A Linear Time Solution to the Partition Problem in a Cellular Tissue-Like Model

Daniel Díaz-Pernil[1], Miguel A. Gutiérrez-Naranjo[2], Mario J. Pérez-Jiménez[2], and Agustín Riscos-Núñez[2,*]

[1] *Research Group on Computational Topology and Applied Mathematics,* [2] *Research Group on Natural Computing, University of Sevilla, Avda. Reina Mercedes s/n, 41012, Sevilla, Spain*

Tissue-like P systems with cell division is a computing model in the framework of membrane computing that is based on the intercellular communication and cooperation between neurons. In such a model, the structure of the devices is a network of elementary cells. Tissue-like P systems with cell division have the ability of increasing the number of cells during the computation. In this paper we exploit this ability and present a polynomial-time (actually, linear-time) solution to the **NP**-complete Partition problem via a uniform family of such P systems.

Keywords: Membrane Computing, Partition Problem, Tissue-Like P Systems.

1. INTRODUCTION

In this paper, *Tissue-like P systems with cell division*[1] are investigated. This is a model of computation in the framework of membrane computing that is inspired by the way living cells communicate and cooperate in tissues. In the computational devices of this type we have processor units (called *cells*) that process some pieces of information in parallel and send signals to other processor units along connecting links. Such links can follow any scheme, and this is one of the features which distinguishes these models from the original model in membrane computing, the cell-like model, where membranes are hierarchically arranged in a tree-like structure (see Ref. [2]).

In tissue-like P systems with cell division the membrane structure is tissue-like and the links between cells form a general undirected graph. The edges of this graph are not given explicitly, but they are deduced from the set of rules, as will be explained later. The communication among cells is based on symport/antiport rules, well-known in the P systems field. Symport rules move several objects across a membrane in one direction, whereas antiport rules move objects across a membrane in opposite directions. Tissue P systems were first considered in Ref. [3], and then in Ref. [4]; after that, several research directions have been developed and other variants have arisen (see, for example, Refs. [5–8]) One of the most interesting variants of tissue P systems was presented in Ref. [1]. In that paper, tissue P systems are endowed with the ability of creating new cells based on *mitosis* or cellular division, yielding *tissue-like P systems with cell division*. The ability of cell division allows us to obtain an exponential amount of cells in linear time and to design polynomial time cellular solutions to **NP**-complete problems.

In this paper we present a solution to the Partition problem via a family of recognizer tissue-like P systems with cell division. 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 the Partition problem in the framework of tissue-like P systems with cell division.

The paper is organized as follows. First we recall some preliminaries. Next, recognizer tissue-like P systems with cell division are briefly described in Section 3. A linear-time solution to the Partition problem is presented in Section 4, including a short overview of the computation and necessary resources. Finally, some conclusions and new open research directions are presented.

2. 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 (of length 0) is 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

[*]Author to whom correspondence should be addressed.

doi:10.1166/jctn.2010.1435

of Σ^*. A *multiset* over a set A is a pair (A, f) where $f\colon A \to \mathbb{N}$ is a mapping. If $m = (A, f)$ is a multiset then its *support* is defined as $supp(m) = \{x \in A \mid f(x) > 0\}$ 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, then it is denoted by $m = a_1^{f(a_1)} a_2^{f(a_2)} \cdots a_k^{f(a_k)}$, where $supp(m) = \{a_1, \dots, a_k\}$, and for each element a_i, $f(a_i)$ is called the multiplicity of a_i. 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 an (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). The *degree* of $v \in V$ is the number of adjacent vertices to v.

In what follows we assume that the reader is already familiar with the basic notions and the terminology underlying P systems. For details, see Ref. [9].

3. TISSUE-LIKE P SYSTEMS WITH CELL DIVISION

3.1. Definition of the Model

Formally, a *tissue-like P system with cell division* of degree $q \geq 1$ is a tuple of the form $\Pi = (\Gamma, \varepsilon, m_1, \dots, m_q, \mathcal{R}, i_0)$ where Γ is a finite *alphabet*, whose symbols are called *objects*; $\varepsilon \subseteq \Gamma$; $m_1, \dots, m_q$ are the multisets of objects associated with the cells in the initial configuration; $i_0 \in \{0, 1, 2, \dots, q\}$; and $\mathcal{R}$ is a finite set of rules of the following forms:

(a) *Communication rules*: $(i, u/v, j)$, for $i, j \in \{0, \dots, q\}, i \neq j$, $u, v \in \Gamma^*$.

(b) *Division rules*: $[a]_i \to [b]_i[c]_i$, where $i \in \{1, 2, \dots, q\}$, $a \in \Gamma$ and $b, c \in \Gamma \cup \{\lambda\}$.

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) labelled by $1, 2, \dots, q$. 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). We interpret that $\varepsilon \subseteq \Gamma$ is the set of objects placed in the environment, each one of them in an arbitrarily large amount of copies. If, during a computation, the systems send some objects not in ε to the environment, then they will not have unbounded multiplicity.

The communication rule $(i, u/v, j)$ can be applied to two cells i and j (with $i, j \in \{0, \dots, q\}$, and $i \neq j$) 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. In the case where either i or j are equal to 0, then the objects are moved to/from the environment.

In the cell-like models of P systems, one usually explicitly defines the initial membrane structure. However, in tissue-like P systems the structure is not given. The communication rules implicitly determine a graph, where the nodes are the cells and the edges indicate if it is possible for pairs of cells to communicate directly. Note that this is a dynamic graph, because new nodes that are produced by the application of division rules can appear.

The application of a division rule $[a]_i \to [b]_i[c]_i$ to a cell $i \in \{1, \dots, q\}$ containing object a divides this cell into two new cells with the same label. All the objects in the original cell are replicated and copied to 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 c in the other one. The rules to be used for the new membranes are precisely identified by the label of the membranes.

Rules are applied in a way that is standard in the framework of membrane computing, that is, in a maximally parallel way (a universal clock is assumed). In one step, each object in a membrane can only be used for one rule (nondeterministically chosen when there are several possibilities), but any object which can participate in a rule of any form must do it, i.e., 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 other than the one involved in the division rule inside that cell do not evolve in that step.

3.2. Recognizer Systems

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 computing efficiency of solving **NP**-complete decision problems, a special class of tissue P systems with cell division is introduced in Ref. [1]: *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 Ref. [10] 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 whether or not a string belongs 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 that encodes 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 the output to the environment, in the last step of the computation.

A recognizer tissue-like P system with cell division of degree $q \geq 1$ is a tuple $\Pi = (\Gamma, \Sigma, \varepsilon, m_1, \ldots, m_q, \mathcal{R}, i_{\text{in}}, i_0)$ where

- $(\Gamma, \varepsilon, m_1, \ldots, m_q, \mathcal{R}, i_0)$ is a tissue-like P system with cell division, of degree $q \geq 1$ (as defined in the previous section), $i_0 = 0$ and $m_1, \ldots, m_q$ 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 $m_1, \ldots, m_q$, but not present in ε.
- Σ is an (input) alphabet strictly contained in Γ.
- $i_{\text{in}} \in \{1, \ldots, q\}$ 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 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 $(m_1, m_2, \ldots, m_{i_{\text{in}}} m, \ldots, m_q)$, 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 to the corresponding halting configuration of $\mathcal{C}$.

Definition 3.1. We say that a decision problem $X = (I_X, \theta_X)$ is solvable in polynomial time by a family $\Pi = \{\Pi(n) \mid n \in \mathbb{N}\}$ of recognizer tissue-like P systems with cell division if the following holds:

- The family Π 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 Π) 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 Π 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 $cod(u)$ halts and, moreover, it performs at most $p(|u|)$ steps;

—the family Π 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 Π 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 *confluence* on every P system $\Pi(n)$, 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 reduction and under complement.

4. A SOLUTION TO THE PARTITION PROBLEM

Let us recall that a partition of a set V is a family of non-empty pairwise disjoint subsets of V such that the union of the subsets of the family is equal to V. The Partition problem (`PART`) can be stated as follows: Let V be a finite set and let w be a weight function on V, w: $V \to \mathbb{N}$ (that is, an additive function). Decide whether or not there exists a partition $\{V_1, V_2\}$ of V such that $w(V_1) = w(V_2)$.

We shall prove that the Partition problem can be solved in a linear time (in $\{n, \lg k\}$ where $k = w_1 + \cdots + w_n$) by a family of recognizer tissue-like P systems with cell division (in the sense of Definition 3.1). Given an instance $u = (V, w)$ of the Partition problem, we denote $V = \{v_1, v_2, \ldots, v_n\}$. Such an instance will be represented by $u = (n, (w_1, \ldots, w_n))$, where $w_i = w(v_i)$, for each i $(1 \leq i \leq n)$.

The initial configuration of each system of the family presented below has two cells (labelled by 1 and 2). We shall address the resolution via a brute force algorithm, which consists of the following stages:

(1) *Generation Stage*: All the possible subsets of V are generated by the application of cell division rules;

(2) *Pre-checking Stage*: In this stage, the weight of each of the subsets of V is calculated;

(3) *Checking Stage*: We compare for each subset if its weight and the weight of its complementary set are equal; and

(4) *Answer Stage*: According to the previous stage, an affirmative or negative response is obtained.

For each $n, k \in \mathbb{N}$ we consider the recognizer tissue-like P system with cell division $\Pi(\langle n, k\rangle) = (\Gamma, \Sigma, \varepsilon, m_1, m_2, R, i_{\text{in}})$ defined as follows

- $\Gamma = \{A_i, \overline{A}_i, B_i', B_i \mid 1 \leq i \leq n\} \cup$
 $\{a_i \mid 1 \leq i \leq \lceil \lg n \rceil + \lceil \lg k \rceil + 14\} \cup$
 $\{c_i, v_i \mid 1 \leq i \leq n\} \cup \{d_i, g_i \mid 1 \leq i \leq \lceil \lg n \rceil + 1\} \cup$
 $\{e_i \mid 1 \leq i \leq \lceil \lg n \rceil + \lceil \lg k \rceil + 5\} \cup$
 $\{A_{ij}, B_{ij} \mid 1 \leq i \leq n \wedge 1 \leq j \leq \lceil \lg k \rceil + 1\} \cup$
 $\{b, d, d_0, p, q, f_0, f_1, f_2, T, S, N, \texttt{yes}, \texttt{no}\}$
- $\Sigma = \{v_1, \ldots, v_n\}$
- $\varepsilon = \Gamma \setminus \{a_1, b, c_1, \texttt{yes}, \texttt{no}, d, A_1, \ldots, A_n, \overline{A}_1, \ldots, \overline{A}_n\}$.
- $m_1 = a_1\, b\, c_1\, \texttt{yes}\, \texttt{no}$ and $m_2 = d\, A_1 \ldots A_n\, \overline{A}_1 \ldots \overline{A}_n$.

We consider that in the environment there are infinitely many copies of each object from ε, and no copies of any element in $\Gamma \setminus \varepsilon$.

- R is the following set of rules:

(1) *Division rules:*

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

(2) *Communication rules:*

$r_{2,i} \equiv (1, a_i/a_{i+1}, 0)$, for $i = 1, \ldots, n + \lceil \lg n \rceil + \lceil \lg k \rceil + 11$

$r_{3,i} \equiv (1, c_i/c_{i+1}^2, 0)$, for $i = 1, \ldots, n$

$r_4 \equiv (1, c_{n+1}/d, 2)$

$r_5 \equiv (2, c_{n+1}/d_0 g_1, 0)$

$r_{6,i} \equiv (2, g_i/g_{i+1}^2, 0)$, for $i = 1, \ldots, \lceil \lg n \rceil$

$r_7 \equiv (2, d_0/d_1 e_2, 0)$

$r_{8,i} \equiv (2, d_i/d_{i+1}^2, 0)$, for $i = 1, \ldots, \lceil \lg n \rceil$

$r_9 \equiv (2, d_{\lceil \lg n \rceil + 1}/d_{\lceil \lg n \rceil + 2}, 0)$

$r_{10,i} \equiv (2, e_i/e_{i+1}, 0)$, for $i = 1, \ldots, \lceil \lg n \rceil + \lceil \lg k \rceil + 4$

$r_{11,i} \equiv (2, g_{\lceil \lg n \rceil + 1} B_i/B_i', 0)$, for $i = 1, \ldots, n$

$r_{12,i} \equiv (2, B_i' \overline{A}_i/B_{i1}, 0)$, for $i = 1, \ldots, n$

$r_{13,i} \equiv (2, d_{\lceil \lg n \rceil + 2} \overline{A}_i/A_{i1}, 0)$, for $i = 1, \ldots, n$

$r_{14,ij} \equiv (2, B_{ij}/B_{ij+1}^2, 0)$, for $i = 1, \ldots, n$ and $j = 1, \ldots, \lceil \lg k \rceil$

$r_{15,ij} \equiv (2, A_{ij}/A_{ij+1}^2, 0)$, for $i = 1, \ldots, n$ and $j = 1, \ldots, \lceil \lg k \rceil$

$r_{16,i} \equiv (2, B_{i,\lceil \lg k \rceil + 1} v_i/p, 0)$, for $i = 1, \ldots, n$

$r_{17,i} \equiv (2, A_{i,\lceil \lg k \rceil + 1} v_i/q, 0)$, for $i = 1, \ldots, n$

$r_{18} \equiv (2, pq/\lambda, 0)$

$r_{19} \equiv (2, e_{\lceil \lg n \rceil + \lceil \lg k \rceil + 5}/f_0 f_1, 0)$

$r_{20} \equiv (2, f_0 p/\lambda, 0)$

$r_{21} \equiv (2, f_0 q/\lambda, 0)$

$r_{22} \equiv (2, f_1/F_2, 0)$

$r_{23} \equiv (2, f_0 f_2/T, 0)$

$r_{24} \equiv (2, T/\lambda, 1)$

$r_{25} \equiv (1, bT/S, 0)$

$r_{26} \equiv (1, S\texttt{yes}/\lambda, 0)$

$r_{27} \equiv (1, a_{n+\lceil \lg n \rceil + \lceil \lg k \rceil + 12} b/N, 0)$

$r_{28} \equiv (1, \texttt{no}N/\lambda, 0)$

- $i_{\text{in}} = 2$, is the label of the input cell.

This family of recognizer tissue-like P systems with cell division consists of *non-deterministic* systems, since several division rules can be applied in the cells labelled by 2. Nonetheless, if a division rule has not been applied yet to a cell labelled by 2, then it will be applied in the subsequent steps since in the initial configuration, the unique cell labelled by 2 contains the objects $A_1, A_2, \ldots, A_n$, i.e., with respect to the division rules, the systems are confluent.

In order to prove that the family $\Pi = (\Pi(t))_{t \in \mathbb{N}}$ defined above provides a linear solution to the Partition problem we need a polynomial encoding (cod, s) of the set of instances of such a problem in the family Π.

We consider a polynomial encoding (cod, s) defined as follows: for each instance $u = (n, (w_1, \ldots, w_n))$ we define $s(u) = \langle n, w_1 + \cdots + w_n \rangle$ (where $\langle \cdot, \cdot \rangle$ denotes the bijection from $\mathbb{N}^2$ to $\mathbb{N}$ defined as $\langle x, y \rangle = (((x+y)(x+y+1))/2) + x$) and $cod(u) = v_1^{w_1} \ldots v_n^{w_n}$.

In this way, the instance $u = (n, (w_1, \ldots, w_n)) \in I_{\text{PART}}$ is processed by the tissue-like P system $\Pi(s(u))$ with the multiset $cod(u)$ provided in the corresponding input cell.

Next, we provide an informal description of the computations of the system $\Pi(s(u))$ with input $cod(u)$ for a general instance u of the Partition problem, and we show that the family defined above is polynomially uniform by deterministic Turing machines.

4.1. An Overview of the Computation

We informally describe here how the recognizer tissue-like P system with cell division $\Pi(s(u))$ with input $cod(u)$ works.

Let us start with the *generation stage*. In this stage we have two parallel processes. On the one hand, in the cell labelled by 1 we have two counters: a_i, which will be used in the output stage, and c_i, which will be multiplied until step n, where 2^n copies of c_{n+1} are obtained. On the other hand, in the cell labelled by 2, the division rules are applied. For each object A_i we produce two cells labelled by 2, one of them containing a new object B_i and the other one not. After the appropriate divisions, in the step n we obtain exactly 2^n cells with label 2, each of them encoding a different subset of V.

The *pre-checking stage* starts at the step $(n+1)$, where each cell labelled by 2 trades the object d for the counter c_{n+1} from cell 1 (by applying the rule r_4 in parallel). From that moment on, only the evolution of the counter a_i will be performed in cell 1, until step $n + \lceil \lg n \rceil + \lceil \lg k \rceil + 13$, via the rules $r_{1,i}$ $(n+2 \leq i \leq n + \lceil \lg n \rceil + \lceil \lg k \rceil + 12)$.

Note that in the next step, the objects c_{n+1} in the cells labelled by 2 will trigger the rules r_5 and r_7 in the next two steps, thus bringing in the counter g_i in step $n+2$, and the counters d_i and e_i in step $n+3$.

From step $n+3$ to step $n + \lceil \lg n \rceil + 3$ the counter g_i duplicates itself (with the rules $r_{6,i}$) until producing at least n copies of the object $g_{\lceil \lg n \rceil + 1}$, and in a further step, it yields the trading of the objects B_i in each cell with label 2 for the objects B_i' from the environment (by the application of the rules $r_{11,i}$).

In step $n + \lceil \lg n \rceil + 5$, each pair of objects B_i' and $\overline{A}_i$ that appear in a cell labelled by 2 are traded for an object B_{i1} by applying the rules $r_{12,i}$.

In parallel, from step $n+4$ to step $n + \lceil \lg n \rceil + \lceil \lg k \rceil + 8$ the counter e_i is evolving until reaching the object $e_{\lceil \lg n \rceil + \lceil \lg k \rceil + 5}$ (by applying the rules $r_{10,i}$). Moreover, from step $n+4$ to step $n + \lceil \lg n \rceil + 4$ the counter d_i duplicates itself (by the rules $r_{8,i}$) until getting at least n copies of the object $d_{\lceil \lg n \rceil + 1}$. In the next step, the rule r_9 trades the objects $d_{\lceil \lg n \rceil + 1}$ in the cells with label 2 for the objects $d_{\lceil \lg n \rceil + 2}$. The arrival of these objects to a cell with label 2 produces the trading of the objects $\overline{A}_i$ (which remain in the cell after the application of the rules $r_{12,i}$) for objects A_{i1} in step $n + \lceil \lg n \rceil + 6$ (by applying the rules $r_{13,i}$).

In this way, we have in each cell with label 2 a pair of complementary subsets, encoded by the objects B_{i1} and A_{i1}, respectively.

From step $n+\lceil \lg n\rceil+7$ to step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+7$ the number of objects B_{i1} and A_{i1} are multiplied by 2 (by application of the rules $r_{14,ij}$ and $r_{15,ij}$, respectively) to reach, at least, k copies of the objects $B_{i,\lceil \lg k\rceil+1}$ and $A_{i,\lceil \lg k\rceil+1}$ $(1\le i\le n)$. Recall that $k=w_1+\cdots+w_n$ represents the total weight of the initial set.

In order to obtain the weight of each one of the subsets, we take each pair of objects $B_{i,\lceil \lg k\rceil+1}$ and v_i (respectively, $A_{i,\lceil \lg k\rceil+1}$ and v_i) that appear in a cell with label 2, and they are traded for an object p (respectively, for an object q) according to the rules $r_{16,i}$ (respectively, $r_{17,i}$). The *checking stage* starts in step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+8$ with the application of the rule r_{18} which removes from the cells labelled by 2 as many pairs of objects p and q as possible. Therefore, if a cell 2 encodes a pair of subsets of weight k, then all the objects p and q will be deleted in this cell. Otherwise, at least one object p or q will remain in this cell.

The *answer stage* starts in step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+9$. In the cells with label 2, the object $e_{\lceil \lg n\rceil+\lceil \lg k\rceil+5}$ is traded for the objects f_0 and f_1 by the rule r_{19}. From this step on, there are two possible situations:

Case 1: Let us suppose that there exists a couple of complementary subsets of V with weight k. In this case, there will exist a cell 2 such that it does not contain any object p or q after step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+9$. Therefore, in the next step, neither rule r_{20} nor r_{21} can be applied in such cell. However, rule r_{22} is applied, allowing the evolution of the counter f_1 to f_2. This object together with the object f_0 produces the object T that, in step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+12$ goes to the cell labelled by 1.

In the next step, such object T and object b that is waiting in cell 1 from the beginning produce the object S. This object allows to send an object `yes` to the environment in step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+14$, which ends the computation. In this case, we have an accepting computation.

Case 2: Let us suppose now that there does not exist a pair of complementary subsets of V such that their weights are both equal to $k/2$. In this case, all the cells labelled by 2 contain either objects p or q (but not both of them simultaneously). Then, in step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+10$, the object f_0 is removed from these cells labelled by 2 together with a copy of p or q (by application of the rules r_{20} or r_{21}). In the meantime, the object f_1 evolves to f_2 (by the rule r_{22}). In this way, after step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+13$ the object b remains in the cell 1. This object together with the object $a_{n+\lceil \lg n\rceil+\lceil \lg k\rceil+14}$ produces an object N, which is sent to the environment together with an object `no` in step $n+\lceil \lg n\rceil+\lceil \lg k\rceil+15$. This step ends the computation with a negative answer.

4.1.1. Polynomial Uniformity of the Family

In order to stablish that the family $\Pi=(\Pi(t))_{t\in\mathbb{N}}$ is polynomially uniform by deterministic Turing machines firstly we note that the set of rules associated with the system $\Pi(\langle n,k\rangle)$ is described in a recursive way. Hence, we only need to show that the amount of necessary resources for defining the system is polynomial in $\max\{n,\lceil \lg k\rceil\}$. The necessary resources for building $\Pi(\langle n,k\rangle)$ are the following:

Size of the alphabet: $2n\cdot\lceil \lg k\rceil+7n+2\lceil \lg k\rceil+3\lceil \lg n\rceil+36\in\theta(n\cdot\lceil \lg k\rceil)$,

Initial number of cells: $2\in\theta(1)$,

Initial number of objects: $2n+6\in\theta(n)$,

Number of rules: $2n\cdot\lceil \lg k\rceil+6n+2\lceil \lg k\rceil+5\lceil \lg n\rceil+33\in\theta(n\cdot\lceil \lg k\rceil)$,

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

Then, we have the following result:

THEOREM 4.1. PART$\in$ $\mathbf{PMC}_{TD}$.

Taking into account that PART is an **NP**-complete problem, we can deduce the following result.

COROLLARY 4.1. $\mathbf{NP}\subseteq\mathbf{PMC}_{TD}$.

5. CONCLUSIONS AND FUTURE WORK

In a similar way as in other P system models, the ability of obtaining an exponential amount of new cells during the computation can be used for trading space for time and obtaining polynomial-time solutions to **NP**-hard problems. Moreover, the solution to the Partition problem presented here runs in a number of cellular steps which is in $O(n,\lg k)$, improving (as far as the number of cellular steps used in the model is concerned) the previous result obtained in the cell-like model in Ref. [11], where the number of steps was in $O(n,k)$.

Comparing the cell-like model with the tissue-like model is not a straightforward task. Both approaches have several important differences, apart from the way of arranging the membrane structure. For example, in the (cell-like) P systems with active membranes model, membranes have electrical charges associated with them, and the communication rules are applied in a sequential way (rewriting rules are also allowed, and are applied in a maximally parallel way). In the tissue-like case, cells do not have polarizations, and only communication rules are allowed (and they are applied in a maximally parallel way).

There are still open problems related to the complexity classes in tissue-like P systems. For example, it has been proved that in the cell-like model, allowing rules for membrane creation yields computing power of at least *PSPACE*.[12] However, it remains to be investigated whether the complexity class associated with tissue-like P systems with membrane creation contains **NP** or *PSPACE*.

Another interesting issue is to get rid of the assumption of having an unbounded amount of objects in the environment. This could be done by taking advantage of the fact that when a cell divides, it replicates all of the objects within it.

Acknowledgment: The authors 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. Gh. Păun, M. J. Pérez-Jiménez, and A. Riscos-Núñez, Tissue P System with Cell Division, Second Brainstorming Week on Membrane Computing, Sevilla, Report RGNC 01/2004 (**2004**), pp. 380–386.
2. Gh. Păun, *Journal of Computer and System Sciences* 61, 108 (**2000**).
3. Gh. Păun, Y. Sakakibara, and T. Yokomori, *Publ. Math. Debrecen*, 60, 635 (**2002**).
4. C. Martín Vide, J. Pazos, Gh. Păun, and A. Rodríguez Patón, *Lecture Notes in Computer Science* 2387, 290 (**2002**).
5. A. Alhazov, R. Freund, and M. Oswald, *Lecture Notes in Computer Science* 3572, 100 (**2005**).
6. F. Bernardini and M. Gheorghe, *Soft Computing* 9, 640 (**2005**).
7. R. Freund, Gh. Păun, and M. J. Pérez-Jiménez, *Theoretical Computer Science* 330, 101 (**2005**).
8. V. J. Prakash, On the Power of Tissue P Systems Working in the Maximal-One Mode, *Preproc. Workshop on Membrane Computing*, Tarragona, Report RGML 28/03 (**2003**), pp. 356–364.
9. Gh. Păun, Membrane Computing. An Introduction, Berlin, Springer-Verlag (**2002**).
10. M. J. Pérez-Jiménez, A. Romero-Jiménez, and F. Sancho-Caparrini, A polynomial complexity class in P systems using membrane division, *Proc. 5th Workshop on Descriptional Complexity of Formal Systems, DCFS* (**2003**), pp. 284–294.
11. M. A. Gutiérrez-Naranjo, M. J. Pérez-Jiménez, and A. Riscos-Núñez, *Soft Computing* 9, 673 (**2005**).
12. M. A. Gutiérrez-Naranjo, M. J. Pérez-Jiménez, and F. J. Romero-Campero, *Lecture Notes in Computer Science* 3850, 241 (**2006**).

Received: 12 November 2008. Accepted: 2 February 2009.