MULTI-AGENT MEMBRANE SYSTEMS

Cristian-Ioan Vasile[1] and Ioan Dumitrache[2]

In this paper, we discuss the basic concepts of network systems and the problem of robot coordination. Moreover, we propose a membrane system model for multi-agent coordination. Membrane systems, also known as P systems, are a computational paradigm inspired by the structure of living cells [14]. Membrane systems were successfully used to model basic robot controllers, such as obstacle avoidance, localization, follower behavior etc. However, the use of membranes systems to model the multi-agent behavior was not explored in the literature. This study shows that membrane systems can be used as a high-level framework for multi-agent coordination. The membranes can simulate swarm behaviors, such as communication between agents, communication between groups of agents, transport of agents between groups, transport of agents between subswarm, split of the groups and regrouping.

Keywords: Membrane Computing, Numerical P Systems, Multi-agent Systems, Dynamical Systems, Consensus Protocol

1. Introduction

P systems are a computational paradigm inspired by the functioning of living cells. The framework of membrane computing has been used so far to model and simulate real-life problems, such as robot behaviors, modeling of dynamical systems and species evolution, applications in economy etc. [12], [13], [15]

Numerical P systems (NP systems) are a class of P systems in which numerical variables evolve inside the compartments by means of programs [14]. A program (or rule) is composed of a production function and a repartition protocol. The variables have an initial value and the production function is a multivariate polynomial. The value of the production function for the current values of the variables is distributed among variables in certain compartments according to a repartition protocol.

[1] Department of Automatic Control and Systems Engineering, Politehnica University of Bucharest, Splaiul Independenţei 313, 060042 Bucharest, Romania, e-mail: cvasile@ics.pub.ro

[2] Department of Automatic Control and Systems Engineering, Politehnica University of Bucharest, Splaiul Independenţei 313, 060042 Bucharest, Romania, e-mail: idumitrache@ics.pub.ro

In this paper, we show that membrane systems can also be used as a high-level framework for multi-agent coordination. We define a membrane system to model the problem of reaching *consensus* [1, 10, 11, 20, 4, 9, 8, 6]. This is a fundamental problem in networked system. Many tasks such as flocking (reaching the same heading), rendezvous (reaching the same position), leader election and many others, can be recast as a consensus problem [7, 5, 19].

2. **Numerical P systems**

Numerical P (NP) systems were first defined in [14]. P systems present a *membrane structure*, the membranes being labeled in a one-to-one manner with elements of an alphabet H. The compartments contain the *variables*, their initial values and rules.

The formal definition of NP systems is the following:

$$\Pi = (m, H, \mu, (Var_1, Pr_1, Var_1(0)), \ldots, (Var_m, Pr_m, Var_m(0))) \quad (1)$$

where:

- m is the number of membranes used in the system, degree of Π; $m \geq 1$;
- H is an alphabet that contains m symbols (the labels of the membranes);
- μ is a membrane structure;
- Var_i is the set of variables from compartment i, and the initial values for these variables are $Var_i(0)$;
- Pr_i is the set of programs (rules) from compartment i. Programs process variables and have two components, a production function and a repartition protocol.

  The j-th program has the following form:

$$Pr_{j,i} = (F_{j,i}(x_{1,i}, ..., x_{k_i,i}), c_{j,1}|v_1 + ... + c_{j,n_i}|v_{n_i}) \quad (2)$$

  where:
  - $F_{j,i}(x_{1,i}, ..., x_{k_i,i})$ is the production function;
  - k_i represents the number of variables in membrane i;
  - $c_{j,1}|v_1 + ... + c_{j,n_i}|v_{n_i}$ is the repartition protocol;
  - n_i represents the number of variables contained in membrane i, plus the the number of variables contained in the parent membrane of i, plus the number of variables contained in the children membranes of i.

The variables $c_{j,1}, \ldots, c_{j,n_i}$ are natural numbers (they may be also 0, case in which it is omitted to write $+0|x$) [14]. These coefficients specify the proportion of the current production distributed to each variable $v_1, ..., v_{n_i}$. Let,

$$C_{j,i} = \sum_{n=1}^{n_i} c_{j,n} \quad (3)$$

A program $Pr_{j,i}$ is executed as follows. At any time t, the function $F_{j,i}(x_{1,i}, ..., x_{k_i,i})$ is computed. The value:

$$q = \frac{F_{j,i}(x_{1,i}, ..., x_{k_i,i})}{C_{j,i}} \tag{4}$$

represents the *unitary portion* to be distributed to variables $v_1, \ldots, v_{n_i}$, according to coefficients $c_{j,1}, \ldots, c_{j,n_i}$ in order to obtain the values of these variables at time $t+1$. Specifically, variable v_s which belongs to the repartition protocol of program j, receives:

$$q * c_{j,s}, for 1 \leq s \leq n_i \tag{5}$$

The variables which receive new values from a rule must be contained within the current, the parent or a child membrane. If a variable belongs to membrane i, it can appear in the repartition protocol of the parent membrane of i and also in the repartition protocol of the child membranes of i. After applying all the rules, if a variable receives such *contributions* from several neighboring compartments, then they are added in order to produce the next value of the variable.

A production function which belongs to membrane i may depend only on some of the variables from membrane i. Those variables which appear in the production function become 0 after the execution of the program.

Deterministic NP systems have only one rule per membrane ($card(Pr_i) = 1$) or must have a selection mechanism that can decide which rule to apply [16]. The NP system with multiple rules per membrane is non-deterministic. Deterministic NP systems are well suited for applications which involve numerical variables and require a deterministic behavior, such as control systems for mobile robots.

3. **Consensus Protocol**

In the following, we review average consensus [1, 10, 11, 20] and present for completeness the main results with proofs. Although the results are presented only for continuous systems, similar results for discrete systems are well established [10].

Consider a networked system with N agents and let $G = (V, E)$, $|V| = N$, be the communication graph between the agents, where each agent is associated with a vertex from V. Furthermore, each agent has a state $x_i \in \mathbb{R}$.

Now, consider the following dynamics for the agents which in is called average consensus protocol:

$$\dot{x}_i = \sum_{j \in N_i} a_{i,j}(x_j - x_i), \ \forall i \in V \tag{6}$$

where $a_{i,j}$ are some positive weights associated with each edge of the communication graph G such that $a_{i,j} = a_{j,i}$. If we concatenate the states of all

agents into a column vector $x = [x_1, \ldots, x_N]^T$ we can rewrite the dynamics in vectorized for in the following way:

$$\dot{x} = -Lx \tag{7}$$

where L is the weighted Laplacian of the communication graph G between the agents.

Next, lets define the set $\Omega = \{a\mathbf{1}_N | a \in \mathbb{R}\}$ which is also called the set of consensus states, because in this case all agents' states are equal. Notice, that Ω is an invariant set the consensus system, Eq. (6) and Eq. (7). Moreover, Ω is the set of equilibria of the consensus system.

The *disagreement function* for the consensus protocol is defined in this case as the potential function

$$\Phi(x) = \frac{1}{2} \sum_{(i,j) \in E} (x_i - x_j)^2$$

which captures the difference in *opinion* between agents. Using the Laplacian of the communication graph we can write the disagreement function as

$$\Phi(x) = x^T L x$$

which shows that it is a p.s.d. function. Moreover, we can express the consensus dynamics as a gradient system in the following way

$$\dot{x} = -\frac{\partial \Phi}{\partial x}$$

An interesting property of the consensus protocol is that the mean of the states is time-invariant.

Proposition 3.1. *Consider the consensus protocol Eq. 7 with communication graph $G = (V, E)$. Let $\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$, then*

$$\dot{\bar{x}} = 0$$

where $N = |V|$.

Proof. The proof is straightforward since we can write $\bar{x} = \frac{1}{N} \mathbf{1}_N^T x$. If we compute the derivative of the mean state we obtain

$$\dot{\bar{x}} = \frac{1}{N} \mathbf{1}_N^T \dot{x} = -\frac{1}{N} \mathbf{1}_N^T L x$$

We know that $\mathbf{1}_N$ is an eigenvector of the Laplacian matrix L corresponding to the eigenvalue 0. Thus, $\mathbf{1}_N^T L = (L^T \mathbf{1}_N)^T = (L\mathbf{1}_N)^T = 0$, because L is a symmetric matrix. □

Now, we are ready to state the main theorem of for consensus protocols:

Theorem 3.1 (Theorem 1 from [10])**.** *Consider the consensus protocol Eq. 7 with communication graph $G = (V, E)$. Then all agents' states converge to a consensus state for any initial state $x(0)$, that is $x_i(t) \to \alpha$ for all $i \in V$ as $t \to \infty$, where $\alpha \in \mathbb{R}$. Moreover, we have the following cases:*

(1) *if G is undirected and connected, then* $\alpha = \frac{1}{N}\sum_{i=1}^{N} x_i(0) = \bar{x}(0)$;
(2) *if G is directed, connected and balanced, then* $\alpha = \frac{1}{N}\sum_{i=1}^{N} x_i(0) = \bar{x}(0)$;
(3) *if G is directed and strongly connected, then* $\alpha = \frac{1}{N}\sum_{i=1}^{N} \gamma_i x_i(0)$, *where* $\gamma_i \in \mathbb{R}$ *and* $\sum_{i=1}^{N} \gamma_i = 1$.

where a balanced graph is defined by the property that for every vertex the number of outgoing edges is equal to the number of incoming ones.

We provide a proof for the first case dealing with undirected graphs. The directed cases are more involved, but follow in a similar manner.

Proof. Consider the Lyapunov function

$$V(x) = \frac{1}{2}\|x\|^2 = \frac{1}{2}x^T x$$

Clearly, $V(x)$ is p.s. and radially unbounded. The total derivative of $V(x)$ is

$$\dot{V}(x) = x^T \dot{x} = -x^T L x$$

and since L is p.s.d. it follows that $\dot{V}(x)$ is n.s.d. Moreover, the total derivative is zero on the linear space spanned by $\mathbf{1}_N$, which defines the equilibria set of consensus states Ω. Thus, by LaSalle's invariance principle, we can conclude that the set Ω is asymptotically stable, i.e. all trajectories converge to some point in Ω. This concludes the proof. □

The convergence rate of the consensus protocol is characterize by the following result

Theorem 3.2 (Corollary 1 from [10])**.** *The convergence rate of the consensus protocol Eq. 7 with communication graph G is exponential and the smallest exponential convergence rate is given by $\lambda_2(L)$, where L is the Laplacian matrix of G and $\lambda_2(L)$ is the second smallest eigenvalue of L.*

Proof. Lets use the coordinate transform $\delta = x - \mathbf{1}_N \bar{x}(0)$ and define the Lyapunov function

$$V(\delta) = \frac{1}{2}\|\delta\| = \frac{1}{2}\delta^T \delta$$

Notice that δ is orthogonal to $\mathbf{1}$,

$$\mathbf{1}_N^T \delta = \mathbf{1}_N^T x - \mathbf{1}_N^T \mathbf{1}_N \bar{x}(0) = N\bar{x}(t) - N\bar{x}(0) = 0$$

where the last equality follows from Proposition 3.1. Next, we compute the total derivative of V

$$\begin{aligned} \dot{V}(\delta) &= -\delta^T L \delta \\ &\leq -\lambda_2(L)\|\delta\|^2 \end{aligned}$$

On the other hand,

$$\dot{V}(\delta) = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{2}\|\delta\|^2\right) = \frac{1}{2}\|\delta\|\frac{\mathrm{d}}{\mathrm{d}t}\|\delta\|$$

Thus it follows that

$$\frac{\mathrm{d}}{\mathrm{d}t}\|\delta\| \leq -\lambda_2(L)\|\delta\|$$

or more explicitly

$$\|\delta(t)\| \leq \|\delta(0)\| e^{-\lambda_2(L)t},\ t \geq 0$$

This concludes the proof. □

Lastly, we want to finish this section by presenting the straightforward extension of consensus to the vector case. In this setup, the state of each agent is a vector in some Euclidean space of dimension $n \geq 2$ instead of scalar values.

Let the networked system with communication graph $G = (V, E)$ such that the state of each agent i is $x_i \in \mathbb{R}^n$. The system dynamics are defined by Equation. 6, where, in this case, the equations describe the evolution of the n-dimensional states of agents. Again, if we concatenate the states of all agents into a column vector $x = [x_1^T, \ldots, x_N^T]^T$ we can rewrite the dynamics in vectorized for in the following way:

$$\dot{x} = -(L \otimes I_n)x \tag{8}$$

where L is the Laplacian matrix of G and $\otimes$ is the Kronecker product.

Remark 3.1. *Similar results hold for discrete systems as well.*

4. Membrane systems as a framework for multi-agent coordination

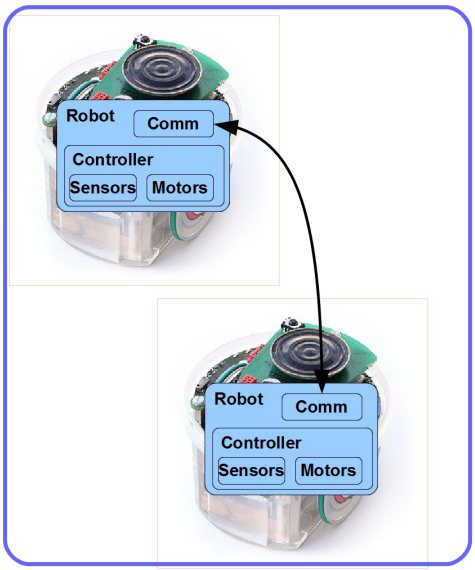
Membrane systems were successfully used to model robot controllers Figure 1(a). We further show that membrane systems can also be used as a high-level framework for multi-agent coordination. Figure 1(b) shows that the membranes can simulate swarm behaviors, such as communication between agents, communication between groups of agents, transport of agents between groups, transport of agents between subswarm, split of the groups and regrouping (membrane splitting and merging).

A membrane system that implements consensus protocol is proposed. The objective of consensus protocol is for all agents to acheive the same state while running the network control system. Figure 1(c) illustrates the consensus membrane system.

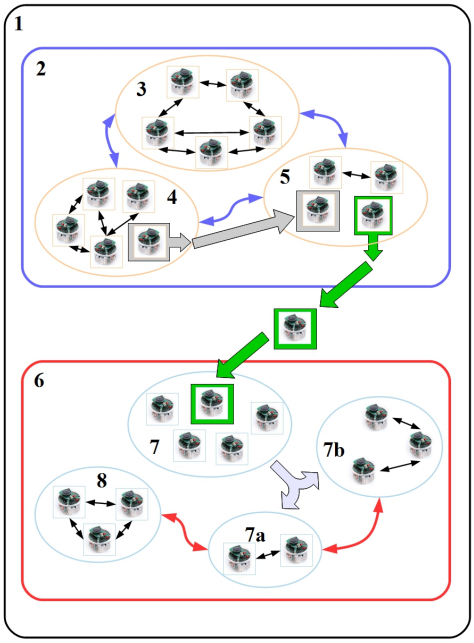
The membrane system is composed of N *Agent* membranes, each corresponding to a robot. All these membranes are contained in the *Consensus* membrane which is responsible for computing the control input for each agent. The communication between the robots and the computation of their inputs is performed in a distributed manner.

The variables of the system have the following interpretation, where $i \in \{1, \ldots, N\}$ is an agent:

- x_i is the state of the agent at the discrete time t;
- y_i is the output of the agent and it is equal to x_i;



(a) Membrane systems as a framework to model robot controllers



(b) Membrane systems as a framework for multi-agent coordination

Consensus

$y_i[x_{i0}], 1 \leq i \leq N$

$Pr_{i,C} : \sum_{j \in N_j} a_{ij}(y_j - y_i) \rightarrow 1|u_i, \ \forall i \in \{1, \ldots, N\}$

Agent$_i$, $i \in \{1, \ldots, N\}$

$x_i[x_{i0}], u_i[0]$

$Pr_{1,A_i} : 2x_i \rightarrow x_i + y_i$

$Pr_{2,A_i} : u_i \rightarrow x_i$

(c) The NP system for consensus protocol

FIG. 1. Membrane systems as a framework for multi-robot systems

- u_i is the control input signal for the agent and it is computed using the consensus protocol;
- x_{i0} is the initial state of the agents.

In the *Consenus* membrane, we define a *consensus* rule for each agent. The *consensus* rule uses the consenus protocol to compute the input of the corresponding agent by summing the difference between the agent's state and

its neighbors' states with given weights. The weights represent the importance of the data received from a neighbor and remain constant during the mission.

Each *Agent* membrane corresponding to a robot contains two rules. Rule Pr_{2,A_i} adds the contribution of the input signal u_i to the state of the robot x_i, and consumes it. Rule Pr_{1,A_i} copies the state of the robot x_i to the next time step $t+1$ and also sets the output of the agent y_i.

In this framework the implementation of the control input on the physical robot is performed using lower level membrane systems, such as obtacle avoidance, position control and position estimation [3, 2, 18, 21, 17]. Figure 1(a) shows the generic structure of a membrane controller for multiple robots and also highlights that they can communicate with each other. Rule Pr_{2,A_i} initiates the execution of the low level membrane controller in order to perform the correponding physical motion.

In the proposed framework shown in figure 1 we define a top level of robot management which is responsible with allocating robots to groups based on mission tasks and with dinamically reassigning robots to groups based on online demands and constraints during the execution of the mission. This coordination is performed in a distributed and self-organizing manner. Figure 1(b) shows the robots organized in two groups and seven subgroups. It also shows how agents can be moved between subgroups of the same group or between groups. Each group performs a specific task such as grouping at a position. Grouping at a position is implemented using the *Consensus* membrane shown in figure 1(c).

This study proves the utility of membrane computing as a modeling framework for distributed and descentralized control systems. We further investigate the membrane computing modeling framework in the context of networked systems.

5. **Conclusions**

We propose a membrane system to model multi-agent cooperation. This module can be easily integrated in a robot swarm architecture to facilitate communications between agents. We will further explore the benefits of membrane systems in the context of networked systems. However, these preliminary results show that the utility of membrane systems can be extended beyond the low level control of autonomous robots. We model consenus protocol for communication in a simple membrane structured module. This module can be easily integrated into the proposed multi-agent framework. Future work includes integrate and test the proposed membrane module on simulated and real robots.

REFERENCES

[1] *R. W. Beard*, *J. Lawton*, and *F. Y. Hadaegh*. A coordination architecture for spacecraft formation control. *IEEE Transactions on control systems technology*, **9**(6):777–790, 2001.

[2] *C. Buiu*, *A. B. Pavel*, *C. I. Vasile*, and *I. Dumitrache*. Perspectives of Using Membrane Computing in the Control of Mobile Robots. In *Proceedings of Beyond AI: Interdisciplinary Aspects of Artificial Intelligence(BAI 2011), Pilsen, Czech Republic*, pages 21–26, December 2011.

[3] *C. Buiu*, *C. I. Vasile*, and *O. Arsene*. Development of Membrane Controllers for Mobile Robots. *Information Sciences*, **187**:33–51, March 2012.

[4] *M.C. De Gennaro* and *A. Jadbabaie*. Formation control for a cooperative multi-agent system using decentralized navigation functions. In *American Control Conference*, pages 1346–1351, Minneapolis, MN, June 2006.

[5] *M. Egerstedt* and *X. Hu*. Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, **17**(6):947–951, 2001.

[6] *V. Gazi* and *K.M. Passino*. Stability analysis of swarms. *IEEE Transactions on Automatic Control*, **48**(4):692–697, April 2003.

[7] *A. Jadbabaie*, *Jie Lin*, and *AS. Morse*. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, **48**(6):988–1001, June 2003.

[8] *N.E. Leonard* and *E. Fiorelli*. Virtual leaders, artificial potentials and coordinated control of groups. In *IEEE Conference on Decision and Control*, volume **3**, pages 2968–2973, 2001.

[9] *P. Ogren*, *E. Fiorelli*, and *N.E. Leonard*. Formations with a mission: Stable coordination of vehicle group maneuvers. In *Symposium on Mathematical Theory of Networks and Systems*, page 15, July 2002.

[10] *R. Olfati-Saber*, *J.A. Fax*, and *R.M. Murray*. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, **95**(1):215–233, Jan 2007.

[11] *R. Olfati-Saber* and *R.M. Murray*. Consensus problems in networks of agents with switching topology and time-delays. *Automatic Control, IEEE Transactions on*, **49**(9):1520–1533, Sept 2004.

[12] *Gh. Păun*. Computing with Membranes. *Journal of Computer and System Sciences*, **61**:108–143, 2000.

[13] *Gh. Păun*. *Membrane Computing - An Introduction*. Springer-Verlag, Berlin, 2002.

[14] *Gh. Păun* and *R. Păun*. Membrane Computing and Economics: Numerical P Systems. *Fundamenta Informaticae*, pages 213–227, 2004.

[15] *Gh. Păun*, *G. Rozenberg*, and *A. Salomaa*, editors. *The Oxford Handbook of Membrane Computing*. Oxford University Press, 2010.

[16] *A. B. Pavel*, *O. Arsene*, and *C. Buiu*. Enzymatic Numerical P Systems - a New Class of Membrane Computing Systems. In *The IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2010) Liverpool*, pages 1331–1336, September 2010.

[17] *A. B. Pavel* and *C. Buiu*. Using Enzymatic Numerical P Systems for Modeling Mobile Robot Controllers. *Natural Computing*, 2012.

[18] *A. B. Pavel*, *C. I. Vasile*, and *I. Dumitrache*. Robot Localization Implemented with Enzymatic Numerical P Systems. In *Living Machines 2012: The International Conference on Biomimetic and Biohybrid Systems*, Proceedings in Lecture Notes in Artificial Intelligence, Barcelona, Spain, July 2012. Springer.

[19] *Wei Ren*. Consensus based formation control strategies for multi-vehicle systems. In *American Control Conference (ACC)*, June 2006.

[20] *H.G. Tanner*, *A. Jadbabaie*, and *G.J. Pappas*. Stable flocking of mobile agents part I: dynamic topology. In *IEEE Conference on Decision and Control*, volume **2**, pages 2016–2021, Dec 2003.

[21] *C. I. Vasile*, *A. B. Pavel*, *J. Kelemen*, and *I. Dumitrache*. Implementing obstacle avoidance and follower behaviors on Koala robots using Numerical P Systems. In *Proceedings of the 10th Brainstorming Week on Membrane Computing*, volume **2**, pages 215–228, Seville, Spain, January-February 2012.