Solving the Partition Problem by Using Tissue-like P Systems with Cell Division

Daniel Díaz-Pernil, Miguel A. Gutiérrez-Naranjo,
Mario J. Pérez-Jiménez, Agustín Riscos-Núñez

Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla
Avda. Reina Mercedes s/n, 41012, Sevilla, Spain
E-mails: {sbdani,magutier,marper,ariscosn}@us.es

Summary. Tissue-like P systems with cell division is a computing model in the framework of Membrane Computing that shares with the spiking neural P system model a similar biological inspiration. Namely, both models are based on the intercellular communication and cooperation between neurons, respectively. Due to this fact, in both models the devices have the same structure: a network of elementary units (cells in a tissue and interconnected neurons, respectively). Nonetheless, the two models are quite different. One of the differences is the ability of tissue-like P systems with cell division for increasing the number of cells during the computation. In this paper we exploit this ability and present a polynomial-time solution for the (**NP**-complete) Partition problem via a uniform family of such P systems.

1 Introduction

Tissue-like P systems with cell division [13] is a computing model in the framework of membrane computing based on inter-cellular communication and cooperation between neurons. It shares some common features with another emerging membrane computing model based on spiking neurons, the *spiking neural P systems* [15]. Their main common feature is that in the computational devices of both models we have certain processor units (called *cells* or *neurons*, respectively) that process in parallel some pieces of information and send signals to other processor units along links that connect some of them. Such links do not follow any scheme, and this is one of the features which distinguishes these models from the initial model in membrane computing, the cell-like model, where membranes are hierarchically arranged in a tree-like structure (see [10]). The biological inspiration for this cell-like model is the morphology of cell, where small vesicles are surrounded by larger ones.

In spiking neural P systems and in tissue-like P systems with cell division the membrane structure is tissue-like and the links between cells form a general graph (directed graph for spiking neural P systems and undirected graph for tissue-like P systems). Nonetheless there are important differences between both models. For instance, in spiking neural P systems only one type of object (called *spike*) is used to encode the information in the cells. Specific rules are used for evolving populations of spikes and the time is used as a support of information[1].

As we said above, in tissue-like P systems we can picture the cells as nodes of a general undirected graph. The edges of such graph are not given explicitly, but they are deduced from the set of rules, as it will be explained later. The communication among cells is based on symport/antiport rules in P systems[2]. Symport rules move objects across a membrane together in one direction, whereas antiport rules move objects across a membrane in opposite directions.

From the seminal definition of tissue P systems [7, 8], several research lines have been developed and other variants have arisen (see, for example, [1, 2, 3, 4, 6, 14]). One of the most interesting variants of tissue P systems was presented in [13]. 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*, and the underlying graph is implicitly described by the rules.

This cellular division is other of the main differences between the model followed in this paper and spiking neural P systems. The ability of cell division allows us to obtain an exponential amount of cells in linear time and to design cellular solutions to **NP**-complete problems in polynomial time. Nonetheless, the solutions to **NP**-complete problems in the spiking neural P systems literature need an exponential amount of pre-computed devices (see [5]).

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

The paper is organized as follows: first we recall some preliminaries and the definition of tissue-like P systems with cell division. Next, recognizing tissue-like P systems with cell division are briefly described in section 3. A linear–time solution to the Partition problem is presented in the section 4, including a short overview of the computation and of the necessary resources. 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

[1] A detailed description can be found in [16] and the references therein.
[2] This way of communication for P systems was introduced in [12].

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

A *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). The *degree* of $v \in V$ is the number of adjacent vertices to v.

In what follows we assume the reader is already familiar with the basic notions and the terminology underlying P systems. For details, see [11].

3 Tissue-like P Systems with Cell Division

In the first definition of the model of tissue P systems [7, 8] the membrane structure did not change along the computation. Based on the cell-like model of P systems with active membranes, Gh. Păun et al. presented in [13] a new model of tissue P systems *with cell division*. The biological inspiration is clear: alive 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 that cells have not polarizations (the contrary holds in the cell-like model of P systems with active membranes, see [11]); the cells obtained by division have the same labels as the original cell and if a cell is divided, its interaction with other cells or with the environment is blocked during the mitosis process. 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. $\mathcal{E} \subseteq \Gamma$.
4. $\mathcal{R}$ is a finite set of rules of the following form:
   (a) *Communication rules*: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \ldots, q\}, i \neq j$, $u, v \in \Gamma^*$.
   (b) *Division rules*: $[a]_i \to [b]_i[c]_i$, where $i \in \{1, 2, \ldots, q\}$ and $a, b, c \in \Gamma$.

5. $i_0 \in \{0, 1, 2, \dots, q\}$.

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) 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).

The communication rules determine a virtual graph, where the nodes are the cells and the edges indicated if it is possible for pairs of cells to communicate directly. This is a dynamical graph, because of new nodes can appear produced by the application of division rules.

The strings $w_1, \dots, w_q$ 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 over 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 $[a]_i \to [b]_i[c]_i$ is applied over 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 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), 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 inside that cell do not evolve in that step.

3.1 Recognizing 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 computing efficiency for solving **NP**-complete decision problems, a special class of tissue P systems with cell division is introduced in [13]: *recognizing tissue P systems*. The key idea of such recognizing systems is the same one as from recognizing P systems with cell-like structure.

Recognizing cell-like P systems were introduced in [9] 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, recognizing cell-like P systems are associated with P systems with *input* in a natural way. The data encoding to 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, and 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 recognizing tissue-like P system with cell division of degree $q \geq 1$ is a tuple

$$\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, \dots, w_q, \mathcal{R}, i_{in}, i_0)$$

where

- $(\Gamma, \mathcal{E}, w_1, \dots, w_q, \mathcal{R}, i_0)$ is a tissue-like P system with cell division of degree $q \geq 1$ (as defined in the previous section), $i_0 = env$ and $w_1, \dots, w_q$ 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 $w_1, \dots, w_q$, but not present in $\mathcal{E}$.
- Σ is an (input) alphabet strictly contained in Γ.
- $i_{in} \in \{1, \dots, q\}$ 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 $(w_1, w_2, \dots, w_{i_{in}} w, \dots, w_q; \mathcal{E})$, 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 to 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 recognizing tissue-like 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 (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.*

In the above definition we have imposed to every P system $\Pi(n)$ to be *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 recognizing tissue-like P systems with cell division in polynomial time. This class is closed under polynomial reduction and under complement.

4 A solution for 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 settled 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 $w(V_1) = w(V_2)$.

Next, we shall prove that the Partition problem can be solved in a linear time (in $\{n, \lg k\}$ where $k = \omega_1 + \cdots + \omega_n$) by a family of recognizing tissue-like P systems with cell division (in the sense of Definition 1).

Given an instance $u = (V, w)$ of the Partition Problem, we will denote $V = \{v_1, v_2, \ldots, v_n\}$. Such instance will be represented by $u = (n, (w_1, \ldots, w_n))$, where $w_i = w(v_i)$, for each i $(1 \leq i \leq n)$.

Next, we present a family of recognizing tissue-like P systems with cell division where at the initial configuration each system of the family has two cells (labelled by 1 and 2). We shall address the resolution via a brute force algorithm, which consists in the following stages:

- *Generation Stage*: All the possible subsets of V are generated by the application of cell division rules.
- *Pre–checking Stage*: In this stage, the weight of each of the subsets of V is calculated.
- *Checking Stage*: We compare for each subset if its weight and the weight of its complementary set are equal.

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

For each $n, k \in \mathbb{N}$ we will consider the recognizing tissue-like P system with cell division and symport/antiport rules

$$\Pi(< n, k >) = (\Gamma, \Sigma, \mathcal{E}, w_1, w_2, R, i_{in})$$

defined as follows

- $\Gamma = \{A_i, \overline{A}_i, B'_i, B_i \,:\, 1 \leq i \leq n\} \cup$
  $\{a_i \,:\, 1 \leq i \leq \lceil \lg n \rceil + \lceil \lg k \rceil + 14\} \cup \{c_i, v_i \,:\, 1 \leq i \leq n\} \cup$
  $\{d_i, g_i \,:\, 1 \leq i \leq \lceil \lg n \rceil + 1\} \cup \{e_i \,:\, 1 \leq i \leq \lceil \lg n \rceil + \lceil \lg k \rceil + 5\} \cup$
  $\{A_{ij}, B_{ij} \,:\, 1 \leq i \leq n \wedge 1 \leq j \leq \lceil \lg k \rceil + 1\} \cup$
  $\{b, D, D_1, p, q, E_1, F_1, F_2, T, S, N, \texttt{yes}, \texttt{no}\}$
- $\Sigma = \{v_1, \ldots, v_n\}$
- $\mathcal{E} = \Gamma \setminus \{a_1, b, c_1, \texttt{yes}, \texttt{no}, D, A_1, ..., A_n, \overline{A_1}, \ldots, \overline{A_n}\}$.
- $w_1 = a_1\, b\, c_1\, \texttt{yes}\,\texttt{no}$ and $w_2 = D\, A_1 \ldots A_n, \overline{A}_1\, \ldots, \overline{A}_n$.
  Also, we consider that in the environment there are infinitely many copies of each object from $\mathcal{E}$, and no copies of any element in $\Gamma \setminus \mathcal{E}$.
- R is the following set of rules:
  1. *Division rules:*
     $r_{1,i} \equiv [A_i]_2 \,\rightarrow\, [B_i]_2[\lambda]_2$, for $i = 1, \ldots, n$
  2. *Communication rules:*
     $r_{2,i} \equiv (1, a_i/a_{i+1}, 0)$, for $i = 1, \ldots, n + \lceil \lg n \rceil + \lceil \lg k \rceil + 11$
     $r_{3,i} \equiv (1, c_i/c^2_{i+1}, 0)$, for $i = 1 \ldots, n$
     $r_4 \equiv (1, c_{n+1}/D, 2)$
     $r_5 \equiv (2, c_{n+1}/D_1 g_1, 0)$
     $r_{6,i} \equiv (2, g_i/g^2_{i+1}, 0)$, for $i = 1, \ldots, \lceil \lg n \rceil$
     $r_7 \equiv (2, D_1/d_1 e_2, 0)$
     $r_{8,i} \equiv (2, d_i/d^2_{i+1}, 0)$, for $i = 1, \ldots, \lceil \lg n \rceil$
     $r_9 \equiv (2, d_{\lceil \lg n \rceil}/d_{\lceil \lg n \rceil + 1}, 0)$
     $r_{10,i} \equiv (2, e_i/e_{i+1}, 0)$, 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 $i = 1, \ldots, n$
     $r_{14,ij} \equiv (2, B_{ij}/B^2_{ij+1}, 0)$, for $i = 1, \ldots, n$ and $j = 1, \ldots, \lceil \lg k \rceil$
     $r_{15,ij} \equiv (2, A_{ij}/A^2_{ij+1}, 0)$, 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 $i = 1, \ldots, n$
     $r_{18} \equiv (2, pq/\lambda, 0)$
     $r_{19} \equiv (2, e_{\lceil \lg n \rceil + \lceil \lg k \rceil + 5}/E_1 F_1, 0)$
     $r_{20} \equiv (2, E_1 p/\lambda, 0)$
     $r_{21} \equiv (2, E_1 q/\lambda, 0)$
     $r_{22} \equiv (2, F_1/F_2, 0)$
     $r_{23} \equiv (2, E_1 F_2/T, 0)$

$r_{24} \equiv (2, T/\lambda, 1)$
$r_{25} \equiv (1, bT/S, 0)$
$r_{26} \equiv (1, S\texttt{yes}/\lambda, 0)$
$r_{27} \equiv (1, a_{n+\lceil \lg n \rceil + \lceil \lg k \rceil + 12} b/N, 0)$
$r_{28} \equiv (1, noN/\lambda, 0)$

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

This family of recognizing tissue-like P systems with cell division and symport/antiport rules 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 next steps since in the initial configuration, the unique cell labelled by 2 contains the objects $A_1, A_2, \dots, A_n$, i.e., with respect to the division rules, the systems are confluent.

In order to justify that the family $\mathbf{\Pi} = (\Pi(t))_{t\in\mathbb{N}}$ 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 $\mathbf{\Pi}$.

We will consider a polynomial enconding (cod, s) defined as follows: for each instance $u = (n, (w_1, \dots, w_n))$ we define $s(u) =< n, w_1 + \cdots + w_n >$ and $cod(u) = v_1^{w_1}, \dots, v_n^{w_n}$.

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

Next, we will provide an informal description of the computations of the system $\Pi(s(u))$ with input $cod(u)$ for a generic instance u of the Partition problem, and we justify 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 recognizing 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, and each of them encode 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 against the counter c_{n+1} from the cell 1 (by applying in parallel the rule r_4). From that moment on, only the evolution of the counter a_i will be performed in cell 1, till the 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 the step $n + 2$, and the counters d_i and e_i in the step $n + 3$.

From the step $n + 3$ to the 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 against the objects B'_i from the environment (by the application of the rules $r_{11,i}$).

In the 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 against an object B_{i1} by applying the rules $r_{12,i}$.

In parallel, from the step $n+4$ to the 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 the step $n + 4$ to the 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 against 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 $r_{12,i}$) against objects A_{i1} in the 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 the step $n + \lceil \lg n \rceil + 7$ to the 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 $A_{i,\lceil \lg k \rceil + 1}$ $(1 \leq i \leq n)$. 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 against an object p (respectively, against an object q) according to the rules $r_{16,i}$ (respectively, $r_{17,i}$).

The *checking stage* starts in the 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 the 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 against the objects E_1 and F_1 by the rule r_{19}. From this step on, there are two possible situations:

- 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 the 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 E_1 produces the object T that, in the step $n + \lceil \lg n \rceil + \lceil \lg k \rceil + 12$ goes to the cell labelled by 1.
In the next step, the objects T and b that initially were in the cell 1 produce the object S. This object allows to send an object `yes` to the environment in the step $n + \lceil \lg n \rceil + \lceil \lg k \rceil + 14$, which ends the computation. In this case, we have an accepting computation.

- Let us suppose now that there does not exist a pair of complementary subsets of V such that its weights are both equal to k. In this case, all the cells labelled by 2 contain either objects p or q (but not both of them simultaneously). Then, in the step $n + \lceil \lg n \rceil + \lceil \lg k \rceil + 10$, the object E_1 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 the 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 the step $n + \lceil \lg n \rceil + \lceil \lg k \rceil + 15$. This step ends the computation with a negative answer.

Polynomial Uniformity of the Family

In order to stablish that the family $\mathbf{\Pi} = (\Pi(t))_{t \in \mathbb{N}}$ is polynomially uniform by deterministic Turing machines firstly we note that the set of rules associated with the system $\Pi(< n, k >)$ is described in a recursive way. Hence, we only need to justify that the amount of necessary resources for defining the system is polynomial in $\max\{n, \lceil \lg k \rceil\}$. The necessary resources for building $\Pi(< n, k >)$ 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 1. `PART`$\in$ $\mathbf{PMC}_{TD}$.

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

Corollary 1. $\mathbf{NP} \subseteq \mathbf{PMC}_{TD}$.

5 Conclusion and Future Work

Tissue-like P systems with cell division is a computing model in the framework of Membrane Computing that shares with the spiking neural P system model the tissue-like structure of cells and the biological inspiration, since both models are based on the intercellular communication and cooperation between neurons. Nonetheless, both models are quite different. One of the main differences is the treatment of the information and how the flow of information between rules is handled. The second main difference is the ability of tissue-like P systems with cell division for increasing the number of cells during the computation. In a similar way to other P system models, this ability can be used for trading space against time and obtaining polynomial-time solutions for **NP** problems by obtaining an exponential amount of new cells during the computation.

One of the main drawbacks of spiking neural P systems in order to design solutions for **NP** problems is that the cell structure cannot change along the computation, so in order to get solutions of hard problems, the design needs to use precomputed resources. An open research line for the future is to study if some of the features of tissue-like P systems can be adapted to spiking neural P systems in order to get new applications to these new systems.

Acknowledgment

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. A. Alhazov, R. Freund, M. Oswald: Tissue P Systems with Antiport Rules and Small Numbers of Symbols and Cells. *Lecture Notes in Computer Science* **3572**, (2005), 100–111.
2. F. Bernardini, M. Gheorghe: Cell Communication in Tissue P Systems and Cell Division in Population P Systems. *Soft Computing* **9**, 9, (2005), 640–649.
3. R. Freund, Gh. Păun, M.J. Pérez-Jiménez: Tissue P Systems with Channel States. *Theoretical Computer Science* **330**, (2005), 101–116.
4. S.N. Krishna, K. Lakshmanan, R. Rama: Tissue P Systems with Contextual and Rewriting Rules. *Lecture Notes in Computer Science* **2597**, (2003), 339–351.
5. A. Leporati, C. Zandron, C. Ferretti, G. Mauri: Solving Numerical NP-Complete Problems with Spiking Neural P Systems. 336-352. *Lecture Notes in Computes Science* **4860**, (2007), 336–352.
6. K. Lakshmanan, R. Rama: On the Power of Tissue P Systems with Insertion and Deletion Rules. In A. Alhazov, C. Martín-Vide and Gh. Păun (eds.) *Preproceedings of the Workshop on Membrane Computing*, Tarragona, Report RGML 28/03, (2003), 304–318.

7. C. Martín Vide, J. Pazos, Gh. Păun, A. Rodríguez Patón: A New Class of Symbolic Abstract Neural Nets: Tissue P Systems. *Lecture Notes in Computer Science* **2387**, (2002), 290–299.
8. C. Martín Vide, J. Pazos, Gh. Păun, A. Rodríguez Patón: Tissue P Systems. *Theoretical Computer Science*, **296**, (2003), 295–326.
9. M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini: A Polynomial Complexity Class in P Systems Using Membrane Division. *Proceedings of the 5th Workshop on Descriptional Complexity of Formal Systems, DCFS 2003*, (E. Csuhaj-Varjú, C. Kintala, D. Wotschke, Gy. Vaszyl, eds.), (2003), 284–294.
10. Gh. Păun: Computing with Membranes. *Journal of Computer and System Sciences*, **61**, 1, (2000), 108–143.
11. Gh. Păun: *Membrane Computing. An Introduction.* Springer–Verlag, Berlin, (2002).
12. A. Păun, Gh. Păun: The Power of Communication: P Systems with Symport/Antiport. *New Generation Computing*, **20**, 3, (2002), 295–305.
13. Gh. Păun, M.J. Pérez-Jiménez, A. Riscos-Núñez: Tissue P System with Cell Division. In Gh. Păun, A. Riscos-Núñez, A. Romero-Jiménez and F. Sancho-Caparrini (eds.), *Second Brainstorming Week on Membrane Computing*, Sevilla, Report RGNC 01/2004, (2004), 380–386.
14. V.J. Prakash: On the Power of Tissue P Systems Working in the Maximal-One Mode. In A. Alhazov, C. Martín-Vide and Gh. Păun (eds.). *Preproceedings of the Workshop on Membrane Computing*, Tarragona, Report RGML 28/03, (2003), 356–364.
15. M. Ionescu, Gh. Păun and T. Yokomori: Spiking Neural P Systems. *Fundamenta Informaticae*, **71**, 2-3 (2006), 279–308.
16. Gh. Păun: Twenty Six Research Topics About Spiking Neural P Systems. In *Fifth Brainstorming Week on Membrane Computing*, (M.A. Gutiérrez-Naranjo, Gh. Păun, A. Romero-Jiménez, A. Riscos-Núñez, eds.) Fénix Editora, Sevilla, 2007, 263–280.