Tissue-like P Systems Without Environment

Hepzibah A. Christinal[1], Daniel Díaz-Pernil[1],
Miguel A. Gutiérrez-Naranjo[2], Mario J. Pérez-Jiménez[2]

[1] Research Group on Computational Topology and Applied Mathematics
Department of Applied Mathematics I
{hepzi,sbdani}@us.es
[2] Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
{magutier,marper}@us.es

University of Sevilla. Avda. Reina Mercedes s/n, 41012, Sevilla, Spain

Summary. In this paper we present a tissue-like P systems model with cell division the environment has been replaced by an extra cell. In such model, we present a uniform family of recognizer P systems which solves the Subset Sum problem. This solution establishes a new frontier for the tractability of computationally hard problems in Membrane Computing, since it proves that **NP**-complete problems can be solved without an arbitrarily large amount of objects in the environment.

1 Introduction

In Membrane Computing, the *environment* is the spatial location where the P system is placed. It appears in the description of all P system models in an explicit or implicit way. In this paper, we focus on its role in the framework of tissue-like P systems.

In cell-like models, it is defined as a region surrounding the skin (and therefore the whole P system) with no rules associated. Its role is inactive. It consists exclusively on holding objects, generally sent out by the P system. Occasionally, the objects in the environment can be sent into the P system if the skin has associated a send-in rule, but this is not the usual situation. If we consider the membrane structure of a cell-like P systems as a tree with the processor units (the membranes) on the nodes, the environment can be seen as a new node, linked uniquely with the skin and able to contain multisets of objects, but no rules.

The most common point of view is considering the cell-like P system as a black box where the computation takes place and where an external observer has no access. Such observer can only watch the skin and the surrounding region from a point out of the P system. Bearing in mind this point of view, the resulting

product of the computation must be expelled to the environment in order to be observed.

In Spiking Neural P system, the environment is also considered the region surrounding the whole system. It does not belong to the formal description of the system, but it is implicitly considered, since one neuron is marked to send spikes *out* of the system.

In spite of the membrane structure of a SN P system is a general graph instead of a tree as in the cell-like model, they share a common property with respect to the environment. In both models only one membrane (neuron, in the usual terminology of SN P systems) is linked to the environment: In cell-like models, it is the skin and in the spiking model, it is the *output neuron*. Beyond this similarity, the role of the environment is even more restrictive in the case of the SN P systems. According to this model, the information is encoded in time, so the important question is to consider the moment in which the spikes are sent out by the output neuron. Such spikes are not stored and can be forgotten.

The role of the environment changes in tissue-like P systems [13, 14]. In such P systems, the cells are placed in a general graph[3], and, potentially, all of them can trade objects against the environment. The main feature of the environment is the arbitrarily large amount of objects placed in it. These objects can participate on the computation according to the symport/antiport rules associated to cells of the system. The biological inspiration it is clear, a living tissue can take from outside as much oxygen and nutrients as it needs without limitation.

This arbitrarily large amount of objects in the environment has been widely exploited in the design of efficient solutions to **NP**-problems by recognizer tissue-like P systems with cell division (see, e.g., [4, 5, 6]). In such designs, the initial resources of the devices are polynomial in the size of the input and the number of objects taken from the environment along the computation is not considered in the initial description.

From this starting point, it is natural to wonder if this singularity can be avoided. In other words, we wonder if tissue-like P systems in which environment is empty on the input can also solve **NP**-problems.

In this paper we give a positive answer to this question. We present a tissue-like P systems model with cell division where environment is supplied by a cell. To do this, we divide this cell so many time as we need. In this manner, we generate so copies of initial objects of this cell as we want.

In such model, we present a uniform family of recognizer tissue-like P systems which solves the Subset Sum problem. This solution establishes a new frontier for the tractability of computationally hard problems in Membrane Computing [8], since it proves that **NP**-complete problems can be solved without an arbitrarily large amount of objects in the environment.

Bearing in mind these considerations, if the initial amount of objects in the environment is fixed in a similar way to the cells, then the environment can be

[3] In fact, a *virtual* graph, as we will see below.

seen as a new cell w_0. The particular feature of this distinguished cell is that it cannot be divided.

The paper is organized as follows: In Section 2 we recall some basic concepts which will be used later. In Section 3 we present the model of tissue-like P systems without environment and cell division and in Section 4 a solution to the Subset Sum problem is this framework is shown. The paper finishes with some final remarks and comments on the future work.

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 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. The set of all multisets on A will be denoted by $\mathcal{M}(A)$. 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 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. If $f(a_i) = 1$, we will write a_i instead of a_i^1. In what follows we assume the reader is already familiar with the basic notions and the terminology underlying P systems[4].

3 Tissue-like P Systems without Environment

Tissue P systems were defined in [13, 14] under two biological inspirations: intercellular communication and cooperation between neurons. The common mathematical model of these two mechanisms is a net of processors dealing with symbols and communicating these symbols along channels specified in advance. From the initial definition, several research lines have been developed and other variants have arisen (see, for example, [2, 3, 7, 11, 12, 18]). Based on the cell-like model of P systems with active membranes, Gh. Păun et al. presented in [16] a new model of tissue P systems endowed 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. In this model, the tissue (of cells) is formed by the cells and a region called *environment* containing all of them. Moreover, this model deals with

[4] We refer to [15] for basic information in this ares, to [17] for a comprehensive presentation and the web site [19] for the up-to-date information.

an arbitrarily large amount of objects in the environment, and it can not divided along a computation.

Next, we present a variant of this model, in which we drop one ingredient: the arbitrary large amount of objects in the environment. The key idea is to consider a set of initial cells $w_1, \ldots, w_n$ plus an extra cell w_0. This extra cell will have the same behavior as the other ones, but it will assume the role of the environment. As we pointed out above, the resources in this cell will be also computed as initial resources and must be polynomially generated.

Formally, a *tissue-like P system without environment* (or simplifying *tissue-like P system*$_{WE+D}$) of degree $q \geq 1$ is a tuple of the form

$$\Pi = (\Gamma, env, w_1 \ldots, w_q, \mathcal{R}, i_0),$$

where:

1. Γ is a finite *alphabet*, whose symbols will be called *objects*.
2. $env(= w_0)$, is a string over Γ representing the multisets of objects associated with the environment in the initial configuration.
3. $w_1, \ldots, w_q$ are strings over Γ representing the multisets of objects associated with the cells in the initial configuration.
4. $\mathcal{R}$ is a finite set of rules of the following form:
   (a) *Communication rules*: $(i, u/v, j)$, for $i, j \in \{0, 1, \ldots, q\}, i \neq j$, $u, v \in \Gamma^*$ and 0 represents to the environment.
   (b) *Division rules*: $[a]_i \rightarrow [b]_i[c]_i$, where $i \in \{1, \ldots, q\}$ and $a, b, c \in \Gamma$. Note that the environment (labeled by 0) cannot divide.
5. $i_0 \in \{0, 1, 2, \ldots, q\}$ denotes the output region, which can be the environment ($i_0 = 0$) or the region inside a cell ($1 \leq i_0 \leq q$).

In tissue-like P systems, the graph structure of the cells is not given in an explicit way. The links between regions are provided by the set of symport/antiport rules. It is known as a *virtual graph*. In such way, two cells are linked if and only if there is a rule that allows the interchange of objects between them. In a similar way, any cell can trade objects against the environment if there exists a rule for this purpose. Notice that the rules are associated to the labels. In such way, the graph is dynamical, since new nodes can appear produced by the application of division rules.

The application of rules in this new model is the same as in usual tissue-like P systems with cell division:

- The communication rule $(i, u/v, j)$ can be applied over two regions i and j such that u is contained in cell i and v is contained in region 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 over a cell $i \in \{1, \ldots, q\}$ 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

the new cell, 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 cannot be communicated in that step.

The cells obtained by division have the same labels as the original cell and if a cell is divided, its interaction with other cells 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.

A *configuration* is an instantaneous description of the system Π, and it is represented as a tuple $(w_0, w_1, \ldots, w_q)$. Given a configuration, we can perform a computation step and obtain a new configuration by applying the rules in a parallel manner as it is shown above. A sequence of computation steps is called a *computation*. A configuration is *halting* when no rules can be applied to it. Then, a computation halts when the system reaches a halting configuration. In the literature, the output of a computation is collected from its halting configuration by reading the objects contained in the output cell.

3.1 Recognizer Tissue-like P Systems$_{WE+D}$

Complexity classes within Membrane Computing have 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 for solving **NP**-complete decision problems, a special class of P systems were introduced in [1]: recognizer P systems. The original definition corresponds to *cell-like* P systems, but it was extended in a natural way in [16] to *tissue-like* ones.

Recognizer cell-like P systems 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 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 output region, in the last step of the computation.

A recognizer tissue-like P system$_{WE+D}$ of degree $q \geq 1$ is a tuple

$$\Pi = (\Gamma, \Sigma, w_0, w_1, \ldots, w_q, \mathcal{R}, i_{in}, i_0)$$

where

- $(\Gamma, w_0, w_1, \ldots, w_q, \mathcal{R}, i_0)$ is a tissue-like P system$_{WE+D}$ of degree $q \geq 1$ (as defined in the previous section), $M(\sigma)$ is a string over $\Gamma \setminus \Sigma$, for each $\sigma \in V \cup \{w_0\}$.
- The working alphabet Γ has two distinguished objects `yes` and `no`, present in at least one copy in some initial multisets.
- 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 output region, 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 $(w_0, w_1, w_2, \ldots, w_{i_{in}} \cup w_i, \ldots, w_q)$, that is, after adding the multiset w to the contents of the input cell i_{in}.

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$_{WE+D}$ 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.*

We denote by $\mathbf{PMC}_{TD-E}$ the set of all decision problems which can be solved by means of recognizer tissue-like P systems$_{WE+D}$ in polynomial time. This class is closed under polynomial reduction and under complement.

4 A Solution for the Subset Sum Problem

The Subset Sum problem is very well-known. It can be settled as follows: *Given a finite set V, a weight function, $w : V \to \mathbb{N}$, and a constant $k \in \mathbb{N}$, determine whether or not there exists a subset $B \subseteq V$ such that $w(B) = k$.*

Next, we prove that the Subset Sum problem can be solved in a linear time (in $\{n, \log k\}$) by a family of recognizer tissue-like P systems$_{WE+D}$. An instance $u = (V, w, k)$ of the Subset Sum Problem with $V = \{v_1, v_2, \ldots, v_n\}$ will be represented by $u = (n, (w_1, \ldots, w_n), k)$, where $w_i = w(v_i)$, for each i $(1 \leq i \leq n)$. Such an instance will be encoded as the multiset $cod(u) = \{\{v_i^j : w(A_i) = j \wedge i \in \{1, \ldots, n\}\}\} \cup \{\{q^k\}\}$.

Next, we present a family of recognizer tissue-like P systems$_{WE+D}$ with cell division where at the initial configuration each system of the family has four regions (labeled by 0,1,2 and 3).

We will address the resolution via a brute force algorithm, which consists in the following stages:

- *generation stage*: all possible subsets of V are generated by successive application of division rules;
- *pre-checking stage*: the weight of each subset of V is calculated;
- *checking stage*: It is check if there exists a subset of V with weight equal to k;
- *output stage*: an affirmative or negative answer to the problem is given, according to the results of the previous stage.

For each $(n, k) \in \mathbb{N}^2$ we will consider the system

$$\Pi(n, k) = (\Gamma, \Sigma, V, env, L, M, \mathcal{R}, \mathcal{E}, i_{in}, i_0),$$

where

- $\Gamma = \Sigma \cup \{A_i, B_i : 1 \leq i \leq n\}$
  $\cup \{G_i : 1 \leq i \leq n + \lceil \log(k+1) \rceil - 2\}$
  $\cup \{a_i : 1 \leq i \leq 2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 9\}$
  $\cup \{\bar{c}_i : 1 \leq i \leq n + \lceil \log(k+1) \rceil - 1\}$
  $\cup \{c_i : 1 \leq i \leq n+1\}$
  $\cup \{d_i : 1 \leq i \leq \lceil \log n \rceil + \lceil \log(k+1) \rceil + 3\}$
  $\cup \{e_i : 1 \leq i \leq \lceil \log n \rceil + 1\}$
  $\cup \{B_{ij} : 1 \leq i \leq n \wedge 1 \leq j \leq \lceil log(k+1) \rceil + 1\}$
  $\cup \{\alpha, b, D, p, q, g_1, g_2, f_1, T, S, N, \texttt{yes}, \texttt{no}\}$
- $V = \{\sigma_1, \sigma_2, \sigma_3\} \cup \{env\}$
- $\Sigma = \{v_i : 1 \leq i \leq n\} \cup \{q\}$
- $L(\sigma_1) = 1, L(\sigma_2) = 2, L(\sigma_3) = 3, L(env) = 0$
- $M(\sigma_1) = a_1\, b\, \bar{c}_1\, \texttt{yes}\, \texttt{no}$
- $M(\sigma_2) = D\, A_1 \ldots A_n$

- $M(\sigma_3) = \{\{G_1 \dots G_{n+\lceil \log(k+1)\rceil -2}\, c_2^2 \dots c_{n+1}^2\, e_1\, e_2^2 \dots e_{\lceil \log n\rceil +1}^2$
  $d_1 \dots d_{\lceil \log n\rceil +\lceil \log(k+1)\rceil +3}\, p^k\, T\, S\, N\, g_1\, g_2\, f_1\}\}$
  $\cup\ \{\{B_{i1}\ 1 \le i \le n\}\}$
  $\cup\ \{\{B_{ij}^2\ 1 \le i \le n \wedge 2 \le j \le \lceil log(k+1)\rceil + 1\}\}$
- $M(env) = a_2\ \dots a_{2n+\lceil \log n\rceil +2\lceil \log(k+1)\rceil +9}\, \bar{c}_2\ \dots \bar{c}_{n+\lceil \log(k+1)\rceil -1}\, c_1$
- $\mathcal{R}$ is the following set of rules:
  1. *Division rules:*
     $r_{1,i} \equiv [G_i]_3 \rightarrow [\alpha]_3[\alpha]_3$ for $i = 1, \dots, n + \lceil \log(k+1)\rceil - 2$
     $r_{2,i} \equiv [A_i]_2 \rightarrow [B_i]_2[\alpha]_2$ for $i = 1, \dots, n$
  2. *Communication rules:*
     $r_{3,i} \equiv (1, a_i/a_{i+1}, 0)$ for $i = 1, \dots, 2n + \lceil \log n\rceil + 2\lceil \log(k+1)\rceil + 8$
     $r_{4,i} \equiv (1, \bar{c}_i/\bar{c}_{i+1}, 0)$ for $i = 1, \dots, n + \lceil \log(k+1)\rceil - 2$
     $r_5 \equiv (1, \bar{c}_{n+\lceil \log(k+1)\rceil -1}/c_1, 0)$
     $r_{6,i} \equiv (1, c_i/c_{i+1}^2, 3)$ for $i = 1, \dots, n$
     $r_7 \equiv (1, c_{n+1}/D, 2)$
     $r_8 \equiv (2, c_{n+1}/d_1e_1, 3)$
     $r_{9,i} \equiv (2, e_i/e_{i+1}^2, 3)$ for $i = 1, \dots, \lceil \log n\rceil$
     $r_{10,i} \equiv (2, d_i/d_{i+1}, 3)$ for $i = 1, \dots, \lceil \log n\rceil + \lceil \log(k+1)\rceil + 2$
     $r_{11,i} \equiv (2, e_{\lceil \log n\rceil +1}B_i/B_{i1}, 3)$ for $i = 1, \dots, n$
     $r_{12,i,j} \equiv (2, B_{ij}/B_{ij+1}^2, 3)$ for $i = 1, \dots, n$, $j = 1, \dots, \lceil \log(k+1)\rceil$
     $r_{13,i} \equiv (2, B_{i\lceil \log(k+1)\rceil +1}v_i/p, 3)$ for $i = 1, \dots, n$
     $r_{14} \equiv (2, pq/\lambda, 0)$
     $r_{15} \equiv (2, d_{\lceil \log n\rceil +\lceil \log(k+1)\rceil +3}/g_1f_1, 3)$
     $r_{16} \equiv (2, f_1p/\lambda, 0)$
     $r_{17} \equiv (2, f_1q/\lambda, 0)$
     $r_{18} \equiv (2, g_1/g_2, 3)$
     $r_{19} \equiv (2, g_2f_1/T, 3)$
     $r_{20} \equiv (2, T/\lambda, 1)$
     $r_{21} \equiv (1, bT/S, 3)$
     $r_{22} \equiv (1, S\texttt{yes}/\lambda, 0)$
     $r_{23} \equiv (1, a_{2n+\lceil \log n\rceil +2\lceil \log(k+1)\rceil +9}b/N, 3)$
     $r_{24} \equiv (1, N\texttt{no}/\lambda, 0)$

- $i_{in} = 2$, is the input cell
- $i_0 = 0$, is the output cell

4.1 An Overview of the Computation

First of all, we recall the polynomial encoding of the Subset Sum problem in the family Π constructed in the previous section. Let $u = (n, (w_1, \dots, w_n), k)$ be an instance of the problem, $s(u) =< n, k >$ and $cod(u) = \{\{v_i^j\ :\ w(A_i) = j \wedge 1 \le i \le n\}\} \cup \{\{q^k\}\}$.

Next, we describe informally how the recognizer tissue P system with cell division $\Pi(s(u))$ with input $cod(u)$ works. Let us start with the *generation stage.*

Recall that if a division rule is triggered, the communication rules cannot be simultaneously applied. In this stage we have three parallel processes:

- The first one occurs in the region labeled by 1, where we have two counters: a_i, which will be used in the answer stage, and $\bar{c}_i$, which will be used to delay the start of the communication rules.
- The second one occurs in the region labeled by 2, where the second group of division rules are applied. For each object A_i (which codifies a member of the set V) we obtain two cells labeled by 2: One of them has an element B_i and the other one has an object α. Such object will not be used any more in the computation.
- The third one occurs in the cell labeled by 3, where the first group of division rules are applied for $n + \lceil \log(k+1) \rceil - 2$ steps. For each object G_i, we obtain two cells labeled by 3: both of them have an object α.

When all divisions have been done, we will have 2^n cells with label 2, in which each one of them will contain the encoding of a subset of V and $2^{n+\lceil \log(k+1) \rceil - 2}$ cells with label 3.Then $\bar{c}_{n+\lceil \log(k+1) \rceil}$ is replaced by c_1 in cell 1. After this step, c_i will be multiplied until getting exactly 2^n copies in n steps. At this moment, the generation stage ends and the pre-checking stage begins.

For each cell 2, an object D is traded against a copy of the counter c_i. In this way, 2^n copies of D will appear in the region 1 and, in each cell labeled by 2 there will be an object c_{n+1}. The occurrence of such object c_{n+1} in the cells 2 will bring two counters:

(a) The counter d_i lets the checking stage start, since it produces the occurrence of the objects g_1 and f_1 in cells 2.
(b) The counter e_i will be multiplied for obtaining n copies of $e_{\lceil \log n \rceil + 1}$ in the step $n + \lceil \log n \rceil + 5$ from cell 3. Then, we trade objects $e_{\lceil \log n \rceil + 1}$ and B_i against B_{i1} for each element A_i in the subset codifying a possible solution associated with the membrane. After that, for each $1 \leq i \leq n$ we get $k+1$ copies of $B_{i\lceil \log(k+1) \rceil + 1}$ from cell 3. Then for each element A_i, we get w_i copies of object p, in the step $2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 3$.

The checking takes place in the step $2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 4$, when all pairs of objects p and q from any cell labeled by 2 are sent to the environment. In this way, if the weight of the subset associated with a cell is equal to k, then no object p or q remains in this cell in the next step. Otherwise, if the encoding is not exactly of weight k, then at least one object p or q will remain in the cell. In the next step the answer stage starts. Two cases must be considered for each cell:

- If no object p or q remains in the cell, the object f_1 keeps in the cell, g_1 evolves to g_2, and in the step $2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 6$ the objects f_1 and g_2 are traded against T from the cell three. In the next step T is sent to the cell 1, and in the step $2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 8$, the objects T and b are sent to the cell labeled by 3 traded against S. Finally in the step $2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 9$ the objects S and `yes` are sent to the environment.

- If any object p or q remains in the cell, such object is sent to the environment together with the object f_1. This causes that the object b still remains in the cell 1 after the step $2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 8$. In this way, the objects b and $a_{2n+\lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 9}$ are traded against the object N with the cell labeled 3, and in the step $2n + \lceil \log n \rceil + 2\lceil \log(k+1) \rceil + 10$ the objects N and `no` are sent to the environment.

4.2 Some Technical Considerations

In order to establish that the family $\mathbf{\Pi}$ is polynomially uniform by deterministic Turing machines we firstly note that the sets of rules associated with the systems $\Pi(n,k)$ are recursively defined. Hence, it suffices to justify that the amount of necessary resources for defining the systems is polynomial in $\max\{n, \lceil \log k \rceil\}$.

- Size of the alphabet: $n \cdot \lceil \log(k+1) \rceil + 8n + 5\lceil \log(k+1) \rceil + 3\lceil \log n \rceil + 30 \in O(n \cdot \log k)$
- Initial number of cells: $4 \in \theta(1)$.
- Initial number of objects: $5n + 3\lceil \log n \rceil + 3\lceil \log(k+1) \rceil + 31 \in \theta(n)$.
- Number of rules: $n \cdot \lceil \log(k+1) \rceil + 6n + 3\lceil \log(k+1) \rceil + 3\lceil \log n \rceil + 27 \in O(n \cdot \log k)$
- Maximal length of a rule: 3.

So, we can claim the following result.

Theorem 1. $SS \in \mathbf{PMC}_{TD-E}$

Taking into account that `SS` is an **NP**–complete problem, and that the class $\mathbf{PMC}_{TD-E}$ is closed under complement, the following is deduced.

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

5 Conclusions and Future Work

The search of biologically inspired frontiers for tractability has been an active research area in the last years. Since the problem **P** vs. **NP** is still open and it seems that will remain open for a long time, the research faces the problem of finding new frontiers between these classes of problems. Current research on complexity in Membrane Computing focuses on looking for the minimum amount of ingredients of a P system model able to solve a **NP**-complete problem.

One of these steps was the discovery of the role of the dissolution rules (a rule apparently innocent) as the key stone for solving **NP**-complete problems in the framework of P systems with active membranes [10, 9].

In this paper we give a new step in the same direction. We have prove that the use of an arbitrarily large amount of objects in the environment can be removed from tissue-like P systems with cell division in order to solve **NP**-complete problems. The next steps in this research area will try to reduce the initial ingredients in order to make the frontier of tractability thinner and thinner.

Acknowledgement

The first author acknowledge the support of the project "Computational Topology and Applied Mathematics" PAICYT research project FQM-296. The second,third and four author acknowledge the support of the projects TIN2008-04487-E and TIN-2009-13192 of the Ministerio de Ciencia e Innovación of Spain and the support of the Project of Excellence with *Investigador de Reconocida Valía* of the Junta de Andalucía, grant P08-TIC-04200.

References

1. M.J. Pérez–Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini, F. A polynomial complexity class in P systems using membrane division. In E. Csuhaj-Varjú, C. Kintala, D. Wotschke and Gy. Vaszyl (eds.) *Proceedings of DCFS 2003*, (2003), 284–294.
2. Alhazov, A., Freund, R. and Oswald, M. Tissue P Systems with Antiport Rules ans Small Numbers of Symbols and Cells. *Lecture Notes in Computer Science* **3572**, (2005), 100–111.
3. Bernardini, F. and Gheorghe, M. Cell Communication in Tissue P Systems and Cell Division in Population P Systems. *Soft Computing*, **9**(9), (2005), 640–649.
4. Díaz-Pernil, D. *Sistemas P de Tejido: Formalización y Eficiencia Computacional.* PhD Thesis, University of Seville, (2008).
5. Díaz-Pernil, D., Gutiérrez, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A. Solving Subset Sum in linear time by using tissue P systems with cell division. *Lecture Notes in Computer Science*, **4527**, (2007), 170-179.
6. Díaz-Pernil, D., Gutiérrez, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A. A uniform family of tissue P systems with cell division solving 3-COL in a linear time. *Theoretical Computer Science*, **404**, (1-2), (2008), 76-87.
7. Freund, R., Păun, Gh. and Pérez-Jiménez, M.J. Tissue P Systems with channel states. *Theoretical Computer Science*, **330**, (2005), 101–116.
8. Gutiérrez-Escudero, R., Pérez-Jiménez, M.J., M. Rius-Font. Characterizing tractability by tissue-like P systems. Lecture Notes in Computer Science, **5957**, (2010), 289-300.
9. Gutiérrez, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A., Romero-Campero, F.J. *Lecture Notes in Computer Science*, **3850**, (2006), 224-240.
10. Gutiérrez, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A., Romero-Campero, F.J. Computational efficiency of dissolution rules in membrane systems. *International Journal of Computer Mathematics*, **83**, (7), (2006) 593 - 611.
11. Krishna, S.N., Lakshmanan K. and Rama, R. Tissue P Systems with Contextual and Rewriting Rules. *Lecture Notes in Computer Science* **2597**, (2003), 339–351.
12. Lakshmanan K. and Rama, R. On the Power of Tissue P Systems with Insertion and Deletion Rules. In A. Alhazov, C. Martín-Vide and Gh. Păun (eds.) *Pre-proceedings of the Workshop on Membrane Computing*, (2003), 304-318.
13. Martín Vide, C. Pazos, J. Păun, Gh. and Rodríguez Patón, A. A New Class of Symbolic Abstract Neural Nets: Tissue P Systems. *Lecture Notes in Computer Science* **2387**, (2002), 290–299.
14. Martín Vide, C. Pazos, J. Păun, Gh. and Rodríguez Patón, A. Tissue P systems. *Theoretical Computer Science*, **296**, (2003), 295–326.

15. Păun, G. *Membrane Computing. An Introduction.* Springer–Verlag, Berlin, (2002).
16. Păun, G., Pérez-Jiménez, M.J. and Riscos-Núñez, A. 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*, (2004), 380–386.
17. Păun, G. Rozenberg, A. Salomaa, eds: Handbook of Membrane Computing. Oxford University Press, (2009).
18. Prakash, V.J. On the Power of Tissue P Systems Working in the Maximal-One Mode. In A. Alhazov, C. Martín-Vide and Gh. Păun (eds.). *Pre-proceedings of the Workshop on Membrane Computing*, (2003), 356-364.
19. P systems web page `http://ppage.psystems.eu/`