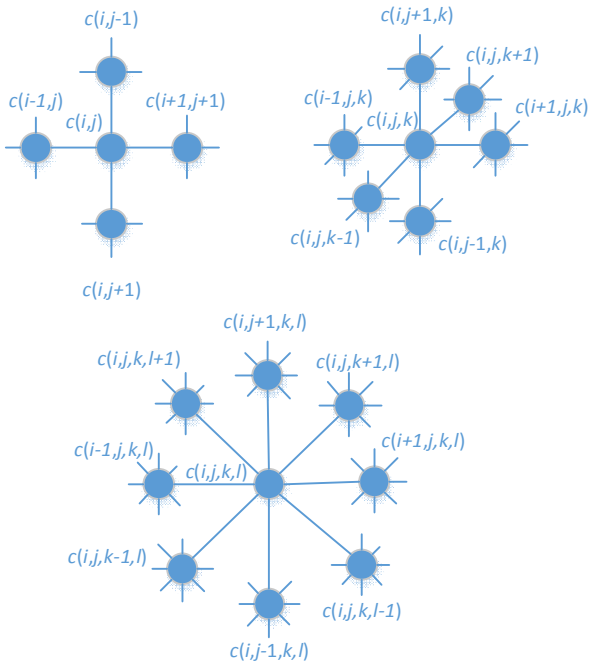

Fig. 1. Examples of structural connections in cellular neural networks

Each of the data presented in Fig. 1 characterized by the number of possible structures on the site link neuron (cells) with neighbors, resulting environment index α. If the value of α exceeds the number of nearest neighbors, such a CNN may be set with a multiple functions of the neighborhood:

$$N_r : H \to H_\alpha, N_r(g) = \{z \mid d(g,z) \le r\},$$

where H – is the set of coordinates of all cells, H_α – the set of coordinates of neighboring cells, g, z – are coordinate vectors, r –is the degree of neighborhood.

The degree of neighborhood r is a maximum coordinate distance between the considered neuron and its neighbors. Method of determining the distance is set up by the function $d(g, z)=|g-z|$.

Nodular CNN neuron is a specialized processor with the number of external connections, relevant to environment index α. When implementation of the classical algorithm CNN its operation is reduced mainly to the calculation of functional for data in an explicit or implicit form with continuous or discrete time. The general form of the dynamic equation for discrete time

$$\begin{aligned} x_g(n+1) &= x_g(n) + \sum_{z \in N_r(g)} A_{g,z} y_z + \sum_{z \in N_r(g)} B_{g,z} u_z + I_g(n), \\ y_g(n) &= f(x_g), \end{aligned} \quad (1)$$

where x_g is an internal state of the cell $c(g)$; y_z is output signal of the cell $c(z)I$; u_z is external input vector of cell $c(z)$; I_g is offset; $A_{g,z}$, $B_{g,z}$ are functionals corresponding to relations between CNN cells $c(g)$ and $c(z)$.

IV. Locally Asynchronous Method

Let us consider locally asynchronous methods [5], specifically targeted at the implementation of a cellular neural networks. Problem for solving by the mentioned methods can be represented as an operator equation

$$L\mathbf{X} = U, \quad (2)$$

where L is the differential operator; $\mathbf{X}$ is matrix of state of CNN; U is function of the right side.

To solve this equation, use one of the well-known iterative methods [6], which suggests the reduction of equation (2) to a system of difference equations on homogeneous net Ω with the step h:

$$L_h \mathbf{X}_h = U_h, \quad (3)$$

where L_h is the difference operator; $\mathbf{X}_h$ is the difference matrix operator; U_h is network function.

In the simplest case, is set in direct correspondence neuron cell network to node of the grid area. Then the dynamics of cell should provide a solution to a differential equation by an iterative formula

corresponding to the selected method. When choosing a suitable iterative formula plays an important role view of the difference operator L_h. The main selection criteria is the difference operator form template on which it is defined, and requirements for the order update on the specified template.

Locally asynchronous method [5], is oriented for application in cellular neural networks, allows to generate a template of a difference operator, limited degree of neighborhood r, and to provide an asynchronous transfer mode, removing a hard limit on the update of all data on the template before executing the next iteration. Dynamic cell in this case is determined by the iterative

$$x_g(n+1) = x_g(n) - \frac{w_g}{h^2}\left[L_g x_g(n) - u_g\right], \qquad (4)$$

where L_g is part of a difference operator; $x_g(n)$ is state of cell $c(g)$ at the iteration step n; h is value of the discretization step; w_g is internal parameter; u_g is value of the grid function.

How to update the data on the pattern of each component is determined by the difference operator L_g chaotic sequence of nonempty sets {1,2, ... η}. Using this sequence, we construct a sequence of iterations by the rule

$$x_g(n) = \begin{cases} x_g(n-1), & g \notin J_n, \\ L_g\left[x_1\left(z_1(n)\right), x_2\left(z_2(n)\right), \ldots, \right. & \\ \qquad\qquad \left. x_\eta\left(z_\eta(n)\right)\right], & g \in J_n. \end{cases} \qquad (5)$$

Maximum efficiency is achieved by asynchronous algorithm

$$\Re = \left\{g \in \{1,2,\ldots,\eta\} \,\forall n \in N \exists m \geq n : g \in J_n\right\}. \qquad (6)$$

In this case we say that the chaotic sequence has a maximum residue due to the presence of the minimum sequence of sets $\left\{m_g\right\}_{g=0}^{\infty}$.

Increasing sequence of sets is called a minimum under the following conditions:

1. $m_0 = 0$;
2. $\bigcup_{i=m_g+1}^{m_{g+1}} J_i = \{1,2,\ldots,n\}$, $g = 0,1,2,\ldots$;
3. $\bigcup_{i=m_g+1}^{m_{g+1}-1} J_i \subset \{1,2,\ldots,n\}$, $g = 0,1,2,\ldots$.

Hence the set $\left\{z_g(n)\right\}_{n=1}^{\infty}$, $g = \{1, 2,\ldots\eta\}$ asynchronous algorithm (5) must satisfy the conditions of: $z_g \leq (n-1), z_i = (n) \to \infty$.

In order to simplify the algorithm (5) the formation of iterative sequence $\left\{x_g(n)\right\}_{n=1}^{\infty}$ must be entered the condition of pseudo compressibility on the differential operator L in the form of inequality

$$\|LX - \varphi\| < \|X - \varphi\|. \qquad (7)$$

In this case, the formation of iterative sequence $\left\{x_g(n)\right\}_{n=1}^{\infty}$ will occur by the rule

$$x_g(n+1) = L_g\left(x_g\left(z_1^g(n)\right), \ldots, x_g\left(s_{k_{z(n)}}^g(n)\right)\right), \qquad (8)$$

where $\{g(n) \mid n = 1, 2, \ldots\}$ – a sequence of elements $g(n) \in H$; $Z^g = \left\{z_{k_g}^g(n)\right\}_{n=0}^{\infty}$, $g = 1,\ldots H$, $k_g = 1,\ldots k$ – set of non-negative integers satisfying

$$\begin{cases} 0 < z_{k_{g(n)}}^g(n) \leq n, n > 0, \\ -M \leq z_{k_{g(n)}}^g(n) \leq 0, n = 0. \end{cases} \qquad (9)$$

From (8) we can see that to start iteration we can use data from the earlier started neurons with some restriction M. Neuron, ahead of its neighbors may use the outdated data lag with depth with a maximum value equal to the number of iterations of this neuron. It follows that locally asynchronous algorithm is quite efficient in terms of physical asynchrony. However, its maximum efficiency can only be achieved with a maximum draft of chaotic sequence $\left\{J_n\right\}_{n=1}^{\infty}$, i.e. a minimum delay of one with respect to another neuron.

V. SIMULATION

Neuronal communications simulation model for said computational process must meet the following conditions [2]:

– Input streams coming from the neurons function defined neighborhood $N_r(g)$, independent;

– Basic parameters for modeling of each neuron is its condition $\sigma(c)$ and $\tau(c)$ – the next time the data modifications that could potentially lead to a change in state.

In accordance with this definition, communication model of CNN for each neuron sets two main functions:

– GetState() – function of forming a new state;

– GetTime() – formation function of point in time for the new state.

Condition arbitrary neuron CNN at time t can be expressed by dependence

$$\sigma_t(c) = \mathrm{GetState}\left(c, \left\{\sigma_x(c_i \in H_\alpha)\right\}_{i=1}^{\alpha}, t \mid x \to t,\ x < t\right).$$

Time of the next modification

$$\tau(c) = \mathrm{GetTime}\left(c, \left\{\sigma_x(c_i \in H_\alpha)\right\}_{i=1}^{\alpha}, t \mid x \to t, x < t\right).$$

In both formulas, following designations are used: x is time, directly prior to the time t; $\{\sigma_x(c_i \in H_\alpha)\}_{i=1}^{\alpha}$ is state of neurons belonging to the set neighborhood H_α.

Suppose that in a multiprocessor system, which we model each neuron cell network is responsible for the modeling of a single processing element. Consider an asynchronous approach to the implementation of interneuron communication for such a model. This approach is in contrast to synchronous, does not require any additional resources, and at the same time maintains a high rate of similarity computing process. (Fig. 2).

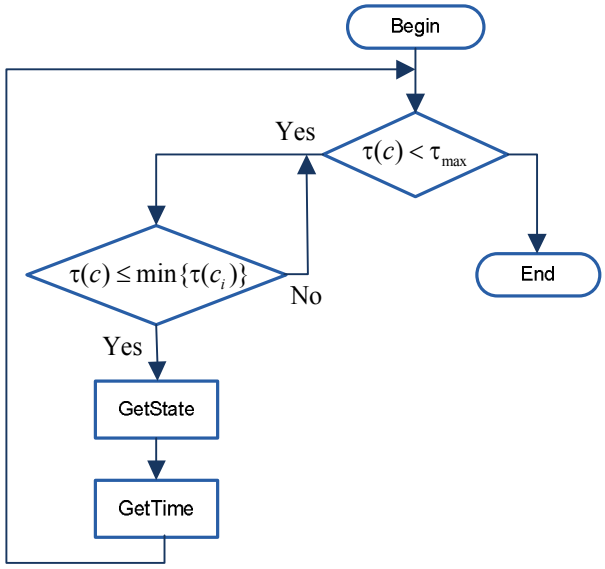

Fig. 2. Asynchronous simulation algorithm

The essential feature of this approach is that the model time $\tau(c)$ of each of neurons is not divided into periods of calculation and exchange, and is continuous. Due to this cellular network neurons can be simultaneously at different stages of the calculations. The key conditions for promotion of the local model time $\tau(c)$ a single neuron in this case are the values of model local time $\tau(c_i)$, topologically related neighboring neurons. However, the condition $\tau(c) \le \min_{c_i \in H_\alpha}\{\tau(c_i)\}_{i=1}^{\alpha}$ does not imply its discovery immediately after the occurrence. To this condition was found to PE function modeling the neuron cell network, it needs to get information about the state of the local and neighboring time in any order with an arbitrary delay and regardless of how the neighboring PE work.

Neuron model time $\tau(c)$ has no direct connection with the physical time flow of simulation process. It is important, however, to note that the model time as a physical characteristic is the constant increase of property.

The set of variables $\{\sigma(c), \tau(c)\}$, for asynchronous algorithm does not have a counterpart of the local. Therefore, the values of state and local time potentially become available neighboring neurons as soon as they update. In this regard, there is a problem accessing shared resources stored values of these variables. One of the traditional methods can solve this problem. Therefore, for simplicity, we assume that this algorithm moments reading and writing data in the shared resource is never the same.

Despite the apparent randomness of this algorithm is deprived of clinch situations. Freedom from clinch stems from the fact that a neuron with the minimum time throughout the network always has the opportunity to promote their local time. Assume that there is no single neuron from set $\{c_i\}_{i=1}^{\alpha}$ environment of the neuron c, for which the steps are carried out to promote the condition of the local time for the reason that a neuron c has the minimum local time $\tau(c)$. In this case, the execution of functions GetState and GetTime can be safe: none of the next neuron changes its state or local time to moment of the closure computation neuron c. This fact guarantees a minimum performance CNN which is average much higher.

Using the properties of locally asynchronous method to ensure the convergence of asynchronous physical conditions, combined with asynchronous communication described algorithm can virtually eliminate the simple computing resources during problem solving. Sign of the asynchronous algorithm is local in nature and provides a stop when the local model time a certain critical value, selected from the conditions of the rate of convergence locally asynchronous method.

In the above algorithms for a single neuron cell network corresponds to one PE. Such a structure can be very inaccurate because of the difficulty of designing PE. Computational process will be much more credible if the description of PE by a subset of neurons.

VI. Example

Let us consider work of aggregate simulation model of the example of the two-dimensional cellular network C with dimension $n \times n$. Lets $n > m$, $n{:}m$ and available for use $(n/m)^2$ of processor elements, each of which contains a subnetwork $C_{k,l} \mid 0 \le k, l \le n/m - 1$ from $m \times m$ neurons (Fig. 3).

Set of neurons $C = \{C_{k,l}\}_{k,l=0}^{n/m-1}$, interpreted as aggregative structure can be roughly divided into two subsets:

– external subset of neurons $O = \left\{ \left\{ c_{i \cdot m,j}, c_{(i+1) \cdot m-1,j} \right\}_{i=0}^{n/m-1} \right\}_{j=0}^{n-1}$, consists of neurons for which one or more neighbors are in one or two of neighboring subnets;

– internal subset of neurons $I = C \setminus O$, for which all the neighboring neurons belong in the same subnet.

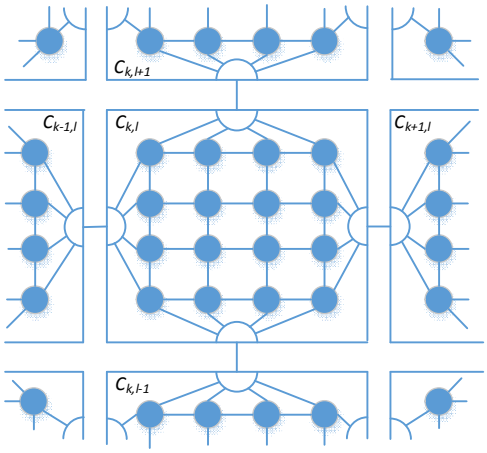

Fig. 3. Aggregate simulation model ($m = 4$)

The main difference of aggregative data structure of the algorithm is that each of the neurons further stores data on the subnets which exist for that the neuron communication topology according CNN. For the topology presented in Fig. 3, the data can be automatically generated in the form of sets $w_{i,j} \in W$ under the following conditions:

$$C_{k-1,l} \in \left\{ w_{km,j} \right\}_{j=lm}^{m(l+1)-1}; \; C_{k+1,l} \in \left\{ w_{m(k+1)-1,j} \right\}_{j=lm}^{m(l+1)-1};$$

$$C_{k,l-1} \in \left\{ w_{i,\mathrm{lm}} \right\}_{j=km}^{m(\mathrm{k}+1)-1}; \; C_{k,l+1} \in \left\{ w_{i,m(\mathrm{l}+1)-1} \right\}_{i=km}^{m(\mathrm{k}+1)-1}.$$

Aggregative structure $c_{k,l}$, has one channel of communication with each of the adjacent aggregative structure. This raises the question of priority access to the resources of the channel. To solve this problem, we introduce a variety of channel times:

$$T\left(C_{k,l}\right) = \left\{ \tau\left(C_{k-1,l}\right), \tau\left(C_{k+1,l}\right), \tau\left(C_{k,l-1}\right), \tau\left(C_{k,l+1}\right) \right\}.$$

The current value of channel time is always equal to the minimum local time defined on the set of neurons containing in its set $w_{i,j}$ and corresponding to aggregative structure.

For Example, $W_{k,l}$ of non-empty subsets $w_{i,j}$ for aggregative structure $C_{k,l}$, illustrated in Fig. 3, contains the following subsets

$$W_{k,l} = \left\{ \begin{matrix} w_{4k,4l}, w_{4k+1,4l}, w_{4k+2,4l}, w_{4k+3,4l}, \\ w_{4k,4l+1}, w_{4k,4l+2}, w_{4k,4l+3}, w_{4k+1,4l+3}, \\ w_{4k+2,4l+3}, w_{4k+3,4l+3}, w_{4k+3,4l+1}, w_{4k+3,4l+2} \end{matrix} \right\}.$$

When forming the channel time $\tau\left(C_{k-1,l}\right)$ should be used a subset of local times $\left\{ \tau\left(c_{4k,4l}\right), \tau\left(c_{4k,4l+1}\right), \tau\left(c_{4k,4l+2}\right), \tau\left(c_{4k,4l+3}\right) \right\}$, as aggregative structure $C_{k-1,l}$ is an element of relevant to subsets

$$w_{4k,4l} = \left\{ C_{k-1,l} C_{k,l-1} \right\}, w_{4k,4l+1} = \left\{ C_{k-1,l} \right\},$$

$$w_{4k,4l+2} = \left\{ C_{k-1,l} \right\}, w_{4k,4l+3} = \left\{ C_{k-1,l} C_{k,l+1} \right\}.$$

That is why,

$$\tau\left(C_{k-1,l}\right) = \min\left\{ \tau\left(c_{4k,4l}\right), \tau\left(c_{4k,4l+1}\right), \tau\left(c_{4k,4l+2}\right), \tau\left(c_{4k,4l+3}\right) \right\}.$$

This approach to the promotion of channel time frees aggregation algorithm from clinch situations because the neuron with minimum time will always be able to move forward.

Block diagram of aggregative algorithm is shown in Fig. 4. The main difference between this algorithm from the previously considered is the presence of branching which allows different promotions to realize simulation time depending on the membership of a neuron to the set O or set I. As the characteristic $c \in O$ selected condition of the non emptiness of set w, associated with the corresponding neuron c. Promotion of local time for this type of neurons is the group nature and is determined by the expression $\tau(C) = \min_{c_i \in O} \left\{ \tau(c_i) \right\}$, which indicates that the local time of the group $\tau(C)$ equal to the minimum local time of a neuron included in the appropriate group of aggregative exchange. Resolution on the further evolution can thus be obtained in the case where this group is the local time at the minimum of all the aggregative groups attached to a given communication channel.

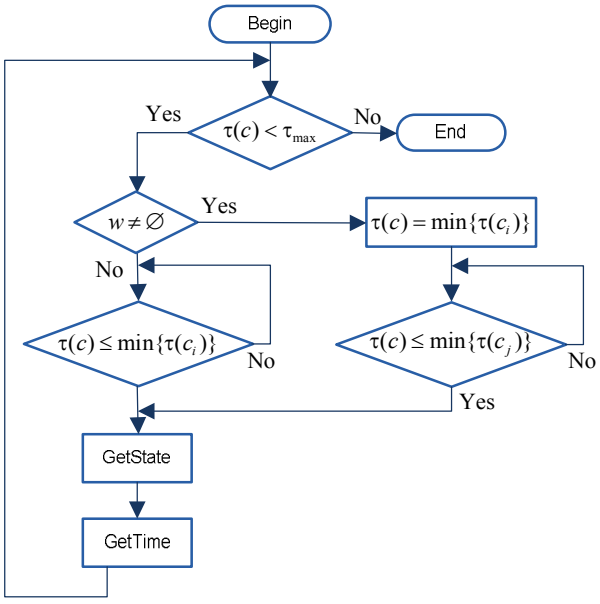

Fig. 4. Aggregative simulation algorithm

In case $w = \varnothing$ we obtain the condition of belonging to a plurality of the neuron *I*. Progress in promoting of local time for this type of neurons is analogous to that discussed earlier in the description of an asynchronous simulation algorithm.

VII. Conclusion

In this work we tested the possibility of using neural networks for simulation of complex multiprocessor systems to improve their performance characteristics. Cellular neural networks were used for modeling, as soon as among all variants of neural networks architectures they suit best for parallel computing. Suggested to use locally asynchronous methods specifically oriented toward the implementation of a cellular neural networks. Operation of this approach to the organization of processes of interaction between neurons allows implementing a variety of models of asynchronous multiprocessor systems through cellular neural networks in applications with parallel structures by locally asynchronous method.

References

[1] Barskyi, A. B. 1980. Planning parallel computing processes. Moscow, Mashinostroenie. 192 p. (in Russian).

[2] Vavilov, A. A.; Nikolaev, E. S. 1983. Simulation of manufacturing systems. Moscow, Mashinostroenie. 416 p. (in Russian).

[3] Grytsyk, V. V.; Zlatogurskyi, E. R.; Mikhoilovskyi, V. N. 1977. Parallelization of algorithms for data processing. Kyiv, Naukova dumka. 123 p. (in Russian).

[4] Nesterenko, B. B.; Novotarskyi, M. A. 1995. Multiprocessor systems. Kyiv, Institute of Mathematics. 408 p. (in Russian).

[5] Novotarskyi, M. A.; Nesterenko, B. B. 2004. Artificial neural networks: the calculation. Kyiv, Institute of Mathematics. 408 p. (in Ukrainian).

[6] Samarsky, A. A.; Nikolaev, E. S. 1978. Methods for solving grid equations. Moscow, Nauka. 592 p. (in Russian).

[7] Chua, L. O. 1998. CNN: A paradigm for complexity. – Singapore, World Scientific. 380 p.

Received 18 November 2013

Chumachenko Olena. Candidate of engineering. Associate Professor.
Technical Cybernetic Department, National Technical University of Ukraine "Kyiv Polytechnic Institute", Kyiv, Ukraine
Education: Georgian Politechnic Institute, Tbilisi, Georgia (1980).
Research area: system analysis, artificial neuron nets.
Publications: 46.
E-mail: chumachenko@tk.kpi.ua

Luzhetskyi Andrii. Student.
Technical Cybernetic Department, National Technical University of Ukraine "Kyiv Polytechnic Institute", Kyiv, Ukraine
Education: National Technical University of Ukraine "Kyiv Polytechnic Institute", Kyiv, Ukraine (2012).
Research area: artificial neuron nets, simulation modeling.
Publications: 2.
E-mail: a.lyzhetskiy@mail.ru

О. І. Чумаченко, А. Ю. Лужецький. Імітаційне моделювання мультипроцесорних систем на основі нейромереж

Розглянуто задачу імітаційного моделювання мультипроцесорних систем на основі використання клітинних нейронних мереж. Кожен процесорний елемент описується деякою підмножиною нейронів штучної нейромережі. В якості алгоритмічного забезпечення розглядаються локально-асинхронні методи, спеціально орієнтовані на використання у клітинний нейронних мережах.

Ключові слова: мультипроцесорні системи; клітинні нейронні мережі; локально-асинхронний алгоритм; імітаційне моделювання.

Чумаченко Олена Іллівна. Кандидат технічних наук. Доцент.
Кафедра технічної кібернетики, Національний технічний університет України «Київський політехнічний інститут», Київ, Україна.
Освіта: Грузинський політехнічний інститут, Тбілісі, Грузія (1980).
Напрям наукової діяльності: системний аналіз, штучні нейронні мережі.
Кількість публікацій: 46.
E-mail: chumachenko@tk.kpi.ua

Лужецький Андрій Юрійович. Студент.
Кафедра технічної кібернетики, Національний технічний університет України «Київський політехнічний інститут», Київ, Україна
Освіта: Національний технічний університет України «Київський політехнічний інститут», Київ, Україна (2012).
Напрям наукової діяльності: штучні нейронні мережі, імітаційне моделювання.
Кількість публікацій: 2.
E-mail: a.lyzhetskiy@mail.ru

Е. И. Чумаченко, А. Ю. Лужецкий. Имитационное моделирование мультипроцессорных систем на основе нейросетей

Рассмотрена задача имитационного моделирования мультипроцессорных систем на основе использования клеточных нейронных сетей. Каждый процессорный элемент описывается некоторым подмножеством нейронов искусственной нейросети. В качестве алгоритмического обеспечения рассматриваются локально-асинхронные методы, специально ориентированные на использование в клеточный нейронных сетях.
Ключевые слова: мультипроцессорные системы; клеточные нейронные сети; локально-асинхронный алгоритм; имитационное моделирование.

Чумаченко Елена Ильинична. Кандидат технических наук. Доцент.
Кафедра технической кибернетики, Национальный технический университет Украины «Киевский политехнический институт», Киев, Украина.
Образование: Грузинский политехнический институт, Тбилиси, Грузия (1980).
Направление научной деятельности: системный анализ, искусственные нейронные сети.
Количество публикаций: 46.
E-mail: chumachenko@tk.kpi.ua

Лужецкий Андрей Юрьевич. Студент.
Кафедра технической кибернетики, Национальный технический университет Украины «Киевский политехнический институт», Киев, Украина.
Образование: Национальный технический университет Украины «Киевский политехнический институт», Киев, Украина (2012).
Направление научной деятельности: искусственные нейронные сети, имитационное моделирование.
Количество публикаций: 2.
E-mail: a.lyzhetskiy@mail.ru