# Integer Linear Programming for Tissue-like P Systems

Raúl Reina-Molina[1], Daniel Díaz-Pernil[1], Miguel A. Gutiérrez-Naranjo[2]

[1]Research Group on Computational Topology and Applied Mathematics
Department of Applied Mathematics
University of Sevilla
raureimol@alum.us.es, sbdani@us.es

[2]Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla
magutier@us.es

**Summary.** In this paper we report a work-in-progress whose final target is the implementation of tissue-like P system in a cluster of computers which solves some instances of the segmentation problem in 2D Digital Imagery. We focus on the theoretical aspects and the problem of choosing a maximal number of application of rules by using Integer Linear Programming techniques. This study is on the basis of a future distribution of the parallel work among the processors.

## 1 Introduction

Membrane systems[1] are distributed and parallel computing devices processing multisets of objects in compartments delimited by membranes. Computation is carried out by applying given rules to every membrane content, usually in a maximal non-deterministic way, although other semantics are being explored.

In spite of some recent efforts (see [6]), there are neither *in vivo*, *in vitro* nor *in silico* implementations of such devices and the unique way to get a mechanical application of the rules is by the development of software tools capable of performing simulations of such devices on current computers [4, 7].

In this paper we report a work-in-progress about the implementation of tissue-like P system in a cluster of computers. This is not the first attempt. In 2003, Ciobanu and Wenyuan presented in [3], a parallel implementation of transition P systems. The program was designed for a cluster of 64 dual processor nodes and

[1] We refer to [13] for basic information in this area, to [14] for a comprehensive presentation and the web site [15] for the up-to-date information.

it was implemented and tested on a Linux cluster at the National University of Singapore.

We will focus on the problem of finding a maximal amount of applications of rules from a given configuration in the framework of tissue-like P systems. The contribution of this paper is the use of a matrix representation for configuratios and rules and the use of Integer Linear Programming.

In this paper we will consider a matrix representation of tissue-like P systems. This new representation will be useful for considering Integer Linear Programming for automatically searching a maximal set of rules.

We also start using Operation Research techniques for calculating the rules to be applied in each computing step.

A linear program, LP for short, is an Operation Research problem consisting in optimizing a linear function subject to linear restrictions. Without lost of generality we may assume that a LP is a problem like the following:

$$(LP)\begin{cases} \text{maximize: } \displaystyle\sum_{k=1}^{n} c_k x_k \\ \text{subject to:} \\ \qquad \displaystyle\sum_{k=1}^{n} a_{1k} x_k \leq b_1 \\ \qquad \cdots \\ \qquad \displaystyle\sum_{k=1}^{n} a_{mk} x_k \leq b_m \end{cases}$$

When the additional constraint of integrality of $x_k$, the LP is called Integer Linear Program, ILP for short.

The paper is organized as follows: First we briefly recall some basic definitions related to multisets and tissue-like P systems. Next, we show a theoretical study on how the tissue-like P systems can adopt a matrix representation. We show that this representation can be useful for using Integer Linear Programming for finding a maximal set of applications which will be used for a future distribution of the work among different processors. We illustrates this definition with an explicative example. Finally, some clues for the future work are presented.

## 2 Preliminaries

An *alphabet*, $\Sigma$, 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 $\lambda$. The set of strings of length $n$ built with symbols from the alphabet $\Sigma$ is denoted by $\Sigma^n$ and $\Sigma^* = \cup_{n\geq 0}\Sigma^n$. A *language* over $\Sigma$ is a subset from $\Sigma^*$.

A *multiset over a set* $A$ is a pair $(A, f)$ where $f : A \to \mathbb{N}$ is a mapping. If $m = (A, f)$ is a multiset then its *support* is defined as $supp(m) = \{x \in A \mid f(x) > 0\}$ 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 $m = a_1^{f(a_1)} a_2^{f(a_2)} \cdots a_k^{f(a_k)}$ or $\{\{a_j^{f(a_j)} ; 1 \leq j \leq k\}\}$ where $supp(m) = \{a_1, \ldots, a_k\}$, and for each element $a_i$, $f(a_i)$ is called the multiplicity of $a_i$. furthermore the multiplicity of an element $a_i \in m$ is denoted as $\text{mult}(a_i, m)$. In what follows we assume the reader is already familiar with the basic notions and the terminology underlying P systems. For details, see [14].

In the initial definition of the cell-like model of P systems [12], membranes are hierarchically arranged in a tree–like structure. Its biological inspiration comes from the morphology of cells, where small vesicles are surrounded by larger ones. This biological structure can be abstracted into a tree–like graph, where the root represents the skin of the cell (i.e., the outermost membrane) and the leaves represent membranes that do not contain any other membrane.

In *tissue P systems*, the tree-like membrane structure is replaced by a general graph. This model has two biological inspirations (see [9, 10]): 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. The communication among cells is based on symport/antiport rules. In symport rules, objects cooperate to traverse a membrane together in the same direction, whereas in the case of antiport rules, objects residing at both sides of the membrane cross it simultaneously but in opposite directions.

Formally, a *tissue-like P system* of degree $q \geq 1$ with input is a tuple of the form

$$\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, \ldots, w_q, \mathcal{R}, i_\Pi, o_\Pi)$$

where

1. $\Gamma$ is a finite *alphabet*, whose symbols will be called *objects*;
2. $\Sigma (\subset \Gamma)$ is the input alphabet;
3. $\mathcal{E} \subseteq \Gamma$ (the objects in the environment);
4. $w_1, \ldots, w_q$ are strings over $\Gamma$ representing the multisets of objects associated with the cells at the initial configuration;
5. $\mathcal{R}$ is a finite set of communication rules of the following form: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \ldots, q\}, i \neq j$, $u, v \in \Gamma^*$;
6. $i_\Pi \in \{1, 2, \ldots, q\}$ is the input cell;
7. $o_\Pi \in \{0, 1, 2, \ldots, q\}$ is the output cell.

A tissue-like P system of degree $q \geq 1$ can be seen as a set of $q$ cells labeled by $1, 2, \ldots, q$. We will use 0 to refer to the label of the environment, $i_\Pi$ and $o_\Pi$ denote the input region and the output region (which can be the region inside a cell or the environment) respectively.

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

The communication rule $(i, u/v, j)$ can be applied over two cells labeled by $i$ 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. Note that if either $i = 0$ or $j = 0$ then the objects are interchanged between a cell and the environment.

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 than can participate in a rule of any form must do it, i.e., at each step a maximal set of rules is applied.

A *configuration* is an instantenous description of the system $\Pi$, and it is represented as a tuple $\langle w_0, w_1, \ldots, w_q \rangle$. Given a configuration, we can perform a computational 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. The output of a computation is collected from its halting configuration by reading the objects contained in the output cell.

## 3 Encoding Tissue-like P Systems by Using Matrices

In this section we define the formal framework for a new way of calculating maximal set of rules to be applied in tissue-like P systems. First of all let us suppose that we have the alphabet indexed, so $\Gamma = \{\gamma_j : 1 \leq j \leq |\Gamma|\}$. In the same sense let $\mathcal{R} = \{r_k : 1 \leq k \leq |\mathcal{R}|\}$ be the set of communication rules. By using the order in $\Gamma$ settled by the indexation, we can consider the *vector representation* [1], $\underline{u} \in \mathbb{N}^{|\Gamma|}$ of the multiset $u$ as the $|\Gamma|$-dimensional vector $\underline{u}$ with $\underline{u}_j = \text{mult}(\gamma_j, u)$ for each $j = 1, 2, \ldots, |\Gamma|$. Moreover, for technical reasons, we will extend this vectorial representation to the environment by including the symbol $\infty$ for the objects with an arbitrary amount of copies. In this way, we will consider a vector $\underline{u}$ with coordinates in $\mathbb{N}_\infty = \mathbb{N} \cup \{\infty\}$ with

$$\underline{u}_j = \begin{cases} \text{mult}(\gamma_j, u) \text{ if } \gamma_j \in \mathcal{E} \\ \infty \text{ if } \gamma_j \in \mathcal{E} \end{cases}$$

We can extend elementary operations in $\mathbb{N}$ to $\mathbb{N}_\infty$ with

$$\infty \pm n = \infty, \forall n \in \mathbb{N}$$
$$\infty \cdot n = \infty, \forall n \in \mathbb{N}, n \neq 0$$

By using this extension to $\mathbb{N}_\infty$, we can use a vector representation for the multisets inside the cell as as well as the multiset in the environment in each configuration.

The *configuration matrix* is the $(q+1) \times |\Gamma|$ matrix of non-negative integers whose $i$-th row is the vector representation of the multiset $w_i$. Let us recall that the rows is indexed from 0, to take in count the multiset for environment.

In the following, we do not lose generality if we consider the communication rule $(i, u/v, j)$ written with $i < j$.

Let $r = (i_u, u/v, i_v)$ be a communication rule interchanging the elements in $u$ with the elements in $v$. From this characterizaction we define two matrices:

$$M_r^- = \begin{pmatrix} \vdots \\ i_u : \underline{u} \\ \vdots \\ i_v : \underline{v} \\ \vdots \end{pmatrix}, M_r^+ = \begin{pmatrix} \vdots \\ i_u : \underline{v} \\ \vdots \\ i_v : \underline{u} \\ \vdots \end{pmatrix} \tag{1}$$

for $0 \leq i \leq q, 1 \leq j \leq |\Gamma|$, where $M_r^-$ has all the rows $\underline{0} \in \mathbb{N}^{|\Gamma|}$ except the $i_u$-th and $i_v$-th, which are, respectively, $\underline{u}$ and $\underline{v}$, and so on. Both matrices defined above makes the *matrix representation* for rule $r$, $M(r) = \langle M_r^-, M_r^+ \rangle$.

For example, let $\Gamma = \{a, b, c, d\}$ be an alphabet with environment $\mathcal{E} = \{c, d\}$. The multiset $u = \{\{a^2, c, d^3\}\}$ is encoded by $\underline{u} = (2, 0, 1, 3)$. The rule $r = (1, ab^2/c, 0)$ in a tissue-like P system with three cells (and the environment) is encoded by

$$M_r^- = \begin{pmatrix} 0 : (0,0,1,0) \\ 1 : (1,2,0,0) \\ 2 : (0,0,0,0) \\ 3 : (0,0,0,0) \end{pmatrix}, M_r^+ = \begin{pmatrix} 0 : (1,2,0,0) \\ 1 : (0,0,1,0) \\ 2 : (0,0,0,0) \\ 3 : (0,0,0,0) \end{pmatrix}$$

If we have a configuration matrix given by, for example,

$$M = \begin{pmatrix} 0 : (0,0,\infty,\infty) \\ 1 : (3,2,0,0) \\ 2 : (0,0,1,0) \\ 3 : (0,1,0,3) \end{pmatrix}$$

then the application of rule $r$ gives the configuration matrix given by

$$M' = M + M_r^+ - M_r^- =$$

$$\begin{pmatrix} 0 : (0,0,\infty,\infty) \\ 1 : (3,2,0,0) \\ 2 : (0,0,1,0) \\ 3 : (0,1,0,3) \end{pmatrix} + \begin{pmatrix} 0 : (1,2,0,0) \\ 1 : (0,0,1,0) \\ 2 : (0,0,0,0) \\ 3 : (0,0,0,0) \end{pmatrix} - \begin{pmatrix} 0 : (0,0,1,0) \\ 1 : (1,2,0,0) \\ 2 : (0,0,0,0) \\ 3 : (0,0,0,0) \end{pmatrix} =$$

$$\begin{pmatrix} 0 : (1,2,\infty,\infty) \\ 1 : (2,0,1,0) \\ 2 : (0,0,1,0) \\ 3 : (0,1,0,3) \end{pmatrix}$$

If a computation step consists on applying the rule $r_k$ for $m_k$ times, $1 \leq k \leq |\mathcal{R}|$, $M$ is the configuration matrix for a given configuration of the system and $M_k^-, M_k^+$ are the matrices defined in Equation 1 for rule $r_k$ then

$$M + \sum_{k=1}^{|\mathcal{R}|} m_k (M_k^+ - M_k^-)$$

is the configuration matrix for the configuration after the computation step. Hence, if $\mathbb{M}$ is a multiset of rules, the result of applying all the rules in $\mathbb{M}$ to the configuration given by $M$, is the configuration obtained from matrix

$$M' = M + \mathbb{M} = M + \sum_{r \in \mathbb{M}} \text{mult}(r, \mathbb{M}) M_r$$

where $M_r = M_r^- + M_r^+$

A rule $r$ can be applied if there are enough objects in each cell to be communicated. Thus, if $M$ is a configuration matrix and $\langle M_r^-, M_r^+ \rangle$ is the matrix representation of the communication rule $r$, it is clear that the rule can be applied if and only if $M + M(r)^- \geq \mathbf{0}$, where $\mathbf{0}$ is the null matrix and $\geq$ is considered elementwise. Thus, if $\mathbb{M}$ is a multiset of rules, then

$$\mathbb{M} \text{ can be applied } \Leftrightarrow M + \sum_{r \in \mathbb{M}} \text{mult}(r, \mathbb{M}) M_r^- \geq \mathbf{0} \tag{2}$$

We will consider the following definition of maximality. A maximal multiset of rules is a multiset $\mathbb{M}$ of communication rules such that no other applicable rule can be added. Hence we have

$$\mathbb{M} \text{ is maximal } \Leftrightarrow \forall r \in \mathcal{R} \setminus \mathbb{M}, M + \mathbb{M}^- + M_r^- \ngeq 0 \tag{3}$$

where $\mathbb{M}^-$ is the multiset $\{\{M_r^{+ m_r} : r \in \mathbb{M} \wedge m_r = \text{mult}(r, \mathbb{M})\}\}$.

Given a configuration matrix $M$, finding a maximal set of rules is equivalent to finding non-negative integers, $m_k, 1 \leq k \leq |\mathcal{R}|$, such that the multiset $\{\{r_k^{m_k} : r_k \in \mathcal{R} \wedge m_k > 0\}\}$ is applicable and it cannot be extended by another applicable rule. In membrane computer literature there are many approaches to the way maximality is defined. As we are trying to apply membrane computing techniques to silicon computers, we are interested in a definition of maximality that fits well to this kind of devices. Therefore, we choose maximality in the sense of number of rules applied.

In this way, we find one of the possible maximal sets of rules from all of the available. For example, given $\Gamma = \{a, b\}$, the rules $r_1 = (1, a/b, 2)$ and $r_2 = (1, a/b^2)$ with multisets $w_1 = \{\{a^2\}\}$ and $w_2 = \{\{b^2\}\}$, the multisets $\mathbb{M}_1 = \{\{r_1^2\}\}$ and $\mathbb{M}_2 = \{\{r_2\}\}$ are both maximals. However, we prefer $\mathbb{M}_1$ because of the higher number of rule applications.

Hence our problem can be reduced to find non-negative integers $m_k$ (thought to be multiplicities of rules) such that they maximize the sum $\sum_k m_k$ (which is

the total number of rule applications) subject to applicability condition. Therefore, this problem can be exposed as the following Integer Linear Programming problem

$$\begin{cases} \textbf{maximize:} \sum_{k=1}^{|\mathcal{R}|} m_k \\ \textbf{subject to:} \\ \sum_{k=1}^{|\mathcal{R}|} |M_k^-[i,j]| m_k \leq M[i,j], \text{ for } 0 \leq i \leq q, 1 \leq j \leq |\Gamma| \end{cases} \tag{4}$$

Although the Integer Linear Programming (ILP) problem in Equation 4 can be solved in parallel, it is a well known that it is a **NP** problem (with respect to the number of decision variables) [5, 8]. However, it can be solved with a reasonable speed in ordinary computers when the number of variables is relatively small. Hence it is important to decrease the number of decision variables. In order to do that we will divide the entire ILP problem in Equation 4 is several smaller problems. However, we must bear in mind the dependence between communication rules. Hence, we say that two communication rules $r = (i_u, u/v, i_v)$ and $r' = (i'_u, u'/v', i'_v)$ are *dependent* if they share some cells and, in the common ones, the multisets being communicated have non empty intersection. It is equivalent to the following condition

$$\begin{array}{c} i_u = i'_u \wedge u \cap u' \neq \emptyset \\ \vee \\ i_v = i'_v \wedge v \cap v' \neq \emptyset \\ \vee \\ i_u = i'_v \wedge u \cap v' \neq \emptyset \\ \vee \\ i_v = i'_u \wedge v \cap u' \neq \emptyset \end{array} \tag{5}$$

We can define a partition of the set of communication rules, $\mathcal{R}$, in several sets such that every two rules in distinct sets are independent. More formally, let $\sim$ be the relation in $\mathcal{R}$ given by $r \sim r'$ if and only if $r$ and $r'$ are dependent rules. Clearly $\sim$ is reflexive and symmetric. Let $\simeq$ be the transitive closure of $\sim$. With the definitions above, $\simeq$ is an equivalence relation and the quotient set $\mathcal{R}/\simeq$ defines a partition in the set of rules such that each rule in any set is independent with any rule in other partition set.

For each partition set defined above, we can define an ILP problem for rule selection as in Equation 4. Hence, we have a solution of the problem for the whole set of rules by considering the partial solution of each subproblem. This technique, together with an appropriate design of the rules, ensures an upper bound on the number of decision variables for each partial ILP making them available to be solved in reasonable time.

## 4 Example

In the following section, we will illustrate the techniques shown in section 3 (Encoding tissue-like P Systems using matrices) by the application of them to an example.

Let consider the following tissue-like P System with two cells

$$\Pi = (\Gamma, \Sigma, \mathcal{E}, w_1, w_2, \mathcal{R}, i_\Pi, o_\Pi)$$

where

1. $\Gamma = \{a, b, c, d\}$ is the alphabet of objects;
2. $\Sigma = \emptyset$ is the input alphabet;
3. $\mathcal{E} = \{a, c, d\}$ represents the objects in the environment;
4. $w_1 = \{\{a^{10}, b^5, d\}\}$, $w_2 = \{\{a^4, c^7\}\}$ are strings over $\Gamma$ representing the multisets of objects associated with the cells at the initial configuration;
5. $\mathcal{R}$ is the finite set of communication rules below:
   a) $r_1 = (0, c/ab, 1)$
   b) $r_2 = (0, c^5/a, 2)$
   c) $r_3 = (0, c^2/a^2b^3, 1)$
   d) $r_4 = (1, d/c^3, 2)$
   e) $r_5 = (0, a^2d^3/d, 1)$
6. $i_\Pi = 1$ is the input cell;
7. $o_\Pi = 2$ is the output cell.

Let $M^k$ denote the configuration matrix of the $k$-th configuration of $\Pi$. Trivially,

$$M^0 = \begin{pmatrix} \infty & 0 & \infty & \infty \\ 10 & 5 & 0 & 1 \\ 4 & 0 & 7 & 0 \end{pmatrix}$$

If $\langle M_k^-, M_k^+ \rangle$ denotes the matricial form for rule $r_k$, for $k = 1, 2, 3, 4, 5$, then

$$M_1^- = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} M_1^+ = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} M_2^- = \begin{pmatrix} 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} M_2^+ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 5 & 0 \end{pmatrix}$$

$$M_3^- = \begin{pmatrix} 0 & 0 & 2 & 0 \\ 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} M_3^+ = \begin{pmatrix} 2 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} M_4^- = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 3 & 0 \end{pmatrix} M_4^+ = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 5 & 1 \end{pmatrix}$$

$$M_5^- = \begin{pmatrix} 2 & 0 & 0 & 3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} M_5^+ = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Rule dependency must be studied before ILP problem definition. Hence, the rule–dependency equivalence relation $\simeq$ decomposes $\mathcal{R}$ in two sets, being $[r_1] = \{r_1, r_2, r_3\}$ and $[r_4] = \{r_4, r_5\}$. Therefore, two ILP must be solved in order to find

a maximal set of rules to be applied for going from configuration $i$ to $i+1$. These ILP will be respectively denoted as $ILP^{(1)}_{i\to\imath+1}$ and $ILP^{(4)}_{i\to\imath+1}$, and are represented below.

$$ILP^{(1)}_{i\to i+1}\begin{cases}\textbf{maximize:} m_1+m_2+m_3\\ \textbf{subject to:}\\ m_1+2m_3\leq \text{mult}(a,w_1)\\ m_1+3m_3\leq \text{mult}(b,w_1)\\ m_2\leq \text{mult}(a,w_2)\end{cases} \tag{6}$$

$$ILP^{(4)}_{i\to i+1}\begin{cases}\textbf{maximize:} m_4+m_5\\ \textbf{subject to:}\\ m_4+m_5\leq \text{mult}(d,w_1)\\ m_4\leq \text{mult}(c,w_2)\end{cases} \tag{7}$$

We firstly solve $ILP^{(1)}_{0\to 1}$, obtaining the solution $m_2=1$, $m_2=4$ and $m_3=1$. The problem $ILP^{(4)}_{0\to 1}$ has two maximal solutions, being $m_4=0, m_5=1$ and $m_4=1, m_5=0$. In such cases, we will choose one non deterministically. For example, let $m_4=0, m_5=1$ be the solution selected.

Both partial solutions make a maximal multiset $\mathbb{M}_{0\to 1}=\{\{r_1^2, r_2^4, r_3, r_5\}\}$, whose application arises the configuration given by matrix

$$M^1=\begin{pmatrix}\infty & 5 & \infty & \infty\\ 8 & 0 & 4 & 3\\ 0 & 0 & 35 & 0\end{pmatrix}$$

Analogously, at this step $ILP^{(1)}_{1\to 2}$ has only the solution $m_1=m_2=m_3=0$, while $ILP^{(4)}_{1\to 2}$ has multiple solutions given in the set $\{(0,3),(1,2),(2,1),(3,0)\}$, where the first one is $m_4$ and the last one is $m_5$. Again, one solution is non deterministically choosen, for example, $m_4=2, m_5=1$. Hence, the multiset of rules to be applied is $\mathbb{M}_{1\to 2}=\{\{r_4^2, r_5\}\}$, and the new configuration is given by matrix

$$M^2=\begin{pmatrix}\infty & 5 & \infty & \infty\\ 10 & 0 & 10 & 3\\ 0 & 0 & 29 & 2\end{pmatrix}$$

Next computation step involves solving $ILP^{(1)}_{2\to 3}$ and $ILP^{(4)}_{2\to 3}$. These Integer Linear Programs have the same solutions as previous ones. Again, in the second ILP one only solution must be non deterministically choosen. Let $m_4=2, m_5=1$ be that solution. Application of the multiset of rules $\mathbb{M}_{2\to 3}=\{\{r_4^2, r_5\}\}$ gives next configuration, settled as

$$M^3=\begin{pmatrix}\infty & 5 & \infty & \infty\\ 12 & 0 & 16 & 3\\ 0 & 0 & 23 & 4\end{pmatrix}$$

Next computation step is similar to the previous one and a solution from $\{(3,0),(2,1),(1,2),(0,3)\}$ must be non determistically choosen. Let it be $m_4 = 3, m_5 = 0$, for example. Therefore, the configuration matrix obtained is

$$M^4 = \begin{pmatrix} \infty & 5 & \infty & \infty \\ 12 & 0 & 25 & 0 \\ 0 & 0 & 14 & 7 \end{pmatrix}$$

On the next step, the only solution is the trivial one, settled as $m_k = 0, k = 1,2,3,4,5$, which is interpreted as halting condition.

## 5 Final Remarks

Parallelism is on the basis of Membrane Computing. All the theoretical efforts for designing membrane computing algorithms which use parallelism in an efficient way has the same problem in realistic simulations. Most of the current computers are sequential the simulations performed in such computers have the same bottleneck.

In this paper we report a preliminary work focused on a distributed simulation of tissue-like P systems in a cluster of computers. In a general view, if the number of cells does not increase along the computation it seems quite natural to plan a distribution of the work among several processors. The first steps in this line have led us to consider new representations (matrix representation) and new techniques (Integer Linear Programming) to solve the problems.

Many open research lines are open form this preliminary work. One of them is to consider different notions of parallelism [2], or introducing new features to our P system model in order to make it suitable for the hardware implementation.

### Acknowledgements

DDP and MAGN 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. Busi, N., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J.: Efficient computation in rational-valued P systems. Mathematical Structures in Computer Science 19, 1125–1139 (2009), http://dx.doi.org/10.1017/S0960129509990144
2. Ciobanu, G., Marcus, S., Păun, Gh.: New strategies of using the rules of a P system in a maximal way. Power and complexity. Romanian Journal of Information Science and Technology (ROMJIST) 12, 157–173 (2009)

3. Ciobanu, G., Wenyuan, G.: P systems running on a cluster of computers. In: Martín-Vide, C., Mauri, G., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) Workshop on Membrane Computing. Lecture Notes in Computer Science, vol. 2933, pp. 123–139. Springer (2003)
4. Díaz-Pernil, D., Graciani, C., Gutiérrez-Naranjo, M.A., Pérez-Hurtado, I., Mario J. Pérez-Jiménez, M.: Software for P systems. In: Păun et al. [14], pp. 437–454.
5. Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York (1979)
6. Gershoni, R., Keinan, E., Păun, Gh., Piran, R., Ratner, T., Shoshani, S.: Research topics arising from the (planned) P systems implementation experiment in Technion. In: Díaz-Pernil, D., Graciani, C., Gutiérrez-Naranjo, M.A., Păun, Gh., Pérez-Hurtado, I., Riscos-Núñez, A. (eds.) Sixth Brainstorming Week on Membrane Computing. pp. 183–192. Fénix Editora, Sevilla, Spain (2008)
7. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Available membrane computing software. In: Ciobanu, G., Pérez-Jiménez, M.J., Păun, Gh. (eds.) Applications of Membrane Computing, pp. 411–436. Natural Computing Series, Springer (2006)
8. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W. (eds.) Complexity of Computer Computations. pp. 85 – 104. Plenum Press (1972)
9. Martín-Vide, C., Pazos, J., Păun, Gh., Rodríguez-Patón, A.: A new class of symbolic abstract neural nets: Tissue P systems. In: Ibarra, O.H., Zhang, L. (eds.) COCOON. Lecture Notes in Computer Science, vol. 2387, pp. 290–299. Springer (2002)
10. Martín-Vide, C., Păun, G., Pazos, J., Rodríguez-Patón, A.: Tissue P systems. Theoretical Computer Science 296(2), 295–326 (2003)
11. Păun, Gh.: Computing with membranes. Tech. Rep. 208, Turku Centre for Computer Science, Turku, Finland (November 1998)
12. Păun, Gh.: Computing with membranes. Journal of Computer and System Sciences 61(1), 108–143 (2000), see also [11]
13. Păun, Gh.: Membrane Computing. An Introduction. Springer-Verlag, Berlin, Germany (2002)
14. Păun, Gh., Rozenberg, G., Salomaa, A. (eds.): The Oxford Handbook of Membrane Computing. Oxford University Press (2010)
15. P system web page. http://ppage.psystems.eu