# Using of Nature Inspired Computing Models for Mobile Robot Control


Viorica Sudacevschi
*Department of Computer Sciences and Systems Engineering*
*Technical University of Moldova*
Chisinau, Republic of Moldova
viorica.sudacevschi@calc.utm.md

Silvia Munteanu
*Department of Computer Sciences and Systems Engineering*
*Technical University of Moldova*
Chisinau, Republic of Moldova
silvia.munteanu@calc.utm.md

Victor Ababii
*Department of Computer Sciences and Systems Engineering*
*Technical University of Moldova*
Chisinau, Republic of Moldova
victor.ababii@calc.utm.md

Viorel Cărbune
*Department of Computer Sciences and Systems Engineering*
*Technical University of Moldova*
Chisinau, Republic of Moldova
viorel.carbune@calc.utm.md

Olesea Borozan
*Department of Computer Sciences and Systems Engineering*
*Technical University of Moldova*
Chisinau, Republic of Moldova
olesea.borozan@ia.utm.md

Iulian Lungu
*Department of Computer Sciences and Systems Engineering*
*Technical University of Moldova*
Chisinau, Republic of Moldova
iulian.lungu@iis.utm.md



***Abstract*— This paper addresses a significant challenge that has emerged over the last few decades: the application of nature-inspired computing to develop advanced control systems in robotics. By drawing on concepts and algorithms derived from biological phenomena, the research seeks to enhance the behavior and performance of mobile robots in diverse environments and operational scenarios. This study focuses on the application of living cell functions and communication models to design reconfigurable control systems that leverage parallel and concurrent data processing. To achieve this, the paper proposes the structure of a computing cell, a Venn diagram of the control system grounded in the membrane computing model, and a functional diagram of the control system. These foundations support the prototyping and deployment of a sensor array for managing the position of mobile robots within their workspace.**



***Keywords—mobile robot, control, nature inspired computing, membrane computing, P-Systems, cell computing, fuzzy logic, neural network, parallel computing, FPGA, development board DE0-nano.***


## I. INTRODUCTION

The membrane computing model, also known as P-Systems, is a computational paradigm inspired by living cell biology that offers significant potential for the design of algorithms for parallel and concurrent computing [1-4]. These models are an extension of DNA computing that offers advantages in modeling discrete, distributed, parallel, pipeline and multiset systems and evolve based on rewritten rules. The membrane computing paradigm was first proposed by Gheorghe Paun in 1998 as an approach to calculus inspired by nature [5]. These systems are based on the concept of processing information using membranes, just as biological organisms' process chemicals through their cell membranes [6-9].

Nature, as an inherently aggressive environment, has provided optimal conditions for the evolution of human understanding and interpretation of biological, physical, and chemical phenomena in nature. As a result of the evolutionary and selection processes, today we are diverse and very complex in nature [10, 11].

Natural systems, from individual organizations to entire ecosystems, are complex products of evolution and interactions among their components. Having an extremely high degree of complexity, they exhibit unpredictable and imprecise behavior, at the same time ensures the survival and adaptability of living organisms and ecosystems within their environment [12, 13].

Natural systems, especially living organisms and the ecosystems in which they live, offer a number of fundamental properties that have inspired numerous applications in various technological fields. Dynamics, flexibility, robustness, self-organization, simplicity of basic objects and decentralization are just some of these essential properties that have been adapted and applied in different technologies [5, 10, 11].

The definition of natural computing is based on the concept of the P-System, which is a computational model inspired by the functionality and structure of biological cells. This model of abstraction of computational processes is based on the interaction of chemicals and cell membranes, representing a way to understand and simulate computational mechanisms in the biological world. At the core of a P-System are membranes, which represent an autonomous functional logical unit, including a set of objects, rules and/or a set of other membranes. The activity of a P-System is interdependent on the environment and the context in which it evolves and can only be analyzed in an environment that provides input data and consumes the results of its processing. An important aspect of P-Systems is their ability to adapt and evolve through dissolution, where its contents migrate to other membranes, or division, where offspring retain all or part of their original properties [14, 15].

Membrane computing systems are a computational model inspired by the functionality and structure of biological cells, which uses membranes and chemical interactions to simulate computational processes. The functional logic of such systems is determined by a set of rules, which define how objects and input conditions are processed and transformed into objects and output conditions. Rules are the basic mechanisms by which information is processed and manipulated within the system. Some rules may take precedence over other rules. In this way, priority cognitive computing models can be developed, in which less dominant rules are activated only under certain specific conditions and contexts [16, 17].

979-8-3503-4929-0/24/$31.00 ©2024 IEEE

The algorithmic complexity of a P-System depends primarily on the set of rules defined for each membrane. These rules govern how input objects and conditions are processed and transformed into output objects and conditions. By defining complex and interdependent rules, a more sophisticated and adaptable operation of the membrane computing system can be achieved. Membrane computing systems can be defined with both asynchronous and synchronous processing, depending on how objects and rules are applied and processed within the system. This flexibility in processing allows the system to be adapted to different types of problems and applications, offering a wide range of possibilities in the design and implementation of membrane computing systems [18-20].

Membrane computing models are a computational paradigm inspired by biology that provides a formal and structural framework for designing and implementing computational and control systems of various complexities. These models provide the description and analysis of computing systems in a detailed and rigorous way, allowing developers to design and implement solutions tailored to the specific needs of different applications. By using membrane computing models, developers can gain a deeper understanding of computing and control systems and design more efficient and adaptable solutions for diverse applications. These models provide a formal and structural framework that facilitates the analysis and development of complex systems and contributes to advancing the field of nature-inspired computing.

At the same time, membrane computing cells can be considered systems with artificial intelligence and cognitive properties, which implement a variety of models and algorithms for data processing, offering innovative and adaptable solutions for various applications. The set of rules can include both mathematical and logical models, as well as models based on neural networks, Fuzzy logic, and evolutionary computation. By using these models, membrane computing cells can perform a wide range of operations and processes, adapting to the specific needs and requirements of the scope, for example: image processing [21]; biology, ecology, robotics or engineering [22, 23]; decision making systems [24, 25]; collaborative systems and Swarm Intelligence [25, 26]; trusted diagnostic systems [27] and others.

It is known that commanding mobile robots with full autonomy is a complex process and requires the involvement of essential hardware and software resources. The technical and technological complexity is determined by the fact that the mobile robot must perform in parallel/concurrent data processing mode a lot of computing operations related to: navigation and location, route planning and obstacle avoidance, interaction with the environment and other objects (mobile robots), etc. The application of traditional single-processor computing systems does not allow achieving the goals of parallel data processing, respectively, the activity of the mobile robot is slower and monotonous. A classic solution can be considered the application of multi-processor systems, where each processor solves one or part of the control problem with the mobile robot. The disadvantage of these systems is the communication between processors and the very high cost, which does not inspire developers of mobile robotic systems to implement them. Another solution is to apply methods of formal description of parallel computing processes, develop autonomous data processing modules and implement them in reconfigurable FPGA architectures. An example of the implementation of the control system of a mobile robot is presented in the papers [28, 29].

In this paper are presented the results of design and implementation based on membrane computing models (models inspired by nature) and reconfigurable FPGA architectures of a control system with a mobile robot, which has capabilities to move in a space limited by walls and static or dynamic obstacles. The research and design process includes: synthesis of the structure of the computing cell, elaboration of the Venn diagram and model for the formal description of the membrane computing system, synthesis of the functional diagram and prototype model with optimal placement of the ultrasonic sensor set for the development of a mobile robot control system.

The objectives of the research carried out in the paper are to demonstrate the capacity of nature-inspired models, especially membrane computing models, to model, organize, structure and simulate parallel computational processes. For implementation, the design of a control system with a mobile robot was selected.

## II. The Structure of Computing Cell

Computing cells present an intelligent computing structure that can evolve over time. The cell's evolution overtime is determined by its ability to perceive and accumulate new knowledge. Figure 1 illustrates the structure of a computing cell and the structure of the living membrane, where: *Environment* – is the activity environment of the cell, and supports actions defined by the state vector *X,* which delivers input data, for the cell's activity, and supports the actions as a result of processing this data; *Input Port* – is the entrance port that filters the data from the activity environment and accepts to enter only those that meet the condition $X \in X[T]$; *Output Port* - is the port that selects the data processed by the cell $Y$ [T] and passes it to the activity environment; *Membrane Region* – is the internal structure of the cell separated from the membrane; $X[T]$ - is the data set accepted by the *Input Port* for processing; *Rules: R1, R2,..., Rn* – is the set of rules applied to process the input data $X[T]$ and transform it into output data $Y[T]$ ; *Knowledge & Properties* – is the set of knowledge specific to the respective cell and its properties;

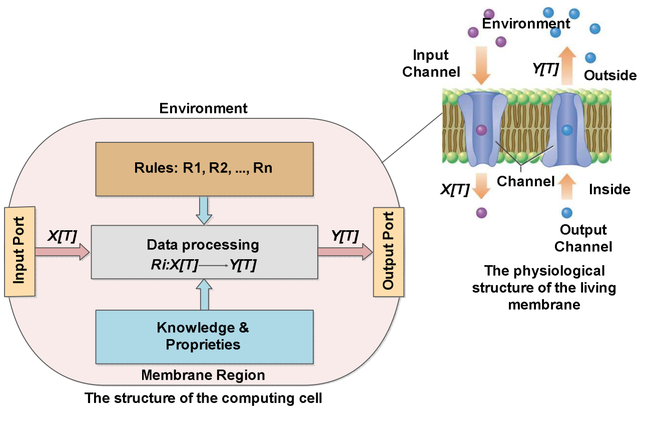


Fig. 1. The structure of computing cell and the physiological structure of living membrane.

*Data processing* - $Ri: X[T] \to Y[T]$ - is the block that applies the set of rules $Ri$ to transform the input data $X[T]$ into output data $Y[T]$; *Outside* – is the external part of the live cell; *Inside* – is the internal part of the live cell; *Channel* – are communication channels between the inside and outside of the live cell; *Input Channel* – are channels associated with the *Input Port*; *Output Channel* – are channels associated with *Output Port* of the computing cell.

## III. The Membrane Computing Model of the Mobile Robot Control System

The membrane computing model (P-Systems) consists of a set of discrete cells with specific functions. The algorithmic complexity achieved by the membrane computing model is determined by its topology and the functions performed by each cell (Figure 1).

Figure 2 shows the Venn diagram of the membrane computing model of the control system with the mobile robot. The Venn diagram models control system components that process data and operate in parallel/concurrent mode and are implemented based on an FPGA circuit. The Venn diagram of the membrane computing model of the mobile robot control system includes: *f=50MHz* – is the clock signal generating cell for synchronizing the calculation processes; *Trigger Generator* – is the cell for calculating the synchronization signal of ultrasonic sensors; *Trig* - is the cell generating the signal for starting the process of calculating the distance to the obstacle; (*Echo 1 – Echo 4*) – there are four cells for recording signals generated by ultrasonic sensors; *DC Motor Control* – is the calculation block for *DC motors* control; *LDO, FDO, BDO, RDO* – are four modules for calculating (*Left, Forward, Right, Back*) the distance to the obstacle, which consist of: *Counter L, F, B, R* - calculates the distance to the obstacle and *D1 / F1 - D4 / F4* - fuzzification blocks that transform the distance to the obstacle code into Fuzzy qualifiers codes; *NNL 1, NNL 2, NNR 1, NNR 2* – are 4 modules of *Neural Networks* that calculate command signals with *DC motors*; *INL1, INL2, INR1, INR2* – are cells generating *PWM* signals for *DC motors* controlling.

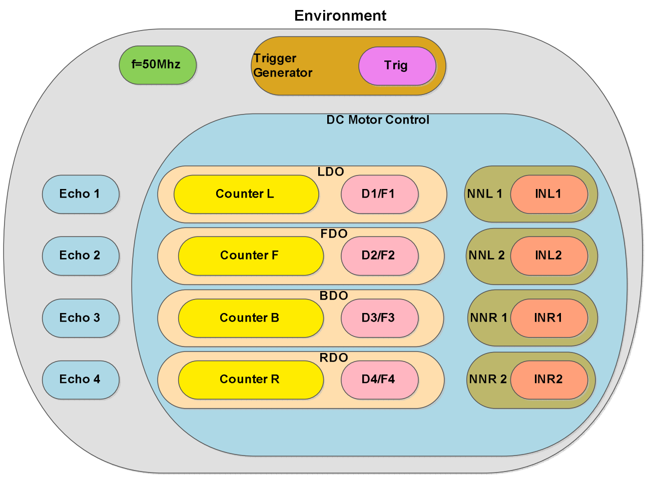


Fig. 2. Membrane Computing Venn diagram of the Mobile Robot Control System.

The formal model of the membrane computing system $\Pi$ is defined by expression (1) [9, 30]:

$$\Pi = \left(V, \mu_1, \ldots, \mu_m, \omega_1, \ldots, \omega_m, R_1, \ldots, R_m, O_j\left(\omega_j\right)\right) \quad (1)$$

where: $V$ is the set of objects (variables) with which the computing cells operate, or the domain of definition of the membrane computing system; $\mu_1, \ldots, \mu_m$ are a set of computing cells, the topological structure between them determines the algorithmic complexity achieved by the membrane computing model; $\omega_j, \forall j = \overline{1,m}$ is the set of objects (variables) that are part of the computing cell $j$, where $\omega_j \subset V$ ; $R_j, \forall j = \overline{1,m}$ are sets of processing/transformation rules of the objects/data associated with the computing cell $j$, where $O_j\left(\omega_j\right), \forall j = \overline{1,m}$ are the set of the computing cells that generate intermediate or final results $\omega_j$ of the membrane computing system $\Pi$.

## IV. Functional Diagram of the Mobile Robot Control System

The functional diagram of the control system with the mobile robot is shown in Figure 3. The multitude of ultrasonic sensors (*Left, Forward, Back, and Right*) assists in calculating the distance from the mobile robot to the obstacle. The Trig signal triggers the process of generating ultrasonic waves, upon their turmoil the sensor generates the *Echo signal (Echo1, Echo2, Echo3, and Echo4)*. The *Control Block* with mobile robot is made based on *FPGA development board DE0-Nano* which ensures the implementation of: *Trigger* signal generator, four *Counters (L, F, B, R)*, four fuzzification elements (*D1/F1, D2/F2, D3/F3, D4/F4*) and four *neural networks (NNL1, NNL2, NNR1, NNR2)* for *DC motor* control signal calculation (*INL1, INL2, INR1, INR2*).

**How it works.** The *50 MHz* generator provide the *Clk* signal that is applied to the timing inputs of the *Trigger* generator and the *Counters (L, F, B, and R)* to estimate the distance to obstacles. The *Trigger* generator issues the *Trig signal* connected in parallel to all ultrasonic sensors. This signal triggers the process of emitting ultrasonic waves by the sensor. Simultaneously with the *Trig signal*, the *RST signal* is generated, which initializes the *Counters (L, F, B, R)* to calculate the distance to the obstacle. The result of reflection by obstacles, ultrasonic waves, are received by the sensor and lead to the generation of the *Echo signals (Echo1, Echo2, Echo3, Echo4)*, connected to the inputs *(En1, En2, En3, En4)* of the *Counters (L, F, B, R)*. These signals stop the process of calculating the distance to the obstacle in that direction. The result of the calculations performed by the *Counters (L, F, B, R)* are transmitted by buses *D1, D2, D3, D4* to the inputs of the fuzzification elements (*D1/F1, D2/F2, D3/F3, D4/F4*) which associates the value of the distance to the obstacle with a linguistic qualifier (*e.g. large, medium, small, or very small*). Fuzzification results are applied to neural network inputs of *NNL1, NNL2, NNR1, and NNR2* that calculate command signal values (*INL1, INL2, and INR1, INR2*). These signals, in order to be plied, are connected to the *Driver Left Engine* and *Driver Right Engine*. The ratio of control signals (*INL1, INL2, and INR1, INR2*) determines the speed and direction of rotation of *DC motors*, respectively, moving the mobile robot in that direction.

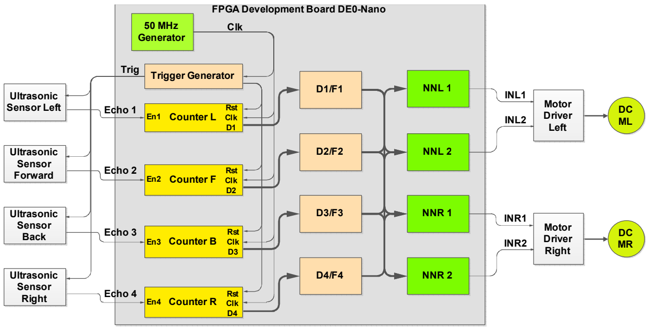


Fig. 3. Functional Diagram of the Mobile Robot Control System.

## V. Mobile Robot Prototype

The design the mobile robot control system based on nature-inspired models (membrane computing) is carried out using the *DE0-Nano development and education board*, which provides the developer with the following resources for implementing and testing projects: Cyclone IV EP4CE22F17C6N FPGA (22,320 LEs, 594 Kb embedded memory, 66 embedded 18x18 multipliers, 4 general-purpose PLLs, 153 FPGS I/O pins); memory devices (32MB SDRAM and 2Kb I2C EEPROM); general user input/output (8 green LEDs, 2 debounced push-button, 4 dip switches); clock system (50MHz); power supply (USB Type mini-AB port 5V, Two DC 5V pins of the GPIO headers) [31].

The Mobile Robot Control System prototype is shown in Figure 4 and includes: four *HC-SR04 Ultrasonic Sensors (01 – 04)* used for measuring distance to obstacle, *Logic Level Converter TXS0108E* intended to connect the logic level of ultrasonic sensors to the logic level of the GPIO pins of the *DE0-Nano Development and Education Board*, *L298N Motor Driver* intended for controlling the speed and direction of rotation of motors *L DC Motor Pololu* and *R DC Motor Pololu* for movement mobile robot, and *DC Power Sources* for power supply to electronic devices and *DC motors*.

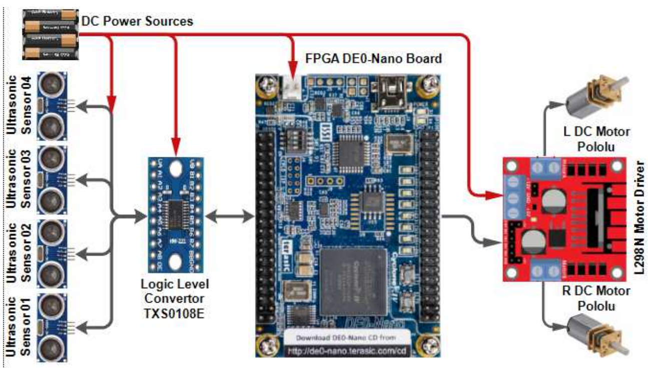


Fig. 4. Mobile Robot Control System Prototype.

The housing for prototyping and the placement of the ultrasonic sensor set is shown in Figure 5, where: *MR Housing* – the housing of the mobile robot for assembly; *Direction of movement* – the basic direction of movement of the mobile robot; *Ultrasonic Sensor Forward* – identifies and ensures the calculation of the distance to obstacles located on the basic direction of movement of the mobile robot; *Ultrasonic Sensor Right* – identifies and ensures the calculation of the distance to the obstacles located on the right side in relation to the direction of movement of the mobile robot; *Ultrasonic Sensor Back* - identifies and ensures the calculation of the distance to obstacles located behind the mobile robot; *Ultrasonic Sensor Left* - identifies and ensures the calculation of the distance to obstacles located on the left side in relation to the direction of movement of the mobile robot; *Right DC Motor* – DC Popolu motor placed on the right side in relation to the direction of travel of the mobile robot; *Left DC Motor* - Popolu DC motor placed on the left side relative to the direction of travel of the mobile robot; *Battery fixing bar* – bar for fixing the battery block for powering the mobile robot; *FPGA fixing kit bar* – bar for fixing *FPGA DE0-Nano Board*.

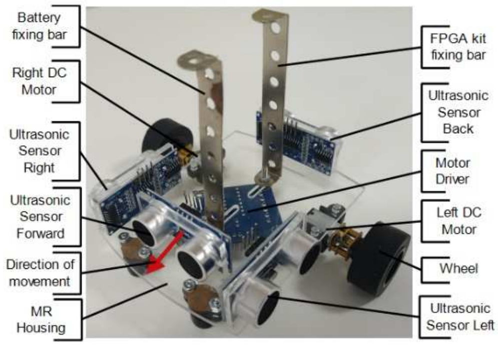


Fig. 5. The Placement Mode of the Ultrasonic Sensor Set.

## VI. Conclusion

Nature-inspired computing models are an essential source in the design and implementation of complex control systems. At first glance, they present simple systems, oriented to perform a small set of operations, but in cooperation with other homogeneous or heterogeneous elements, they can solve very complex problems. These advantages of nature-inspired calculation models have ensured the development of efficient command, control and decision-making systems with application in various fields of the economy: management of complex technological and production processes, decision support systems for management and marketing services, classification systems and image and speech recognition and others.

In this paper it is proposed to design and implement a control system of a mobile robot based on models inspired by nature, especially membrane computing (P-Systems). The structure of the computational cell was proposed, along with a description of its functionality and its resemblance to the structure of living cells.

The topology of the control system for the mobile robot was developed based on the membrane computing model, utilizing insights from the Venn diagram. This process included synthesizing the functional diagram for the control system, which in turn guided the prototyping and placement of the ultrasonic sensor set. The placement of these sensors was strategically designed to minimize interference between ultrasonic waves and to create a navigable space for the mobile robot within its operational environment.

The application of formal description models, such as Venn diagrams to define membrane computing models, and their subsequent implementation in FPGA circuits, ensures the parallel representation and efficient organization of processes involved in calculating distances to obstacles and generating command signals for the DC motors controlling the robot's movement.

However, a limitation of utilizing the membrane computing model in the control system design arises in scenarios with modified spatial dimensions and numerous obstacles, where interference from ultrasonic waves can occur during navigation. This challenge could potentially be addressed by employing a scanning algorithm with a selective order approach.

Looking ahead, future research aims to develop a mathematical model and algorithm for translating the membrane computing model into HDL code, with the possibility of automatic implementation in FPGA circuits. This endeavor seeks to further enhance the efficiency and adaptability of the control system for mobile robots.

## ACKNOWLEDGMENT

The scientific research presented in this paper was realized within the Laboratory "Artificial Intelligence and Multi-Agent Systems" and is part of the topic of doctoral theses at the Department of Computer Science and Systems Engineering.

## REFERENCES

[1] G. Zhang, Z. Shang, S. Verlan, M.A. Martinez-Del-Amor, Ch. Yuan, L. Valencia-Cabrera, M.J. Perez-Jimenez, ”An Overview of Hardware Implementation of Membrane Computing Models,” ACM Computing Surveys, Vol. 53, No. 4, Article 90, August 2020, 38p., https://doi.org/10.1145/3402456.

[2] H. Nan, J. Jiang, J. Zhang, R. Liu., A. Wang. ”Conversion between Number Systems in Membrane Computing,” Appl. Sci. 2023, 13, 9945. https://doi.org/ 10.3390/app13179945.

[3] B. Song, K. Li, D. Orellana-Martín, M. J. Pérez-Jiménez, and I. Pérez-Hurtado, “A Survey of Nature-Inspired Computing: Membrane Computing,” ACM Comput. Surv. 54, 1, Article 22 (February 2021), 31 pages. https://doi.org/10.1145/3431234.

[4] K. Li, L. Jia, and X. Shi, “IPSOMC: An Improved Particle Swarm Optimization and Membrane Computing based Algorithm for Cloud Computing,” International Journal of Performability Engineering, vol. 17, no. 1, January 2021, pp. 135-142, DOI: 10.23940/ijpe.21.01.p13.135142.

[5] G. Păun, G. Rozenberg, A. Salomaa, ”DNA Computing: New Computing Paradigm,” Texts in Theoretical Computer Science (An EATCS Series). Springer, Berlin, Heidelberg. 1998, 409p. ISBN: 978-3-540-64196-4.

[6] A. Biswas, A.P. Tonda, R. Patgiri, K.K. Mishra, ”Advances in Computers: Application of Nature-Inspired Computing and Optimization Techniques,” Elsevier, Vol. 135, 2024, 546p., ISBN: 978-0-323-95768-7.

[7] X.-S. Yang, ”Nature-Inspired Computation in Engineering,” Springer, 2016, 276p., ISBN: 978-3-319-30233-1, DOI: 10.1007/978-3-319-30235-5.

[8] X.-S. Yang, ”Nature-Inspired Computation and Swarm Intelligence. Algorithms, Theory and Applications,” Elsever, 2020, 417p., ISBN: 978-0-12-819714-1, DOI: 10.1016/C2019-0-00628-0.

[9] J.K. Mandal, S. Mukhopadhyay, T. Pal, ”Handbook of Research on Natural Computing for Optimization Problems. Introduction to Molecular Computation: Theory and Applications – DNA and Membrane Computing,” 2016, 1015p., ISBN: 978-1-52250-058-2, DOI: 10.4018/978-1-52250-058-2.

[10] K.R. Skene, ”Systems theory, thermodinamics and life: Integrated thinking across ecology, organization and biological evolution,” in BioSystems, Elsevier, Vol. 236, 2024, Article 105123, Online ISSN: 1872-8324.

[11] D.J. Futuiama, ”Evolution,” Third edition, Sinauser Associates, 2013, 655p.

[12] N. Siddique, H. Adeli, ”Nature Inspired Computing: An Overview and Some Future Directions,” in Cognitive Computing, 2015, 7: 706-714. DOI: 10.1007/s12559-015-9370-8.

[13] N.E. Shklovskiy-Kordi, K. Matsuno, P.C. Marijuán, A.U. Lgamberdiev, ”Fundamental principles of biological computation: From molecular computing to evolutionary complexity,” Biosystems, Elsevier, Vol. 219, 2022, Article 104719, ISSN: 0303-2647, https://doi.org/10.1016/j.biosystems.2022.104719.

[14] Gh. Păun, G. Rozenberg, ”A guide to Membrane Computing,” in Theoretical Computer Science. 287 (1): 73-100, 2002. DOI: 10.1016/S0304-3975(02)00136-6. ISSN: 0304-3975.

[15] Gh. Păun, ”Introduction to Membrane Computing,” Applications of Membrane Computing. Springer Berlin Heidelberg. pp. 1-42. ISBN: 978-3-540-29937-0.

[16] A. Alhazov, ”Communication in Membrane Systems with Symbol Objects,” Ph.D. Thesis. Tarragona, Spain: Universitat Rovira i Virgili, 2006, 218p.

[17] A. Alhazov, ”Small Abstract Computers,” Habilitation Thesis. Chișinău (RM): Academy of Sciences of Moldova, 2013, 255p.

[18] M. J. Dinneen, Y.-B. Kim, and R. Nicolescu, ”Synchronization in P Modules,” In: Unconventional Computation, volume 6079 of Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg, 2010, pp. 32–44.

[19] M. J. Dinneen, Y.-B. Kim, and R. Nicolescu, ”An Adaptive Algorithm for P System Synchronization,” In: Proceedings of the Twelfth International Workshop on Membrane Computing, (CMC11), Fontainebleau/Paris, France, 2011, pp. 1–26.

[20] J.-L. Giovitto, O. Michel, ”The Topological Structures of Membrane Computing,” LaMI technical report Nr. 70-2001. November 2001, France.

[21] D. Diaz-Pernil, M.A. Gutierrez-Maranjo, H. PENG, ”Membrane computing and Image Processing: a Short Survey,” in Journal of Membrane Computing, 1(1), pp. 58-73, 2019, ISSN: 2523-8914, DOI: 10.1007/s41965-018-00002-x.

[22] E. Sanchez-Karhunen, L. Valencia-Cabrera, ”Modelling Complex Market Interaction using PDP Systems,” in Journal of Membrane Computing, 1(1), pp. 40-51, 2019, ISSN: 2523-8914, DOI: 10.1007/s41965-019-00008-z.

[23] V. Ababii, V. Sudacevschi, S. Munteanu, O. Borozan, A. Nistiriuc, V. Lasco, ”IoT based on Membrane Computing Models,” In: Proceedings of the 13th International Conference on Electromechanically and Energy Systems (SIELMEN-2021), 7-8 October, 2021, Chisinau, Republic of Moldova, pp. 010-014, ISBN: 978-1-6654-0078-7. DOI: 10.1109/SIELMEN53755.2021.9600341. (IEEE Catalog Number: CFP21L58-ART).

[24] P. Sosik, ”P-Systems Attacking Hard Problems Beyond NP: a survey,” in Journal of Membrane Computing, 1(3), pp. 198-208, 2019, ISSN: 2523-8914, DOI: 10.1007/s41965-019-00017-y.

[25] S. Munteanu, V. Sudacevschi, V. Ababii, O. Borozan, C. Ababii, V. Lasco, ”Multi-Agent Decision Making System based on Membrane Computing,” In: The 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. 22-25 September, 2021, Cracow, Poland, Vol. 2. pp. 851-854. ISBN: 978-1-6654-4210-7, DOI: 10.1109/IDAACS53288.2021.9660971.

[26] D. Orellana-Martin, L., Valencia-Cabrera, A. Riscos-Nunez, M.J. Perez-Jimenez, ”Minimal Cooperation as a way to Achieve the Efficiency in Cell-Like Membrane Systems,” in Journal of Membrane Computing, 1(2), pp. 85-92, 2019, ISSN: 2523-8914, DOI: 10.1007/s41965-018-00004-9.

[27] N. Zhou, A. Wang, ”Fault Diagnosis of Transmission Circuit Based on Triangular Interval Valued Fuzzy Spike Neural Network P-System,” Energy Reports, 2022, Vol. 8, pp. 776-784. DOI: 10.1016/j.egyr.2021.11.086.

[28] Anjitha Gavin, ”Obstacle avoidance robot using FPGA.” Project Report, 2016, 19p., DOI: 10.13140/RG.2.2.28792.47368. URL: https://www.researchgate.net/publication/341480237.

[29] R.I. Mohammed, S.M. Abbas, A.H. Issa, ”FPGA Cyclone II based the Mobile Robot Control System,” SCEE-2018, pp. 186-191, ISBN: 978-1-7281-1587-0.

[30] S. Munteanu, V. Sudacevschi, V. Ababii, ”Computer Systems Synthesis Inspired from Biologic Cells Structures,” in Journal of Engineering Science, June, 2022, Vol. XXIX (2), pp. 91-107, ISSN: 2587-3474 / E-ISSN: 2587-3482, DOI: 10.52326/jes.utm.2022.29(2).09.

[31] DE0-Nano Development and Education Board, URL: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=165&No=593#contents.