Date of publication SEP-30, 2025, date of current version APR-06, 2025.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.19.3.XX

Multi-Coalition Multi-Agent Decision Making System Synthesis

VICTOR ABABII, VIORICA SUDACEVSCHI, SILVIA MUNTEANU, VIOREL CARBUNE, OLESEA BOROZAN

Technical University of Moldova, Chisinau, Republic of Moldova

Corresponding author: Victor Ababii (e-mail: victor.ababii@calc.utm.md).

The research presented in this paper supports the objectives of the doctoral thesis. The design and evaluation of the scientific results were facilitated by the expertise and facilities of the 'Artificial Intelligence and Multi-Agent Systems' Laboratory, within the Department of Informatics and Systems Engineering at the Technical University of Moldova.

ABSTRACT This research explores the interdisciplinary field of nature-inspired computing, which relies on biological models and processes to develop innovative algorithms and computational systems. The paper analyzes the main categories in this field: evolutionary computing, collective intelligence, biological systems, as well as advanced approaches, such as cellular and membrane models. These paradigms provide robust and scalable solutions to complex problems that are difficult to address by traditional methods. The research places particular emphasis on cell computing, which reproduces the structure and functionality of biological cells, and on membrane computing, which introduces concepts of hierarchy and distributed processing. At the same time, the paper proposes an innovative methodology for the design of Multi-Agent systems, based on these biological models, including the dynamic formation of coalitions and the optimization of interactions between autonomous agents. The main contribution lies in the development of a mathematical model and a functional architecture for the integration of these paradigms, promoting collaborative, resilient, and innovative solutions for the future of distributed artificial intelligence.

KEYWORDS nature-inspired computing; membrane computing; cell computing; P-systems; swarm computing; collective decision making; multi-agent systems; multi-criteria optimization; genetic algorithm; goal; strategy; reconfigurable architecture.

I. INTRODUCTION

Nature-inspired Computing is an interdisciplinary field of computer science, which relies on patterns, processes and behaviors observed in nature to develop algorithms, methods and computational systems. Inspired by natural phenomena, such as evolution, population dynamics, colony formation or biological and physical processes, this type of computing offers efficient and creative solutions to complex problems.

The main categories of nature-inspired calculus include:

Evolutionary calculus – based on Darwin's theory of biological evolution, it uses mechanisms such as selection, crossover, mutation, and reconfiguration for optimization (e.g., genetic algorithms, genetic programming, evolutionary strategies);

Computation based on collective intelligence – inspired by the coordinated behavior of groups of organisms, with applications in logistics, computing networks and robotics (e.g. Ant Colony Optimization or Particle Swarm Optimization);

Biological systems-based computing – successfully applied in the modeling of biological processes, particularly in optimization and data analysis (e.g., artificial neural networks, immuno-inspired algorithms, bioinformatics).

These methods offer new paradigms for approaching complex problems that are difficult to solve by traditional means [1].

Cellular computing, based on biological processes and cellular organization in living organisms, deserves special attention in the design of computing systems inspired by nature. It explores models and methods that simulate the interaction and functioning of biological cells, using these principles to develop efficient algorithms and computational systems.

Nature-inspired methods used in cell computing include:

Cellular automata – inspired by the local interaction between cells in a network or grid, in which each cell applies a set of simple rules to determine its future state based on the state of its neighbors [31];

Gene networks and biochemical reactions – inspired by the interactions between genes and biological molecules, they shape cell population dynamics or metabolic behavior;

Membrane computing – inspired by the

compartmentalized structure of biological cells, which uses membranes to divide computing space into compartments, each with its own rules for transforming data;

Cellular genetic algorithms – based on selection, mutation and crossing, they use a localized space of a cellular system to adapt solutions to optimization problems;

DNA-based systems – inspired by genetic encoding and decoding processes, they use the manipulation of DNA sequences to store data and solve problems.

These methods provide innovative tools for the development of efficient algorithms and computational systems, mentioned in the papers [2, 3, 28, 29].

II. RELATED WORK

Nature-inspired computing offers multiple advantages by using the principles, processes, and behaviors observed in nature. Depending on the field of research, some of the essential benefits offered by nature-inspired computing can be mentioned [4 - 7]: robustness and ability to adapt to dynamic environments; scalability and decentralization; the ability to provide effective solutions to complex problems, which are often unsolvable by classical methods; Collaboration and distribution in multi-agent systems modeling.

One of the directions of research inspired by nature is collective decision-making computing (swarm computing) [8, 9, 32], an interdisciplinary field that uses the mechanisms, algorithms and strategies of groups of agents (humans, software or robots) [10, 11] to make decisions collectively. These computational models are based on the idea that a collective, through information exchange, cooperation, data collection and aggregation, negotiation, consensus, and evaluation of alternatives, can make better decisions than an autonomous individual [12, 13].

At the basis of these computing methods and techniques are: collective optimization algorithms (multicriteria optimization) [13 - 15]; distributed learning models [16]; game theory and decision analysis [17].

By using collective decision-making computing, more efficient, innovative and resilient solutions are obtained, with promising potential for future developments, including their integration into artificial intelligence and global distributed systems [18].

III. STATEMENT OF THE RESEARCH PROBLEM

The synthesis of Multi-Agent Multi-Coalition decision-making systems is a complex process, requiring theoretical and practical knowledge from various fields of science and technology. The paper proposes the following: synthesis of the computing cell model; definition of the topology of the Multi-Agent system based on membrane computing models (P-Systems); elaboration of a mathematical model for the dynamic configuration of the membrane computing architecture, to model coalition formation processes in Multi-Agent systems.

IV. SYNTHESIS OF THE COMPUTATIONAL CELL MODEL

The computing cell model is a fundamental concept in the field of nature-inspired computing, used for the development of distributed systems that are both efficient and scalable. Inspired by the structure and functioning of biological cells, this model aims to create computational architectures that mimic how natural units tackle and solve complex problems. A computing cell is defined as a basic unit in a distributed computational system, characterized by autonomy, collaboration, communication, and adaptability to environmental changes or specific requirements.

The synthesis of the computing cell model requires multidisciplinary knowledge that integrates concepts from biology, mathematics, and computer science. In order to ensure synchronization and cooperation between the computing cells, it is necessary to observe the following essential steps:

Defining the set of local behavioral rules – These rules express the autonomy of cells by performing basic functions, such as data storage and processing;

Establishing interaction rules – Rules that allow cells to communicate with each other and exchange information efficiently;

Optimization of load distribution – The distribution of loads must ensure a balance between complexity and cost, avoiding overloading of cells and enhancing system performance;

Integrating self-healing mechanisms – These mechanisms detect and correct errors locally and, in exceptional cases, redistribute loads to other cells or replace defective cells.

These steps contribute to the development of a robust, scalable, and adaptable computing system. Figure 1 presents the functional diagram of the computing cell, highlighting its key components and interactions.

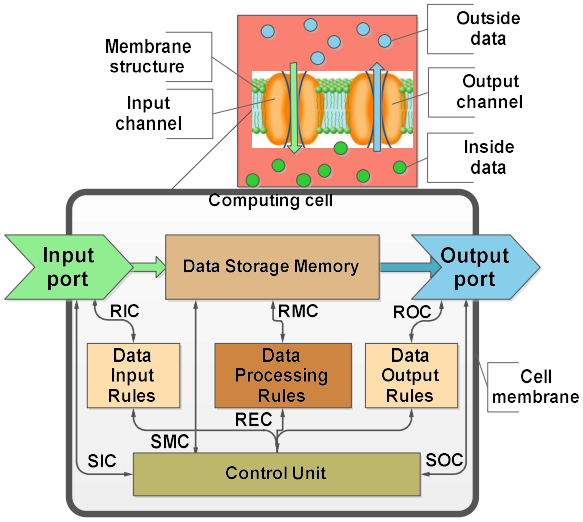

Figure 1. Functional diagram of a computing cell.

The functional diagram of the computing cell includes:

Input port and Output port – for data communication with the environment;

Data Storage Memory – stores input data from the environment, the current state of the computing cell, and output information to be transmitted back to the environment;

Control Unit – generates synchronization signals required to execute operations defined by the set of rules;

Data Input Rules – the set of rules that manage access to data from the environment into the internal memory (Data Storage Memory);

Data Output Rules – the set of rules that manage the transmission of data from the Data Storage Memory to the environment;

Data Processing Rules – the set of rules intended for processing data stored in the Data Storage Memory.

The cell membrane ensures the structure and autonomy of the computing cell.

The ***membrane structure*** includes the following components: ***Input channel*** – channels that belong to the Input port structure; ***Output channel*** – channels that belong to the Output port structure; ***Outside data*** – data originating from the environment; ***Inside data*** – data stored and processed within the calculation cell.

The activity of the stone cell, inspired by the metabolism of living cells, operates based on the following principles:

The ***Control Unit*** generates the following signals: ***SIC*** (Synchronization of Input Control) – synchronizes data acquisition from the environment; ***REC*** (Rules Execution Control) – manages the execution of rules; ***SOC*** (Synchronization of Output Control) – synchronizes data transmission to the environment.

According to the defined rules: The ***Data Input Rules*** block generates ***RIC*** (Rules Input Control) signals, which manage data acquisition through the Input port (Input channel); The ***Data Output Rules*** block generates ***ROC*** (Rules Output Control) signals, which manage data transmission through the Output port (Output channel); The ***Data Processing Rules*** block generates ***RMC*** (Rules Memory Control) signals, which access the ***Data Storage Memory*** and process the data according to the defined rules.

V. SYNTHESIS OF MULTI-AGENT DECISION MAKING SYSTEM BASED ON CELL COMPUTING MODEL

The compute cell model provides a solid foundation for defining the structure and functionality of computational units. However, to efficiently model complex systems and architectures, ***Membrane Computing*** (P-Systems) models can be utilized.

Cell Computing and ***Membrane Computing*** are two biologically inspired paradigms that share common principles while exhibiting distinct characteristics. These paradigms are complementary: ***Cellular Computing*** offers an ideal framework for spatial simulations with simple rules; ***Membrane Computing*** enables hierarchical, parallel, and concurrent processes, making it particularly suitable for problems involving multi-level communication and processing. Both approaches contribute to the development of innovative and effective solutions to complex problems across various fields [2, 19].

Figure 2 shows a Venn diagram illustrating the key characteristics of an abstract membrane computing system, emphasizing hierarchy, concurrency, and parallelism [20].

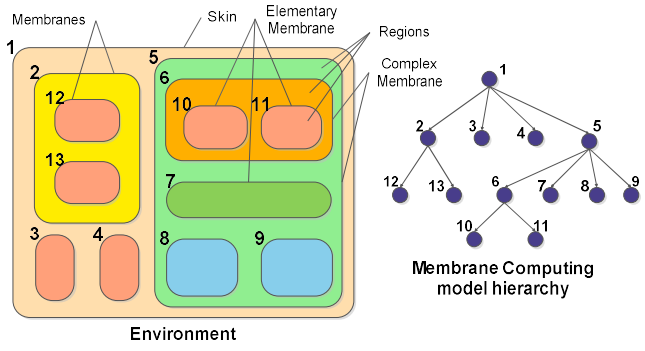

Figure 2. Venn diagram of the Membrane computing model.

The components of the ***Venn diagram*** for defining the membrane computing system are as follows: ***Skin*** (outer membrane): Forms the structure and topology of the computing system, enabling data exchange with the external environment; ***Membranes:*** Structures that separate the functional logic and ensure the autonomy of computing cells, exchanging data with the environment based on defined rules; ***Elementary Membranes:*** Simple computing cells that perform basic data processing operations; ***Complex Membranes:*** Complex structures integrating multiple elementary computing cells or other complex membranes; ***Regions:*** Refer to the internal architecture of computing cells, as described in Figure 1.

The ***hierarchy*** of the membrane computing model explains the connection between membranes and their mode of interaction: At the top level is Membrane ***1***; Membranes ***2, 3, 4, and 5*** operate in parallel or concurrent mode; Similarly, membranes ***6–9, 10–11***, and ***12–13*** also operate in parallel or concurrently.

Membrane calculus is inspired by the biological structures and processes that occur in living cells. It uses the concept of ***compartments bounded by membranes*** to process information in a ***parallel*** and ***distributed*** manner, simulating processes that occur simultaneously in multiple cells, a characteristic feature of nature.

The formal model of the membrane computing system (P-System) Π is defined by expression (1) [2, 3, and 20]:

$$\Pi = \left(O, C, \mu, \omega_1, ..., \omega_M, R_1, ..., R_M, \mu_j\left(\omega_j\right)\right), \quad (1)$$

where: $O = \bigcup_{m=1}^{M}\left(O_m\right)$ - is the set of objects (variables) operated by the computing cells, where, $O \in R^N$ and $\bigcap_{m=1}^{M}\left(O_m\right) \neq \varnothing$; $C = \left\{c_m, \forall m = \overline{1,...M}\right\}$ - is the set of catalyst elements (representing a collection of synchronization signals that validate the operations planned by the rule set R; μ - is the structure (topology) of the membrane computing system consisting of $\mu_1, ..., \mu_M$ membranes (see Figure 2); $\omega_1, ..., \omega_M$ - is the set of objects (variables) and state data belonging to the computing cell $\mu_m, \forall m = \overline{1,M}$, where $O_m \subset \omega_m$; $R_1, ..., R_M$ - is the set of rules for processing/transforming the objects (variables) and state data associated with computing cell $m = \overline{1,M}$; $\mu_j\left(\omega_j\right), j = \overline{1,M}$ - is the set of computing cells that, based on the set of objects (variables) and state data ω_j generate intermediate or final results, thereby acting upon the activity environment and controlling its evolution.

Let the Multi-Agent System (MAS) $A = \left\{A_i, i = \overline{1,M}\right\}$ (Figure 3) consisting of M Autonomous Agents AA operating in a N dimensional space based on model (2) and $A = \bigcup_{i=1}^{M}\left(A_i\right)$:

$$\begin{cases} E_i(X_i) \in R^N, \\ Q(X) \to opt(X), \\ S_i \left| X \to opt(X_i), \right. \\ f_i \left| X \to Y_i. \right. \end{cases} \quad \forall i = \overline{1...M}, \tag{2}$$

where: $E_i(X_i)$ - is the domain of definition for Agent A_i, determining the set of objects (variables) used by the Agent, where $E(X) = \bigcup_{i=1}^{M} (E_i(X_i))$; $Q(X) \to opt(X)$ - the objective function of the Multi-Agent System for global optimization in the definition space R^N; $S_i \left| X \to opt(X_i) \right.$ - the set of strategies defined for each Agent A_i ensuring the optimization condition for variables X_i, respecting the condition $X_i \in E(X_i)$; $f_i \left| X \to Y_i \right.$ - the set of functions for computing the action values applied to the activity environment $E(X)$.

Figure 3 shows the initial state of the Multi-Agent System. In this state, each Agent operates autonomously, focusing solely on its own objectives as defined in its individual strategy. The results of an Agent's actions become visible to others only through their effect on the shared environment. This mode of operation is inefficient because the Agents' actions are uncoordinated, making it impossible to achieve the overall optimization objectives within the specified time frames.

The proactive formation of coalitions, which is necessary to improve the efficiency of the Multi-Agent System, becomes impossible as each Agent adapts its strategy in response to changes in the environment over time and space. A viable solution to forming effective coalitions involves applying dynamic coalition methods that consider both overall objectives and individual Agent strategies [13, 21, 27].

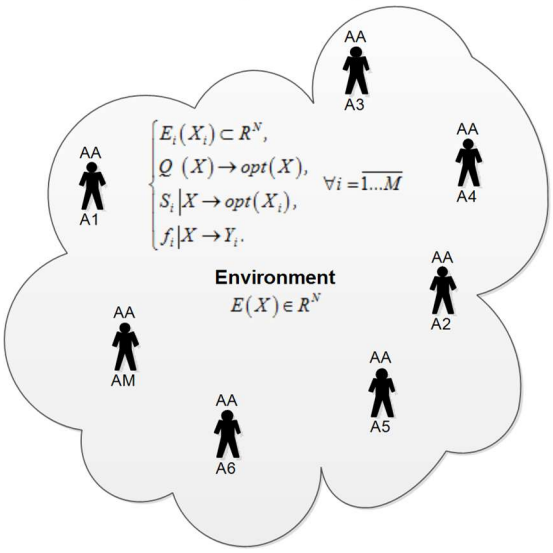

Figure 3. The initial state of the Multi-Agent System.

In Figure 4 is presented diagram of the membrane computing model (P-Systems) of the Multi-Agent System - Agent $A = \left\{ A_i, i = \overline{1,M} \right\}$ for the initial condition of autonomous activity of the Agents.

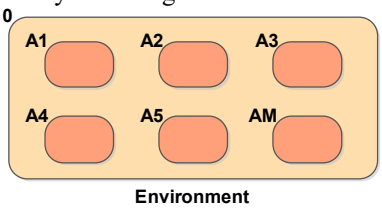

Figure 4. Venn Diagram for Multi-Agent System Membrane Computing Model for Initial Condition.

Figure 5 shows the diagram of the calculation cell for modeling the Multi-Agent System based on membrane computing.

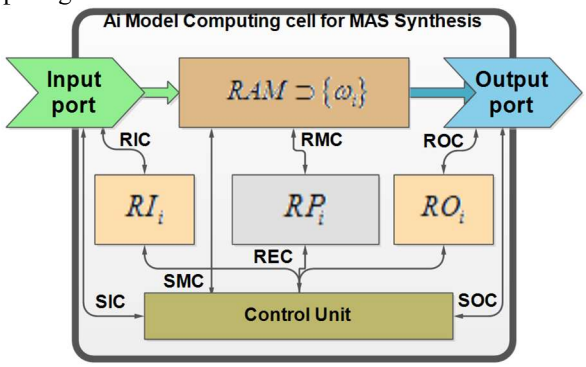

Figure 5. Calculation cell diagram for modeling the Multi-Agent System (MAS) based on membrane computing.

Specifications of the compute cell used for the modeling of the Multi-Agent System based on membrane computing. This architecture is oriented to be implemented using Micro-Controller ESP32 [22, 23], Genuino 101 [24] or Raspberry Pi [25, 26] devices:

Input/Output port – ports for perceiving the environment, acting on the activity environment and communicating with other Agents (depending on the environment and sensor technology, the following may be recommended: analog ports; digital GPIO (General Purpose Input/Output), PWM (Pulse Width Modulation) ports; serial ports based on RS-232/RS-485 protocols, UART (Universal Asynchronous Receiver-Transmitter), SPI (Serial Peripheral Interface), I²C (Inter-Integrated Circuit); USB ports; Ethernet ports; wireless Wi-Fi (IEEE-802.11), Bluetooth, ZigBee, Lora WAN ports; or CAN (Controller Area Network) ports). ;

RI_i - rules for controlling the process of data acquisition from the surrounding environment and forming coalitions with other Agents:

$$\begin{cases} \forall E_i(X_i) \in R^N, \forall i = \overline{1,...,M}, \forall j = \overline{1,...,M}: \\ IF \left(\left(Q_i(X_i) \to \min \right) \& \left(\frac{\partial (Q_i(X_i))}{\partial (X_j)} < 0 \right) \right), \\ THEN \left(A_j \subset C(A_i) \right). \end{cases} \tag{3}$$

where

$$\begin{cases} \forall E_i(X_i) \in R^N, \forall i = \overline{1,...,M}, \forall j = \overline{1,...,M}: \\ IF\left(\left(Q_i(X_i) \to \max\right) \& \left(\frac{\partial(Q_i(X_i))}{\partial(X_j)} > 0\right)\right), \\ THEN\left(A_j \subset C(A_i)\right). \end{cases} \quad (4)$$

where $\left(Q_i(X_i) \to \min\right) \& \left(\frac{\partial(Q_i(X_i))}{\partial(X_j)} < 0\right)$ and $\left(Q_i(X_i) \to \max\right) \& \left(\frac{\partial(Q_i(X_i))}{\partial(X_j)} > 0\right)$ is the condition of the Agent's membership A_j to belong to the coalition formed by Agent A_i where $A_j \subset C(A_i)$.

$RAM \supset \{\omega_i\}$ - The storage of working data in memory for the Agent A_i, where $\omega_i \left| A_j \subset C(A_i), \forall j = \overline{1,...,M}\right.$ it is the set of data obtained from all Agents A_j which form the coalition for the Agent A_i, including the state data of the Agent;

RP_i - it is the set of rules that implement the strategy. $S_i \left| \omega_i \to opt\right.$ and function $f_i \left| \omega_i \to Y_i\right.$ for computing the output data and the state of the computing cell;

RO_i - it is the set of rules that identify the set of data for output and communication with all the Agents forming the coalition $C(A_i)$.

Membrane computing (P-systems) offers advantages in modeling computational architectures, ranging from functional logic circuits (such as registers, counters, adders, encoders, decoders, multiplexers, and demultiplexers) to multi-core microprocessor architectures and global computing networks (e.g., the Internet and Intranets) [5, 6, 25, 26, 30]. In this paper, the notion of an Agent is used to abstract and encapsulate specific functionalities into a functional logic entity enhanced with artificial intelligence.

The mathematical expressions (3) and (4) define the dynamics of the Multi-Agent decision-making system architecture in relation to the objectives of each Agent. If an Agent's actions accelerate the convergence of the objective function toward the optimal value, that Agent is included in the coalition. The inclusion of an Agent in the coalition ensures efficient organization of data exchange among Agents by applying the multicast data transfer method.

The Multi-Agent Decision System is a reconfigurable distributed computing architecture characterized by the following advantages: dynamic scalability, fault tolerance and robustness, flexibility in involving Agents in solving complex problems, collective decision-making, and support for the implementation of distributed artificial intelligence.

The specific areas of application for the Multi-Agent decision-making system with multiple coalitions, which provide the aforementioned advantages, include industrial and technological processes, smart agriculture, transport and logistics, smart cities, and the gaming industry.

VI. EXAMPLE OF COALITIONS FORMATION

To validate the results, we will examine a Multi-Agent decision-making process applied in Smart Agriculture, with two contradictory objectives:

a) Minimization of resources used in the production process: involving labor - $A_1(X_1)$, water consumption - $A_2(X_2)$, and fertilizer expenses - $A_3(X_3)$;

b) Maximization of total profit (***Profit*** = ***Income*** - ***Expenses***): involving a market specialist in the agricultural products production market - $A_4(X_4)$, energy expenditure - $A_5(X_5)$, and logistics expenses - $A_6(X_6)$.

Based on the data we will define the Multi-Agent system A, where:

$$A = \begin{cases} M = 6, \\ Q(X_1, X_2, X_3) \to \min, \\ Q(X_4, X_5, X_6) \to \max,, \\ S = \{S_1, ..., S_6\}, \\ Y_1 = f_1(X_1, X_2, X_3), \\ Y_4 = f_4(X_4, X_5, X_6). \end{cases} \quad (5)$$

In the process of the evolution of the topology of the membrane computing system, two coalitions were formed: $C(A_1) = \cup(A_1, A_2, A_3)$ and $C(A_4) = \cup(A_4, A_5, A_6)$ with two action variables upon the activity environment Y_1 and Y_4.

Figure 6 shows the Venn diagram for the membrane computational model (P-Systems) of the Multi-Agent System defined by the mathematical model (5).

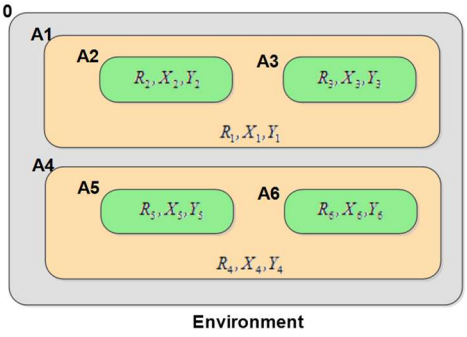

Figure 6. Venn Diagram for a Multi-Agent System in a Membrane Computing Model Defined by Mathematical Equation (5).

The coalitions formed by the Agent A_1 and A_4 ensure more efficient collaboration in order to solve the objectives defined by applying those strategies.

VII. CONCLUSIONS

This paper addresses one of the global challenges in the design of collaborative computing systems, focusing on Multi-Agent systems and models inspired by nature. The concept of nature-inspired computing, through its interdisciplinary approach, provides a solid framework for

developing innovative, efficient, and scalable algorithms and computational systems. This methodology offers solutions to complex problems that classical methods cannot solve, providing benefits such as robustness, adaptability, and distribution.

The computing cell model, proposed here, is a central concept in the design of distributed systems inspired by biological processes. It integrates mechanisms for autonomy, collaboration, and communication, enabling the development of systems capable of handling complex tasks efficiently and robustly. By utilizing the principles of hierarchy, parallelism, and concurrency, membrane computing extends the applicability of biology-inspired models, facilitating distributed and parallel processing of information. This model is particularly suited for problems involving interactions across multiple levels of complexity.

The integration of membrane computing models into Multi-Agent systems offers a systematic approach for dynamic coalition formation, agent coordination, and overall strategy optimization. This framework enables objectives to be achieved through collaboration and adaptability in complex, variable environments. To validate the ideas developed in the paper, we propose a mathematical model for the synthesis of multi-coalition Multi-Agent systems, demonstrating the potential of nature-inspired computation to provide efficient solutions to optimization problems through dynamic strategies for system configuration and reconfiguration.

The research reveals that coalition building and the application of multi-agent systems based on membrane computing show significant promise for integrating these models into emerging fields such as artificial intelligence, distributed networks, and robotics. These research directions offer new opportunities for the design of complex, autonomous systems.

References

[1] N. Dey, A. S. Ashour & S. Bhattacharyya, *Applied nature-inspired computing: algorithms and case studies,* Springer Singapore, 2020, 275 p., https://doi.org/10.1007/978-981-13-9263-4.

[2] G. Paun, *Membrane computing: an introduction.* Springer Berlin, Heidelberg, 2012, 420 p, https://doi.org/10.1007/978-3-642-56196-2.

[3] S. Patnaik, X. S. Yang & K. Nakamatsu, *Nature-inspired computing and optimization. Theory and Application (Vol. 10).* Heidelberg: Springer, 2017, 494 p., https://doi.org/10.1007/978-3-319-50920-4.

[4] N. Siddique & H. Adeli, "Nature inspired computing: an overview and some future directions," *Cognitive computation*, vol. 7, pp. 706-714, 2015. https://doi.org/10.1007/s12559-015-9370-8.

[5] B. Song, K. Li, D. Orellana-Martín, M. J. Pérez-Jiménez & I. Pérez-Hurtado, "A survey of nature-inspired computing: Membrane computing," *ACM Computing Surveys (CSUR)*, vol. 54, issue 1, pp. 1-31, 2021, https://doi.org/10.1145/3431234.

[6] S. Kaul, Y. Kumar, U. Ghosh, et al. "Nature-inspired optimization algorithms for different computing systems: novel perspective and systematic review," *Multimed Tools Appl*, vol. 81, pp. 26779–26801, 2022, https://doi.org/10.1007/s11042-021-11011-x.

[7] L. Jiao, J. Zhao, C. Wang, X. Liu, F. Liu, & S. Yang, "Nature-Inspired Intelligent Computing: A Comprehensive Survey," *Research,* vol. 7, Article 0442, 2024. https://doi.org/10.34133/research.0442.

[8] S. Garnier, & M. Moussaïd, "We the swarm – Methodological, theoretical, and societal (r)evolutions in collective decision-making research," *Collective Intelligence,* vol. 1, issue 2, 2022, https://doi.org/10.1177/26339137221133400.

[9] A. Almansoori, M. Alkilabi & E. Tuci, "On the evolution of mechanisms for three-option collective decision-making in a swarm of simulated robots," *Proceedings of the Genetic and Evolutionary Computation Conference, 2023 (GECCO'23),* pp. 4-12, https://doi.org/10.1145/3583131.3590385.

[10] A. Dorri, S. S. Kanhere & R. Jurdak, Multi-agent systems: A survey. *IEEE Access,* vol. 6, pp. 28573-28593, 2018, https://doi.org/10.1109/ACCESS.2018.2831228.

[11] J. Qin, Q. Ma, Y. Shi & L. Wang, "Recent advances in consensus of multi-agent systems: A brief survey," *IEEE Transactions on Industrial Electronics*, vol. 64, issue 6, pp. 4972-4983, 2016, https://doi.org/10.1109/TIE.2016.2636810.

[12] A. M. Uhrmacher & D. Weyns (Eds.), *Multi-Agent systems: Simulation and applications.* CRC press, 2018, 543 p., ISBN: 978-1-4200-7023-1.

[13] R. Melnic, V. Ababii, V. Sudacevschi, O. Sachenko, O. Borozan & T. Lendiuk, "Multi-objective based multi-agent decision-making system," *Proceedings of the 2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS),* vol. 1, 2023, pp. 834-839, https://doi.org/10.1109/IDAACS58523.2023.10348725.

[14] R. Rădulescu, P. Mannion, D. M. Roijers & A. Nowé, "Multi-objective multi-agent decision making: a utility-based analysis and survey," *Autonomous Agents and Multi-Agent Systems*, vol. 34, issue 1, Article 10, 2020. https://doi.org/10.1007/s10458-019-09433-x.

[15] A. S. Akopov & M. A. Hevencev, "A multi-agent genetic algorithm for multi-objective optimization," *Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics*, 2013, pp. 1391-1395, https://doi.org/10.1109/SMC.2013.240.

[16] E. Lavrov, N. Pasko, A. Tolbatov and N. Barchenko, "Development of adaptation technologies to man-operator in distributed E-learning systems," *Proceedings of the 2017 2nd International Conference on Advanced Information and Communication Technologies (AICT)*, Lviv, Ukraine, 2017, pp. 88-91, https://doi.org/10.1109/AIACT.2017.8020072.

[17] E. N. Barron, *Game theory: an introduction,* Third Edition, John Wiley & Sons, 2024, 547 p.

[18] R. Casado-Vara, F. Prieto-Castrillo & J. M. Corchado, "A game theory approach for cooperative control to improve data quality and false data detection in WSN," *International Journal of Robust and Nonlinear Control*, vol. 28, no. 16, pp. 5087-5102, 2018. https://doi.org/10.1002/rnc.4306.

[19] P. Frisco, *Computing with cells: Advances in membrane computing.* OUP Oxford, 2009, 336 p. https://doi.org/10.1093/acprof:oso/9780199542864.001.0001.

[20] S. Munteanu, V. Sudacevschi, V. Ababii, "Computer systems synthesis inspired from biologic cells structures," *Journal of Engineering Science,* Vol. XXIX (2), pp. 91-107, 2022. https://doi.org/10.52326/jes.utm.2022.29(2).09.

[21] V. Ababii, V. Sudacevschi, A. Turcan, R. Melnic, V. Carbune, I. Cojuhari, "Multi-objective decision making system based on spatial-temporal logics," *Proceedings of the 24th International Conference on Control Systems and Computer Science (CSCS-2023)*, 24-26 May 2023, Bucharest, Romania, pp. 6-10, https://doi.org/10.1109/CSCS59211.2023.00010.

[22] N. Cameron, "ESP32 Microcontroller," *In: ESP32 Formats and Communication. Maker Innovations Series. Apress, Berkeley, CA, 2023*, pp. 1-54, https://doi.org/10.1007/978-1-4842-9376-8_1.

[23] A. Maier, A. Sharp and Y. Vagapov, "Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things," *Proceedings of the 2017 Internet Technologies and Applications (ITA)*, Wrexham, UK, 2017, pp. 143-148, https://doi.org/10.1109/ITECHA.2017.8101926.

[24] D. de Santana Nunes, J. L. V. de Brito and G. N. Doz, "A low-cost data acquisition system for dynamic structural identification," *IEEE Instrumentation & Measurement Magazine*, vol. 22, no. 5, pp. 64-72, 2019. https://doi.org/10.1109/IMM.2019.8868280.

[25] S. Gunde, A. K. Chikaraddi and V. P. Baligar, "IoT based flow control system using Raspberry Pi," *Proceedings of the 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, Chennai, India, 2017, pp. 1386-1390, https://doi.org/10.1109/ICECDS.2017.8389671.

[26] A. Pajankar, "Introduction to single board computers and Raspberry Pi," *In: Raspberry Pi Supercomputing and Scientific Programming. Apress, Berkeley, CA,* 2017, pp. 1-25, https://doi.org/10.1007/978-1-4842-2878-4_1.

[27] O. Dunets, C. Wolff, A. Sachenko, G. Hladiy and I. Dobrotvor, "Multi-agent system of IT project planning," *Proceedings of the 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, Bucharest, Romania, 2017, pp. 548-552, https://doi.org/10.1109/IDAACS.2017.8095141.

[28] P. Bykovyy, V. Kochan, A. Sachenko and G. Markowsky, "Genetic algorithm implementation for perimeter security systems CAD,"

Proceedings of the 4th IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Dortmund, Germany, 2007, pp. 634-638, https://doi.org/10.1109/IDAACS.2007.4488498.

[29] P. Bykovyy, Y. Pigovsky, V. Kochan, A. Sachenko, G. Markowsky and S. Aksoy, "Genetic algorithm implementation for distributed security systems optimization," *Proceedings of the IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, Istanbul, Turkey, 2008, pp. 120-124, https://doi.org/10.1109/CIMSA.2008.4595845.

[30] N. Dziubanovska, "Modeling of war-induced Ukrainian migration's impact on Poland's trade using machine learning," *Proceedings of the 4th International Workshop on Information Technologies: Theoretical and Applied Problems*, 2024, pp. 494-508. [Online]. Available at: https://ceur-ws.org/Vol-3896/paper29.pdf.

[31] M. Mohammed Ibrahim, R. Venkatesan, Kavikumar Jacob, "Investigating the feasibility of elementary cellular automata based scrambling for image encryption," *International Journal of Computer Network and Information Security (IJCNIS)*, vol. 17, no. 1, pp. 28-38, 2025. https://doi.org/10.5815/ijcnis.2025.01.03.

[32] A. Novikov, S. Yakovlev, I. Gushchin, "Exploring the possibilities of MADDPG for UAV swarm control by simulating in Pac-Man environment," *Radioelectronic and Computer Systems*, vol. 2025, no. 1, pp. 327-337, 2025. https://doi.org/10.32620/reks.2025.1.21.

***VICTOR ABABII** received the PhD degree in computer engineering from Technical University of Moldova, Republic of Moldova, in 2000. Currently, he is an Associate Professor with the Department of Computer Science and Systems Engineering, Technical University of Moldova. His research interests include modeling and design of reconfigurable real-time decision making multi-agent systems.*

***VIORICA SUDACEVSCHI** graduated from Technical University of Moldova in Computer Science in 1988. In 2010 she received her PhD degree in Engineering, specialty "Computers, computer systems and Networks" from Technical University of Moldova. Her research interests focus on computer systems design, modelling and analyzing based on Petri nets, reconfigurable computer architectures, cyber security and cyber incident management.*

***SILVIA MUNTEANU** graduated from the Technical University of Moldova in Computer Science in 2002. She is currently a PhD student in Engineering, specializing in "Control Systems, Computers, and Networks" at the Technical University of Moldova. Her research interests focus on the design, modeling, and analysis of computer systems based on nature-inspired models and reconfigurable computing architectures.*

***VIOREL CARBUNE** graduated from the Technical University of Moldova in Computer Science in 2007. In 2020, he received his PhD degree in Engineering, specializing in 'Control Systems, Computer and Networks' from the Technical University of Moldova. His research interests include modeling and design of reconfigurable and Artificial Intelligence-based systems.*

***OLESEA BOROZAN** graduated from the Technical University of Moldova in Computer Science in 2005. She is currently a PhD student in Engineering, specializing in "Control systems, Computer and Networks" at the Technical University of Moldova. Her research interest focuses on the design, modeling, and expertise of decision-making computer systems based on natural language processing and speech recognition.*