The application of artificial immune system to solve recognition problems

I F Astachova, A E Zolotukhin, E Yu Kurklinskaya, N V Belyaeva
Department of Applied Mathematics, Informatics and Mechanics, Voronezh State University, 1, University Square, Voronezh, 394018, Russia
E-mail: astachova@list.ru

Abstract. The present paper is devoted to the creation of a model and algorithm of an artificial immune system (AIS). The proposed model and algorithm are designed to solve the problems of single symbols recognition, symbolic regression, managing autonomous systems (robot). The taxonomy of the proposed artificial immune system model and algorithm of research methodology is described in terms of the natural immune system. The adopted concept of B-cells are responsible for producing antibodies, a special class of complex proteins on the surface of B-lymphocytes, capable of binding to certain types of molecules (antigens) is used. The full set of antigenic receptors of all B-cells is a plurality of antibodies that can be produced by the organism. The developed program complex allows to solve all three problems: single symbols recognition, symbolic regression, managing autonomous systems (robot). Accuracy of calculations is compared to that for other methods, i.e. methods of genetic programming. Computing experiments for each considered task are made.

1. Introduction
Genetic programming [1] is the oldest method of automatic program creation by means of genetic algorithm, developed by American computer scientist J. R. Koza. This method is based on computer language LISP, which is able to manipulate symbolic expressions. During existence of GP the numerous processes such as: data fitting, logical expressions synthesis, robot trajectory optimization, synthesis of a program for artificial ant movement, system identification, etc. were realized.

The description of the first artificial immune system models was presented in the work of Farmer, Packard, and Perelson [1]. It is the most recent concept that has been introduced to informatics from biology. The very basis of the artificial immune systems was created only in the middle of the 1990s. In 1994 Kepkhart [2] described the negative selection algorithm. The first collective monograph on artificial immune systems was published in 1999 [3]. Later in 2000s some other models of artificial immune systems appeared, however the majority of them have been dealing with the concept of only one type of cells, namely, B-lymphocytes, without taking into consideration such features of immune systems as distribution and decentralization.

The natural immune system is a complex system consisting of several parts with different functionality [4]. The immune system uses multilevel protection against external hostile antigens through nonspecific (congenital) and specific (acquired) protective mechanisms. The main role of the immune system is to classify the cells and molecules as "friends" and "foes". The immune system recognizes a variety of different antigens, from viruses to parasitic worms, and...
distinguishes them from the genuine cells. Recognition of the antigens is complicated by their adaptation and evolutionary development of new methods of successful infecting of the host’s organism. When the foreign cell is detected, it is further classified and, depending on the result, the immune system triggers the protective mechanism in order to destroy the foreign molecules.

Let us consider the human immune system, what it consists of and how it functions. The human organism contains a great number of immunocompetent cells, circulating throughout the body. The immune system cells, which play the key role in the acquired immunity, are lymphocytes that are a subtype of leukocytes. The most part of lymphocytes are responsible for the specific acquired immunity since they provide recognizing of the infectious antigens inside and outside the cells, in tissues or in blood. Other cells, phagocytes, are auxiliary cells, are capable of destroying foreign molecules or cells. The lymphocytes are divided into B-lymphocytes and T-lymphocytes. All lymphocytes are formed in the bone marrow, and T-lymphocytes additionally undergo a stage of differentiation in the thymus. Both B- and T-cells contain on their surfaces receptor molecules that recognize specific proteins. The receptors represent a “counterimpression” of a certain part of a foreign molecule capable of joining it. In this one cell may contain receptors of only one antigen type [3, 4].

The books [5, 6] summarize the fundamental concepts of immunology to understand the computational models based on immunology. Some theoretical models of immune processes are presented. The immune system has ability to ”remember” its encounters with antigens to achieve a faster response The immune networks and danger theory are discussed. Some immune models algorithms, based on clonal selection and very closely related to immune networks, are specified. Such models are based on danger theory, cytokine network models and MHC-based models. But these books don’t contain the universal model of artificial immune system. The main purpose is to develop a universal model and algorithm for the functioning of the immune system and apply them to solving recognition problems. A computational experiment should show the applicability of the developed models and algorithms to solve recognition problems by comparing with known results.

2. The model and algorithm of artificial immune system

We suggest the mathematical expression of artificial immune systems, which could be presented as a set of the following elements [7–9]:

\[
AIS = \langle L, G, A, m, S \rangle,
\]

where

- \(AIS\) is the artificial immune system;
- \(L\) is the space of all possible lymphocytes; in so doing a lymphocyte can represent a line, a list of coordinates, or an expression tree;
- \(G\) is a set of all possible antigens; in so doing \(G\) can be a line, a matrix of logical values, and/or a list of values of a function in the certain points;
- \(A : L \times G \rightarrow [0, 1]\) is the given measure of affinity which assigns to each lymphocyte and each anti-gen a certain number from the segment \([0, 1]\) ( this number shows how ”well” this lymphocyte reacts on the given anti-gen);
- \(\mu : L \rightarrow L\) is the mutation operator which is applied to a certain lymphocyte for the improvement of its recognition properties;
- \(S : A \subset L \rightarrow B \subset A \subset L\) is the selection operator leaving the best lymphocytes in the current immune system, supporting the network size.

Then algorithm could be presented as a sequence of the following steps:

**Step 1** is to create initial immune system \(ImSystem \subset L\).

In this step, the given number of admissible lymphocytes for a particular problem is generated randomly to form an initial system.
Step 2 is to get 
\[ g \in G, \forall l \in ImSystem : a_l = A(L,g) . \]

In this step, the affinity (fitness) of all lymphocytes of the current immune system is calculated (using the corresponding antigens approach to all lymphocytes).

Step 3 is to define the best lymphocyte \( l^* = \arg \max(a_l) . \)

Step 4 is to apply the mutation operator \( M = \{\mu(l), l \in ImSystem\} \) to lymphocytes.

The mutation operator can be applied not to all lymphocytes, but to a certain subset (more frequently to those possessing the higher value of affinity). The mutation operator makes equal small changes in the value or structure of the lymphocyte.

Step 5 is to use the selection operator for preserving the network size; the selection operator selects and keeps lymphocytes with the greatest values of affinity (from the current set of lymphocytes and from the set of the mutated lymphocytes obtained in step 4).

\[ ImSystem = S(ImSystem \cup M); \]

Step 6 is to decide whether \( l^* \) matches the given criteria or the maximum number of iterations is achieved. If yes, then go to the exit, otherwise return to Step 2.

Summarizing, it could be noted that the immune system solves a problem of optimization of the function, which represents the affinity. For various technical tasks this model and algorithm would change, depending on a given problem.

3. Model of an artificial immune system for the problem of symbol images recognition

The algorithm of artificial immune systems can be used for the recognition of symbol images, thus the problem of the single symbol recognition is considered first.

Initial data related to this problem is a single symbol image. The input information is a file with an image in supported format (.bmp, .jpg). In order not to limit the set of images, let’s assume that they may be of any color and size. On the other hand, the algorithm perceives an image as the matrix of Boolean values. Thus, we face the problem of image pre-processing, i.e. its scaling and binarization [8, 9].

At first the color image set in the RGB format needs to be converted into the gray scale image (using the procedure described in [8, 9]). This procedure will give us the image with 256 tones of gray. Then it needs to be converted into the two-color image. Binarization of the image represents the process of image transformations consisting of the gradation of one color (gray) to a bitmap, i.e., the image with each pixel can have only two colors (in our case it is black and white colors). The result of such conversions is the desired image representation.

To solve the problem of binarization, let’s use the Otsu method [8], which is the most effective from all the methods of global binarization both by its quality (mistakes up to 30 % or less) and by processing speed. This method adopts the histogram of values distribution of raster image pixel brightness. The histogram is constructed according to the values \( p_i = n_i/N \), where \( N \) is a number of image pixels, and \( n_i \) is a number of pixels with the brightness of \( i \)-level. The brightness range is divided into two classes by means of brightness threshold value of \( k \)-level, where \( k \) is the integer value from 0 to 255. Relative frequencies refer to each class. We will obtain \( \eta \)-threshold, which is a border between two classes, those shades, which are less than a threshold value, will be referred as 0, while the others will be assigned to 1.

Moreover all images to be recognized should be reduced to pre-assigned dimensions. For the solution of this problem, we will use the scaling algorithm based on the Brezenkhem algorithm [9]. Thus, from the image of arbitrary size it is possible to obtain the transformed image of the desired size. First, we need to define the borders of the rectangular area occupied by the symbol image.
itself. Then, if the detected that size of this area does not coincide with the desired size, the scaling operation needs to be performed again.

In the natural immune system the classification takes place through chemical reactions. For the artificial immune system we recommend the following network organization and lymphocytes representation:

**Step 1** — preprocessing of the image. The monochrome image can be presented as $M \times N$ matrix of Boolean values in which 'true' occupies those places which correspond to black pixels and actually format the image.

**Step 2** — representation of the B-lymphocytes concept. A concept of lymphocyte from the natural immune system can be presented in terms of artificial immune system as an array of $P$ couples of numbers. Each of such couples represents the pixel coordinates in the image. However, we will use a bit more complicated conceptual representation of the lymphocyte, namely a lymphocyte in which the information about those pixels that should be white (blank) is still stored. Thus, our lymphocyte contains two lists of coordinates: one of which stores coordinates of black pixels and the other stores coordinates of white pixels. Then number $S$ may be presented as a sum $S = S_1 + S_2$, where $S_1$ is a number of black pixels of the input image (antigens) which are defined by coordinates from the first list of the lymphocyte, and $S_2$ is a number of white pixels of the input image (antigens) which are defined by coordinates from the second list of the lymphocyte. The structural model of the lymphocyte is given in figure 1.

**Step 3** — calculation of the affinity. Let's introduce a certain number $S < P$ in order to describe the lymphocyte reaction to the antigen, as well as the affinity measure to this antigen for each according to the following formula: $f = S/N$, where $N$ is the number of pixels in both lists of the lymphocyte (the number of pixels, which the lymphocyte stores information about).

**Step 4** — composition of the network. To simplify the natural immune system model we suggest considering only B-lymphocytes for classification of each antigen. T-lymphocytes in this artificial system will not be considered.

As a result of all these transformations, we will obtain the matrix of given dimensions, consisting of logical values, representing a recognizable image.

To solve the matching problem of functioning of various algorithms for handwritten symbols recognition, the data from MNIST database (Mixed National Institute of Standards and Technology database) were used [10–15].

Here is the test example aiming on the result accuracy proof. The set of symbols for training consists of 60,000 images of handwritten digits from 0 to 9. The size of each image is 28 by 28 pixels, each character is scaled to this size. The color image has 255 shades of gray, where white is background and black color is the symbol itself.

As a test sample a set of 10,000 characters of the same size is used.

For the solution of this problem the artificial immune system with the following parameters is used:

1) The lymphocyte size is 200 pixels.
2) The threshold value of affinity (at which it is considered that the lymphocyte reacted to

![Figure 1. Structural model of a lymphocyte.](image-url)
the given image) is equal to 0.8.
3) The amount of the training selection is 60000 symbols.
4) The amount of test selection is 10000 symbols.
5) During the training for each fed image 80 lymphocytes reacting to it are generated.

Table 1. Comparative accuracy of various recognition methods.

| Method                                                                 | Percentage of correctly recognized images of the test set |
|-----------------------------------------------------------------------|----------------------------------------------------------|
| Linear classifier (neural network with the 1st level)                 | 12.0                                                    |
| K of the nearest neighbors, Euclidean distance                        | 5.0                                                     |
| Method of principal components and quadratic classifier               | 3.3                                                     |
| Method of reference vectors (Gaussian kernel)                         | 1.4                                                     |
| Two-layer neural network, 300 neurons in the hidden layer             | 4.7                                                     |
| Two-layer neural network, 1000 neurons in the hidden layer             | 4.5                                                     |
| Three-layer neural network, 300 and 100 neurons in the hidden layer respectively | 3.05                                                   |
| Three-layer neural network, 500 and 150 neurons in the hidden layer respectively | 2.95                                                   |
| Six-layer neural network with the number of neurons 784-2500-2000-1500-1000-500-10, parallel version functioning on video accelerators | 0.35                                                   |
| Artificial immune system                                              | 3.04                                                    |

Now, let us calculate the number of operations for each image out of the training set. Change of weights will affect all weights of the network; let us indicate the number of operations for their calculation as $W_{\text{weights}}$, and note that the computational complexity is linear in $W$: $O(W)$. After the using of the artificial immune network with the above-stated parameters to the test set, the algorithm showed the accuracy of 3.04 [6]. Data is taken from works [10–15].

Also, for each neuron in the network its induced field and local gradient will be calculated; let us indicate this number of operations as $W_{\text{neurons}}$.

And, since the training is performed interactively (by epochs), for each sample out of the training set these operations will be performed not once, but several times, until the algorithm reaches the stop criteria. As the stop criteria we use reaching by gradient vector Euclidean norm of sufficiently small values, or a sufficiently small absolute intensity of changes of mean square error during the epoch. What is to be considered as "sufficiently small" shall be determined prior to starting the algorithm. Let us indicate the average number of iterations of the back propagation of error algorithm to one element of the training set as $N_{\text{iter}}$. It can be noticed that $N_{\text{iter}} > 1$ in the general case, since the multilayer turns out that the average number of operations for one training example is: $E_{\text{example}} = N_{\text{iter}} \cdot (W_{\text{weights}} + W_{\text{neurons}})$.

Then the total number of operations can be calculated by formula

$$E = N \cdot E_{\text{example}} = N \cdot N_{\text{iter}} (W_{\text{weights}} + W_{\text{neurons}}).$$

Let us consider the three-layer neural network with 300 and 100 neurons with the hidden layers respectively. Since the example of the rendered image is 28 by 28 pixels, the system has 784 inputs. Thus, we have that
\[ W_{\text{weights}} = 784 \cdot 300 + 300 \cdot 100 + 100 \cdot 10 = 266\,200 \]
\[ W_{\text{neurons}} = 2 \cdot (300 + 100 + 10) = 820 \]
\[ E_{\text{example}} = N_{\text{iter}} \cdot (468\,500 + 1\,320) = 469\,820N_{\text{iter}} \]

In an artificial immune system, the training set depends on two parameters: the number of pixels, information of which is stored by lymphocyte \((N_{\text{pixels}})\), and the number of lymphocytes created for each rendered image out of training set \((N_{\text{lymph}})\). For each pixel the generation of two random numbers is required, so we multiply lymphocyte \((N_{\text{pixels}})\) and training set \((N_{\text{lymph}})\) by 2. Then, the number of operations for one sample out of the training set is equal to:

\[ E_{\text{example}} = 2 \cdot N_{\text{pixels}}N_{\text{lymph}}. \]

For the considered immune system \(N_{\text{pixels}} = 200, N_{\text{lymph}} = 80, E_{\text{example}} = 32\,000\), which is 8 times less than that of the neural network with 300 and 100 neurons and 14 times less than that of the neural network with 500 and 150 neurons. It should also be noted that the artificial immune system does not have a notion of the number of iterations (epochs), unlike the immune networks.

4. Model of an artificial immune system for resolving the problem of symbolic regression

There is a set of values of free variables \(\{x_1, \ldots, x_r\}\), where \(x_i \in \mathbb{R}^n\) and corresponding to them values of function \(\{y_1, \ldots, y_r\}\). These two sets give rise to a set of source data — \(D\).

Also there is a given set of functions, which will be used for the construction of superposition. Let us consider here only continuously differentiable functions: \(g : \mathbb{R}^n \to \mathbb{R}\) such as \(\sin, \cos, \text{Polynomials}\).

We consider any superposition of function \(f\) consisting of \(m\) of functions \(g\). It is required to find such superposition, which would provide maximum or minimum for the functional \(p(f, D)\). This functional defines objective function. For the construction of superposition it is possible to use various metrics and distances, the relative and absolute error — the difference of initial values and the values of the resulting function at the same points.

Thus, the input data for the calculation is the set of points of \(n\)-dimensional space and values of function in this point. The solution of this task is the function depending on \(n\) variables, which approximates initial function the best way [16–19].

![Figure 2. The expression tree sample.](image-url)

Since the number of variables is known beforehand, a lymphocyte also stores a list of valid variables.

All lymphocytes should support the following operations:
• Calculation of the value of the given function at a point.
• Calculation of the affinity of the given lymphocyte (discussed below).
• Simplification of the corresponding expression.
• Return of the line representation of expression (with the correctly placed brackets).

**Step 1.** Thus, the function presented by the lymphocyte can be written as [9, 16–19]:

\[ F = f_1(f_2(\ldots f_m(x_1, x_2, \ldots, x_n))), \]

where \( f_1, f_2, \ldots, f_m \) — functions from the given set, \( m \) — number of functions, less than or equal to the maximum allowable height of an expression tree, \( x_i \) — free variables.

**Step 2.** Affinity (objective function). Let us introduce the concept of affinity. In biology affinity of antibodies is the binding strength of the active centers of the antibody molecule with antigen determinant groups. The antigen is a set of input data — values of functions at given points. In the given model affinity characterizes the degree of ”success” of approximation, i.e. represents the value of the objective function. Let us use the following function:

\[ \sqrt{\sum_{i=1}^{r} (f(x_i) - y_i)^2}, \]

where \( \{x_1, \ldots, x_r\}, \ x_i \in R^n \) — a specified set of values of free variables; \( \{y_1, \ldots, y_r\} \) — a specified set of values of the search function at appropriate locations \( x_i \);

\( f(x_i), \ i = 1, \ldots, r \) — multiple values of the function represented by the given lymphocyte in corresponding locations \( x_i \).

• The value of the function at a point would be calculated;
• The measure of affinity would be calculated;
• The corresponding expression would be simplified;
• The representation of expression would be returned with correctly placed brackets.

In immunology, affinity is a measure of how well selected B cell interacts with the antigen. In our model affinity is a measure of how close the function (represented by the lymphocyte) is to solution. We present the affinity function using the Euclidian metrics.

**Step 3.** Immune system or immune network consists of many lymphocytes. Because of decentralized nature of the immune system, artificial immune systems can have many groups of lymphocytes, which can be placed on different computational nodes. These groups of lymphocytes can communicate with the others, share good solutions and maintain a variety of solutions.

**Step 4** is to apply the mutation operator \( M = \{\mu(l), \ l \in ImSystem\} \) to lymphocytes.

The mutation operator may be applied not to all lymphocytes, but to a certain subset (more frequently to those possessing the higher value of affinity). The mutation operator inserts little changes in the value or structure of a lymphocyte.

**Step 5** is to use the selection operator \( ImSystem = S(ImSystem \cup M) \), which selects and remains lymphocytes with the greatest values of affinity from the current set of lymphocytes and from the set of the mutated lymphocytes obtained on step 4.

**Step 6** is to decide whether \( l^* \) satisfies to the given criterion or whether the maximum number iteration is achieved. If yes, then go to the exit, otherwise return to Step 2.
5. Result accuracy. Test examples, comparison between used model and other existing methods

Let’s compare the work of genetic programming algorithms with the offered the immune system. Let’s consider the functions and the convergence of methods described in [20], where authors show some experimental results of the genetic programming algorithm based on the Trees Addition Grammar (TAG). Trees Addition Grammar is a formal grammar, created by Aravind Joshi. This grammar summarizes the context-free grammar in such a way that the elementary unit in the rules of inference becomes not the individual characters, but trees. Thus, the grammar defines the rules of tree nodes replacement to subtrees. In their article authors compared the results of the new algorithm (TAG3P) with the results of a simple algorithm of genetic programming (GP) and genetic programming algorithm based on grammars (GGGP).

To conduct a test authors use the following functions:

\[ x^2 + x; \]
\[ x^3 + x^2 + x; \]
\[ x^4 + x^3 + x^2 + x; \]
\[ \cos(2x). \]

For genetic programming algorithms [20] the authors use the following settings:

1) Terminal expression of the used expression trees: \( x \) — independent variable for the searched function — polynomial, \( x \), and real numbers for the function — the cosine.

2) Supported operators: binary \{+, -, *, /\}; unary: \{\sin, \cos, \exp, \log\} polynomials for functions, \{\sin\}, for the function — the cosine.

3) Initial data: the values of 20 randomly selected points in the interval \([-1, 1]\) for polynomials and 20 randomly selected points in the interval \([0, 2]\) for the cosine.

4) The objective function: the amount of errors in the 20 specified points.

5) Genetic operation: tournament selection, a one-point crossover and replacement of genes — for algorithm TAG3P. Tournament selection, standard crossover and mutation for GP and GGGP algorithms.

6) Parameters of genetic algorithms: the probability of crossover for GP, GGGP and TAG3P equals to 0.9. The probability mutation for GP and GGGP equals to 0.1. The probability of gene replacement algorithm TAG3P equals to 0.01 for the functions of polynomials, and 0.04 for the cosine. The number of individuals participating in the tournament equals 3. Maximum number of generations is 30 for the functions and polynomials, and 200 for the cosine. The population size is 500 individuals.

7) The criterion of the successful work of the algorithm: in all 20 points achieved the accuracy of 0.01.

8) The goal is to find a function of a single variable that fits into the initial data in the best way (calculating values of 20 points).

For each algorithm and for each function the algorithm run 50 times. The obtained results are shown in the table below (table 2) [20].

Now let’s consider the characteristics of the used immune system:

1) Permitted functions: \{+, -, *, /, \sin, \cos\}.

2) Terminal elements: variable \( X \) and the real numbers (unlike to genetic algorithms).

3) The maximum height of the tree: 4 (can be changed to more complex functions).

4) The number of lymphocytes in the system is 250 (2 times less than the number of individuals in the genetic algorithms).

5) Maximum number of steps is 150 (steps in the immune system consist only of a simple selection and mutation, so their number can be increased).
Table 2. Results of the genetic programming algorithms.

| Function          | GP       | GGGP     | TAG3GP   |
|-------------------|----------|----------|----------|
| $x^2 + x$         | 47 (94%) | 46 (92%) | 50 (100%)|
| $x^3 + x^2 + x$   | 30 (60%) | 32 (64%) | 50 (100%)|
| $x^4 + x^3 + x^2 + x$ | 21 (42%) | 24 (48%) | 48 (96%) |
| $\cos(2x)$       | 0 (0%)   | 10 (20%) | 18 (36%) |

6) The criterion that a solution is found (stopping criterion) is the accuracy 0.01. In contrast to the criteria of successful work of genetic programming algorithms, in the immune system successfully stop means that the objective function has a value of less than 0.01.

Let’s recall that as an objective function the following formula is used:

$$\sqrt{\sum_{i=1}^{r} (f(x_i) - y_i)^2},$$

where $x_i \in [-5, 5]$ — free variable; $\{y_i, i = 1, r\}$ — specified set of values of the search function at the corresponding points $x_i$ ($r$ = amount of points); $f(x_i), i = 1, r$ — multiple values of function represented by this lymphocyte at the corresponding points $x_i$.

As a result we obtained that the immune system uses a more stringent criterion for the successful solution of the problem, so we extend the segment, from which the values of the independent variable are taken, to $[-5, 5]$. On the segment with small values the immune system, already at the first steps, finds a reasonably well approximating function given with an accuracy of less than 0.5, so we extend the segment.

In conclusion we found that artificial immune system in all cases was able to solve the problem better than the 2 algorithm of genetic programming (GP and GGGP), showing comparable results with TAG3GP algorithm — better for the cosine, and a little worse for polynomials.

Based on the examples described above and comparison with existing methods one can makes a conclusion about the competitiveness of artificial immune systems application in solving the problem of symbolic regression.

6. Model of an artificial immune system for resolving the problem of managing autonomous systems

The method of artificial immune systems is applied to the problem of managing autonomous systems (robots) [21]. The main goal of autonomous robots is to maintain the operability and ability to function in a dynamically changing environment [22, 23].

Let’s consider the problem statement, the constraints, and then describe the artificial environment.

We will use the problem statement described in [24, 25], according to which mobile robot makes decisions by applying the algorithm of an artificial immune system. Plan of it’s actions in a dynamic environment is the solution of this problem [22].

The requirements for the environment are: 1) the presence of obstacles; 2) the availability of a base for recharging the robot; 3) the existence of targets — “garbage”; 4) the medium is a map of finite size.
Requirements for the robot can be follows:
1) the robot spends energy during its movement on the environment;
2) the robot must identify obstacles and avoid them;
3) the robot determines the moment to return to the base for recharging;
4) the robot has a goal — to collect garbage.

The main properties of artificial immune systems allow us to anticipate behavior in an unfamiliar situation, based on the experience of past contacts. A compact immune memory with the property of "forgetfulness" allows to realize rapid adaptation to the new environment.

An artificial immune system is advisable to use in the following robot blocks: 1) the decision block on the return to the base for recharging; 2) the block for determining the obstacle and the choice of the point to its displacement.

The artificial environment is a virtual map in which the robot is placed. A map is a rectangle with obstacles and objects.

There are different types of objects on the map: obstacles, base, goals, the robot.

An obstacle is a section of the map on which the robot can not be located. The concept "obstacles" can describe a sufficiently large set of objects. In the room there are a table, a chair, a wardrobe, a bed and similar interior items. The projections of these objects on the floor are perceived as obstacles.

Visually obstacles differ from free areas in color, obstacles are high highlighted in black. It is really possible to describe any space by the representation above. All obstacles are static and do not move during the experiment, that is, the main part of the considered environment is not dynamic.

The artificial environment has a certain base, on which the robot can replenish its energy supply. The obstacles do not move during the experiment, and can not also be displayed anywhere on the map.

Visually the base is blue.

The base is the starting point for the robot: the robot starts its movement while being on the base.

The purpose is to collect garbage by the robot. The target appears unexpectedly in any free place on the map. The number of goals that can appear on the map at one time is limited only by the amount of free space on the map.

The current battery level of the robot can be calculated using the following formula [5]:

\[ E(t) = E(t - 1) - E_m - k_1 E'_m - k_2 E_c; \]

\[ k_1 = \begin{cases} 
1, & \text{as robot has garbage} \\
0, & \text{else} 
\end{cases} \]

\[ k_2 = \begin{cases} 
1, & \text{as robot has collision with the obstacle} \\
0, & \text{else,} 
\end{cases} \]

where \( E(t) \) (energy units) is battery charge at time \( t \), \( E_m \) is quantity unit of energy, which robot expends during a moment of time, \( E'_m \) is quantity unit of energy, which it expends on garbage transfer, \( E_c \) is quantity unit of energy, which it expends while collision with obstacles.

Two modules can be responsible for the logic and decision making by robot: the decision module of further movement; the decision module of the return to the base for recharging.

The most important thing for a robot is not to discharge, it returns to the base through the shortest path in case of any threat. If the charge is normal, the robot collects garbage and simply walks on the map, but it rapidly returns in case of any threat.

To make a decision, the robot needs to analyze a set of characteristics: 1) current energy level; 2) the path made; 3) distance to the base in a straight line.
The data set of these characteristics makes it possible to take a decision on the expediency to apply the return-to-base algorithm and replenish the energy charge.

The module preserves the previous situations and makes a choice based on the saved experience in which the decision was made. The best tool to achieve these aims is the immune memory.

Each lymphocyte (figure 3) stores information about the characteristics. The lymphocyte determines the corresponding response of the system, which takes a positive or negative value to answer to the question: "Do I need to return to the base or not?"

![Figure 3. Structure of the lymphocyte for the decision module of the return to the base.](image)

Let’s consider the decision module about the further movement.

Let’s define the classes to solve the problem:
1) the class of transition to the left cell;
2) the class of transition to the left upper cell;
3) the class of transition to the upper cell;
4) the class of transition to the upper right cell;
5) class of transition to the right cell;
6) the class of transition to the right lower cell;
7) the class of transition to the lower cell;
8) the class of transition to the left lower cell.

We take, as the structure of the lymphocyte: 1) the current position of the robot; 2) the level of stimulation of the nodes.

The immune memory must preserve successful transitions in various situations, the system remembers the location of obstacles and allows robot to determine where it should not move.

The system considers the current readings and compares them with the samples in the immune memory in order to determine the position of the obstacle.

Affinity in both cases will be the level of stimulation. During initialization, all lymphocytes by default have the same level of stimulation, equal to the pre-set number. The level of stimulation of lymphocytes increases to ensure a timely return to the base and bypassing obstacles, the likelihood of being selected further position increases.

7. A program complex for realization of the offered algorithms
If we look at the structure of a program complex more in detail [26, 27], we’ll see that it involves the subsystem for solving the problem of recognition of single symbols, this subsystem uses programming language C# and consists from the following modules:

1. The recognition module, which comprises classes of lymphocytes, artificial immune system, and affinity calculation methods.
2. The pre-processing module, which realizes algorithms for binarization of the image and scaling, and other functions for solving other problems.
3. The module of preservation and loading of already trained artificial immune system from the file.
4. The module of the user’s graphic interface.

The developed program complex allows to solve all three problems of single symbols recognition, of symbolic regression, of managing autonomous systems (robot).
8. Conclusion

The model and algorithm of an artificial immune system are considered in this paper. These models conclude the presentation of the lymphocyte, affinity measure. This model and algorithm were applied to solve three problems of single symbols recognition, of symbolic regression, of managing autonomous systems (robot). The program product involving the algorithms described above developed, and computing experiments for each considered task are made. The developed program complex allows to solve all three problems of single symbols recognition, of symbolic regression, of managing autonomous systems (robot).

References

[1] Farmer J D, Packard N and Perelson A 1995 Physica D 2 187–204
[2] Kephart J O 1994 Proceedings of Artificial Life IV: The Fourth International Workshop on the Synthesis and Simulation of Living Systems 130–139
[3] Petrov P V 1987 Immunologia (Moscow: Medicina)
[4] Harty J A, Twinnereim A and White D 2000 Annual Review of Immunology 18 275–308
[5] Dasgupta D and Nino F 1958 Immunological Computation. Theory and Applications (Taylor & Francis Group)
[6] Dasgupta D, Yua S and Nino F 2011 Applied Soft Computing 11 1574–1587
[7] Astakhova I F, Ushakov S A and Hitskova Y V 2015 Proceeding of the 15-th International Conference on Evolutionary computing (EC’15), Proceeding of the 6-th European Conference of Computer Science (ECC S’15) (Rome, Italy) pp 127–131
[8] Otsu N A 1979 IEEE Transactions on Pattern Analysis and Machine Intelligence 9 62–66
[9] Rodzhers D D 1989 Algorithmic basis for graphics (Moscow: Mir)
[10] LeCun Y, Cortes C C and Burges J C The mnist database of handwritten digits URL http://yann.lecun.com/exdb/mnist/
[11] Ciresan D C, Meier U, Gambardella L M and Schmidhuber J 2010 Neural Computation 22(12) 3207–3220
[12] Keysers D, Deselaers T, Gollan C and Ney H 2007 IEEE Transactions on Pattern Analysis and Machine Intelligence 29(8) 1422–1435
[13] Kussul E and Baidyk T 2004 Image and Vision Computing 22(12) 971–981
[14] LeCun Y, Bottou L, Bengio Y and Haffner P 1998 Proceedings of the IEEE 86(11) 2278–2324
[15] Zhang B and Srihari S N 2004 IEEE Transactions on Pattern Analysis and Machine Intelligence 26(4) 525–528
[16] Johnson C G, Ryan C, Soule T and Keijzer M 2003 EuroGP 2003. LNCS 2610 345–353
[17] Farmer J D, Packard N and Perelson A 1986 Physica D 2 187–204
[18] Garrett S 2005 Evolutionary Computation 13(2) 145–178
[19] Cutello V and Nicosia G 2002 Lecture Notes in Computer Science, Springer 2527 361–370
[20] Hoai N X 2002 Evolutionary Computation, CEC ‘02. Proceedings of the 2002 Congress 2 1326–1331
[21] Ishiguro A, Kondo T, Watanabe Y, Shirai Y and Uchikawa Y 1997 IEEE International Conference on Systems, Man and Cybernetics 1222–1228
[22] Shrivastava K, Jha S S and Nair S B 2013 Proceedings of Conference on Advances In Robotics 1–7
[23] Khan M T Robust and Autonomous Multi-Robot Cooperation Using an Artificial Immune System (Vancouver: The University Of British Columbia)
[24] Ishiguro A A, Watanabe Y and Kondo T 1997 Proceedings of ISADS 231–237
[25] Pfeifer R 1995 Technical report, AI Lab, University of Zurich 2–17
[26] Roelsen Y 2013 C# Programming Language (Moscow: Vilams)
[27] Makoviy K, Proskurin D, Khitskova Y and Metelkin Y 2017 CEUR Workshop Proceedings — Mathematical Modeling. Information Technology and Nanotechnology 2017 (Samara, Russia: International Conference Information Technology and Nanotechnology) pp 178–183