ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **11**, Number 3, 2008, 229–241

Computational Efficiency of Cellular Division in Tissue-like Membrane Systems

D. DÍAZ–PERNIL, M. J. PÉREZ–JIMÉNEZ,
A. RISCOS–NÚÑEZ, Á. ROMERO–JIMÉNEZ

Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Seville

E-mail: sbdani@us.es, marper@us.es,
ariscosn@us.es, Alvaro.Romero@cs.us.es

Abstract. Tissue-like P systems with cell division are computing models in the framework of membrane computing. They are inspired by the intercellular communication and neuronal synaptics, their structures being formalized by underlying graphs. As usual in membrane computing, division rules allow the construction of an exponential workspace (described by the number of cells) in a linear time. In this paper this ability is used for presenting a uniform linear–time solution for the (**NP**–complete) `Vertex Cover` problem via a uniform family of such systems. This solution is compared to other ones obtained in the framework of cell-like membrane systems.

1. Introduction

This paper lies in the framework of membrane computing, a theoretical model of computation inspired by the structure and functioning of cells as living organisms able to process information. The key elements of this model are the membranes, which in the living cells delimit the regions where chemical reactions take place, and also act as selective channels of communication between different compartments, as well as between the cell and its environment [1].

Membrane computing is an emergent cross-disciplinary branch of natural computing introduced by Gh. Păun in [13]. It has received important attention from

the scientific community since then, with contributions by computer scientists, biologists, formal linguists and complexity theoreticians, enriching each other with results, open problems and promising new research lines. In fact, already in 2003 membrane computing was selected by the Institute for Scientific Information, USA, as a *Fast Emerging Research Front* in Computer Science.

The computational devices in membrane computing are called *P systems*. Roughly speaking, a P system consists of a membrane structure, in the compartments of which one places multisets of objects which evolve according to given rules in a synchronous non-deterministic maximally parallel manner.

In the last years, many different variants of this model have appeared. One of the most important is known as tissue P systems, first considered in [16] and then in [11]. In these systems the structure is defined by a general graph, while in the original model of P systems membranes are arranged in a tree-like structure.

There exist several tissue-like models of P systems (see, for example, [2, 4, 8, 9, 10, 19]). One of the most interesting variants of tissue P systems was presented in [15] (and was studied in depth in [5]). In that paper, tissue P systems are endowed with the ability of getting new cells based on the *mitosis* or cellular division, yielding *tissue-like P systems with cell division*.

Some **NP**–complete problems have been efficiently solved with tissue-like P systems: `SAT` [15], `3-coloring` [6], `Subset Sum` [7]. In this paper, a uniform linear–time solution to the `Vertex Cover` problem is presented.

In the literature we can find uniform solutions to this problem in the cell-like model of P systems with active membranes, but this is the first solution to `Vertex Cover` in the framework of tissue-like P systems.

The paper is organized as follows: in Sections 2 and 3 we recall some preliminaries and the definition of tissue-like P systems with cell division, respectively. Next, recognizer tissue P systems are briefly described. A linear–time solution to the `Vertex Cover` problem with the necessary resources and the main results are presented in the following section. In the next section is included a short overview of the computations. In Section 7 we show a comparative study between the solution to the `Vertex Cover` problem presented in this paper and a solution using P systems with active membranes [3]. Finally, some conclusions and new open research lines are presented.

2. Preliminaries

In this section we briefly recall some of the concepts used later on in the paper.

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 $\Sigma^* = \bigcup_{n\geq 0} \Sigma^n$. A *language* over Σ is a subset from Σ^*.

A *multiset* over a set A is a pair (A, f) where $f : A \to \mathbb{N}$ is a mapping. If $m = (A, f)$ is a multiset then its *support* is defined as $supp(m) = \{x \in A : f(x) > 0\}$

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, then it will be denoted as $m = a_1^{f(a_1)} a_2^{f(a_2)} \cdots a_k^{f(a_k)}$, where $supp(m) = \{a_1, \ldots, a_k\}$, and for each element a_i, $f(a_i)$ is called the multiplicity of a_i.

Given $m_1 = (A, f_1)$ and $m_2 = (A, f_2)$ two multisets over A, the union of m_1 and m_2 is defined as $m_1 m_2 = (A, f_1 + f_2)$.

An *undirected graph* G is a pair $G = (V, E)$ where V is the set of vertices and E is the set of edges, each one of which is a (unordered) pair of (different) vertices. If $\{u, v\} \in E$, we say that u is *adjacent* to v (and also v is *adjacent* to u).

In what follows we assume the reader is already familiar with the basic notions and the terminology underlying P systems (see [14] for details).

3. Tissue-like P Systems with Cell Division

In the first definition of the model of tissue P systems [11] the membrane structure did not change along the computation. Gh. Păun et al. presented in [15] a new model of tissue P systems *with cell division* inspired by the cell-like model of P systems with active membranes. The biological inspiration is clear: live tissues are not *static* network of cells, since cells are duplicated via mitosis in a natural way.

The main features of this model, from the computational point of view, are: (a) the initial objects in the environment are present in an arbitrary number of copies; (b) cells have no polarizations; (c) the division rules do not change the labels and they inhibit the application of other rules to a dividing cell. In some sense, this means that while a cell is dividing it closes the communication channels with other cells and with the environment.

Formally, a *tissue-like P system with cell division* of degree $q \geq 1$ is a tuple of the form $\Pi = (\Gamma, \mathcal{E}, w_1, \ldots, w_q, \mathcal{R}, i_0)$, where:

1. Γ is a finite *alphabet*, whose symbols will be called *objects*.
2. $w_1, \ldots, w_q$ are strings over Γ representing the multisets of objects associated with the cells in the initial configuration.
3. $\mathcal{E} \subseteq \Gamma$ are the initial objects of the environment.
4. $\mathcal{R}$ is a finite set of rules of the following form:
   - **(a)** *Communication rules*: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \ldots, q\}, i \neq j$ and $u, v \in \Gamma^*$.
   - **(b)** *Division rules*: $[a]_i \rightarrow [b]_i [c]_i$, for $i \in \{1, 2, \ldots, q\}$ and $a \in \Gamma, b, c \in \Gamma \cup \{\lambda\}$.
5. $i_0 \in \{0, 1, 2, \ldots, q\}$.

A tissue-like P system with cell division of degree $q \geq 1$ can be seen as a set of q cells (each one consisting of an elementary membrane) labeled by $1, 2, \ldots, q$. We

shall use 0 to refer to the label of the environment, and i_0 denotes the output region (which can be the region inside a cell or the environment).

The communication rules determine a virtual graph, where the nodes are the cells and the edges denote the pairs of cells able to communicate directly. This is a dynamical graph, because new nodes can appear by the application of division rules.

The strings $w_1, \ldots, w_q$ describe the multisets of objects placed in the q cells of the system. We interpret that $\mathcal{E} \subseteq \Gamma$ is the set of objects placed in the environment, each one of them in an arbitrary large amount of copies.

The communication rule $(i, u/v, j)$ can be applied to two cells i and j such that u is contained in cell i and v is contained in cell j. The application of this rule means that the objects of the multisets represented by u and v are interchanged between the two cells.

The division rule $[a]_i \rightarrow [b]_i[c]_i$ is applied to a cell i containing object a. The application of this rule divides this cell into two new cells with the same label. All the objects in the original cell are replicated and copied in each of the new cells, with the exception of the object a, which is replaced by the object b in the first one and by the object c in the other one.

Rules are used as usual in the framework of membrane computing, that is, in a maximally parallel way (a universal clock is considered). In one step, each object in a membrane can only be used for one rule (non-deterministically chosen when there are several possibilities), and in each step we apply a maximal set of rules. 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 move in that step.

4. Recognizer Tissue-like P Systems with Cell Division

NP–completeness has been usually studied in the framework of *decision problems*. Let us recall that a decision problem is a pair (I_X, θ_X) 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.

In order to study the computational efficiency of solving computationally hard decision problems, a special class of tissue P systems with cell division is introduced in [15]: *recognizer tissue P systems*. The key idea of such recognizer systems is the same one as from recognizer P systems with cell-like structure.

Recognizer cell-like P systems were introduced in [17] and they are the natural framework to study and solve decision problems within membrane computing, since deciding whether an instance of a given problem has an affirmative or negative answer is equivalent to deciding if a string belongs or not to the language associated with the problem.

In the literature, recognizer cell-like P systems are associated with P systems with *input* in a natural way. The data encoding an instance of the decision problem has to be provided to the P system in order to compute the appropriate answer. This is done by codifying each instance as a multiset placed in an *input membrane*. The

output of the computation (`yes` or `no`) is sent to the environment in the last step of the computation. In this way, cell-like P systems with input and external output are devices which can be seen as black boxes, in the sense that the user provides the data before the computation starts, and then waits *outside* the P system until it sends to the environment the output in the last step of the computation.

A recognizer tissue-like P system with cell division of degree $q \geq 1$ is a tuple $\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, \ldots, w_q, \mathcal{R}, i_{in})$ where

- $(\Gamma, \mathcal{E}, w_1, \ldots, w_q, \mathcal{R}, 0)$ is a tissue-like P system with cell division of degree $q \geq 1$ (as defined in the previous section), and $w_1, \ldots, w_q$ are strings over $\Gamma \setminus \Sigma$.
- The working alphabet Γ has two distinguished objects `yes` and `no`, present in at least one copy in some initial multisets $w_1, \ldots, w_q$, but not present in $\mathcal{E}$.
- Σ is an (input) alphabet strictly contained in Γ.
- $i_{in} \in \{1, \ldots, q\}$ is the input cell.
- All computations halt.
- If $\mathcal{C}$ is a computation of Π, then either the object `yes` or the object `no` (but not both) must have been released into the environment, and only in the last step of the computation.

The computations of the system Π with input $w \in \Sigma^*$ start from a configuration of the form $(w_1, w_2, \ldots, w_{i_{in}} w, \ldots, w_q; \mathcal{E})$, that is, after adding the multiset w to the contents of the input cell i_{in}. We say that the multiset w is *recognized* by Π if and only if the object `yes` is sent to the environment, in the last step of the corresponding computation. We say that $\mathcal{C}$ is an accepting computation (respectively, rejecting computation) if the object `yes` (respectively, `no`) appears in the environment associated with the corresponding halting configuration of $\mathcal{C}$.

Definition 1. We say that a decision problem $X = (I_X, \theta_X)$ is *solvable in polynomial time* by a family $\mathbf{\Pi} = \{\Pi(n) : n \in \mathbb{N}\}$ of recognizer tissue-like P systems with cell division if the following hold:

- 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 (called a polynomial encoding of I_X in $\mathbf{\Pi}$) 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.

From the soundness and completeness conditions above we deduce that every P system $\Pi(n)$ is *confluent*, in the following sense: every computation of a system with the *same* input multiset must always give the *same* answer.

We denote by $\mathbf{PMC}_{TD}$ the set of all decision problems which can be solved by means of recognizer tissue-like P systems with cell division in polynomial time. This class is closed under polynomial–time reduction and under complement (see [18] for a similar result for cell-like P systems).

5. A Solution for the `Vertex Cover` Problem

Let us recall that a *vertex cover* of a non-directed graph is a subset of its vertices such that for each edge of the graph at least one of its endpoints belongs to that subset. The number of vertices in the subset is called the size of the vertex cover. The `Vertex Cover (VC)` problem can be formulated as follows: given a non-directed graph, $G = (V, E)$, and a natural number $k \leq |V|$, decide whether or not G has a vertex cover of size at most k.

Next, we shall prove that `VC` can be solved in linear time (in the number of nodes and edges of the graph) by a family of recognizer tissue-like P systems with cell division. To this aim, we are going to construct a family $\mathbf{\Pi} = \{\Pi(\langle n, m, k\rangle) : n, m, k \in \mathbb{N}\}$ where each system of the family will process every instance u of the problem given by a graph with n vertices and m edges, and by a size k of the vertex cover (that is, $s(u) = \langle n, m, k\rangle$, where $\langle a, b\rangle = \frac{(a+b)(a+b+1)}{2} + a$ and $\langle a, b, c\rangle = \langle\langle a, b\rangle, c\rangle$). To provide a suitable encoding of these instances into the systems, we will use the objects A_{ij}, with $1 \leq i < j \leq n$, to represent the edges of the graph, and we will provide $cod(u) = \{A_{ij} : 1 \leq i < j \leq n \wedge (v_i, v_j) \in E\}$ as the initial multiset for the system. From the definitions of the functions and the definition of the systems of the family that will be given next, it can be easily seen that (cod, s) is a polynomial encoding of I_{VC} in $\mathbf{\Pi}$.

Then, given an instance u of the `VC` problem, the system $\Pi(s(u))$ with input $cod(u)$ decides that instance by a brute force algorithm, implemented in the following four stages: *generation stage*: all the possible subsets of vertices are generated by the application of division rules; *pre–checking stage*: only the subsets of size k are selected; *checking stage*: for each of these subsets it is checked if there exists an edge of the graph for which none of its endpoints is in the subset; *answer stage*: an affirmative or negative answer to the problem is given, according to the results of the previous stage.

The family $\mathbf{\Pi} = \{\Pi(\langle n, m, k\rangle) : n, m, k \in \mathbb{N}\}$ of recognizer tissue-like P sys-

tems with cell division is defined as follows: for each $n, m, k \in \mathbb{N}, \Pi(\langle n, m, k\rangle) = (\Gamma, \Sigma, \mathcal{E}, w_1, w_2, \mathcal{R}, i_{in}, i_0)$, where

- $\Gamma = \{A_i, B_i, \overline{B}_i, B'_i, C_i, C'_i : 1 \leq i \leq n\} \cup \{e_i : 1 \leq i \leq 2n+1\} \cup$
  $\{a_i : 1 \leq i \leq 3n + m + \lceil \lg n \rceil + 14\} \cup \{c_i, d_i : 1 \leq i \leq n+1\} \cup$
  $\{g_i : 1 \leq i \leq \lceil \lg n \rceil + 1\} \cup \{h_i : 1 \leq i \leq \lceil \lg m \rceil + 1\} \cup$
  $\{l_i : 1 \leq i \leq m + \lceil \lg n \rceil + 7\} \cup \{p_i : 1 \leq i \leq m + \lceil \lg n \rceil + 6\} \cup$
  $\{B_{ij}, C_{ij} : 1 \leq i \leq n, 1 \leq j \leq m+1\} \cup \{D_{ij} : 1 \leq i, j \leq n\} \cup$
  $\{A_{ij}, P_{ij} : 1 \leq i < j \leq n\} \cup \{b, D, F_0, F_1, F_2, T, S, N, \texttt{yes}, \texttt{no}\}$
- $\Sigma = \{A_{ij} : 1 \leq i < j \leq n\}$
- $\mathcal{E} = \Gamma \setminus \{a_1, b, c_1, \texttt{yes}, \texttt{no}, D, A_1, \ldots, A_n\}$
- $w_1 = a_1\, b\, c_1\, \texttt{yes}\, \texttt{no}$ and $w_2 = D\, A_1 \cdots A_n$. Also, we consider that in the environment there are always infinitely many copies of each object from $\mathcal{E}$, and, initially, no copies of any element in $\Gamma \setminus \mathcal{E}$
- $\mathcal{R}$ is the following set of rules:

  Division rules:

$$r_{1,i} \equiv [A_i]_2 \rightarrow [B_i]_2[\lambda]_2, \text{ for } i = 1, \ldots, n$$

  Communication rules:

$$\begin{aligned}
r_{2,i} &\equiv (1, a_i/a_{i+1}, 0), \text{ for } i = 1, \ldots, 3n + m + \lceil \lg n \rceil + 13\\
r_{3,i} &\equiv (1, c_i/c_{i+1}^2, 0), \text{ for } i = 1, \ldots, n\\
r_4 &\equiv (1, c_{n+1}/D, 2)\\
r_5 &\equiv (2, c_{n+1}/d_1 e_1, 0)\\
r_{6,i} &\equiv (2, e_i/e_{i+1}, 0), \text{ for } i = 1, \ldots, 2n\\
r_{7,i,j} &\equiv (2, d_j B_i/D_{ij}, 0), \text{ for } 1 \leq i, j \leq n\\
r_{8,i,j} &\equiv (2, D_{ij}/\overline{B}_i d_{j+1}, 0), \text{ for } 1 \leq i, j \leq n\\
r_9 &\equiv (2, e_{2n+1} d_{k+1}/F_0, 0)\\
r_{10} &\equiv (2, F_0/l_1 F_1, 0)\\
r_{11,i} &\equiv (2, l_i/l_{i+1}, 0), \text{ for } i = 1, \ldots, m + \lceil \lg n \rceil + 6\\
r_{12} &\equiv (2, F_1/p_1 F_2, 0)\\
r_{13} &\equiv (2, F_2/g_1 h_1, 0)\\
r_{14,i} &\equiv (2, p_i/p_{i+1}, 0), \text{ for } i = 1, \ldots, m + \lceil \lg n \rceil + 5\\
r_{15,i} &\equiv (2, g_i/g_{i+1}^2, 0), \text{ for } i = 1, \ldots, \lceil \lg n \rceil\\
r_{16,i} &\equiv (2, h_i/h_{i+1}^2, 0), \text{ for } i = 1, \ldots, \lceil \lg m \rceil\\
r_{17,i,j} &\equiv (2, A_{ij} h_{\lceil \lg m \rceil + 1}/P_{ij}, 0), \text{ for } 1 \leq i < j \leq n
\end{aligned}$$

$$
\begin{aligned}
r_{18,i} &\equiv (2, g_{\lceil \lg n \rceil + 1}\overline{B}_i/C_i, 0), \text{ for } i = 1, \ldots, n \\
r_{19,i} &\equiv (2, C_i/C_{i1}B_{i1}, 0), \text{ for } i = 1, \ldots, n \\
r_{20,i,j} &\equiv (2, B_{ij}/B_{i,j+1}B'_i, 0), \text{ for } i = 1, \ldots, n \text{ and } j = 1, \ldots, m \\
r_{21,i,j} &\equiv (2, C_{ij}/C_{i,j+1}C'_i, 0), \text{ for } i = 1, \ldots, n \text{ and } j = 1, \ldots, m \\
r_{22,i,j} &\equiv (2, B'_i P_{ij}/\lambda, 0), \text{ for } 1 \leq i < j \leq n \\
r_{23,i,j} &\equiv (2, C'_j P_{ij}/\lambda, 0), \text{ for } 1 \leq i < j \leq n \\
r_{24,i,j} &\equiv (2, p_{m+\lceil \lg n \rceil + 6} P_{ij}/\lambda, 0), \text{ for } 1 \leq i < j \leq n \\
r_{25} &\equiv (2, l_{m+\lceil \lg n \rceil + 7} p_{m+\lceil \lg n \rceil + 6}/T, 0) \\
r_{26} &\equiv (2, T/\lambda, 1) \\
r_{27} &\equiv (1, bT/S, 0) \\
r_{28} &\equiv (1, S\texttt{yes}/\lambda, 0) \\
r_{29} &\equiv (1, a_{3n+m+\lceil \lg n \rceil + 14} b/N, 0) \\
r_{30} &\equiv (1, \texttt{no}N/\lambda, 0)
\end{aligned}
$$

- $i_{in} = 2$ is the input cell

In order to show that the family $\mathbf{\Pi}$ is polynomially uniform by deterministic Turing machines we first note that the sets of rules associated with the system $\Pi(\langle n, m, k \rangle)$ are recursive. Hence, it is enough to note that the amount of necessary resources for defining each system is quadratic in $\max\{n, m\}$, since those resources are the following:

1. Size of the alphabet: $nm+2n^2+13n+3m+4\lceil \lg n \rceil + \lceil \lg m \rceil + 42 \in \Theta(nm+n^2) \subseteq O(n^3)$.

2. Initial number of cells: $2 \in \Theta(1)$.

3. Initial number of objects: $n + 6 \in \Theta(n)$.

4. Number of rules: $2nm + 4n^2 + 7n + 3m + 4\lceil \lg n \rceil + \lceil \lg m \rceil + 36 \in \Theta(nm + n^2) \subseteq O(n^3)$.

5. Upper bound for the length of the rules: $3 \in \Theta(1)$.

As we will see in the following section, the family $\mathbf{\Pi}$ is also polynomially (linearly) bounded, sound and complete with regard to $(\texttt{VC}, cod, s)$. So, we get the main result of the paper.

Theorem 1. $\texttt{VC} \in \mathbf{PMC}_{TD}$

Taking into account that VC is an **NP**–complete problem, the following is deduced.

Corollary 1. $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC}_{TD}$

6. An Overview of the Computations

To support our claim that the family **Π** is also linearly bounded, sound and complete with regard to $(\texttt{VC}, cod, s)$, we next describe with detail the steps followed by the system $\Pi(s(u))$ when it is given the input multiset $cod(u)$, for an arbitrary instance u of VC. Observe that the system starts with only two cells, one labeled by 1 and the other labeled by 2, and that the division rules associated with the system are applied only to cells with this latter label. This means that along the computations there will always be a unique cell labeled 1 (which we will call the 1-cell), but that new cells labeled by 2 (which we will call the 2-cells) will appear.

In order for the system $\Pi(s(u))$ to decide the instance of the VC problem encoded by $cod(u)$, it starts with the *generation stage*, where all the possible subsets of the vertices of the graph are generated. This is performed by the successive application of the division rules. These rules take the objects A_i in the 2-cells, which encode the vertices of the graph, and produce two new 2-cells, one of them substituting that object by the object B_i, meaning that we include the vertex in the subset, and the other eliminating it, meaning that we do not include the vertex in the subset. Of course, all the remaining objects contained in the original 2-cells are replicated into the new ones. This way, at the end of this stage, which lasts n steps, the system will have 2^n 2-cells, each of them encoding a different subset of vertices of the graph, by means of the objects B_i.

To control when this stage ends, objects c_i in the 1-cell of the system are used as counters. This 1-cell starts containing an object c_1, that is interchanged, using rule $r_{3,1}$ by two objects c_2 from the environment; each of these objects are again interchanged, using rule $r_{3,2}$, by two objects c_3; and so on. Thus, at the end of the generation stage the 1-cell will contain 2^n objects c_{n+1}.

On the other hand, objects a_i in the 1-cell of the system are used, by means of rules $r_{2,i}$, as global counters of the computation, and, unlike the objects c_i, they are not duplicated.

Note that this generation stage is non–deterministic, but it is easy to check that, independently of the way the division rules are applied, at the end of the stage the same configuration is always reached. Thus, the system is confluent in this stage and performs n steps.

Now that all the subsets of vertices of the graph are generated, the *pre–checking stage* selects only those of size k. This stage is activated by rules r_4 and r_5, which interchange the object D of each 2-cell (recall that there are 2^n of them) by an object c_{n+1} of the 1-cell (recall that there are 2^n of them), and then each of the latter in each 2-cell with an object d_1 and an object e_1 from the environment (recall that there are infinitely many of them). After the $(n+2)$-th step, the 1-cell of the system contains $a_{n+3}D^{2^n}b\,\texttt{yes}\,\texttt{no}$ and each 2-cell encodes a subset of vertices of the graph by means of the objects B_i within it, and it also contains $d_1e_1cod(u)$.

From now on, the only role played by the 1-cell is waiting for the last stage, using the objects a_i, by means of rules $r_{2,i}$, to count the number of steps performed by the computation.

The objects d_1 and e_1 start two processes of counting in each 2-cell. The first

one counts the number of steps of the stage that have been performed, and is controlled by the objects e_i, which are repeatedly interchanged by objects e_{i+1} from the environment using rules $r_{6,i}$.

The second process counts the number of vertices in the subset. It is performed using rules $r_{7,i,j}$ and $r_{8,i,j}$, which interchange the objects B_i in the 2-cells by objects $\overline{B}_i$ (indicating this way that the corresponding vertex has been counted) and increase the counter d_j (the only purpose of the objects D_{ij} is to reduce the length of the rules). Note that this is a non-deterministic process, since the vertex "counted" in each step is chosen in a non-deterministic way. However, as the size of the subsets of vertices is upper bounded by n, after $2n$ steps of this process, the same configuration is always reached, so the system is also confluent in this stage.

Note that for the counter d_j of a 2-cell to increase, it is necessary and sufficient that in that cell there exist objects B_i left. This means that at the end of the process explained in the previous paragraph, the only 2-cells that contain objects encoding subsets of vertices of size k are those containing the object d_{k+1}. At this moment, those cells also contain the counter e_{2n+1}, which then in two steps cause (using rules r_9 and r_{10}, and the intermediate object F_0 for rules size reduction) the object d_{k+1} to be interchanged by objects l_1 and F_1 from the environment.

The total number of steps of the pre-checking stage is $2 + 2n$.

The *checking stage* starts now, but before we can check if any of the subsets of vertices of size k selected in the previous stage is a vertex cover of the graph, we need some preparation steps. First of all, the objects l_i will be used as a counter, controlled by rules $r_{11,i}$, of the number of steps performed. On the other hand, rule r_{12} introduces another counter p_i, controlled by rules $r_{14,i}$, which runs in parallel, but with a delay of one step. Also, in each 2-cell encoding a subset of vertices of size k objects g_1 and h_1 are introduced by rules r_{12} and r_{13}, and are then multiplied by rules $r_{15,i}$ and $r_{16,i}$ until obtaining n copies of the former and m copies of the latter.

The objects $h_{\lceil \lg m \rceil + 1}$ are used by rule $r_{17,i,j}$ to change into objects P_{ij} the objects A_{ij} encoding the edges of the graph, which have been replicated to every 2-cell from the input cell of the system. On the other hand, rules $r_{18,i}, r_{19,i}, r_{20,i,j}$ and $r_{21,i,j}$ produce, from objects $g_{\lceil \lg n \rceil + 1}$ and $\overline{B}_i$ and by successive interchanges of objects between the 2-cells and the environment, m copies of objects B'_i and C'_i for each and all of the vertices in the subset encoded into the 2-cell.

As the copies of objects B'_i and C'_i are being produced, rules $r_{22,i,j}$ and $r_{23,i,j}$ eliminate from the 2-cell, in a non-deterministic way, edges of the graph (encoded by objects P_{ij}) such that at least one of its endpoints is contained in the subset encoded in the corresponding 2-cell. Once this stage has performed $m + \lceil \lg n \rceil + 6$ steps, we are sure that if there is any object P_{ij} left in the 2-cell, then the subset of vertices encoded in that cell is not a vertex cover of the graph, and rule $r_{24,i,j}$ eliminates the counter p in an additional step.

The *answer stage* starts at step $3n + m + \lceil \lg n \rceil + 9$, when the object $l_{m+\lceil \lg n \rceil + 7}$ appears in every 2-cell encoding a subset of vertices of size k. If the counter p has survived in any of these 2-cells, it means that it encoded a vertex cover of the graph, and rule r_{25} interchanges the two counters with an object T from the environment, which is then sent to the 1-cell of the system by rule r_{26}. Then, rules r_{27}, r_{28}, r_{29} and

r_{30} control if this cell has received at least one object T from any of the 2-cells of the system. If this is the case, it is detected at step $3n + m + \lceil \lg n \rceil + 14$, when an object `yes` is sent to the environment and the system halts. Otherwise, it is detected at step $3n + m + \lceil \lg n \rceil + 15$, when an object `no` is sent to the environment and the system halts.

7. Comparing Solutions in Tissue-like Model and Active Membranes Model

The first solutions to **NP**–complete problems in membrane computing were designed in the cell-like model called *P systems with active membranes*. Many kinds of problems have been addressed, as for instance the `Satisfiability` problem, several numerical problems (`Subset Sum`, `Knapsack`, `Partition`, etc.), and also graph problems (`3-Coloring`, `Clique`, `Vertex Cover`, etc.).

We shall focus here on the uniform solution for the `Vertex Cover` problem presented by A. Alhazov et al. in [3], designed within the (cell-like) P systems with active membranes model allowing three electrical charges for the membranes, and using evolution rules, communication rules (send-in and send-out), and 2-division of elementary membranes.

Before going on, it is important to note that we do not intend to compare the computational power of the two models by studying two particular solutions for the same problem. Nevertheless, we believe that this comparison can bring some light on the different strategies used for solving problems in each model.

In both cases a brute force approach has been considered. That is, all possible subsets of the set of vertices are constructed, then subsets of size k are filtered, and after that a checking stage is carried out to test if any of these subsets is actually a vertex cover of the graph. The appropriate answer (`yes` or `no`) is sent to the environment at the end of the computations.

Concerning the number of steps, in both cases there is a linear bound with respect to n (the number of nodes) and m (the number of edges). More precisely, $5n+2m+3$ steps in the active membranes case and $3n + m + \lceil \lg n \rceil + 15$ in the tissue-like case.

Let us summarize next the necessary resources required to build the families of P systems solving the `Vertex Cover` problem in both models.

	Tissue-like	Cell-like
Size of the alphabet	$O(n^3)$	$O(n^4)$
Initial number of cells	$\Theta(1)$	$\Theta(1)$
Initial number of objects	$\Theta(n)$	$\Theta(1)$
Number of rules	$O(n^3)$	$O(n^4)$
Maximum length of the rules	$\Theta(1)$	$\Theta(1)$
Number of steps	$O(n^2)$	$O(n^2)$

8. Conclusions and Future Work

An efficient solution to the 3–coloring problem is presented in [6] by D. Díaz-Pernil et al. Following the ideas used there, a schema for solving **NP**–complete problems of graph theory can be inferred. This schema is used in this paper for presenting a uniform linear–time solution to the `Vertex Cover` problem and could be used for solving other **NP**–complete problems: `Clique`, `Independent Set`, etc.

The efficiency of cell-like P systems has been widely studied in the last years. On the other hand, there are very few works studying the case of tissue-like P systems.

An interesting question is the search for frontiers between tractability and intractability with tissue-like P systems. Is it absolutely necessary to use division rules for efficiently solving **NP**–complete problems? Is it possible to reduce the size of the rules to 2?

One could also consider a tissue P system where the environment always has a finite amount of objects. An interesting question is to know if we could obtain efficient solutions of **NP**–complete problems with this new variant.

Finally, we would like to mention the possibility to consider cell-creation rules in the tissue-like model, which makes a significant difference, because this kind of rules does not perform replication of objects, as is the case with cell-division rules.

Acknowledgement. The authors wish to acknowledge the support of the project TIN2006-13425 of the Ministerio de Educación y Ciencia of Spain, cofinanced by FEDER funds, and the support of the project of excellence TIC-581 of the Junta de Andalucía.

References

[1] ALBERTS B., JOHNSON A., LEWIS J., RAFF M., ROBERTS K., WALTER P., *The molecular biology of the cell*, Garland Publishing, Inc., London, 2002.

[2] ALHAZOV A., FREUND R., OSWALD M., *Tissue P systems with antiport rules and small numbers of symbols and cells*. In: C. de Felice, A. Restivo (Eds.) 9th International Conference, DLT 2005, Developments in Language Theory, Palermo, Lecture Notes in Computer Science, **3572**, 2005, pp. 100–111.

[3] ALHAZOV A., MARTÍN–VIDE C., PAN L., *Solving graph problems by P systems with restricted elementary active membranes*. In: N. Jonoska, Gh. Păun, G. Rozenberg (Eds.) Aspects of molecular computing, essays dedicated to Tom Head on the occasion of his 70th birthday, Lecture Notes in Computer Science, **2950**, 2004, pp. 1–22.

[4] BERNARDINI F., GHEORGHE M., *Cell communication in tissue P systems and cell division in population P systems*, Soft Computing, **9**, 9, 2005, pp. 640–649.

[5] DÍAZ–PERNIL D., *Sistemas P de tejido: formalización y eficiencia computacional*, PhD Thesis, University of Seville, 2008.

[6] DÍAZ–PERNIL D., GUTIÉRREZ–NARANJO M.A., PÉREZ–JIMÉNEZ M.J., RISCOS–NÚÑEZ A., *A linear-time tissue P system based solution for the 3-coloring problem*, Electronic Notes in Theoretical Computer Science, **171**, 2007, pp. 81–93.

[7] DÍAZ–PERNIL D., GUTIÉRREZ–NARANJO M.A., PÉREZ–JIMÉNEZ M.J., RISCOS–NÚÑEZ A., *Solving Subset Sum in linear time by using tissue P systems with cell division*. In: J. Mira, J. R. Alvarez, J. R. Ivarez (Eds.) 2nd International Work-Conference, IWINAC 2007, Interplay between natural and artificial computation, Lecture Notes in Computer Science, **4527**, 2007, pp. 170–179.

[8] FREUND R., PĂUN Gh., PÉREZ-JIMÉNEZ M.J., *Tissue P systems with channel states*, Theoretical Computer Science, **330**, 2005, pp. 101–116.

[9] KRISHNA S.N., LAKSHMANAN K., RAMA R., *Tissue P systems with contextual and rewriting rules*. In: Gh. Păun, G. Rozenberg, A. Salomaa, C. Zandron (Eds.) Workshop on Membrane Computing, WMC3, Curtea de Arges, 2002, Lecture Notes in Computer Science, **2597**, 2003, pp. 339–351.

[10] LAKSHMANAN K., RAMA R., *On the power of tissue P systems with insertion and deletion rules*. In: A. Alhazov, C. Martín–Vide, Gh. Păun (Eds.) Preproceedings of the Workshop on Membrane Computing, RGML Report 28/03, Rovira i Virgili University, Tarragona, 2003, pp. 304–318.

[11] MARTÍN–VIDE C., PAZOS J., PĂUN Gh., RODRÍGUEZ–PATÓN A., *A new class of symbolic abstract neural nets: tissue P systems*. In: H. H. Yin, L. Zhang, O. H. Ibarra (Eds.) 8th Annual International Conference, Cocoon 2002, Lecture Notes in Computer Science, **2387**, 2002, pp. 290–299.

[12] PĂUN A., PĂUN Gh., *The power of communication: P systems with symport/antiport*, New Generation Computing, **20**, 3, 2002, pp. 295–305.

[13] PĂUN Gh., *Computing with membranes*, Journal of Computer and System Sciences, **61**, 1, 2000, pp. 108–143.

[14] PĂUN Gh., *Membrane computing. An introduction*, Springer–Verlag, Berlin, 2002.

[15] PĂUN Gh., PÉREZ–JIMÉNEZ M.J., RISCOS–NÚÑEZ A., *Tissue P system with cell division*, International Journal of Computers, Communications & Control, **Vol. III**, 3, 2008, pp. 295–302.

[16] PĂUN Gh., SAKAKIBARA Y., YOKOMORI T., *P Systems on graphs of restricted forms*, Publicationes Mathematicae Debrecen, **60**, 2002, pp. 635–660.

[17] PÉREZ–JIMÉNEZ M.J., ROMERO–JIMÉNEZ A., SANCHO–CAPARRINI F., *A polynomial complexity class in P systems using membrane division*, Journal of Automata, Languages and Combinatorics, **11**, 4, 2006, pp. 423–434.

[18] PÉREZ–JIMÉNEZ M.J., ROMERO–JIMÉNEZ A., SANCHO–CAPARRINI F., *Complexity classes in cellular computing with membranes*, Natural Computing, **2**, 3, 2003, pp. 265–285.

[19] PRAKASH V.J., *On the power of tissue P systems working in the maximal–one mode*. In: A. Alhazov, C. Martín–Vide, Gh. Păun (Eds.) Preproceedings of the Workshop on Membrane Computing, RGML Report 28/03, Rovira i Virgili University, Tarragona, 2003, pp. 356–364.