Application of Artificial Immune Networks in Continuous Function Optimizations

Petar Čisar[1], Sanja Maravić Čisar[2], Brankica Popović[1], Kristijan Kuk[1], Igor Vuković[3]

[1]University of Criminal Investigation and Police Studies, Cara Dušana 196, 11080 Zemun, Serbia

[2]Subotica Tech – College of Applied Sciences, Marka Oreškovića 16, 24000 Subotica, Serbia

[3]Ministry of the Interior of the Republic of Serbia, Kneza Miloša 101, 11000 Belgrade, Serbia

petar.cisar@kpu.edu.rs, sanjam@vts.su.ac.rs, brankica.popovic@kpu.edu.rs, kristijan.kuk@kpu.edu.rs, igor.vukovic@mup.gov.rs

Abstract: This paper deals with the application of artificial immune networks in continuous function optimizations. The performance of the immunological algorithms is analyzed using the Optimization Algorithm Toolkit. It is shown that the CLIGA algorithm has, by far, the fastest convergence and the best score - in terms of the number of required iterations, for the analyzed continuous function. Also, based on the test results, it was concluded, that the lowest total number of iterations for the defined run time was achieved with the opt-IA algorithm, followed by the CLONALG and CLIGA algorithms.

Keywords: artificial immune networks; Optimization Algorithm Toolkit; continuous function optimization; performance

1 Introduction

Optimization is defined as the procedure for determining the best set of acceptable conditions, to achieve a specific objective and is formulated in mathematical terms. Optimization problems arise in a broad area of real-world applications. This paper presents the specificities of artificial immune-based algorithms to solve continuous optimization problems. The problem of continuous optimization includes determining the extremes of a function of one or many real variables, within a value spectrum, with possible constraints.

The main contribution made by this study is the introduction of methods for implementing the Optimization Algorithm Toolkit (OAT) environment, in order to examine the performance of artificial immune networks in continuous function optimizations.

This work is structured as follows: Section 2 in the form of related work offers an outlining the general principle of how this immune algorithm functions and explaining the categorization of artificial immune system (AIS) algorithms. Section 3 describes the problems of continuous function optimization and presents the implemented test functions. This is followed by Section 4, with the focus on performance measuring of artificial immune algorithms in a suitable software environment. Finally, in Section 5, conclusions are drawn and future study inquiries are suggested.

2 Related Work

The passages below give ample background information so as to enable the comprehension of the immunological algorithms.

There are several definitions for artificial immune systems. One of them was formulated by de Castro and Timmis [1]: artificial immune systems are adaptive systems, inspired by theoretical immunology and observed immune functions, principles and models.

In an artificial immune network (system) a set of integral components called B cells (B lymphocytes, binary-encoded candidate solutions), interact with each other and go through certain cloning and mutation operations. Similar to artificial neural networks, as shown by Dragulescu and Albu [22], artificial immune networks can learn new information and use previously learned information.

The appropriate theoretical approach, as well as different applications of AIS are elaborated in [14-16].

The general functioning principle of the immune algorithm is outlined in the flowchart [2].

An immune algorithm mathematically models the immune diversity, network theory and clonal selection as a multi-modal function optimization problem.

The authors de Castro and von Zuben [3], formulated the functional similarities and differences between the immune system and immune algorithm.

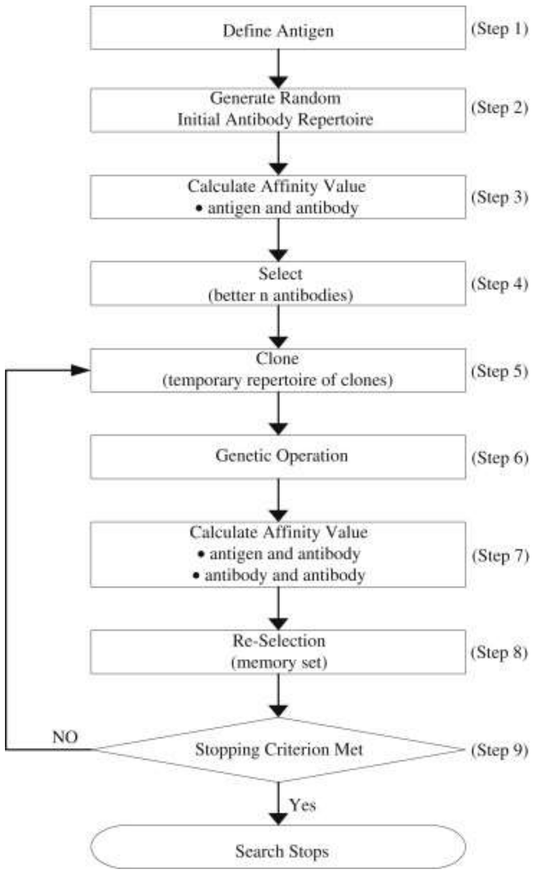

Figure 1
Flowchart of the immune algorithm [2]

Table 1
Immune system vs. immune algorithm

Immune system	**Immune algorithm**
Antigen	Problem to be solved
Antibody	Best solution vector
Recognition of antigen	Identification of the problem
Production of antibody from memory cells	Recalling a past successful solution
Lymphocyte differentiation	Maintenance of good solutions (memory)
T-cell suppression	Elimination of surplus candidate solutions
Proliferation of antibody	Use of genetic operators to create new antibodies

Lopez, Morales and Niño [4] concluded that four major AIS algorithms were under constant development: Negative Selection Algorithms (NSA), Artificial Immune Networks (AINE), Clonal Selection Algorithms (CLONALG) and Dendritic Cell Algorithms (DCA). AIS algorithms can be successfully applied in problems related to clustering, data visualization, control, pattern recognition (intrusion detection [23-27]) as well as, various types of optimizations.

Cutello and Nicosia pointed out in [5] that a simple clonal selection algorithm was named Immunological Algorithm (IA) and later was renamed to Simple Immune Algorithm (SIA). This algorithm analyzes a population of antibodies (B cells) that are exposed to a clonal expansion process. The process involves the cloning of cells with the implementation of a hypermutation parameter.

Table 2

SIA description

Parameter	**Description**
P	Population of antibodies
l	Length of binary string representation
d	Antibody population size
dup	Duplication of the bit string
clone	The number of clones created for each antibody
hypermutation	Modification of a bit string (bit flipping), requires the specification (ρ) of the probability of flipping each bit

The original Immunological Algorithm (IA) was renamed and represented many times by different authors. Other names included Simple Immune Algorithm (SIA), Cloning, Information Gain, Aging (CLIGA), and Optimization Immune Algorithm (opt-IA, opt-IMMALG).

The functioning of SIA can be elaborated by the following pseudocode, as formulated by Brownlee [6]:

```
P <- rand(d, l)
ForEach p of P Do             //presentation
      affinity(p)
EndFor
While Not StopCondition Do
      ForEach p of P Do       //clonal expansion
            Pc <- clone(p, dup)
      EndFor
      ForEach c of Pc Do      //affinity maturation
            hypermutate(c)
      EndFor
      ForEach c of Pc Do      //presentation
            affinity(c)
      EndFor
```

```
P <- select(Pc, P, d)        //clonal selection
EndWhile
```

Listing 1
Simple Immune Algorithm - pseudocode

An AIS combining CLONALG (one of the immune algorithms proposed for pattern recognition and optimization) with the immune network theory resulted in a model named aiNet [17]. This model was successfully applied to data compression and clustering applications, including non-linear separable and high-dimensional problems. The optimization version of aiNet (opt-aiNet) algorithm [12] [13] was applied to several uni/bi - dimensional functions in order to assess its performance. The results illustrated its behavior for some of the problems tested and compared it with results obtained by CLONALG. Three functions were used for testing: the multi-modal function, roots, and Schaffer's function. It was demonstrated that the opt-aiNet located 61 local maximums, while the CLONALG located 18. In addition, the opt-aiNet positions one single individual in each peak, which can overcome the 'waste of resources' disadvantage of the CLONALG.

Ulutas and Kulturel-Konak formulated the general steps of CLONALG [7]:

1) Initialization - randomly initialize a population
2) Evaluation - given a collection of patterns to identify, determine the match of each pattern with each member in the population
3) Selection and cloning - select a few of the best affinity elements and clone (duplicate) them according to their affinity with the antigen
4) Hypermutation - all the clones should be mutated proportional to the affinity with the input sample
5) Editing receptors - add the mutated individuals to the population and reselect a number of the maturated (optimized) individuals as memory
6) Steps 2-5 should be repeated until a stopping criterion is reached

As a representative of AIS algorithms, CLIGA algorithm uses three parameters:

Table 3
CLIGA description

Parameter	Description
d	Antibody population size
dup	Duplication of the bit string
τ_b	B cell's expected mean life (aging operator)

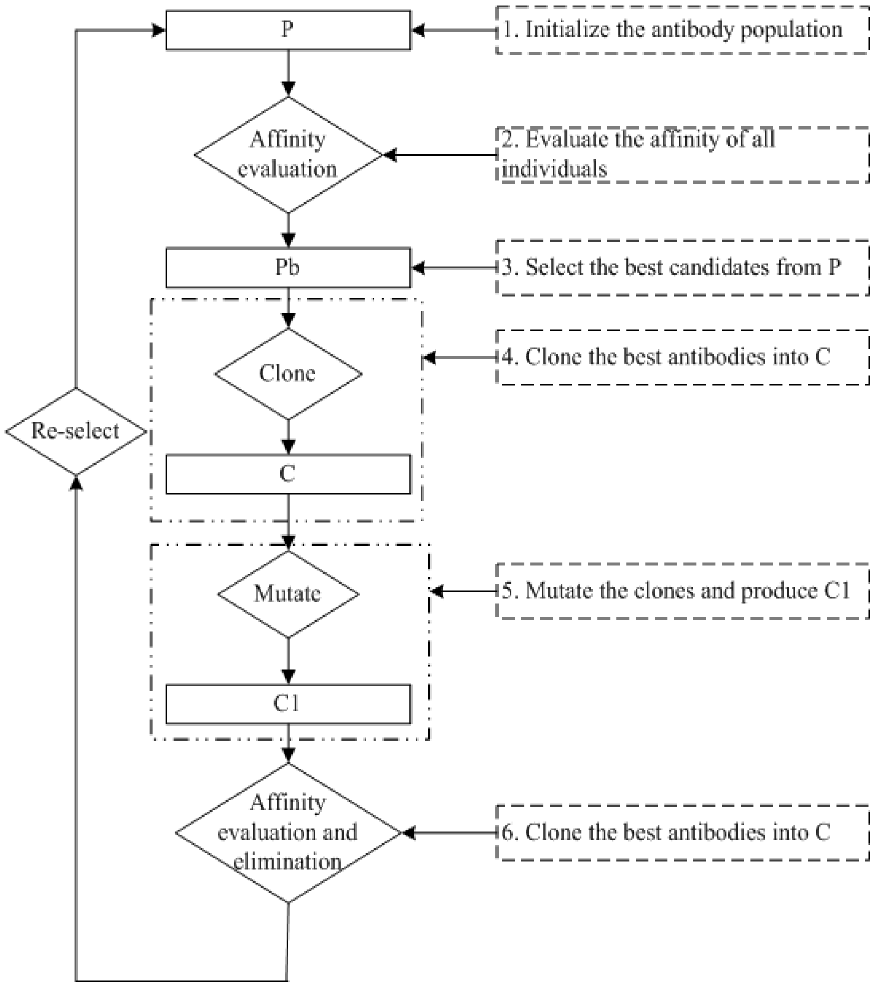
P
1. Initialize the antibody population
Affinity evaluation
2. Evaluate the affinity of all individuals
Pb
3. Select the best candidates from P
Clone
4. Clone the best antibodies into C
Re-select
C
Mutate
5. Mutate the clones and produce C1
C1
Affinity evaluation and elimination
6. Clone the best antibodies into C

Figure 2
Clonal selection method [8]

As its termination condition, CLONALG uses a fixed number of generations, while CLIGA uses the maximum information gain (K) principle - $dK/dt \geq 0$. When $dK/dt = 0$, the learning process ends. For t = 0, the following values are computed: fitness value ($P^{(t)}$), cloning expansion (P^{clo}), variation operator ($P^{hyp} = hypermutation(P^{clo})$), evaluation of fitness value of P^{hyp}, the impact of aging ($P^{(t+1)} = aging(P^{hyp}, P(t), \tau_b)$), information gain ($K(t, t_0)$). At the end, the time is increased ($t = t + 1$) and the previous values are recalculated.

B cells distribution function at time t is $f^{(t),m}$, where m is the fitness value. Information gain can be defined as:

$$\boldsymbol{K(t,t_0) = \sum_m f^{(t),m} \log(f^{(t),m}/f^{0,m})} \quad (1)$$

To evaluate the convergence of algorithms in artificial immune systems, several stopping criteria are used:

- Once a preset number of iterations or function evaluations is reached, the iterative process ceases.
- The iterative process stops as soon as the network attains a preset number of antibodies - convergence of population.
- The average distance between the antibodies and the antigens is examined so as to minimize this value. As soon as the average error is greater than a pre-defined threshold, the iterative process is stopped.
- The network should have converged on condition that its average error increases following *k* consecutive iterations.
- If a defined number of cells do not differ from one network suppression to another, the network is taken to have become stable.
- Once the distance function is inside a specific prescribed distance from the optimum, the algorithm is assumed to have converged.
- The maximum of information gain is reached.

3 Continuous Function Optimization

Continuous optimization is a part of the main mathematical domains for a large number of real-world problems. The problem of continuous optimization includes determining the minimum or maximum value of a function of one or many real variables, subject to constraints (these are, in fact, equations or inequalities). In continuous optimization, the variables in the model can assume any real value within a value spectrum. This feature of the variables is contrary to discrete optimization, where certain variables or all of them may be binary (restricted to the values 0 and 1), integer (for which only integer values are allowed), or more abstract objects drawn from sets with finitely numerous elements [9].

There is a vital difference in continuous optimization between those problems with *no constraints* on the variables and problems where there are *constraints* on the variables. *Unconstrained optimization* problems derive directly from countless practical applications; they further appear in the reformulation of *constrained* optimization problems in which the constraints are replaced by a penalty term in the objective function. *Constrained optimization* problems stem from applications where there are explicit constraints on the variables. There are a great number of subfields of constrained optimization, for which, there are specific algorithms [9].

The applied test functions that belong to the problem domain opt-aiNet are:

- **Multi-function:**

  $\boldsymbol{g(x,y) = x \cdot \sin(4\pi x) - y \cdot \sin(4\pi y + \pi) + 1}$, where x,y ϵ [-2, 2] (2)

- **Roots:**

  $\boldsymbol{g(z) = \frac{1}{1+|z^6-1|}}$, z = x + iy, where x,y ϵ [-2, 2] (3)

- **Schaffer's function:**

  $\boldsymbol{g(z) = 0.5 + \frac{sin^2\left(\sqrt{x^2+y^2}\right)-0.5}{(1+0.001(x^2+y^2))}}$, where x,y ϵ [-10, 10] (4)

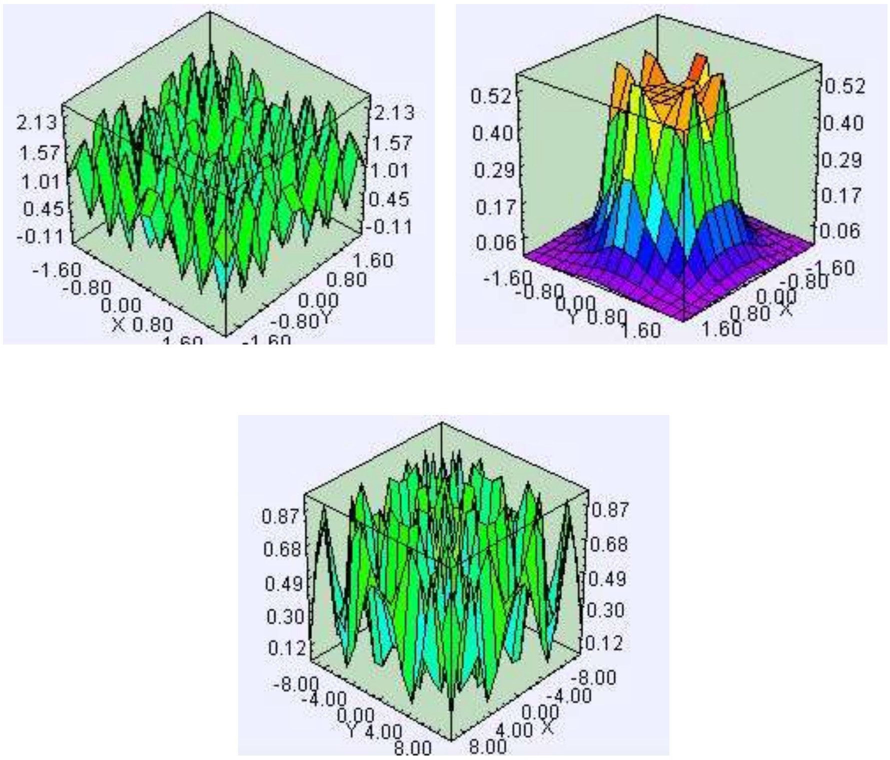

Figure 3

Multi-function, roots, and Schaffer's function

The application of AIS principles to solve different optimization problems is described in [18-21].

4 Case Study - Comparative Analysis of Artificial Immune Algorithms

This paper focuses on examining the performance of artificial immune algorithms in solving continuous function optimization problems using the Optimization Algorithm Toolkit (OAT). OAT is a software environment for developing, evaluating, and experimenting with optimization algorithms on standard benchmark problem domains. There are reference algorithm implementations, graphing, visualizations and various options included in this open-source software. OAT offers a functional library that can be used to investigate existing algorithms and problems, as well as apply new problems and algorithms. This library has a practical explorer and experimenter GUI built on top of it, which ensures that the user can comprehend the functionality in the library. The library's aim is to make easier the best practice of algorithm, problem, and experiment design and implementation, as well as software engineering principles. The GUI supplies a non-technical access that can be used to configure and visualize existing techniques on standard benchmark problem instances [10] [20].

The obtained test results of different artificial immune algorithms are displayed in the following table. The choice of the analyzed algorithms and functions is determined by the built-in capabilities of the OAT software. The run time was set to 180 000 ms (3 minutes) for all analyzed algorithms and the stopping conditions were set at a window width of 1000.

Table 4
Test results of continuous function optimization
(best score/ total evaluations/ iterations for the best score)

	Clonal Selection Algorithm (CLONALG)	**Optimization Immune Algorithm (opt-IA)**	**Optimized Artificial Immune Network (opt-aiNET)**
Multi-function	4.253888443317228/ 5 313 667/22 400	4.253888443317227/ 2 587 598/213 400	4.25388772262681/ 68 334 152/691 000
Roots	0.9999999999999996/ 5 736 550/26 100	1.0/2 517 050/109 000	0.99947323205505/ 28 747 191/1 272 250
Schaffer's function	1.0/5 261 051/21 400	1.0/2 444 295/200 400	0.9999999960496306 /118 526 809/348 900

	Simple Immune Algorithm (SIA)	**CLIGA**	**Optimization Immune Algorithm (opt-IMMALG)**
Multi-function	4.253888443317228/ 7 788 757/15 000	4.2535962571955395/ 5 141 229/4	3.12639125091429/ 71 998 500/919 000
Roots	0.9999999999999996/ 7 764 594/12 700	0.9925992949003002/ 4 761 841/4	0.7162300584615976 /92 545 300/922 300
Schaffer's function	0.9902840901224856/ 7 235 321/18 650	0.9999638750655384/ 5 399 009/1	0.9804511919171228 /86 835 900/1 018 700

By analyzing the results above, it can be concluded that the CLIGA algorithm achieved significantly lower number of required iterations (the third value in the table) for the best score, thus showing the fastest convergence. In addition, having in mind the total number of iterations (the second value in the table), the best result was achieved in the case of the opt-IA algorithm, followed by CLIGA and CLONALG. CLIGA algorithm uses generational aging and information gain stopping criteria. The gain is the quantity of information that the system has learned in relation to the initial distribution function (the randomly generated initial population). This algorithm uses a cloning operator modeled by a cloning potential without memory cells and an aging phase, a stochastic elimination process governed by an exponential negative law, and Kullback's entropy to measure the information gain discovered during the learning process [11]. The Optimization Immune Algorithm (opt-IMMALG) in the cases of multi-function and roots test functions did not achieve the same best score as other tested algorithms.

Conclusions

The main goal of this work is the examination of the continuous function optimization capabilities of artificial immune systems. The performance of these systems, in the form of comparative analysis, was determined using OAT software.

The case study, focused on artificial immune algorithms, showed that the CLIGA algorithm had by far the fastest convergence, the best score - in terms of the number of required iterations, for the analyzed functions. This finding highlighted that with this algorithm, the speed of achieving the used stopping criterion (based on information gain) was the highest. The next algorithms were SIA and CLONALG, which used a specified number of generations, as their stopping criterion.

Further, based on the test results, it can be concluded that the lowest total number of iterations for the defined run-time was achieved with the opt-IA algorithm, followed by the CLIGA and CLONALG algorithms.

Considering the obtained findings, the directions of future research will focus on the application of the CLIGA algorithm, in targeted areas of machine learning.

References

[1] L. de Castro, J. Timmis: Artificial Immune Systems: A New Computational Intelligence Approach, Springer, pp. 57-58, ISBN 978-1-85233-594-6, 2002

[2] C. Chu, M. Lin, G. Liu, Y. Sung: Application of immune algorithms on solving minimum-cost problem of water distribution network, Mathematical and Computer Modelling, Volume 48, Issues 11-12, pp. 1888-1900, 2008

[3] L. de Castro, F. von Zuben, Artificial Immune Systems: Part II - A Survey of Applications, Technical Report DCA-RT 02/00, 2000

[4] G. Q. López, L. A. Morales, L. F. Niño: Immunological computation, Chapter 23, https://www.ncbi.nlm.nih.gov/books/NBK459484/

[5] V. Cutello, G. Nicosia: An Immunological Approach to Combinatorial Optimization Problems, Proceedings of the 8th Ibero-American Conference on AI: Advances in Artificial Intelligence, Seville, Spain, pp. 361-370, 2002

[6] J. Brownlee: Clonal Selection Algorithms, CIS Technical Report 070209A, 2007

[7] B. Ulutas, S. Kulturel-Konak: A Review of Clonal Selection Algorithm and its Applications, Artificial Intelligence Review, Springer, 2011

[8] I. Aydin, M. Karakose, E. Akin: Chaotic-based hybrid negative selection algorithm and its applications in fault and anomaly detection, Expert Systems with Applications, Volume 37, Issue 7, pp. 5285-5294, 2010

[9] Neos Guide, https://neos-guide.org/content/continuous-optimization

[10] Optimization Algorithm Toolkit, https://www.onworks.net/software/app-optimization-algorithm-toolkit-oat

[11] L. de Castro, F. von Zuben: Recent Developments in Biologically Inspired Computing, Idea Group Publishing, 2005

[12] L. de Castro, J. Timmis: An artificial immune network for multimodal function optimization, Proceedings of the 2002 Congress on Evolutionary Computation, Honolulu, HI, USA, IEEE Computer Society, 2002, pp. 699-704, ISBN: 0-7803-7282-4

[13] J. Timmis, C. Edmonds: A Comment on opt-AINet: An Immune Network Algorithm for Optimisation, Proceedings, Part I, Genetic and Evolutionary Computation Conference (GECCO 2004), Seattle, WA, USA, Germany: Springer, 2004, pp. 308-317

[14] L. de Castro, F. von Zuben: Artificial Immune Systems: Part I – Basic Theory and Applications, TR – DCA 01/99, 1999

[15] L. de Castro, J. Timmis: Artificial Immune Systems: A New Computational Intelligence Approach, Springer, pp. 57-58, ISBN 1-85233-594-7, 9781852335946, 2002

[16] E. Hart, J. Timmis: Application areas of AIS: The past, the present and the future, Journal of Applied Soft Computing, Vol. 8, No. 1, pp. 191-201, 2008

[17] L. de Castro, F. von Zuben: aiNet: An Artificial Immune Network for Data Analysis, Data Mining: A Heuristic Approach, Idea Group Publishing, 2001

[18] L. de Castro, F. von Zuben: Learning and optimization using the clonal selection principle, IEEE Transactions on Evolutionary Computation, 2002 Jun, 6(3), pp. 239-251, ISSN: 1089-778X

[19] V. Cutello, G. Nicosia, M. Pavone, G. Narzisi: Real Coded Clonal Selection Algorithm for Unconstrained Global Numerical Optimization using a Hybrid Inversely Proportional Hypermutation Operator, 21st Annual ACM

Symposium on Applied Computing (SAC), Dijon, France, 2006, pp. 950-954, ACM

[20] P. Čisar, S. Maravić Čisar, B. Markoski: Implementation of Immunological Algorithms in Solving Optimization Problems, Acta Polytechnica Hungarica, Vol. 11, No. 4, 2014, pp. 225-240

[21] K. Đuretec: Artificial immune systems, Project: Algorithms based on evolutionary computation, University of Zagreb, Faculty of Electrical Engineering and Computing, www.zemris.fer.hr/~golub/ga/studenti/projekt2008/ais/umjetni_imunoloski_sustavi.pdf

[22] D. Dragulescu, A. Albu: Medical Prediction Systems, Acta Polytechnica Hungarica, Vol. 4, No. 3, 2007, pp. 89-240

[23] J. Jiang, F. Zhang, K. Demertzis: Detecting Portable Executable Malware by Binary Code Using an Artificial Evolutionary Fuzzy LSTM Immune System, Security and Communication Networks, Vol. 2021, pp. 1, 2021

[24] B. Bejoy, T. Bijees et al.: Artificial immune system based frameworks and its application in cyber immune system: A comprehensive review, Journal of Critical Reviews, Vol. 7, No. 2, pp. 52-560, 2020

[25] M. Tabatabaefar, M. Miriestahbanati and J.-C. Grégoire: Network intrusion detection through artificial immune system, Systems Conference (SysCon) 2017 Annual IEEE International, pp. 1-6, 2017

[26] R. Pump, V. Ahlers and A. Koschel: State of the art in artificial immune-based intrusion detection systems for smart grids, 2018 Second World Conference on Smart Trends in Systems Security and Sustainability (WorldS4), pp. 119-126, 2018

[27] R. Pump, V. Ahlers and A. Koschel: Evaluating Artificial Immune System Algorithms for Intrusion Detection, 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), 2020, pp. 92-97, doi: 10.1109/WorldS450073.2020.9210342