On the Efficiency of Cell-Like and Tissue-Like Recognizing Membrane Systems

Miguel A. Gutiérrez-Naranjo, Mario J. Pérez-Jiménez,
Agustín Riscos-Núñez, Francisco J. Romero-Campero
Research Group on Natural Computing, Department of Computer Science and Artificial Intelligence, ETS Ingeniería Informática, University of Sevilla, Avda. Reina Mercedes s/n, Sevilla, Spain

Cell-like recognizing membrane systems are computational devices in the framework of membrane computing inspired from the structure of living cells, where biological membranes are arranged hierarchically. In this paper tissue-like recognizing membrane systems are presented. The idea is to consider that membranes are placed in the nodes of a graph, mimicking the cell intercommunication in tissues.
In this context, polynomial complexity classes associated with recognizing membrane systems can be defined. We recall the definition for cell-like systems, and we introduce the corresponding complexity classes for the tissue-like case. Moreover, in this paper two efficient solutions to the satisfiability problem are analyzed and compared from a complexity point of view.

1. INTRODUCTION

Membrane computing is a young branch of natural computing providing distributed parallel computational devices called *membrane systems*, which are inspired from some basic biological features of living cells, as well as from the cooperation of cells in tissues, organs, and organisms.

In this area, there are basically two ways to consider computational devices: cell-like membrane systems and tissue-like membrane systems. The first one uses membranes arranged hierarchically, inspired from the structure of the cell, and the second one uses membranes placed in the nodes of a graph, inspired from the cell intercommunication in tissues.

In the past years, several computing models using powerful tools from nature have been developed (because of this, they are known as *bioinspired* models) and several solutions in polynomial time to **NP**-complete problems have been presented, making use of nondeterminism and/or of an exponential amount of space. This is the reason why a practical implementation of such models (in biological, electronic, or other media) could provide a significant advance in the resolution of computationally hard problems.

In this paper, we present recognizing membrane systems (both cell-like and tissue-like variants) as a framework to address ways to efficiently solve computationally hard problems, capturing the true concept of algorithm instead of providing a (classical) nondeterministic solution. Also, we present a solution to the satisfiability problem in both variants.

The paper is structured as follows. First, in Section 2, cell-like recognizing membrane systems are defined, together with the complexity classes associated with them. In this context, Section 3 discusses the **P** versus **NP** problem, and Section 4 shows how membrane division rules in the *active membranes* model allow for polynomial-time solutions to **NP**-complete problems. We conclude the cell-like approach by solving the satisfiability problem in a linear time, using polarizationless active membranes. Then, in Section 5 we introduce polarizationless tissue-like recognizing membrane systems with active membranes (adapting the general definition of recognizing systems), and we also introduce the corresponding complexity classes (analogously as for the cell-like case). We also present a quadratic solution to the satisfiability problem in this framework. Finally, conclusions are presented in the last section.

2. CELL-LIKE RECOGNIZING MEMBRANE SYSTEMS

In the structure and functioning of a cell, biological *membranes* play an essential role. The cell is separated from its environment by means of a *skin membrane*, and it is internally compartmentalized by means of *internal membranes*.

The main *syntactic* components of a cell-like membrane system (also called *P system*, see Ref. 1 for details) are a *membrane structure*, *multisets*, and *evolution rules*. A *membrane structure* consists of several membranes arranged in a hierarchical structure inside a main membrane (the *skin*), and delimiting *regions* (the space in-between a membrane and the immediately inner membranes, if any). Each membrane identifies a region inside the system. A membrane structure can be considered as a rooted tree. Regions defined by a membrane structure contain objects corresponding to chemical substances present in the compartments of a cell. The objects can be described by symbols or by strings of symbols, in such a way that *multiset of objects* are placed in regions of the membrane structure. The objects can evolve according to given *evolution rules*, associated with the regions (hence, with the membranes).

The *semantics* of the cell-like membrane systems is defined through a nondeterministic and synchronous model (in the sense that a global clock is assumed) as follows: A *configuration* of a cell-like membrane system consists of a membrane structure and a family of multisets of objects associated with each region

of the structure. At the beginning, there is a configuration called the *initial configuration* of the system. In each time unit, we can transform a given configuration in another configuration by applying the evolution rules to the objects placed inside the regions of the configurations, in a nondeterministic and maximally parallel manner (the rules are chosen in a nondeterministic way, and in each region all objects that can evolve must do it). In this way, we get *transitions* from one configuration of the system to the next one. A *computation* of the system is a (finite or infinite) sequence of configurations such that each one is obtained from the previous one by a transition, and shows how the system is evolving. A computation that reaches a configuration where no more rules can be applied to the existing objects is called a *halting computation*. The result of a halting computation is usually defined through the multiset associated with a specific output membrane (or the environment) in the final configuration.

In the basic definition, cell-like membrane systems can be seen as generative devices, working in a nondeterministic and maximally parallel manner, with output membrane, and without input membrane. However, as we are interested in using cell-like membrane systems for solving decision problems, we can adapt the definition as follows:

DEFINITION 2.1. *A cell-like membrane system with external output is said to be a* recognizing *system if: (a) The working alphabet contains two distinguished elements* `yes` *and* `no`*; (b) all computations halt, and if C is a computation of the system, then either some object* `yes` *or some object* `no` *(but not both) must have been released into the environment in the last step of the computation.*

A computation of a recognizing system is said to be an *accepting computation* (respectively, *rejecting computation*) if the object `yes` (respectively, `no`) appears in the environment associated with the corresponding halting configuration.

We want these kinds of systems (which are nondeterministic devices) to properly solved decision problems according to the true algorithmic concept. With this aim, instead of using classical nondeterministic acceptance (it is enough that *one* computation gives an affirmative answer), it is necessary to require a condition of *confluence*; that is, the system processing an instance of the problem must always give the same answer in all computations. This idea leads us to the concepts of soundness and completeness.

2.1. Soundness and Completeness

A family of cell-like recognizing P systems will provide a solution to a decision problem if for each instance of the problem: (a) if *there exists an* accepting computation of the membrane system processing it, then the instance of the problem has an affirmative answer (*soundness*); and (b) if the instance of the problem has an affirmative answer, then *any* computation of the system processing that instance is an accepting computation (*completeness*).

Next, we formalize these ideas in the following definition:

DEFINITION 2.2. *Let $X = (I_X, \theta_X)$ be a decision problem. Let $\mathbf{\Pi} = (\Pi(w))_{w \in I_X}$ be a family of recognizing membrane systems* without input.

- *The family $\mathbf{\Pi}$ is sound with regard to X if for each instance $w \in I_X$ such that there exists an accepting computation of $\Pi(w)$, we have $\theta_X(w) = 1$.*
- *The family $\mathbf{\Pi}$ is complete with regard to X if for each instance $w \in I_X$ such that $\theta_X(w) = 1$, we have every computation of $\Pi(w)$ is an accepting computation.*

These concepts can be extended to families of cell-like recognizing membranes *with input membrane* in a natural way, but in this case a P system belonging to the family can process several instances of the problem, provided that an appropriate input is supplied to the system.

DEFINITION 2.3. *A* P system with input *is a tuple (Π, Σ, i_Π), where. (a) Π is a P system with working alphabet Γ, with p membranes labeled by $1, \ldots, p$, and initial multisets $\mathcal{M}_1, \ldots, \mathcal{M}_p$ associated with them; (b) Σ is an (input) alphabet strictly contained in Γ, and the initial multisets are over $\Gamma - \Sigma$; and (c) i_Π is the label of a distinguished (input) membrane.*

If m is a multiset over Σ, then the *initial configuration of* (Π, Σ, i_Π) *with input* m is $(\mu, \mathcal{M}_1, \ldots, \mathcal{M}_{i_\Pi} \cup m, \ldots, \mathcal{M}_p)$.

DEFINITION 2.4. *Let $X = (I_X, \theta_X)$ be a decision problem. Let $\mathbf{\Pi} = (\Pi(n))_{n \in \mathbf{N}}$ be a family of cell-like recognizing P systems* with input. *A* polynomial encoding *of X in $\mathbf{\Pi}$ is a pair (cod, s) of polynomial time computable functions over I_X such that for each instance $w \in I_X$, $s(w)$ is a natural number and $cod(w)$ is an input multiset of the system $\Pi(s(w))$.*

DEFINITION 2.5. *Let $X = (I_X, \theta_X)$ be a decision problem. Let $\mathbf{\Pi} = (\Pi(n))_{n \in \mathbf{N}}$ be a family of cell-like recognizing membrane systems* with input. *Let (cod, s) be a polynomial encoding of X in $\mathbf{\Pi}$.*

- *The family $\mathbf{\Pi}$ is sound with regard to (X, cod, s) if for each instance $w \in I_X$ such that there exists an accepting computation of $\Pi(s(w))$ with input $cod(w)$, we have $\theta_X(w) = 1$.*
- *The family $\mathbf{\Pi}$ is complete with regard to (X, cod, s) if for each instance $w \in I_X$ such that $\theta_X(w) = 1$, we have every computation of $\Pi(s(w))$ with input $cod(w)$ is an accepting computation.*

Next, we consider different complexity classes in the framework of cell-like recognizing membrane systems.

2.2. Polynomial Semiuniform Solutions

The first results about solvability of **NP**-complete problems in polynomial time by membrane systems were given by Păun,[2] Zandron et al.,[3] Krishna and Rama,[4] and Obtulowicz,[5] in the framework of membrane systems that lack an input membrane.

Thus, the constructive proofs of such results design *one* system for *each* instance of the problem.

In this context, let us define polynomial complexity classes in cell-like recognizing P systems without input. To solve a decision problem we need, then, to associate with each instance of the problem a system which decides the instance.

DEFINITION 2.6. *Let $\mathcal{R}$ be a class of cell-like recognizing P systems without input membrane. A decision problem $X = (I_X, \theta_X)$ is solvable in polynomial time by a family $\mathbf{\Pi} = (\Pi(w))_{w \in I_X}$, of P systems of $\mathcal{R}$, and we denote it by $X \in \mathbf{PMC}^*_{\mathcal{R}}$, if the following holds:*

- *The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines; that is, there exists a deterministic Turing machine working in polynomial time which constructs the system $\Pi(w)$ from the instance $w \in I_X$.*
- *The family $\mathbf{\Pi}$ is polynomially bounded; that is, there exists a polynomial function $p(n)$ such that for each $w \in I_X$, every computation of $\Pi(w)$ halt in, at most, $p(|w|)$ steps.*
- *The family $\mathbf{\Pi}$ is sound and complete with regard to X.*

We say that the family $\mathbf{\Pi}$ is a *semiuniform solution* to the problem X.

As a direct consequence of working with recognizing membrane systems we have these complexity classes are closed under complement. Moreover, they are closed under polynomial time reduction.

2.3. Polynomial Uniform Solutions

Next, we deal with cell-like recognizing P systems *with input membrane* and we propose to solve problems in an *uniform* way in the following sense: All instances of a decision problem that have the same *size* (according to a prefixed polynomial time computable criterion) are processed by the same system, on which an appropriate input is supplied.

Let us formalize these ideas in the following definition:

DEFINITION 2.7. *A decision problem $X = (I_X, \theta_X)$ is solvable in polynomial time by a family of cell-like recognizing P systems with input $\mathbf{\Pi} = (\Pi(n))_{n \in \mathbf{N}}$, and we denote it by $X \in \mathbf{PMC}_{\mathcal{R}}$, if the following holds:*

- *The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines; that is, there exists a deterministic Turing machine that constructs in polynomial time the system $\Pi(n)$ from $n \in \mathbf{N}$.*
- *There exists a polynomial encoding (cod, s) of X in $\mathbf{\Pi}$ such that*
  - *The family $\mathbf{\Pi}$ is polynomially bounded with regard to (X, cod, s); that is, there exists a polynomial function $p(n)$ such that for each $w \in I_X$, every computation of the system $\Pi(s(w))$ with input $cod(w)$ is halting and, moreover, it performs at most $p(|w|)$ steps.*
  - *The family $\mathbf{\Pi}$ is sound and complete with regard (X, cod, s).*

These complexity classes are closed under complement. Moreover, they are closed under polynomial time reduction.

3. P VERSUS NP PROBLEM IN THE CONTEXT OF CELL-LIKE RECOGNIZING MEMBRANE SYSTEMS

We consider deterministic Turing machines as language recognizing devices. Then, we can associate with each deterministic Turing machine a decision problem, which will allow us to define when such a machine is simulated by a family of P systems (this issue was also addressed, e.g., in Refs. 6 and 7).

DEFINITION 3.1. *Let M be a Turing machine with input alphabet Σ_M. The* decision problem associated with *M is the problem $X_M = (I, \theta)$, where $I = \Sigma_M^*$, and for every $w \in \Sigma_M^*$, $\theta(w) = 1$ if and only if M accepts w.*

Obviously, the decision problem X_M is solvable by the Turing machine M.

DEFINITION 3.2. *We say that a Turing machine, M, is simulated in polynomial time by a family of cell-like recognizing P systems of the class $\mathcal{R}$, if $X_M \in \mathbf{PMC}_{\mathcal{R}}$.*

In cell-like membrane systems, evolution rules, communication rules, and rules involving dissolution are called *basic rules*. Note that by applying this kind of rules the size of the membrane structure does not increase. Hence, it is not possible to construct an exponential number of membranes in polynomial time using only basic rules in a cell-like membrane system.

We recall here a result from Chapter 9 of Ref. 8.

PROPOSITION 3.1. *Let M be a deterministic Turing machine working in polynomial time. Then M can be simulated in polynomial time by a family of cell-like recognizing P systems using only basic rules.*

Reciprocally, in Ref. 7 the following result was proved:

PROPOSITION 3.2. *For every decision problem solvable in polynomial time by a family of cell-like recognizing P systems using only basic rules, there exists a Turing machine solving it in polynomial time.*

Under the hypothesis $\mathbf{P} \neq \mathbf{NP}$, Zandron et al.[3] established the limitations of cell-like membrane systems, which use only basic rules concerning the efficient solution to **NP**-complete problems. This result was generalized by Pérez-Jiménez et al.[7] obtaining the following two characterizations of the $\mathbf{P} \neq \mathbf{NP}$ relation by means of unsolvability results in polynomial time for **NP**-complete problems by families of cell-like recognizing membrane systems using only basic rules.

THEOREM 3.1. *The following assertions are equivalent:*

1. $\mathbf{P} \neq \mathbf{NP}$.
2. *There exists an* **NP***-complete decision problem unsolvable in polynomial time by a family cell-like recognizing membrane systems using only basic rules.*

3. *Each* **NP***-complete decision problem is unsolvable in polynomial time by a family of cell-like recognizing membrane systems using only basic rules.*

Let us denote by $\mathcal{RB}$ the class of cell-like recognizing membrane systems using only basic rules. In Ref. 9 the complexity class **P** has been characterized in terms of cell-like recognizing P systems, by proving the following theorem.

THEOREM 3.2. $\mathbf{P} = \mathbf{PMC}_{\mathcal{RB}}$.

4. RECOGNIZING CELL-LIKE P SYSTEMS WITH ACTIVE MEMBRANES

A particularly interesting class of cell-like membrane systems is the systems with active membranes, where the membrane division can be used to solve computationally hard problems, e.g., **NP**-complete problems, in polynomial or even linear time, by a space-time trade-off.

DEFINITION 4.1. *A recognizing cell-like P system* (Π, Σ, i_Π) *is called with active membranes, if the rules of* Π *are of the following forms* (Γ *being the working alphabet, and* H *the set of labels of* Π)*:*

(a) $[\, a \to \omega \,]_h^\alpha$ *for* $h \in H$, $\alpha \in \{+, -, 0\}$, $a \in \Gamma$, $\omega \in \Gamma^*$*: An object* a *within a membrane labeled with* h *and polarity* α *evolves to a multiset* ω*.*

(b) $a\,[\ \ ]_h^{\alpha_1} \to [\, b \,]_h^{\alpha_2}$ *for* $h \in H$, $\alpha_1, \alpha_2 \in \{+, -, 0\}$, $a, b \in \Gamma$*: An object from the region immediately outside a membrane labeled with* h *is introduced in this membrane, possibly transformed into another object, and simultaneously, the polarity of the membrane can be changed.*

(c) $[\, a \,]_h^{\alpha_1} \to b\,[\ \ ]_h^{\alpha_2}$ *for* $h \in H$, $\alpha_1, \alpha_2 \in \{+, -, 0\}$, $a, b \in \Gamma$*: An object is sent out from membrane labeled with* h *to the region immediately outside, possibly transformed into another object, and simultaneously, the polarity of the membrane can be changed.*

(d) $[\, a \,]_h^\alpha \to b$ *for* $h \in H$, $\alpha \in \{+, -, 0\}$, $a, b \in \Gamma$*: A membrane labeled with* h *is dissolved in reaction with an object. The skin is never dissolved.*

(e) $[\, a \,]_h^{\alpha_1} \to [\, b \,]_h^{\alpha_2}\, [\, c \,]_h^{\alpha_3}$ *for* $h \in H$, $\alpha_1, \alpha_2, \alpha_3 \in \{+, -, 0\}$, $a, b, c \in \Gamma$*: A membrane can be divided into two membranes with the same label, possibly transforming some objects and their polarities.*

These rules are applied according to the following principles:

- All the rules are applied in parallel and in a maximal manner. In one step, one object of a membrane can be used by only one rule (chosen in a nondeterministic way), but any object which can evolve by one rule of any form, must evolve.
- If a membrane is dissolved, its content (multiset and internal membranes) is left free in the surrounding region.
- If at the same time a membrane labeled by h is divided by a rule of type (e) and there are objects in this membrane which evolve by means of rules of type (a), then we suppose that first the evolution rules of type (a) are used, and then the division is produced. Of course, this process takes only one step.
- The rules associated with membranes labeled by h are used for all copies of this membrane. At one step, a membrane can be the subject of *only one* rule of types (b)–(e).

Let us denote by $\mathcal{AM}$ the class of recognizing P systems with active membranes using 2-division. Different polynomial time solutions to **NP**-complete problems have been obtained using this class of cell-like recognizing membrane systems: *Knapsack*,[10] SAT,[11] *Clique*,[12] and *Bin Packing*.[13] Hence, the following proposition holds:

PROPOSITION 4.1. **NP** $\subseteq$ **PMC**$_{\mathcal{AM}}$, *and* **co-NP** $\subseteq$ **PMC**$_{\mathcal{AM}}$.

Moreover, Sosík[14] proved that **PSPACE** $\subseteq$ **PMC**$^{*}_{\mathcal{AM}^0(+d,+ne)}$, where $\mathcal{AM}^0(+d, +ne)$ is the class of recognizing P systems with active membranes allowing division for nonelementary membranes.

Therefore, the complexity class **PMC**$_{\mathcal{AM}}$ does not seem precise enough to describe classical complexity classes below **NP**. Consequently, it is challenging to investigate weaker models of cell-like membrane systems able to characterize classical complexity classes.

Following this line, several efficient solutions to **NP**-complete problems have been obtained within the following variants of cell-like P systems with active membranes:

- P systems with active membranes but using only two electrical charges (Alhazov,[15] Riscos[16];
- P systems with active membranes without polarizations, but using bistable catalysts (Pérez and Romero[17]);
- P systems without polarizations, without cooperation, without priorities, without label changing, without division, but using three types of membrane rules: separation, merging, and release (Pan et al.[18]);
- P systems with separation rules instead of division rules, in two different cases: in the first, using polarizations and separation rules; and in the second one, without polarizations but using separation rules that can change membrane labels (Pan and Ishdorj[19]).

We can define polarizationless cell-like P systems with active membranes in a similar manner by removing electrical charges, that is, considering only rules of the following types:

(a) $[\, a \to u \,]_h$ for $h \in H$, $a \in \Gamma$, $u \in \Gamma^*$.
(b) $a\, [\ \ ]_h \to [\, b \,]_h$ for $h \in H$, $a, b \in \Gamma$.
(c) $[\, a \,]_h \to b\, [\ \ ]_h$ for $h \in H$, $a, b \in \Gamma$.
(d) $[\, a \,]_h \to b$ for $h \in H$, $a, b \in \Gamma$.
(e) $[\, a \,]_h \to [\, b \,]_h\, [\, c \,]_h$ for $h \in H$, $a, b, c \in \Gamma$.

We denote by $\mathcal{AM}^0$ the class of polarizationless cell-like recognizing P systems with active membranes.

At the beginning of 2005, Păun (problem **F** from Ref. 20) wrote: "My favorite question (related to complexity aspects in P systems with active membranes and with electrical charges) is that about the number of polarizations. Can the polarizations be completely avoided? The feeling is that this is not possible—and such a result

would be rather sound: passing from no polarization to two polarizations amounts to passing from non-efficiency to efficiency."

That is, formally we can formulate the so-called Păun's conjecture as follows: "The class of decision problems solvable in polynomial time by families of cell-like recognizing P systems belonging to $\mathcal{AM}^0$ is the standard complexity class **P**."

We denote by $\mathcal{AM}^0(\alpha, \beta)$, where $\alpha \in \{-d, +d\}$ and $\beta \in \{-ne, +ne\}$, the class of all polarizationless cell-like recognizing P systems with active membranes such that (a) if $\alpha = +d$ (resp. $\alpha = -d$) then dissolution rules are permitted (resp. forbidden); and (b) if $\beta = +ne$ (resp. $\beta = -ne$) then division rules for elementary and non-elementary (resp. only division rules for elementary) membranes are permitted.

In the framework of polarizationless cell-like recognizing P systems with active membranes, it turns out that dissolution rules play a surprising role, as they make the difference between efficiency and nonefficiency. More precisely, the following result is presented in Ref. 21:

THEOREM 4.1. *We have the following:*

(1) $\mathbf{P} = \mathbf{PMC}_{\mathcal{AM}^0(-d,\beta)} = \mathbf{PMC}^*_{\mathcal{AM}^0(-d,\beta)}$, for each $\beta \in \{-ne, +ne\}$.
(2) $\mathbf{NP} \cup \mathbf{co} - \mathbf{NP} \subseteq \mathbf{PMC}^*_{\mathcal{AM}^0(+d,+ne)}$.

That is, Păun's conjecture has a negative answer if dissolution rules are allowed (assuming that $\mathbf{P} \neq \mathbf{NP}$), and an affirmative answer otherwise.

The inclusion shown in (2) was improved in Ref. 22, where $\mathbf{PSPACE} \subseteq \mathbf{PMC}_{\mathcal{AM}^0(+d,+ne)}$ is proved.

The inclusion relations between complexity classes can be summarized graphically in Figure 1.

4.1. Solving the SAT Problem by Using Polarizationless Cell-Like P Systems with Active Membranes

The first efficient semiuniform solution to SAT (satisfiability problem) was given by Păun,[2] using division for nonelementary membranes. This result was improved by Păun et al. in Ref. 23 using only division for elementary membranes (in that paper a semiuniform solution to the Hamiltonian path problem using membrane creation is also presented).

In this section, we present a semiuniform solution to the SAT problem in linear time by using polarizationless cell-like recognizing P systems with active membranes.

THEOREM 4.2. *The satisfiability of any propositional formula in the conjunctive normal form, using n variables and m clauses, can be decided in a linear time with respect to n by a polarizationless cell-like recognizing P system with active membranes, constructed in linear time with respect to n and m.*

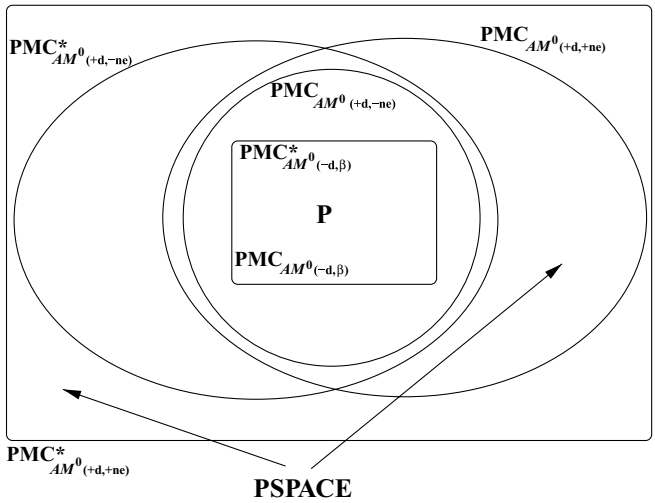

Figure 1. Hierarchy of complexity classes.

Let us consider a propositional formula $\varphi = C_1 \wedge \cdots \wedge C_m$, such that each clause C_j $(1 \leq j \leq m)$, is of the form $C_j = y_{j,1} \vee \cdots \vee y_{j,k_j}$, $k_j \geq 1$, for $y_{j,r} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$. For each $i = 1, 2, \ldots, n$, let us denote

$$t(x_i) = \{c_j \mid \text{there exists } r,\ 1 \leq r \leq k_j, \text{ such that } y_{j,r} = x_i\},$$

$$f(x_i) = \{c_j \mid \text{there exists } r,\ 1 \leq r \leq k_j, \text{ such that } y_{j,r} = \neg x_i\}.$$

These are the sets of clauses that assume the value *true* when x_i is *true*, and when x_i is *false*, respectively.

We construct the cell-like recognizing P system $\Pi(\varphi)$ with the following components.

$$O = \{a_i, f_i, t_i \mid 1 \leq i \leq n\}$$

$$\cup \{c_j, d_j \mid 1 \leq j \leq m\}$$

$$\cup \{p_i \mid 1 \leq i \leq 2n+1\}$$

$$\cup \{r_i \mid 1 \leq i \leq 7\}$$

$$\cup \{b_1, b_2, \texttt{yes}, \texttt{no}\},$$

$$H = \{s, s', p, 0, 1, 2, \ldots, m\},$$

$$\mu = [_s[_{s'}[_0[_p\,]_p[_1\,]_1[_2\,]_2 \cdots [_m\,]_m]_0]_{s'}]_s,$$

$$\mathcal{M}_p = p_1,\ \mathcal{M}_0 = a_1,$$

$$\mathcal{M}_s = \mathcal{M}_{s'} = \mathcal{M}_j = \lambda,\ \text{for all } j = 1, 2, \ldots, m,$$

The set of evolution rules, R, consists of the following rules:

(a) $[\, p_i \to p_{i+1}]_p$, for all $1 \le i \le 2n$,
$[\, r_i \to r_{i+1}]_0$, for all $1 \le i \le 3$,
$[\, r_i \to r_{i+1}]_{s'}$, for all $4 \le i \le 6$.
(b) $[\, a_i]_0 \to [\, f_i]_0 [\, t_i]_0$, for all $1 \le i \le n$,
$[\, f_i \to f(x_i) a_{i+1}]_0$ and $[\, t_i \to t(x_i) a_{i+1}]_0$, for all $1 \le i \le n-1$,
$[\, f_n \to f(x_n)]_0$; $[\, t_n \to t(x_n)]_0$.
(c) $c_j [\ ]_j \to [\, c_j]_j$ and $[\, c_j]_j \to d_j$, for all $1 \le j \le m$.
(d) $[\, p_{2n+1}]_p \to p_{2n+1}$; $[\, p_{2n+1} \to b_1 r_1]_0$.
(e) $b_1 [\ ]_j \to [\, b_1]_j$; and $[\, b_1]_j \to b_2$, for all $1 \le j \le m$,
$[\, b_2]_0 \to b_2$.
(f) $[\, r_4]_0 \to$ `yes` ; $[\,$`yes`$]_{s'} \to$ `yes` ; $[\,$`yes`$]_s \to$ `yes`$[\ ]_s$
$[\, r_7]_{s'} \to$ `no` ; $[\,$`no`$]_s \to$ `no`$[\ ]_s$

4.1.1. An Overview of the Computations

The membranes with labels p, q, and r, with the corresponding objects p_i, q_i, and r_i, respectively, are used as counters, which evolve simultaneously with the *main membrane* 0, where the truth assignments of the n variables $x_1, \ldots, x_n$ are generated. The evolution of the counter p_i is done by the rules of the type (a).

In parallel with these rules, membrane 0 evolves by means of the rules of the type (b). In odd steps (from the step 1 to step $2n$), we divide the (nonelementary) membrane 0 (with f_i, t_i corresponding to the truth values *false, true*, respectively, for variable x_i); in even steps we introduce the clauses satisfied by x_i, $\neg x_i$, respectively. When we divide membrane 0, all inner objects and membranes are replicated; in particular, all membranes with labels $1, 2, \ldots, m$, as well as membrane p, are replicated, hence they are present in all membranes with label 0.

This process lasts $2n$ steps. At the end of this phase, all 2^n truth assignments for the n variables are generated and they are encoded in membranes labeled by 0.

In parallel with the division steps, if a clause C_j is satisfied by the previously expanded variable, then the corresponding object c_j enters membrane j, by means of the first rule of the type (c), permitting their dissolution by means of the second rule of that type and sending objects d_j to membrane 0.

This is done also in steps $2n+1$ and $2n+2$, in parallel with using the rules of type (a) and (d) for evolving membrane p. In step $2n+2$, the second rule of the type (d) produces objects b_1 and r_1 in each membrane 0.

Thus, after $2n+2$ steps, the configuration $\mathcal{C}_{2n+2}$ of the system consists of 2^n copies of membrane 0, each of them containing the membrane p empty, possible objects c_j and d_j, $1 \le j \le m$, as well as copies of only membranes with labels $1, 2, \ldots, m$ corresponding to clauses which were not satisfied by the truth assignment generated in that copy of membrane 0. The clauses satisfied by the truth assignments generated have been dissolved by the corresponding object c_j. Moreover, in that configuration membranes s' and s are empty.

Therefore, formula φ is satisfied if and only if there is a membrane 0 in the configuration $\mathcal{C}_{2n+2}$ without any membrane j $(1, 2, \ldots, m)$ inside it. To check this last condition, we proceed as follows.

In step $2n + 3$, we use the first rule of the type (e) which introduces the object b_1 in each membrane j which has not been dissolved. In parallel, the counter r keeps evolving. The object b_1 in membrane j (step $2n + 4$) dissolves that membrane producing an object b_2 in membrane 0.

In step $2n + 5$, the counter r_3 evolves to r_4 and, simultaneously, each membrane 0 containing an object b_2 is dissolved by the third rule of the type (e). Then, formula φ is satisfied if and only if in the configuration $\mathcal{C}_{2n+5}$ there exists a membrane 0 that has not been dissolved (and so, containing the object r_4).

In the next step, objects r_4 in membrane 0 produce objects `yes`, and objects r_4 in membrane s' produce objects r_5. In step $2n + 7$ the object r_5 in membrane s' evolves to r_6. Simultaneously, if the formula φ is satisfiable then an object `yes` dissolves the membrane s' by applying the second rule of the type (f) producing an object `yes` in the skin, that in the next step is sent to the environment; and the P system halts.

If the formula φ is not satisfiable, then in configuration $\mathcal{C}_{2n+7}$ the membrane s' has not been dissolved and each object r_6 appearing in that membrane, evolves to r_7 in the next step. Then, in step $2n + 9$ an object r_7 dissolves membrane s' producing an object `no` in the skin, by using the fourth rule of the type (f), and in the next step sends to the environment an object `no`; and the system halts.

Therefore, if the formula is satisfiable, then the object `yes` exits the system in step $2n + 8$, and, if the formula is not satisfiable, then the object `no` exits the system in step $2n + 9$. In both cases, this is the last step of the computation.

The system Π uses an alphabet of $5n + 2m + 12$ objects, $m + 4$ initial membranes containing initially only 2 objects, and $5n + 2m + 15$ rules. The length of any rule is bounded by $m + 3$. Clearly, all computations halt (after at most $2n + 9$ steps) and all of them give the same answer, `yes` or `no`, to the question whether formula φ is satisfiable.

5. TISSUE-LIKE RECOGNIZING MEMBRANE SYSTEMS WITH ACTIVE MEMBRANES

In this section, we consider computational devices inspired from the cell intercommunication in tissues, and adding the ingredient of cell division rules of the same form as in the case of polarizationless cell-like membrane systems with active membranes.

DEFINITION 5.1. *A polarizationless tissue-like membrane system with active membranes of the degree $p \geq 1$ is a tuple $\Pi = (\Gamma, \Sigma, \mathcal{M}_1, \ldots, \mathcal{M}_p, E, R, i_{in})$, where*

1. Γ *is the working alphabet containing two distinguished objects* `yes` *and* `no`*;*
2. Σ *is an* (*input*) *alphabet strictly contained in* Γ.

3. $\mathcal{M}_1, \ldots, \mathcal{M}_p$ *are multisets over* $\Gamma - \Sigma$*, describing the objects placed in the cells of the system (we suppose that at least one copy of* `yes` *and* `no` *is in some of these multisets);*
4. $E \subseteq \Gamma$ *is the set of objects present in the environment in arbitrary many copies each (the objects* `yes` *and* `no` *are not present in* E*);*
5. R *is a finite set of developmental rules, of the following forms:*
   (a) $(i, u/v, j)$*, for* $i, j \in \{0, 1, 2, \ldots, p\}$*,* $i \neq j$*, and* $u, v \in \Gamma^*$*; 1, 2, . . . , p identify the cells of the system, 0 is the environment: When applying a rule* $(i, u/v, j)$*, the objects of the multiset represented by* u *are sent from region* i *to region* j *and simultaneously the objects of the multiset* v *are sent from region* j *to region* i*;*
   (b) $[\,a\,]_i \to [\,b\,]_i[\,c\,]_i$*, where* $i \in \{1, 2, \ldots, p\}$ *and* $a, b, c \in \Gamma$*: Under the influence of object* a*, the cell with label* i *is divided in two cells with the same label; in the first copy the object* a *is replaced by* b*, in the second copy the object* a *is replaced by* c*; all other objects are replicated and copies of them are placed in the two new cells.*
6. $i_{in} \in \{1, \ldots, n\}$ *is the label of the input cell.*

Let m be a multiset over Σ. The *initial configuration of* Π *with input* m is the tuple $(\mathcal{M}_1, \ldots, \mathcal{M}_{i_{in}} \cup m, \ldots, \mathcal{M}_p)$.

The rules of a tissue-like membrane system as mentioned above are used in the nondeterministic maximally parallel manner as customary in membrane computing. In each step, we apply a set of rules which is maximal (no further rule can be added), with the following important restriction: if a cell is divided, then the division rule is the only one which is applied for that cell in that step, its objects do not participate in any communication rule.

The computation starts from the initial configuration and proceeds as defined above; only halting computations give a result, and the result is given by the presence of a distinguished object in the environment (i.e., we work with systems with external output).

DEFINITION 5.2. *A tissue-like membrane system is said to be a* recognizing *system if (a) the working alphabet contains two distinguished elements* `yes` *and* `no`*; (b) all computations halt, and if* $\mathcal{C}$ *is a computation of the system, then either some object* `yes` *or some object* `no` *(but not both) must have been released into the environment in the last step of the computation.*

We say that a computation is an accepting (rejecting) computation if the object `yes` (respectively, the object `no`) appears in the environment associated with the corresponding halting configuration.

We denote by $\mathcal{TR}$ the class of polarizationless tissue-like recognizing membrane systems with active membranes.

To present the concept of uniform solvability in the framework of tissue-like membrane systems, we translate the concept of polynomial encoding.

DEFINITION 5.3. *Let* L *be a language, and* $\mathbf{\Pi} = (\Pi(n))_{n \in \mathbf{N}}$ *a family of tissue-like systems of* $\mathcal{TR}$*. A polynomial encoding of* L *in* $\mathbf{\Pi}$ *is a pair* (cod, s) *of polynomial-time computable functions whose domain is* L*, and for each* $u \in L$*,* $s(u)$ *is a natural number and* $cod(u)$ *is an input multiset of the tissue-like system* $\Pi(s(u))$*.*

Next we define the concept of solvability by using polarizationless tissue-like recognizing systems with active membranes.

DEFINITION 5.4. *A decision problem $X = (I_X, \theta_X)$ is solvable in polynomial time by a family $\mathbf{\Pi} = (\Pi(n))_{n\in\mathbf{N}}$, of tissue-like systems of $\mathcal{TR}$, and we denote it by $X \in \mathbf{PMC}_{\mathcal{TR}}$, if the following holds:*

- The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines.
- There exists a polynomial encoding (cod, s) from I_X to $\mathbf{\Pi}$ such that the family $\mathbf{\Pi}$ is polynomially bounded, sound and complete with regard to (X, cod, s).

We also have the class $\mathbf{PMC}_{\mathcal{TR}}$ is closed under polynomial-time reduction and complement.

We shall not explain in detail the way the computations proceed; in particular, they can be nondeterministic, as standard in membrane computing. It is important, however, to note that the systems always stop and they always send out an object which is the correct answer to the instance of the problem that they are processing.

5.1. Solving the SAT Problem by Using Polarizationless Tissue-Like P Systems with Active Membranes

Next, we present a uniform solution to the SAT problem in polynomial time, in a confluent way by using recognizing tissue P systems with active membranes.

THEOREM 5.1. *The satisfiability of any propositional formula in the conjunctive normal form, using n variables and m clauses, can be decided in a quadratic time with respect to n by a polarizationless tissue-like recognizing P system with active membranes, constructed in a quadratic time with respect to n and m.*

Let us consider a propositional formula $\varphi = C_1 \wedge \cdots \wedge C_m$, consisting of m clauses $C_j = y_{j,1} \vee \cdots \vee y_{j,k_j}$, where $y_{j,r} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$ (there are used n variables). Without loss of generality, we may assume that no clause contains two occurrences of some x_i or two occurrences of some $\neg x_i$ (the formula is not redundant at the level of clauses), or both x_i and $\neg x_i$ (otherwise such a clause is trivially satisfiable, hence can be removed).

We consider the family $\mathbf{\Pi} = \{\Pi(\langle n, m\rangle) : n, m \in \mathbf{N}\}$ of tissue-like recognizing membrane systems, being $\langle n, m\rangle = \frac{(n+m)\cdot(n+m+1)}{2} + n$.

The tissue-like recognizing membrane system

$$\Pi(\langle n, m\rangle) = (\Gamma(\langle n, m\rangle), \Sigma(\langle n, m\rangle), \mathcal{M}_1, \mathcal{M}_2, E(\langle n, m\rangle), R(\langle n, m\rangle), i_{in})$$

will process all Boolean formulae in conjunctive normal form with n variables and m clauses, and is defined as follows:

$$\Gamma(\langle n, m\rangle) = \{a_i, t_i, f_i \mid 1 \le i \le n\} \cup \{r_i \mid 1 \le i \le m\}$$
$$\cup \{T_{i,1}, F_{i,1} \mid 1 \le i \le n\}$$
$$\cup \{T'_{i,j}, F'_{i,j} \mid 1 \le i \le n, 1 \le j \le m+1\}$$
$$\cup \{s_{i,j}, s'_{i,j} \mid 1 \le i \le n, 1 \le j \le m\}$$
$$\cup \{b_i \mid 1 \le i \le 3n+m+1\} \cup \{c_i \mid 1 \le i \le n+1\}$$
$$\cup \{d_i \mid 1 \le i \le 3n+nm+m+2\}$$
$$\cup \{e_i \mid 1 \le i \le 3n+nm+m+4\}$$
$$\cup \{f, g, \texttt{yes}, \texttt{no}\},$$
$$\Sigma(\langle n, m\rangle) = \{s_{i,j}, s'_{i,j} \mid 1 \le i \le n, 1 \le j \le m\},$$
$$\mathcal{M}_1 = \texttt{yes no}\ b_1c_1d_1e_1,$$
$$\mathcal{M}_2 = fga_1a_2\ldots a_n,$$
$$E(\langle n, m\rangle) = \Gamma(\langle n, m\rangle) - \{\texttt{yes}, \texttt{no}\},$$
$$i_{in} = 2,$$

and the following rules.

1. $[\, a_i \,]_2 \to [\, T_{i,1} \,]_2\, [\, F_{i,1} \,]_2$, for all $i = 1, 2, \ldots, n$.
2. $(1, b_i/b_{i+1}^2, 0)$, for all $i = 1, 2, \ldots, n+1$.
3. $(1, c_i/c_{i+1}^2, 0)$, for all $i = 1, 2, \ldots, n+1$.
4. $(1, d_i/d_{i+1}^2, 0)$, for all $i = 1, 2, \ldots, n+1$.
5. $(1, e_i/e_{i+1}, 0)$, for all $i = 1, 2, \ldots, 3n+nm+m+3$.
6. $(1, b_{n+1}c_{n+1}/f, 2)$.
7. $(1, d_{n+1}/g, 2)$.
8. $(2, c_{n+1}T_{i,1}/c_{n+1}T'_{i,1}, 0)$.
9. $(2, c_{n+1}F_{i,1}/c_{n+1}F'_{i,1}, 0)$, for each $i = 1, 2, \ldots, n$.
10. $(2, T'_{i,j}/t_iT'_{i,j+1}, 0)$.
11. $(2, F'_{i,j}/f_iF'_{i,j+1}, 0)$, for each $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$.
12. $(2, b_i/b_{i+1}, 0)$.
13. $(2, d_i/d_{i+1}, 0)$, for all $i = n+1, \ldots, (n+1)+(2n+m)-1$.
14. $(2, b_{3n+m+1}t_is_{i,j}/b_{3n+m+1}r_j, 0)$.
15. $(2, b_{3n+m+1}f_is'_{i,j}/b_{3n+m+1}r_j, 0)$, for all $1 \le i \le n$ and $1 \le j \le m$.
16. $(2, d_i/d_{i+1}, 0)$, for all $i = 3n+m+1, \ldots, (3n+m+1)+nm-1$.
17. $(2, d_{3n+nm+m+i}r_i/d_{3n+nm+m+i+1}, 0)$, for all $i = 1, 2, \ldots, m$.
18. $(2, d_{3n+nm+2m+1}/f\ \texttt{yes}, 1)$.
19. $(2, \texttt{yes}/\lambda, 0)$.
20. $(1, e_{3n+nm+2m+2}f\ \texttt{no}/\lambda, 2)$.
21. $(2, \texttt{no}/\lambda, 0)$.

5.1.1. An Overview of the Computations

Membrane 2 is repeatedly divided, each time expanding one object a_i, corresponding to a variable x_i, into $T_{i,1}$ and $F_{i,1}$, corresponding to the values *true* and *false* which this variable may assume. In this way, in n steps, we get 2^n cells with label 2, each one containing one of the 2^n possible truth assignments for the n variables. The objects f, g are duplicated, hence a copy of each of them will appear in each cell.

In parallel with the operation of dividing cell 2, the counters b_i, c_i, d_i, e_i from cell 1 grow their subscripts. In each step, the number of copies of objects of the first three types is doubled, hence after n steps we get 2^n copies of b_{n+1}, c_{n+1}, and d_{n+1}. Objects b_i will check which clauses are satisfied by a given truth assignment, objects c_i are used to multiply the number of copies of t_i, f_i as we will see immediately, d_i are used to check whether there is at least one truth assignment which satisfies all clauses, and if such an assignment does not exist, then e_i will be used in order to produce the object `no` at the end of the computation.

In step $n+1$, the counters $b_{n+1}, c_{n+1}, d_{n+1}$ are brought in cells with label 2, in exchange of f and g. Because we have 2^n copies of each object of these types and 2^n cells 2, each one containing exactly one copy of f and one of g, due to the maximal parallel use of the rules, each cell 2 gets precisely one copy of each of $b_{n+1}, c_{n+1}, d_{n+1}$. Note that cells 2 cannot divide anymore, because the objects a_i were exhausted.

In the presence of c_{n+1}, the objects $T_{i,1}, F_{i,1}$ get primed, which initiates the possibility of introducing m copies of each t_i and f_i in each cell 2. As we have m clauses, then to check their values for a given truth assignment, we need for each clause one set of objects encoding the values of all variables. Note that this phase needs $2n$ steps for priming the objects $T_{i,1}, F_{i,1}$—for each object we need one step, because we have only one copy of c_{n+1} available—then m further steps for each $T'_{i,1}, F'_{i,1}$; all these steps are done in parallel, but for the last primed $T_{i,1}, F_{i,1}$ we have to continue m steps after the $2n$ necessary for priming. Thus, the total number of steps performed in this process is $2n+m$.

In parallel with the previous operations, the counters b_i and d_i increase their subscripts, until reaching the value $3n+m+1$. This is done in all cells 2 at the same time. Simultaneously, e_i increases its subscript in cell 1.

In the presence of b_{3n+m+1}—and not before—we check the values assumed by clauses for the truth assignments from each cell 2. We have only one copy of b_{3n+m+1} in each cell, hence we need at most nm steps for this: each clause contains at most n literals, and we have m clauses. In parallel, d increases the subscript, until reaching the value $3n+nm+m+1$.

In each cell with label 2 we check whether or not all clauses are satisfied by the corresponding truth assignment. For each clause which is satisfied, we increase by one the subscript of d, hence the subscript reaches the value $3n+nm+2m+1$ if and only if all clauses are satisfied.

If one of the truth assignments from a cell 2 has satisfied all clauses, then we reach $d_{3n+nm+2m+1}$, which is sent to cell 1 in exchange of the objects `yes` and f.

In the next step, the object yes leaves the system, signaling the fact that the formula is satisfiable. In cell 1, the counter e will increase one more step its subscript, but after that it will remain unchanged—it can leave cell 1 only in the presence of f, but this object was already moved to cell 2.

If the counter e reaches the subscript $3n + nm + 2m + 2$ and the object f is still in cell 1, then the object no can be moved to a cell 2, randomly chosen, and from there it exits the system, signaling that the formula is not satisfiable.

Next, we justify that the solution provided is an uniform solution.

We consider the polynomial encoding (cod, s) of $I_{\mathtt{SAT}}$ in $\mathbf{\Pi}$, defined as follows: If the formula φ is an instance of SAT with size parameters n (number of variables) and m (number of clauses), then $s(\varphi) = \langle n, m \rangle$ and $cod(\varphi)$ is the set

$$\bigcup_{1 \leq i \leq n, 1 \leq j \leq m, 1 \leq r \leq k_j} \{s_{i,j} \mid y_{j,r} = x_i\} \cup \{s'_{i,j} \mid y_{j,r} = \neg x_i, \}$$

That is, in the multiset $cod(\varphi)$ we replace each variable x_i from each clause C_j with $s_{i,j}$ and each negated variable $\neg x_i$ from each clause C_j with $s'_{i,j}$, then we remove all parentheses and connectives. In this way we pass from φ to $cod(\varphi)$ in a number of steps which is linear with respect to $n \cdot m$.

The presented family of tissue-like recognizing membrane systems is polynomially uniform by Turing machines, because the definition of the family is done in a recursive manner from a given instance of SAT, in particular from the constants n (number of variables) and m (number of clauses). Furthermore the tissue P system $\Pi(\langle n, m \rangle)$ uses an alphabet of $5nm + 17n + 4m + 12$ objects, 2 initial cells containing $n + 8$ objects in all, and n rules. The length of any rule is bounded by 3. Clearly, all computations halt, and the number of steps is bounded by $3n + nm + 2m + 4$ (when the answer is negative; if the answer is affirmative, then the number of steps is $3n + nm + 2m + 2$).

From the above we deduce the following results:

THEOREM 5.2.

1. $\mathtt{SAT} \in \mathbf{PMC}_{\mathcal{TR}}$.
2. $\mathbf{NP} \subseteq \mathbf{PMC}_{\mathcal{TR}}$, and $\mathbf{NP} \cup \mathbf{co} - \mathbf{NP} \subseteq \mathbf{PMC}_{\mathcal{TR}}$.

Remarks. We have presented two solutions to the SAT problem by using polarizationless recognizing membrane systems with active membranes. The first one, in the cell-like framework is semiuniform, does not use cooperation (i.e., the left-hand sides of the rules only have one object), and it is linear in time and in (the initial) space. The second one, in the tissue-like framework is a uniform solution, uses cooperation, and it is quadratic in time and in (the initial) space. Both solutions construct an exponential working space (in terms of membranes or cells) in linear time.

6. CONCLUSIONS

In this paper, we have presented cell-like (inspired from the structure of the cell) and tissue-like (inspired from the cell inter-communication in tissues) recognizing

membrane systems as computational devices specially suitable to attack the efficient solvability of computationally hard problems.

In that new framework, two characterizations of the relation $\mathbf{P} = \mathbf{NP}$ have been described through the solvability of **NP**-complete problems by a family of cell-like recognizing membrane systems using only basic rules.

The main contribution of this paper is to formalize the concept of tissue-like membrane systems, and to present, in the framework of tissue-like recognizing membrane systems, a formal definition of the polynomial complexity class $\mathbf{PMC}_{\mathcal{TR}}$. Besides, two polynomial time solutions to the satisfiability problem by using polarizationless (cell-like and tissue-like) recognizing membrane systems with active membranes are presented.

Finally, we would like to mention two open questions for future research. What happens if in tissue-like P systems division rules are forbidden? What if communication rules are restricted to $(i, a/b, j)$, where i, j are labels and a, b are objects (instead of multisets)?

Acknowledgments

The authors acknowledge the support of the project TIN2005-09345-C04-01 of the Ministerio de Educación y Ciencia of Spain, cofinanced by FEDER funds and of the project of Excellence TIC 581 of the Junta de Andalucía.

References

1. Păun Gh. Membrane computing. An introduction. Berlin: Springer-Verlag; 2002.
2. Păun Gh. P systems with active membranes: attacking NP–complete problems. J Automata, Lang Comb 2001;6(1):75–90.
3. Zandron C, Ferreti C, Mauri G. Solving NP–complete problems using P systems with active membranes. In: Antoniou I, Calude CS, Dinneen MJ, editors. Unconventional models of computation, UMC'2K, Berlin: Springer-Verlag; 2000. pp 289–301.
4. Krishna SN, Rama R. A variant of P systems with active membranes: solving NP–complete problems. Romanian J Inform Sci Technol 1999;2(4):357–367.
5. Obtulowicz A. Deterministic P systems for solving SAT problem. Romanian J Inform Sci Technol 2001;4(1–2):551–558.
6. Pérez–Jiménez MJ. An approach to computational complexity in Membrane Computing. In: Mauri G, Păun, Gh, Pérez–Jiménez MJ, Rozenberg Gr, Salomaa A, editors. Membrane Computing, 5th International Workshop, WMC5, Lecture Notes in Computer Science, Vol. 3365. Berlin: Springer; 2005. pp 85–109.
7. Pérez–Jiménez MJ, Romero–Jiménez A, Sancho–Caparrini F. The P versus NP problem through cellular computing with membranes. In: Jonoska N., Păun Gh, Rozenberg G, editors. Aspects of molecular computing. Lecture Notes in Computer Science, Vol 2950. Berlin: Springer; 2004. pp 338–352.
8. Romero–Jiménez A. Complexity and universality in cellular computing models. PhD. Thesis, University of Seville, Spain, 2003.
9. Gutiérrez–Naranjo MA, Pérez–Jiménez MJ, Riscos–Núñez A, Romero–Campero FJ, Romero–Jiménez A. Characterizing tractability by cell-like membrane systems. In: K.G. Subramanian, K. Rangarajan, M. Mukund, editors. Formal models, languages and applications. MPAI Series, Vol 66. Singapore: World Scientific; 2006. pp 137–154.
10. Pérez–Jiménez MJ, Riscos–Núñez A. A linear-time solution to the knapsack problem using P systems with active membranes. In: Martín-Vide C, Păun Gh, Rozenberg G, Salomaa A,

editors. Membrane computing, Lecture Notes in Computer Science, Vol 2933. Berlin: Springer; 2004. pp. 248–266.
11. Pérez–Jiménez MJ, Romero–Jiménez A, Sancho–Caparrini F. Complexity classes in cellular computing with membranes. Nat Comput 2003;2(3):265–285.
12. Alhazov A, Martín–Vide C, Pan L. Solving graph problems by P systems with restricted elementary active membranes. In: Jonoska N, Păun Gh, Rozenberg G, editors. Aspects of molecular computing, Lecture Notes in Computer Science, Vol 2950. Berlin: Springer; 2004. pp 1–22.
13. Pérez–Jiménez MJ, Romero–Campero FJ. An efficient family of P systems for packing items into bins. J Universal Comput Sci 2004; 10(5):650–670.
14. Sosik P. The computational power of cell division in P systems: Beating down parallel computers? Nat Comput 2003;2(3):287–298.
15. Alhazov A, Pan L, Păun Gh. Trading polarizations for labels in P systems with active membranes. Acta Inform 2004;41(2–3):111–144.
16. Riscos–Núñez A. Cellular programming: efficient resolution of NP–complete numerical problems. Ph.D. Thesis, University of Seville, Spain; 2004.
17. Pérez–Jiménez MJ, Romero–Campero FJ. Trading polarizations for bi-stable catalysts in P systems with active membranes. In: Mauri G, Păun Gh, Pérez–Jiménez MJ, Rozenberg Gr, Salomaa A, editors. Membrane Computing, 5th International Workshop, WMC5, Lecture Notes in Computer Science, Vol. 3365. Berlin: Springer; 2005. pp. 373–388.
18. Pan L, Alhazov A, Ishdorj T-O. Further remarks on P systems with active membranes, separation, merging, and release rules. Soft Comp—A Fusion Found Methodol Appl 2005;9(9):686–690.
19. Pan L, Ishdorj T-O. P Systems with active membranes and separation rules. J Universal Comput Sci 2004;10(5):630–649.
20. Păun Gh. Further twenty six open problems in membrane computing. In: Gutiérrez-Naranjo MA, Riscos-Núñez A, Romero-Campero FJ, Sburlan D, editors. Proceedings of the Third Brainstorming Week on Membrane Computing; 2005. pp 249–262.
21. Gutiérrez–Naranjo MA, Pérez–Jiménez MJ, Riscos–Núñez A, Romero–Campero FJ. On the power of dissolution in P systems with active membranes. In: Lecture Notes in Computer Science. Berlin: Springer; Vol. 3850. 2006; pp 224–240.
22. Alhazov A, Pérez-Jiménez MJ. Uniform solution to QSAT using polarizationless active membranes. In: Durand-Lose J and Margenstern M, editors. Machines, Computations, and Universality, Lecture Notes in Computer Science, Vol. 4664. Berlin: Springer; 2007. pp 122–133.
23. Păun Gh, Suzuki Y, Tanaka H, Yokomori T. On the power of membrane division in P systems. Theor Comput Sci 2004;324(1):61–85.