computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

ARTIFICIAL IMMUNE SYSTEMS APPROACH FOR MALWARE DETECTION: NEURAL NETWORKS APPLYING FOR IMMUNE DETECTORS CONSTRUCTION

Sergei Bezobrazov, Vladimir Golovko

Brest State Technical University,
Moskovskaja str. 267, 224017 Brest, Belarus,
bescase@gmail.com, gva@bstu.by

Abstract: *This paper presents an approach for solving unknown computer viruses detection problem based on the Artificial Immune System (AIS) method, where immune detectors represented neural networks. The AIS is the biologically-inspired technique which have powerful information processing capabilities that makes it attractive for applying in computer security systems. Computer security systems based on AIS principles allow detect unknown malicious code. In this work we are describing model build on the AIS approach in which detectors represent the Learning Vector Quantization (LVQ) neural networks. Basic principles of the biological immune system (BIS) and comparative analysis of unknown computer viruses detection for different antivirus software and our model are presented.*

Keywords: *artificial immune system, computer security system, malicious code detection, LVQ neural network.*

1. INTRODUCTION

Up-to date antivirus software is complex software modules (software scanners, heuristic analyzers, firewalls, disk auditors, emulator et al.) which realize different algorithms for the computer security. These modules are integrated in operating-system kernel and work with operating system bodily. Nevertheless, antivirus software engineers lose struggle against malware developers. Virus writers continuously develop the new infection algorithms and bypass the existing protection. The quality of malware is rise permanently. Hackers pass ahead of antivirus software engineers. The response to new virus from antivirus industry can be late on dozens of hours. In the meantime up-to date malware is capable to infect of thousands computer systems, to produce a viral epidemic and to bring in a huge damage.

At present the most exact method for malware detection is signature analysis [1]. This method based on comparison unknown pattern with virus signatures. Presence of actual virus signature databases is necessary to success malware detection. Antivirus with outdated virus signature databases is powerless in the face of new security threat. The computer users are in need of the regular data base updating. Do not save the situation heuristic analyzers [2] which were developed for detection unknown viruses. To date they are still a long way from perfect and frequently heuristic analyzers find malicious code where it absent (in noninfected files) and vice versa. According to some estimates the heuristic analyzers detect 25-30 percent of all amount malware and have a high level of misoperations [2].

Biologically-inspired methods such as artificial neural networks (ANN), genetic algorithms, and cellular automata proved its own appropriateness and successfully used by solving many problems in science and engineering sphere. The biological immune system (BIS) is unique protective mechanism which defends organism from invaders: harmful bacteria and viruses. The BIS capable to detect foreign cells and destroy them, and based on synthesis of special proteins – antibodies, which capable to bind with foreign material. Every day BIS face with a dozens invaders and successfully struggle against them. If only we could create the same computer security system then we would decide the problem of unknown malicious code detection. The AIS is grounded on basic principles of BIS and has powerful information processing capabilities such as future extraction, pattern recognition, learning, adaptability, memory, and

distributive nature. All aforesaid features made AIS attractive for up-to-date computer security system creation.

This paper presents AIS approach for malicious code detection. Aspect of AIS which we used in our security system consists in ANN application for detectors generation. The paper is organized as follows. Section 2 gives a short review of biological immune system mechanism and malicious code detection method based on AIS. General model of AIS-based security system is also adduced. Section 3 describes the ANN architecture for detectors generation. In section 4 the experimental model of the AIS security system is described. The results of tests and comparative analysis of computer viruses detection for different antivirus software and our model are given in Section 5. Conclusions are discussed in Section 6.

2. THE AIS METHOD FOR INFORMATION SECURITY

The AIS is biological-inspired method which appears owing to the biological immune system. Before the start consideration of AIS's mechanisms let's describe shortly basic principles of the BIS.

2.1. THE BIOLOGICAL IMMUNE SYSTEM REVIEW

The biological immune system is based on capability of antibodies to distinguishes between self (cells of own body) and nonself (antigens, foreign substance) [3]. For complete and successful detection of wide variety of antigens the BIS must generate a large variety of detectors (B-lymphocytes and T-lymphocytes). Lymphocytes are formed from bone marrow stem cells and initially incapable of antigens detect. In order to acquire immunological ability they have to go through maturation process. T-lymphocytes are mature in thymus and B-lymphocytes are mature in lymph nodes. Mature lymphocytes have on the own surface detectors which able to react on specific antigens. They circulate in the body and perform function on antigens detection [4]. When some lymphocyte detects an antigen the process called clonal selection is occurred [5]. The clonal selection process consists in proliferation those lymphocytes who detected a virus; thereby a large population of identical detectors for quick virus eliminating is formed. Another important process in the BIS is immune memory [4]. After elimination of all antigens owing to clonal selection most of cloned lymphocytes died, however some of them move to so-called memory cells. Population of such cells forms the immune memory. By repeated infection antigens can be detected quickly sine the BIS already has lymphocytes which react on this infection. Described processes showed in Figure 1.

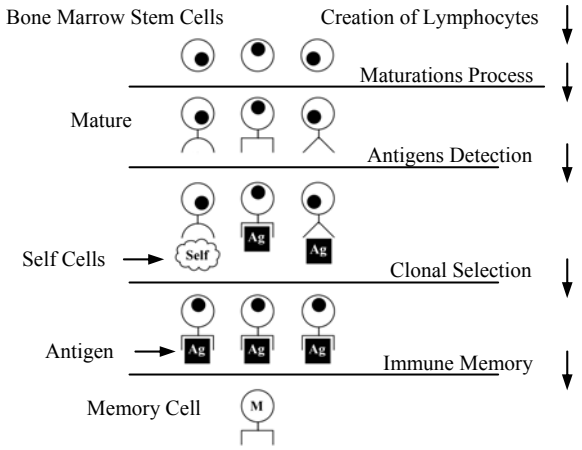

Fig. 1 – Basic principles of biological immune system: stages of lymphocytes evolution.

Described basic principles of BIS are underlined to AIS activity.

2.2. The Artificial Immune Systems Review

The AIS is founded on the same processes as BIS: detectors generation, detectors maturation, detection process, detectors cloning and mutation, immune memory creation. Let's view in detail each process (Fig. 2 shows processes as flow block).

Process of detectors creation in computer system represents a random generation of detectors population. Each of them can be, for example, as binary string of fixed size [6].

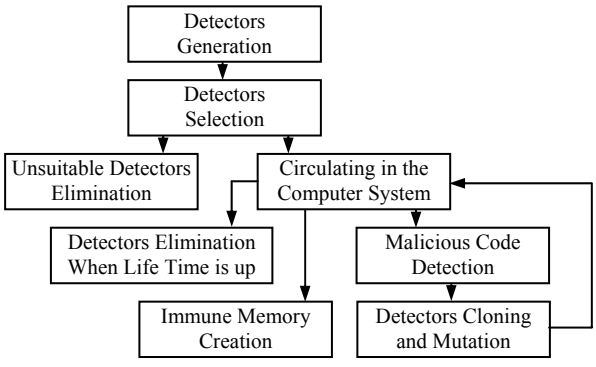

Fig. 2 – Block-diagram model of artificial immune system: AIS interprocess communication.

After generation detectors undergo a selection process. Since detector generations is randomly process, we should defend the computer system from undesirable detectors. During the selection process unsuitable detectors are eliminated and survive only those which able to distinguish between self and nonself. S. Forest at al. [7] proposed negative selection algorithm based on the principles

of self – nonself discrimination in the BIS. According to negative selection algorithm detectors are compared with set of self files. If detector is similar to self files, it is reputed as negative and destroyed. Only those detectors survive which are structurally different from self files. For matching between detectors and files can be applied different rules: bit-by-bit comparison, r-contiguous matching [7] and r-chunk matching [8]. Mature detectors structurally different from self files therefore react only against malicious code.

Mature detectors circulate in computer systems. For maintenance of wide variety of structurally different detectors, each detector has a lifecycle [9]. Lifecycle is a time during detectors can be found in the computer system. When the life time ends the detector is destroyed but if the detector detected malicious code then lifecycle is prolong. Lifecycle mechanism allows the AIS to unload from weak detectors and permanently provide a space for new various detectors.

When malicious code enters the computer system it often infects a large quantity of files. For quick reacting and eliminating virus manifestation we need a great number of similar detectors. A detector which found malicious code undergoes a cloning mechanism. Cloning means a large quantity of similar detectors creation. This mechanism allows the AIS infection elimination in a short space of time. Along with cloning a mutation mechanism is used [10]. Mutation process means small random changes in detectors structure (for example, inverting of several bits in binary string) thereby as much as possible similar structure to finding virus acquires.

When the malicious virus is eliminated then most of cloning detectors die. However the fittest of them are kept as memory detectors. A set of such detectors are formed an immune memory. The immune memory keeps information about all malicious code which a computer system infects. The same as BIS the immune memory allows the AIS to quickly react on repeated infection and to fight against it.

3. THE LEARNING VECTOR QUANTIZATION FOR DETECTORS CONSTRUCTION

In the artificial immune system the detectors act as main element for malware detection. The immune detectors circulate through file system and RAM of computer and detect malware. Successful malware detection depends to a large extent on choice of detectors structure. We considered the detector as a binary string. This structure is comfortable, as it corresponds with data presentation in computer systems, and allows to implement simple matching rules. However, binary structure applies some restrictions. As it is well known bit-by-bit comparison is one of the slowest operations and needs heavy computational power. We propose the ANN applying for the detectors formation. This approach for the detectors generation should remove weaknesses of the binary string structure and should increase a rate of the malicious code detection.

3.1. THE ARTIFICIAL NEURAL NETWORKS FOR VECTOR QUANTIZATION

The ANN for vector quantization was proposed by T. Kohonen in 1982 and named as learning vector quantization (LVQ) [11]. The LVQ is used in classification and image segmentation problems. The LVQ is a feedforward artificial neural network with an input layer, a single hidden competitive Kohonen layer and an output layer (see Fig.3).

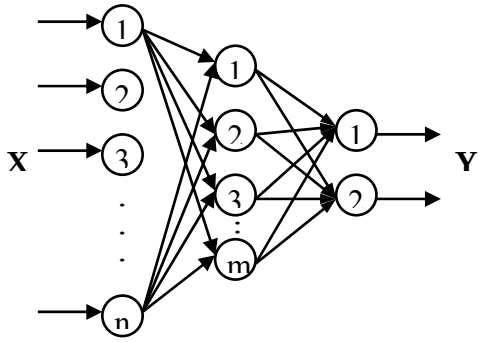

Fig. 3 – The Learning vector quantization architecture: one hidden competitive layer of neurons fully connected with the input layer, and the linear output layer consists of a number of neurons equal of a number of classes.

The output layer has so many elements as there are classes. Processing elements of the hidden (Kohonen) layer are grouped for each of these classes. Each class can be represents as a number of cells of the input space of samples. The centre of each cell corresponds to a codebook vector. One codebook vector represents one class only. The main idea of vector quantization is to cover the input space of samples with codebook vectors. A codebook vector can be seen as a hidden (Kohonen) neuron or a weight vector of the weights between all input neurons and the regarded Kohonen neuron respectively [12].

The learning consists in modifying weights in accordance with adapting rules and, therefore, changing the position of a code vector in the input space. Many methods of training of the LVQ are exists [13]. We used the competitive training with one winner. This method can be represents as follows:

1. Weighting coefficients of the neural network in diapason [0,1] are randomly generated.
2. An initial value $t = 1$ of a point of time are defined.
3. For all input patterns x^l, $l = \overline{1,L}$ (where L is a total amount of code vectors) are sequentially calculated:
   a. a norm of vector is $D_j^l = \left| X^l - W_j \right|$, where $j = \overline{1,m}$;
   b. a winner neuron, which provide for minimal space, is determined $D_k^l = \min_j D_j^L$;
   c. the weighting coefficients of the neural network are modified
   $$w_{ij}(t+1) = w_{ij}(t) + \gamma(t) \cdot (x_i - w_{ij}(t)),$$
   when $j = k$ (1)
   $w_{ij}(t+1) = w_{ij}(t)$, when $j \neq k$
   where $i = \overline{1,n}$, $j = \overline{1,m}$.
   A value $\gamma(t)$ characterized training rate in point of time t. It is constant or is decrease with time by the rule: $\gamma(t) = 1/t$.
4. The value of time $t = t + 1$ is changed and process is repeated starting with the step 3.

The training is continued as long as necessary degree matching between input and weight vectors will be getting, or as long as the weighting coefficients are changed.

3.2. THE PROCESS OF DETECTORS GENERATION

Let's examine the process of detectors generation based on the LVQ. First an initial population of detectors is created. Each detector represents one LVQ. Further we will determine a set of self files consisting of different utilities of operating system, various software files etc, and one or a few malicious code (or signature of malicious code). Both self files and malicious virus will be used for LVQ learning. It is necessary to be sure that files from the set of self's are noninfected (without malicious code). Presence of malicious code or its signature in a learning sample allows a mature detector to tell the difference between self and nonself. Of course the more there are diverse files in the learning sample the more structurally different detectors are got. It is desirable to have all kind of malicious cod (worms, Trojans, file infectors etc.) in the learning sample. However, it is not compulsory condition. As stated above there are differences between malicious software and noninfected files, which influence on the decision of a mature detector. Using difference between malware and software we can detect computer viruses. Owing to generalizing ability of neural network the immune detectors can find differences between self files (different software) and malware and detect it.

A set of mature LVQ form a population of detectors which circulate into the computer system. In process of checking of a file the LVQ identifies unknown pattern and determines its proximity to one or another sample vector. Depending on this the LVQ takes a decision about the nature of files – self or malicious code.

4. Description of Experimental Model of the AIS Security System

We used next structure of the LVQ for detectors formation – 128 neurons of the input layer, 10 neurons of the hidden layer and 2 neurons of the output layer (such detector is illustrated in Fig. 3, where $n = 128$, $m = 10$). A learning sample for one detector is formed as follows:

- four noninfected files from self's and one malicious code are selected randomly;
- from each selected file in fives fragments (binary string with length equal 128 bits) are randomly chosen. Then these fragments step by step will be inputted to the LVQ.

Competitive learning with one winner is used for the LVQ training. It is learning by instruction that is we indicate during training to the neural network where data from noninfected files is and where data from malicious code is. As a result of learning we get 10 code vectors in the hidden layer and they correspond with two output classes. The first class consists from 8 code vectors (noninfected files). The second class consists 2 code vectors (malicious code).

As a result we will have a set of structurally different mature detectors since a random process for files selecting is used for detectors learning. These detectors will be used for file identifications and decision making – is it self file or malicious virus? Experimental results in the next section are described.

An immature detector any input pattern (independently of malicious code or noninfected file) to compares the first class (noninfected files) with probability 80% and to the second class (malicious code) with probability 20% since we divide the input space of samples in proportions 8 to 2 (see above). A mature detector (after the LVQ learning) will correlate an input pattern from a noninfected file with the first class with an expectancy of hitting more then 80%. Accordingly, the mature detector will correlate an input pattern from malicious code with the second class with expectancy of hitting more then 20%. The detector divides the under test file into pieces of 128 bytes

apiece, examines them for malicious code in series and calculates total expectancy of hitting in one or another class:

$$P = X/N * 100\%, \quad (2)$$

where X is a number of pieces running in one of a class, N is a total number of pieces of an under test file.

Let's review an example:

The file diskcopy.com (utility of operation system): file size is 7168 byte – 56 pieces of 128 bytes. A detector correlated 49 pieces with the first class (self) that was $P_S = 49/56 \cdot 100\% = 87{,}5\%$ expectancy of hitting. Accordingly an expectancy of hitting in the second class (malicious code) was $P_M = 7/56 \cdot 100\% = 12{,}5\%$. Detector's decision was noninfected file.

5. EXPERIMENTAL RESULTS

We used Matlab version 6.5 and its neural network toolbox for detectors creation. Our detectors were tested on the next scheme: initially mature detectors examined noninfected selected at random utilities of operation system. Then detectors examined "wild" malicious code, i.e. viruses really had spread all over the world and had damaged computer systems. The results of noninfected files testing are presented in Table 1. Table 2 represents the results of malicious code testing. In these tables P_S is expectancy that the under test file is noninfected and P_M is expectancy that the under test file is malicious code. Malicious code classification is according to Kaspersky antivirus software.

Table 1. The Results of Noninfected Files Testing

File name	Detector 1 P_S / P_M	Detector 2 P_S / P_M	Detector 3 P_S / P_M	Detector 4 P_S / P_M
Cacls.exe	0,78 / **0,22**	**0,93** / 0,07	**0,89** / 0,11	**0,82** / 0,18
ctfmon.exe	**0,81** / 0,19	**0,86** / 0,14	**0,87** / 0,13	**0,89** / 0,11
dbexplor.exe	**0,90** / 0,10	**0,93** / 0,07	**0,94** / 0,06	**0,90** / 0,10
dcomcnfg.exe	**0,91** / 0,09	**0,96** / 0,04	**0,96** / 0,04	**0,96** / 0,04
diskcopy.com	**0,83** / 0,17	**0,93** / 0,07	**0,92** / 0,08	**0,83** / 0,17
dllhost.exe	**0,89** / 0,11	**0,96** / 0,04	**0,98** / 0,02	**0,85** / 0,15
etm70.exe	**0,91** / 0,09	**0,94** / 0,06	**0,95** / 0,05	**0,87** / 0,13
notepad.exe	**0,84 /** 0,16	**0,91 /** 0,09	**0,92** / 0,08	**0,83 /** 0,17
soundman.exe	**0,87** / 0,13	**0,93** / 0,07	**0,94** / 0,06	**0,93** / 0,07
taskman.exe	**0,88** / 0,12	**0,92** / 0,08	**0,95** / 0,05	**0,92** / 0,08
uninlib.exe	0,58 / **0,43**	**0,81** / 0,19	**0,83** / 0,17	**0,82** / 0,18

For training of the first detector we used except utilities the malicious code *Email-Worm.Win32.Mydoom*. As can be seen this detector detected email worms and file infectors (*Gpcode, Hidrag*) very well. However this detector classified *uninlib.exe* and *cacls.exe* as malicious code that are misoperation (false detection). Such detector is undesirable and should be destroyed during selection phase. For training of next three detectors we used different malicious viruses therefore they detected different malicious code.

As can be seen from Table 2 some malicious code (*E-Worm.Mydoom, Virus.Gpcode*, *Virus.Hidrag*) are detected very well and some of them (*Backdoor.Agent, Trojan.Daemonize, Exploit.DebPloit*) are detected with difficulty or not detected. Partly it is because they don't bring damage (*Backdoor.Agent*). Also it is necessary to take into account that we presented results only for four detectors. Generation of a large quantity of different detectors allows to decide such problems. Table 2 also shows capability of one detector for several malicious viruses detection.

The next test represents a comparative analysis of computer viruses detection for different antivirus software and the AIS model. For this test next antivirus products were chosen: Kaspersky antivirus ver. 5 with actual antivirus bases, Kaspersky antivirus ver. 5 with outdated antivirus bases, NOD32 only heuristic analyzer is used and the AIS. The purpose of this test is to show the weakness of signature analysis method and imperfection of heuristic analyzers.

Table 2. The Results of Malicious Code Testing

File name	Detector 1 P_S / P_M	Detector 2 P_S / P_M	Detector 3 P_S / P_M	Detector 4 P_S / P_M
Backdoor.Agent	**0,98** / 0,02	**0,98** / 0,02	**0,98** / 0,02	**0,96** / 0,04
Backdoor.Agobot	**0,91** / 0,09	0,58 / **0,42**	0,68 / **0,32**	**0,83** / 0,17
E-Worm.Bozori	0,64 / **0,36**	0,73 / **0,27**	0,55 / **0,45**	**0,85** / 0,15
E-Worm.Zafi	0,70 / **0,30**	0,58 / **0,42**	0,68 / **0,32**	**0,87** / 0,13
E-Worm.Mydoom	0,67 / **0,13**	0,65 / **0,35**	0,65 / **0,35**	0,79 / **0,21**
E-Worm.NetSky	0,61 / **0,39**	0,68 / **0,32**	0,57 / **0,43**	0,80 / 0,20
Exploit.DebPloit	**0,85** / 0,15	**0,92** / 0,08	**0,92** / 0,08	**0,92** / 0,08
N-Worm.Lovesan	**0,83** / 0,17	**0,81** / 0,19	0,77 / **0,23**	0,71 / **0,29**
Net-Worm.Mytob	**0,84 /** 0,16	0,55 **/ 0,45**	0,63 / **0,37**	0,74 **/ 0,26**
Trojan.Bagle	**0,81** / 0,19	**0,85** / 0,15	0,78 / **0,22**	0,68 / **0,32**
Trojan.Daemoniz	**0,93** / 0,07	**0,84** / 0,16	**0,84** / 0,16	**0,84** / 0,16
Trojan.LdPinch	**0,89** / 0,11	0,60 / **0,40**	0,76 / **0,24**	**0,81** / 0,19
Virus.Gpcode	0,73 / **0,27**	0,54 / **0,46**	0,64 / **0,36**	0,58 / **0,42**
Virus.Hidrag	0,79 / **0,21**	0,76 / **0,24**	0,75 / **0,25**	0,77 / **0,23**

Table 3. Comparative Analysis of Different Antivirus Software

File name	Kaspersky antivirus (actual bases)	Kaspersky antivirus (outdated bases)	NOD32 (heuristic analyzer)	AIS (four detectors)
Backdoor.Agent.lw	Backdoor	OK	OK	OK
Backdoor.Agobot	Backdoor	Backdoor	Agobot	Virus
Email-Worm.Maddas	Email-Worm	Email-Worm	OK	Virus
Email-Worm.Gigger	Email-Worm	Email-Worm	OK	Virus
Email-Worm.Loding	Email-Worm	Email-Worm	OK	Virus
Email-Worm.Zafi.d	Email-Worm	OK	Zafi	Virus
Net-Worm.Bozori.a	Net-Worm	OK	Bozori	Virus
Net-Worm.Mytob.a	Net-Worm	OK	Mytob	Virus
Trojan.Psyme.y	Trojan	OK	OK	Virus
Trojan.Bagle	Trojan	OK	Bagle	Virus
Trojan.Agent	Trojan	Trojan	OK	Virus
Trojan.Daemonize	Trojan	Trojan	OK	OK
Trojan.Mitglieder	Trojan	Trojan	Trojan	Virus
Trojan.LdPinch	Trojan	Trojan	PSW	Virus
Virus.Gpcode.ac	Virus.Win32	OK	OK	Virus
Exploit.DebPloit	Exploit	OK	OK	OK

As can be seen form Table 3 Kaspersky antivirus with actual bases detected all malicious viruses since their signatures present in antivirus bases. Kaspersky antivirus with outdated bases detected only half of the total amount of malicious viruses. Heuristic analyzer showed insufficiently good result. It detected only seven malicious viruses. Only three viruses stayed undetected in the case of AIS implementation. It is necessary to say that only four detectors were used in our test. Increasing a number of detectors allows detect all presented malicious codes.

6. CONCLUSION

In this paper we examined the AIS approach for malicious code detection. The AIS is able to discern between noninfected file of operation system and malicious code. The feature of the AIS consists in capability for unknown malicious viruses detection. Application of the ANN for detectors generation allows us to create the powerful detectors. Undesirable detectors are destroyed during the selection process which allows avoiding false detection appearance. Uniqueness of detectors consists in capability to detect several malicious viruses. That is detector can detect viruses analogous with that malicious code on which training are realized. In that way we significant increased probability of unknown malicious code detection. As experiments show it is necessary to large population

of detectors creation. Presence of random probability by detectors generation enables to create different detectors. However it is significant that detectors ability depends on files on which they are trained. It is desirable for training process a various noninfected files and all types of malicious code to have. If your computer system with outdated antivirus bases can be unprotected in the face of new malicious code attack then the AIS gives you a high probability detect it. Applying of the AIS for malicious code detection will expand the potentialities of existing antivirus software and will increase level of computer systems security.

7. REFERENCES

[1] Traditional antivirus solutions – are they effective against today's threats? – *http://www.viruslist.com*, 2008.

[2] Proactive protection: a panacea for Viruses? – *http://www.viruslist.com*, 2008.

[3] Janeway, C.A. How the Immune System Recognizers Invaders. *Scientific American.* Vol. 269, no. 3, pp. 72–79, 1993.

[4] Kuby, J. Immunology. In Abadi, M., Ito, T. (eds.): Theoretical Aspects of Computer Software. *Lecture Notes in Computer Science, Vol. 1281.* Springer-Verlag, Berlin Heidelberg New York, pp. 415–438, 1997.

[5] Jerne, N. K., Clonal Selection in a Lymphocyte Network. *Raven Press*, pp. 39–48, 1974.

[6] Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd edn. Springer-Verlag, Berlin Heidelberg New York, 1996.

[7] Forest S., Perelson A., Allen L., Cherukuri R. Self-Nonself Discrimination in a Computer. *Proceedings IEEE Symposium on Research in Security and Privacy.* IEEE Computer Society Press, pp. 202–212, 1994.

[8] Balthrop J., Esponda F., Forrest S., Glickman M. Coverage and Generalization in an Artificial Immune System. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO).* Morgan Kaufmann Publishers, pp. 3–10, 2002.

[9] S. Hofmeyr and S. Forrest. Architecture for an artificial immune system. *Evolutionary Computation*, vol. 8, no. 4, pp. 443–473, 2000.

[10] de Castro L. N., Timmis J.I. *Artificial Immune Systems: A New Computational Intelligence Approach.* Springer-Verlag, 2002.

[11] Kohonen T. Self-organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics*, no 43, pp. 59–69, 1982.

[12] Hagan M.T., Demuth H.B., Beale M.H. *Neural Network Design.* 1st edn. PWS Pub. Co., 1995.

[13] Kung S.Y., Mak M.V., Lin S.H. *Biometric Authentication: A Machine Learning Approach.* Prentice Hall, 2004.

Prof. Vladimir Golovko, *head of Intelligence Information Technologies Department and Laboratory of Artificial Neural Networks of the Brest State Technical University, Belarus.*

Research interests: artificial intelligence, neural networks, artificial immune systems, autonomous learning robot, signal processing, chaotic processes, intrusion and epilepsy detection, information security.

Sergei Bezobrazov, *assistant of Intelligence Information Technologies Department and Laboratory of Artificial Neural Networks of the Brest State Technical University, Belarus.*

Research interests: artificial intelligence, neural networks, information security, artificial immune systems, data-mining.