A Uniform Solution to Common Algorithmic Problem by Tissue P Systems with Cell Division

Yunyun Niu, Linqiang Pan
Key Laboratory on Image Processing and Intelligent Control
Department of Control Science and Engineering
Huazhong University of Science and Technology
Wuhan 430074, Hubei, China
niuyunyun1003@163.com,
lqpan@mail.hust.edu.cn (corresponding author)

Mario J. Pérez-Jiménez
Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla
Sevilla 41012, Spain
marper@us.es

***Abstract*—Common algorithmic problem is an optimization problem, which has the nice property that several other NP-complete problems can be reduced to it in linear time. A tissue P system with cell division is a computing model which has two basic characters: intercellular communication and the ability of cell division. The ability of cell division allows us to obtain an exponential amount of cells in linear time and to design cellular solutions to computationally hard problems in polynomial time. We here present an effective solution to the common algorithmic decision problem using a family of recognizer tissue P systems with cell division.**

***Keywords*-membrane computing; tissue P system; cell division; common algorithmic problem**

I. Introduction

Membrane computing is a new branch of natural computing, which is inspired by the structure and the functioning of living cells [1], as well as the organization of cells in tissues, organs, and other higher order structures. The devices of this model, called P systems, provide distributed parallel and non-deterministic computing models. Since being introduced by Gh. Păun in 1998, it has received important attention from the scientific community. As computer scientists, biologists, formal linguists and complexity theoreticians plug into this area, it has definitely become a rich and exciting realm of cross-disciplinary research. Please refer to [2] for an introduction of membrane computing, refer to [3] for the most recent overview of the field of membrane computing, and refer to [4] for further bibliography.

In last years, many different classes of P systems have been investigated. The most studied variants are the cell-like models of P systems, where membranes are hierarchically arranged in a tree-like structure. Most of them are computationally universal (i.e., able to compute whatever a Turing machine can do), as well as computationally efficient (i.e., able to trade space for time and solve in this way presumably intractable problems in a feasible time) (e.g., [5], [6], [7], [8]).

Another interesting class of P system is tissue P system [9], where instead of considering a hierarchical arrangement, membranes are placed in the nodes of a graph. Tissue P systems are abstracted from the intercellular communication and the cooperation between cells in tissues [10]. Here, we focus on a variant of tissue P systems: tissue P system with cell division [11].

The common algorithmic problem (CAP) [12] is an optimization problem. It can be defined as follows. Let S be a finite set and F be a family of subsets of S. Find the cardinality of a maximal subset of S which does not include any set belonging to F. The sets in F are called forbidden sets. Several other NP–complete problems can be reduced to it in linear time (using a logarithmic bounded space), such as independent set problem, vertex cover problem, maximum clique problem, satisfiability problem, Hamiltonian path problem and tripartite matching problem [12], [13], so we can say that they are subproblems of CAP in the sense of linear time reduction.

In [13], an effective solution to the CAP was proposed using a family of recognizer P systems with active membranes. However, there is no known way to transform a recognizer P system with active membranes to a tissue P system. Tissue P systems with cell division can solve some NP-complete problems in polynomial time, e.g., the subset sum problem [14], the partition problem [15], and the 3–coloring problem [7]. But it remains open how to compute the reduction of an NP problem to another NP–complete problem by P systems. So, in this work, we give a direct solution to the CAP in framework of tissue P systems with cell division.

The paper is organized as follows. Some preliminaries are recalled in section II including the definition of recognizer tissue P systems with cell division. A polynomial–time solution to CAP is presented in section III. Some discussion is presented in section IV.

II. Preliminaries

An *alphabet* Σ is a non empty set, whose elements are called *symbols*. An ordered sequence of symbols is a *string*. The number of symbols in a string u is the *length* of the string, and it is denoted by $|u|$. As usual, the empty string (with length 0) will be denoted by λ. The set of strings of length n built with symbols from the alphabet Σ is denoted by Σ^n and $\Sigma^* = \cup_{n\geq 0}\Sigma^n$. A *language* over Σ is a subset from Σ^*.

A *multiset* m over a set A is a pair (A, f), where f is a map from A to the set of natural numbers $\mathbb{N}$. If $m = (A, f)$ is a multiset then its *support* is defined as $supp(m) = \{x \in A \mid f(x) > 0\}$ and its *size* is defined as $\sum_{x \in A} f(x)$. A multiset is empty (resp. finite) if its support is the empty set (resp. finite).

If $m = (A, f)$ is a finite multiset over A, and $supp(m) = \{a_1, \ldots, a_k\}$, then it will be denoted as $m = \{\{a_1^{f(a_1)}, \ldots, a_k^{f(a_k)}\}\}$. That is, superscripts indicate the multiplicity of each element, and if $f(x) = 0$ for any $x \in A$, then this element is omitted. If $m_1 = (A, f)$ and $m_2 = (A, g)$ are multisets over A, then the union of m_1 and m_2 is defined as $m_1 m_2 = (A, h)$, where $h = f + g$.

A *recognizer tissue P system with cell division* of degree $q \geq 1$ is a tuple of the form

$$\Pi = (\Gamma, \Sigma, \Omega, w_1, \ldots, w_q, \mathcal{R}, i_{in}, i_{out}), \text{ where:}$$

- $q \geq 1$ (the initial degree of the system; the system contains q cells, labeled with $1, 2, \ldots, q$; all these q cells are placed in the environment; the environment is labeled with 0);
- Γ is the working alphabet, which contains two distinguished objects `yes` and `no`, at least one copy of them occurring in some initial multisets w_1, ..., w_q, but not occurring in Ω;
- Σ is an input alphabet strictly contained in Γ;
- $\Omega \subseteq \Gamma$ is the set of objects occurring in the environment, each one in arbitrarily many copies;
- $w_1, \ldots, w_q$ are strings over Γ, describing the multisets of objects located in the cells of the system at the beginning of the computation;
- $\mathcal{R}$ is a finite set of rules of the following forms:
  - (a) Communication rules: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \ldots, q\}, i \neq j, u, v \in \Gamma^*$ ($|u| + |v|$ is called the length of the communication rules $(i, u/v, j)$).
  - (b) Division rules: $[a]_i \rightarrow [b]_i[c_i]$, where $i \in \{1, 2, \ldots, q\}, a \in \Gamma$ and $b, c \in \Gamma \cup \{\lambda\}$.
- $i_{in} \in \{1, \ldots, q\}$ is the input cell;
- $i_{out} \in \{0, 1, \ldots, q\}$ indicates the output region, where $i_{out} = 0$ denotes that the output region is the environment;

All computations halt (that is, they always reach a configuration where no further rules can be applied). During a computation of Π, either the object `yes` or the object `no` (but not both) must be released into the environment, and only in the last step of the computation.

When a division rules $[a]_i \rightarrow [b]_i[c]_i$ is applied, all the objects in the original cells are replicated and copies of them are placed in each of the new cells, with the exception of the objects a, which is replaced by $b \in \Gamma \cup \{\lambda\}$ in the first new cell and by $c \in \Gamma \cup \{\lambda\}$ in the second one.

When a rules $(i, u/v, j)$ is applied, 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. For a cell in the system Π, it is possible to have more than one applicable communication rules in a step. There applicable communication rules are used in non-deterministic maximally parallel manner (the system non-deterministically chooses and applies a multiset of communication rules that is maximal, no further rule can be added).

In general, the rules of a system as above are used in the non-deterministic maximally parallel manner. In each step, all cells which can evolve must evolve in a maximally parallel way. This way of applying rules has only one restriction: when a cell is divided, the division rule is the only one which is applied for that cell in that step; the objects inside that cell do not evolve by means of communication rules. Their labels precisely identify the rules which can be applied to them.

A configuration of Π at an instant t is described by the multisets of objects over Γ associated with all the cells present in the system at that moment, and the multiset over Ω associated with the environment at the instant t. All computations start from the initial configuration and proceed as defined above. A computation $\mathcal{C}$ is called an accepting computation (respectively, rejecting computation) if the object `yes` (respectively, `no`) appears in the environment associated to the corresponding halting configuration of $\mathcal{C}$, and only in the last step of the computation.

Definition 2.1: Let $X = (I_X, \theta_X)$ be a decision problem, where I_X is a language over a finite alphabet (whose elements are called instances) and θ_X is a total boolean function over I_X (that is, a predicate). The decision problem X is solvable in polynomial time by a family $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ of recognizer tissue P systems with cell division 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 $\Pi(n)$ from $n \in \mathbb{N}$.
- There exists a pair (cod, s) of polynomial-time computable functions over I_X such that:
  - for each instance $u \in I_X$, $s(u)$ is a natural number and $cod(u)$ is an input multiset of the system $\Pi(s(u))$;
  - the family $\mathbf{\Pi}$ is *polynomially bounded* with regard to (X, cod, s), that is, there exists a polynomial function p, such that for each $u \in I_X$ every computation of $\Pi(s(u))$ with input $cod(u)$ is halting and, moreover, it performs at most $p(|\ u\ |)$ steps;
  - the family $\mathbf{\Pi}$ is *sound* with regard to (X, cod, s), that is, for each $u \in I_X$, if there exists an accepting computation of $\Pi(s(u))$ with input $cod(u)$, then $\theta_X(u) = 1$;
  - the family $\mathbf{\Pi}$ is *complete* with regard to (X, cod, s), that is, for each $u \in I_X$, if $\theta_X(u) = 1$, then every computation of $\Pi(s(u))$ with input $cod(u)$ is an accepting one.

In the above definition we have imposed to every P system $\Pi(n)$ to be confluent, in the following sense: every compu-

tation of a system with the same input multiset must always give the same answer.

We denote by $\mathbf{PMC}_{TDC}$ the set of all decision problems which can be solved by means of recognizer tissue P systems with cell division in polynomial time.

III. A Solution for Common Algorithmic Decision Problem

The common algorithm decision problem (CADP) can be defined as follows. *Given S a finite set, F a family of subsets of S, and $k \in N$, we ask if there exists a subset A of S such that $|A| \geq k$, and which does not include any set belonging to F. The sets in F are called forbidden sets.*

We address the solution of this problem via a brute force algorithm, in the framework of recognizer tissue P systems with cell division. Our strategy will consist in the following phases:

- *Generation Stage*: The initial cell with label 2 is divided into two new cells; and the division is iterated until we have all possible subsets to the problem (one subset of S for each membrane with label 2). Simultaneously, in the membrane with label 1 there is a counter, and it will determine the moment in which the checking stage starts.
- *Checking Stage*: The system checks whether or not there exists a subset A of S such that A does not include any forbidden set in the family F and $|A| \geq k$.
- *Output Stage*: The system sends to the environment the right answer according to the results of the previous stage.

Let us use a tuple $u = (\{s_1, \cdots, s_n\},\ (B_1, \cdots, B_m),\ k)$ to represent an instance of the problem, where $\{s_1, \cdots, s_n\}$ is a finite set S, $B_i, 1 \leq i \leq m$ are forbidden sets, and k is the constant. The input of instant u can be defined as follows:

$$cod(u) = \{s_{i,j} \mid s_j \in B_i\}.$$

Let us consider the polynomial time computable function between $\mathbb{N}^3$ and $\mathbb{N}$, $\langle n, m, k\rangle = \langle\langle n, m\rangle, k\rangle$, induced by the pair function $\langle n, m\rangle = ((n+m)(n+m+1)/2)+n$. We shall construct a family $\Pi = \{\Pi(i) \mid i \in \mathbb{N}\}$ such that each system $\Pi(\langle n, m, k\rangle)$ will solve all instances of CADP with given size paraments: the size n of a finite set S, the size m of the family F of forbidden sets, and target subset size k.

For each $(n, m, k) \in \mathbb{N}^3$,
$\Pi(\langle n, m, k\rangle) = (\Gamma(\langle n, m, k\rangle), \Sigma(\langle n, m, k\rangle), \Omega(\langle n, m, k\rangle), w_1,$ $w_2, \mathcal{R}(\langle n, m, k\rangle), i_{in}, i_{out})$,
with the following components:

- $\Gamma(\langle n, m, k\rangle) = \Sigma \cup \{a_j, T_j, F_j \mid 1 \leq j \leq n\}$
  $\cup\{r_i \mid 1 \leq i \leq m\} \cup \{b_i \mid 1 \leq i \leq 2n + m + 1\}$
  $\cup\{F_{i,j} \mid 1 \leq i \leq m, 1 \leq j \leq n\}$
  $\cup\{d_i \mid 1 \leq i \leq 2n + mn + 2m + k + 1\}$
  $\cup\{e_i \mid 1 \leq i \leq 2n + mn + 2m + k + 3\}$
  $\cup\{c_i \mid 1 \leq i \leq n + 1\} \cup \{f, g, \texttt{yes}, \texttt{no}\}$.
- $\Sigma(\langle n, m, k\rangle) = \{s_{i,j} \mid 1 \leq i \leq m, 1 \leq j \leq n\}$.
- $\Omega(\langle n, m, k\rangle) = \Gamma(\langle n, m, k\rangle) - \{\texttt{yes}, \texttt{no}\}$.
- $w_1 = \{\{b_1, c_1, d_1, e_1, g,\ \texttt{yes},\ \texttt{no}\}\}$.
- $w_2 = \{\{f, a_1, a_2, \cdots, a_n\}\}$.
- $\mathcal{R}(\langle n, m, k\rangle)$ is the set of rules:

1) **Division rule:**
   $r_{1,j} \equiv [a_j]_2 \rightarrow [T_j]_2[F_j]_2$, for $1 \leq j \leq n$.
2) **Communication rules:**
   $r_{2,i} \equiv (1, b_i/b_{i+1}^2, 0)$, for $1 \leq i \leq n$;
   $r_{3,i} \equiv (1, c_i/c_{i+1}^2, 0)$, for $1 \leq i \leq n$;
   $r_{4,i} \equiv (1, d_i/d_{i+1}^2, 0)$, for $1 \leq i \leq n$;
   $r_{5,i} \equiv (1, e_i/e_{i+1}, 0)$, for $1 \leq i \leq 2n + 2m + mn + 2$;
   $r_6 \equiv (1, b_{n+1}c_{n+1}d_{n+1}/f, 2)$;
   $r_{7,j} \equiv (2, c_{n+1}F_j/c_{n+1}F_{1,j}, 0)$, for $1 \leq j \leq n$;
   $r_{8,ij} \equiv (2, F_{i,j}/f_jF_{i+1,j}, 0)$, for $1 \leq i \leq m$, $1 \leq j \leq n$;
   $r_{9,i} \equiv (2, b_i/b_{i+1}, 0)$, for $n + 1 \leq i \leq 2n + m$;
   $r_{10,i} \equiv (2, d_i/d_{i+1}, 0)$, for $n + 1 \leq i \leq 2n + m + mn$;
   $r_{11,ij} \equiv (2, b_{2n+m+1}f_js_{ij}/b_{2n+m+1}r_i, 0)$, for $1 \leq i \leq m$, $1 \leq j \leq n$;
   $r_{12,i} \equiv (2, d_{2n+mn+m+i}r_i/d_{2n+mn+m+i+1}, 0)$, for $1 \leq i \leq m$;
   $r_{13,ij} \equiv (2, d_{2n+mn+2m+i}T_j/d_{2n+mn+2m+i+1},\ 0)$, for $1 \leq i \leq k$, $1 \leq j \leq n$;
   $r_{14} \equiv (2, d_{2n+mn+2m+k+1}/g\ \texttt{yes}, 1)$;
   $r_{15} \equiv (2, \texttt{yes}/\lambda, 0)$;
   $r_{16} \equiv (1, e_{2n+mn+2m+k+3}\ g\ \texttt{no}/\lambda, 2)$;
   $r_{17} \equiv (2, \texttt{no}/\lambda, 0)$.

- $i_{in} = 2$ is the input cell.
- $i_{out} = 0$ is the output region (i.e., the environment).

A. An Overview of a Computation

First of all we define a polynomial encoding for the CADP in $\mathbf{\Pi}$. Let $u = (\{s_1, \cdots, s_n\},\ (B_1, \cdots, B_m),\ k)$ be an instance of the CADP. Let the size mapping be $s(u) = \langle n, m, k\rangle$ and the encoding of instance be $cod(u) = \{s_{i,j} \mid s_j \in B_i\}$, for a given CADP-instance $u = (\{s_1, \cdots, s_n\},\ (B_1, \cdots, B_m),\ k)$. Next we informally describe how the system $\Pi(s(u))$ with input $cod(u)$ works.

Let us start with the generation stage. In cells with label 2, the division rules are applied. Cells with label 2 is repeatedly divided, each time expanding one object a_j, corresponding to s_j in the finite set S, into T_j and F_j, corresponding to the existence or absence of s_j in certain subset. In this way, after n steps we get 2^n cells with label 2, each one respects a subset of S. The object f is duplicated, hence a copy of it will appear in each cell. In parallel with the above operation of dividing cells with label 2, the counters: b_i, c_i, d_i, e_i from cell with label 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} in cell 1. Objects b_i are used to check whether a forbidden set B_i is not included in the corresponding subset A, objects c_i are used to multiply the number of copies of f_j, objects d_i are used to check whether there exists at least one subset A such that A does not include any forbidden set B_i from the family F and $|A| \geq k$. The object e_i will be used to produce the object no, if this will be the case, in the end of the computation.

The checking stage starts when the generation stage is finished after n steps. Note that cells with label 2 cannot divide any more, because the objects a_j were exhausted. At this moment, the content of the cell with label 1 is $\{\{b_{n+1}^{2^n}, c_{n+1}^{2^n}, d_{n+1}^{2^n}, e_{n+1}, g, \texttt{yes}, \texttt{no}\}\}$. In the step $n+1$, the counters b_{n+1}, c_{n+1}, d_{n+1} are brought into cells with label 2, in exchange of f by applying rule r_6. Because we have 2^n copies of each object of these types and 2^n cells with label 2, each one containing exactly one copy of f, due to the maximality of the parallelism of using the rules, each cell 2 gets precisely one copy of each of b_{n+1}, c_{n+1}, d_{n+1}.

Recall that T_j represents that s_j is in the corresponding subset, while F_j represents that s_j is not in the corresponding subset. In the presence of c_{n+1}, the object F_j introduces the object $F_{1,j}$. This phase needs at most n steps, because only one copy of c_{n+1} is available in each cell with label 2. Then further m steps are necessary for $F_{1,j}$ to grow its first subscript. In this way, m copies of f_j are introduced (Object f_j represents element s_j from S is not in the corresponding subset A. In order to check which forbidden sets are not included in A, it is possible to need one copy of f_j for each forbidden set). The counters b_i and d_i increase their subscripts, until reaching the value $2n+m+1$. In parallel, object e_i increases its subscript to $2n+m+2$ in cell with label 1.

In the presence of b_{2n+m+1}, we apply the rules $r_{11,ij}$ to check which forbidden sets are not included in the corresponding subset of S. The objects r_i represents that the forbidden set B_i is not included in the corresponding subset of S. It takes at most mn steps, because there is only one copy of b_{2n+m+1} in each cell with label 2. After step $2n+m+mn+1$, the rule $r_{12,i}$ is used to check whether there exists a subset A which does not include any forbidden set. If and only if it is positive, the subscript of d_i in the corresponding cell with label 2 grows to $2n+2m+mn+1$. After step $2n+2m+mn+1$, in the cell with label 2 whose corresponding subset of S does not include any forbidden set, the rule $r_{13,ij}$ is used to check whether the cardinality of the corresponding subset is not less than k. If and only if it is still positive, the subscript of d_i in the corresponding cell with label 2 grows to $2n+2m+mn+k+1$.

When the checking stage is done, the subscript of object e_i in cell with label 1 grows to $2n+2m+mn+k+2$. The output stage starts from step $2n+2m+mn+k+2$.

- *Affirmative answer*: If there exists at least one subset of set S which does not include any forbidden set, and its cardinality is not less than k, there is an object $d_{2n+2m+mn+k+1}$ in the corresponding cell with label 2 as described above. One of cells with label 2 containing object $d_{2n+2m+mn+k+1}$ gets the objects yes and g in exchange of $d_{2n+2m+mn+k+1}$ by the rule r_{14}. In the next step, the object yes in cell 2 leaves the system by the rule r_{15}, signaling the fact that there exists one subset A of S such that $|A| \geq k$, and A does not include any forbidden set from the family F. At that step, the cell with label 1 contains the object $e_{2n+2m+mn+k+3}$ but no the object g. The computation halts at step $2n+2m+mn+k+3$.
- *Negative answer*: In this case, the subscript of counter e_i reaches $2n+2m+mn+k+3$ and the object g is still in the cell with label 1. The object no can be moved to the environment by the rules r_{16} and r_{17}, signaling that there is no subset A of S such that $|A| \geq k$, and A does not include any forbidden set from the family F. The computation finishes at step $2n+2m+mn+k+4$.

B. Main Results

From the overview of a computation in section III-A, we can find that all computations halt in a polynomial time, and that either an object yes or an object no is sent out exactly in the last step of the computation; that is, the family Π is polynomially bounded, sound and complete. So, in order to show that the family Π built above solves QAP in a polynomial time according to Definition 2.1, we just need to show that the family Π is polynomially uniform by Turing machines.

It is easy to check that the rules of a system $\Pi(\langle n, m, k\rangle)$ of the family are defined recursively from the values n, m and k. Besides, the necessary resources to build an element of the family are of a polynomial order, as shown below:

- Size of the alphabet: $4mn+10n+6m+2k+10 \in \Theta(mn)$.
- Initial number of cells: $2 \in \Theta(1)$.
- Initial number of objects: $n+8 \in \Theta(n)$.
- Number of rules: $4mn+8n+5m+k+7 \in \Theta(mn)$.
- Maximal length of a rule: $5 \in \Theta(1)$.

Therefore, a deterministic Turing machine can build $\Pi(\langle n, m, k\rangle)$ in a polynomial time with respect to n, m and k.

From the discussion in the previous sections and according to the definition of solvability given in Section II, we have the following result:

Theorem 3.1: CADP $\in \mathbf{PMC}_{TDC}$.

Corollary 3.1: $\mathbf{NP} \cup \mathbf{co}-\mathbf{NP} \subseteq \mathbf{PMC}_{TDC}$.

Proof: It suffices to make the following observations: the CADP is NP–complete, CADP $\in \mathbf{PMC}_{TDC}$ and this complexity class is closed under polynomial-time reduction and under complement. ■

IV. Conclusions

In this work, a family of recognizer tissue P system with cell division is designed to solve the CADP. Although the algorithm proposed here is theoretically proved to be efficient to the CADP, the real implementation of such algorithms is a great challenge. In addition, it is open and of great interest how to build a biochemical computer for computing.

A solution to the CADP P systems with active membranes was proposed in [13], where four types of rules were applied in those systems: object evolution rules, communication rules, dissolving rules and division rules for elementary membranes; moreover, three charges $+, -, 0$ are used to control the application of these types of rules. The solution to the CADP given in this work is based on tissue P system with cell division, where two kinds of rules are used: communication rules and division rules. The systems constructed in this work are elegant in the sense of less kinds of rules and without charges.

ACKNOWLEDGMENT

The authors wish to acknowledge the support of the project TIN2009-13192 of the Ministerio de Ciencia e Innovación of Spain, cofinanced by FEDER funds, and the "Proyecto de Excelencia con Investigador de Reconocida Valía" of the Junta de Andalucía under grant P08-TIC04200. The first two authors are also supported by National Natural Science Foundation of China (61033003 and 30870826), Ph.D. Programs Foundation of Ministry of Education of China (20100142110072), Fundamental Research Funds for the Central Universities (2010ZD001), and Natural Science Foundation of Hubei Province (2008CDB113 and 2008CDB180).

REFERENCES

[1] Gh. Păun, "Computing with membranes," *Journal of Computer and System Sciences*, vol. 61, no. 1, pp. 108–143, 2000.

[2] Gh. Păun, *Membrane Computing. An Introduction*. Springer–Verlag, Berlin, 2002.

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

[4] P systems web page `http://ppage.psystems.eu/`

[5] A. Alhazov, C. Martín Vide, and L. Pan, "Solving a PSPACE-complete problem by P systems with restricted active membranes," *Fundamenta Informaticae*, vol. 58, no. 2, pp. 67–77, 2003.

[6] A. Alhazov, and M.J. Pérez-Jiménez, "Uniform solution of QSAT using polarizationless active membranes," *Lecture Notes in Computer Science*, vol. 4664, pp. 122–133, 2007.

[7] D. Díaz-Pernil, M. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, and A. Riscos-Núñez, "A linear-time tissue P system based solution for the 3-coloring problem," *Electronic Notes in Theoretical Computer Science*, vol. 171, pp. 81–93, 2007.

[8] L. Pan, and C. Martín Vide, "Solving multidimensional 0-1 knapsack problem by P systems with input and active membranes," *Journal of Parallel and Distributed Computing*, vol. 65, pp. 1578–1584, 2005.

[9] C. Martín Vide, J. Pazos, Gh. Păun, and A. Rodríguez Patón, "A new class of symbolic abstract neural nets: tissue P systems," *Lecture Notes in Computer Science*, vol. 2387, pp. 290–299, 2002.

[10] C. Martín Vide, J. Pazos, Gh. Păun, and A. Rodríguez Patón, "Tissue P systems," *Theoretical Computer Science*, vol. 296, pp. 295–326, 2003.

[11] Gh. Păun, M.J. Pérez-Jiménez, and A. Riscos-Núñez, "Tissue P systems with cell division," *International Journal of Computers, Communications and Control*, vol. 3, no. 3, pp. 295–303, 2008.

[12] T. Head, M. Yamamura, and S. Gal, "Aqueous computing: writing on molecules," *Proceedings of the Congress on Evolutionary Computation 1999*, IEEE Service Center, Piscataway, NJ, pp. 1006–1010, 1999.

[13] M.J. Pérez-Jiménez, and F.J.R. Campero, "Attacking the common algorithmic problem by recognizer P system," *Lecture Notes in Computer Science*, vol. 3354, pp. 304–315, 2005.

[14] D. Díaz-Pernil, M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, and A. Riscos-Núñez, "Solving subset sum in linear time by using tissue P systems with cell division," *In Proceedings of IWINAC 2007, LNCS 4527*, Springer, pp. 170–179, 2007.

[15] D. Díaz-Pernil, M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, and A. Riscos-Núñez, "Solving the partition problem by using tissue-like P systems with cell division," *In Proceedings of 6th Brainstorming Week on Membrane Computing, Report RGNC 01/08*, Fenix Editoria, pp. 124–134, 2008.

[16] M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez, and F.J. Romero-Campero, "On the power of dissolution in P systems with active membranes," *Lecture Notes in Computer Science*, vol. 3580, pp. 224–240, 2006.