Proceedings of the International Workshop, Automata for Cellular and Molecular Computing, MTA SZTAKI, Budapest, pages 23 - 34, 2007.

Tissue Simulator: A Graphical Tool for Tissue P Systems

Rafael Borrego–Ropero, Daniel Díaz–Pernil, and Mario J. Pérez–Jiménez

Research Group on Natural Computing
Dpt. of Computer Science and Artificial Intelligence. University of Sevilla
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
{rborrego,sbdani,marper}@us.es

Abstract

Recently, different new models of tissue-like P systems have received important attention from the scientific community. This paper is focused in a concrete model: recognizing tissue P system with cell division. A software application allowing to understand better this model is presented. A linear-time solution to an **NP**-complete problem from graph theory, the 3–coloring problem is considered as a case study with this tool.

1 Introduction

Membrane Computing is an emergent branch of Natural Computing introduced by Păun in [10], which, considering as computer process the process that takes place into living cells, constructs a new non-deterministic model of computation. In membrane computing, basically, there are two types of framework: P systems with the membranes structure described by a tree, inspired from the cell, and tissue P systems with the membranes placed in the nodes of an arbitrary graph, inspired in the cellular tissue and in the neuron. The second one corresponds to the idea of forming a network of membranes linked in a specific manner and working together, [11]. In both types, the main idea is having multisets of objects placed in compartments evolving according to given rules in a synchronous non-deterministic maximally parallel manner.

It shall be focused here on tissue P systems. This variant has two biological inspirations (see [8]): intercellular communication and cooperation between cells/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 (this way of communication for P systems was introduced in [12]). Symport rules move objects across a membrane together in one direction, whereas antiport rules move objects across a membrane in opposite directions.

From the definition of tissue P systems [7, 8], several research lines have been developed and new variants have arisen (see, for example, [1, 2, 6, 17]). One of the most interesting variants of tissue P systems was presented in [13]. In that paper, the definition of tissue P systems is combined with the one of P systems with active membranes, yielding *tissue P systems with cell division*.

One of the main features of such tissue P systems with cell division is related to their computational efficiency. Several solutions for NP-complete problems have been published recently. In [13] a polynomial-time solution for the SAT problem is presented, and in [4, 5] lineal-time solutions for 3-coloring problem and Subset Sum problem, respectively, are presented. Polynomial-time solutions to NP-complete problems in Membrane Computing framework are done by trading time by space, in a theoretical way by constructing an exponential workspace in polynomial time. Although real implementations of this systems are no possible to be made, because it should be necessary to implement the maximal parallelism in some way, they are very interesting problems to treat in order to study some complexity aspects of Theoretical Computer Science, as for example $P \neq NP$ conjecture [14].

In recent years, this new field has been addressed in different ways: the study of computational properties such as computational power or complexity classes, definition of new variants of membrane systems closer to biological inspiration, working as a new framework of making biological simulations, etc. In P system web page [19] several software applications can be found. Most of them are thought in order to run an experiment which is built using some kind of membrane system as simulation framework; for example, in order to work with P system as a way of simulating biological system ([15]). In another hand, other group of software applications are thought as an easy way of visualize the computations of a P system ([9]) or spiking neural P system ([18]). Finally, in [3] a visual software application to understand the design of solutions for 3-coloring problem in the framework of recognizing tissue P system with cell division, is described.

In this paper a new visual tool called *Tissue Simulator* is presented. It is an application with a friendly graphical user interface that helps to work and understand tissue P systems with cell division.

The tool allows the user to write in an easy way the rules and the elements of a concrete tissue, run the execution of the system, and shows graphically a trace of the simulation with the rules applied in each computation step.

This paper is organized as follows: in Section 2 tissue P systems with cell division are defined and a solution of 3-coloring problem is described. Section 3 is devoted to describe *tissue simulator* application software is presented and some of the most important implementations aspects are commented. Finally, conclusions and future works are formulated.

2 Preliminaries

In this section, we briefly recall some concepts used in the paper.

An *alphabet*, Σ, is a non empty set, whose elements are called *symbols*. An ordered sequence of symbols is a *string*. The number of symbols in a string u is the *length* of the string, and it is denoted by $|u|$. As usual, the empty string (with length 0) will be denoted by λ. The set of strings of length n built with symbols from the alphabet Σ is denoted by Σ^n and $\Sigma^* = \cup_{n\geq 0}\Sigma^n$. A *language* over Σ is a subset from Σ^*.

A *multiset* m 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 $A = \{a_1, \ldots, a_n\}$, then it will be denoted as $m = a_1^f(a_1) \ldots a_n^f(a_n)$.

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

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 mitotic in a natural way.

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 mitotic 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 P system with cell division* of degree $q \geq 1$ is a tuple of the form

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

where:

1. Γ is a finite *alphabet*, whose symbols will be called *objects.*

2. $w_1, \ldots, w_q$ are strings over Γ.

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. $i_o \in \{0, 1, 2, \ldots, q\}$.

A tissue 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 $1, 2, \ldots, q$. We use 0 to refer to the label of the environment, and i_0 denotes the output region (which can be the region inside a membrane 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, as 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 consider 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 $[a]_i \rightarrow [b]_i[c]_i$ can be 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 new cell and by c in the second one.

Rules are used as usual in the framework of membrane computing, that is, in a maximally parallel way. 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.

4 Recognizing Tissue 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 system is the same one as from recognizing P systems with cell-like structure.

Recognizing cell-like P systems were introduced in [16] and they are the natural framework to study and solve decision problems within Membrane Computing, since deciding whether an instance 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 in a natural way with P systems with *input*. The data related 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 encoding each instance as a multiset placed in an *input membrane*. The output of the computation (`yes` or `no`) is sent to the environment. 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 P system with cell division of degree $q \geq 1$ is a tuple

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

where:

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

The computations of the system Π with input $w \in \Sigma^*$ start from a configuration of the form $(w_1, w_2, \ldots, w_{i_{in}} w, \ldots, w_q; \mathcal{E})$, that is, after adding the multiset w to the contents of the input cell i_{in}. We say that $\mathcal{C}$ 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 4.1 *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 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 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 $cod(u)$ 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 tissue P system $\Pi(n)$ to be *confluent*, in the following sense: every computation of such system with the *same* input multiset must always give the *same* answer.

It is denoted by $\mathbf{PMC}_{TD}$ the set of all decision problems which can be solved by means of recognizing tissue P systems with cell division in polynomial time.

5 A Solution for the 3–Coloring Problem

A k–coloring ($k \geq 1$) of an undirected graph $G = (V, E)$ is a function $f : V \to \{1, \ldots, k\}$, where the numbers are interpreted as colors. We say that G is k–colorable if there exists a k–coloring, f, such that $f(u) \neq f(v)$ for every edge $\{u, v\} \in E$ (such a k–coloring f is said to be *valid*).

The 3–coloring problem is the following: *given an undirected graph G, decide whether or not G is 3-colorable*; that is, if there exists a valid 3–coloring of G.

In [4] it is proved that the 3–coloring problem can be solved in linear time by a family of recognizing tissue P systems with cell division.

For each $n, m \in \mathbb{N}$, we shall consider the system

$$\Pi(g(n,m)) = (\Gamma(g(n,m)), \Sigma(n), w_1, w_2(n), \mathcal{R}(g(n,m)), \mathcal{E}(g(n,m)), i_{in}, i_o)$$

being $g(n,m) = ((n+m)(n+m+1)/2) + n$, and where:

- $\Gamma(g(n,m))$ is the set
  $\{A_i, R_i, T_i, B_i, G_i, \overline{R}_i, \overline{B}_i, \overline{G}_i \,:\, 1 \leq i \leq n\} \,\cup$
  $\{a_i \,:\, 1 \leq i \leq 2n + m + \lceil log\ m \rceil + 11\} \,\cup\, \{c_i \,:\, 1 \leq i \leq 2n+1\} \,\cup$
  $\{d_i \,:\, 1 \leq i \leq \lceil log\ m \rceil + 1\} \,\cup \{f_i \,:\, 2 \leq i \leq m + \lceil log\ m \rceil + 6\} \,\cup$
  $\{A_{ij}, P_{ij}, \overline{P}_{ij}, R_{ij}, B_{ij}, G_{ij} \,:\, 1 \leq i < j \leq n\} \cup \{b, D, \overline{D}, e, T, S, N, \flat, \texttt{yes}, \texttt{no}\}$
- $\Sigma(n) = \{A_{ij} \,:\, 1 \leq i < j \leq n\}$
- $w_1 = \{\{a_1, b, c_1, \texttt{yes}, \texttt{no}\}\}$
- $w_2(n) = \{\{D, A_1, \ldots, A_n\}\}$
- $\mathcal{R}(g(n,m))$ is the set of rules:

  1. **Division rules:**
     $r_{1,i} \equiv [A_i]_2 \to [R_i]_2[T_i]_2$ for $i = 1, \ldots, n$
     $r_{2,i} \equiv [T_i]_2 \to [B_i]_2[G_i]_2$ for $i = 1, \ldots, n$
  2. **Communication rules:**
     $r_{3,i} \equiv (1, a_i/a_{i+1}, 0)$ for $i = 1, \ldots, 2n + m + \lceil log\ m \rceil + 10$
     $r_{4,i} \equiv (1, c_i/c_{i+1}^2, 0)$ for $i = 1, \ldots, 2n$
     $r_5 \equiv (1, c_{2n+1}/D, 2)$

$r_6 \equiv (2, c_{2n+1}/d_1\overline{D}, 0)$
$r_{7,i} \equiv (2, d_i/d_{i+1}^2, 0)$ for $i = 1, \ldots, \lceil log\ m \rceil$
$r_8 \equiv (2, \overline{D}/e\, f_2, 0)$
$r_{9,i} \equiv (2, f_i/f_{i+1}, 0)$ for $i = 2, \ldots, m + \lceil log\ m \rceil + 5$
$r_{10,ij} \equiv (2, d_{\lceil log\ m \rceil+1}A_{ij}/P_{ij}, 0)$ for $1 \le i < j \le n$
$r_{11,ij} \equiv (2, P_{ij}/R_{ij}\overline{P}_{ij}, 0)$ for $1 \le i < j \le n$
$r_{12,ij} \equiv (2, \overline{P}_{ij}/B_{ij}G_{ij}, 0)$ for $1 \le i < j \le n$
$r_{13,ij} \equiv (2, R_iR_{ij}/R_i\overline{R}_j, 0)$ for $1 \le i < j \le n$
$r_{14,ij} \equiv (2, B_iB_{ij}/B_i\overline{B}_j, 0)$ for $1 \le i < j \le n$
$r_{15,ij} \equiv (2, G_iG_{ij}/G_i\overline{G}_j, 0)$ for $1 \le i < j \le n$
$r_{16,j} \equiv (2, \overline{R}_jR_j/\flat, 0)$ for $1 \le j \le n$
$r_{17,j} \equiv (2, \overline{B}_jB_j/\flat, 0)$ for $1 \le j \le n$
$r_{18,j} \equiv (2, \overline{G}_jG_j/\flat, 0)$ for $1 \le j \le n$
$r_{19} \equiv (2, e\,\flat/\lambda, 0)$
$r_{20} \equiv (2, e\ f_{m+\lceil log\ m \rceil+6}/T, 0)$
$r_{21} \equiv (2, T/\lambda, 1)$
$r_{22} \equiv (1, b\,T/S, 0)$
$r_{23} \equiv (1, S\ \texttt{yes}/\lambda, 0)$
$r_{24} \equiv (1, b\ a_{2n+m+\lceil log\ m \rceil+11}/N, 0)$
$r_{25} \equiv (1, N\ \texttt{no}/\lambda, 0)$

- $\mathcal{E}(g(n,m)) = \Gamma(g(n,m)) - \{\texttt{yes}, \texttt{no}\}$
- $i_{in} = 2$ is the *input cell.*
- $i_o = 0$ is the *output region.*

Given an undirected graph $G = (V, E)$ with $V = \{A_1, \ldots A_n\}$ and $|E| = m$, we consider $s(u) = g(n,m)$ and $cod(u) = \{A_{ij} : \{A_i, A_j\} \in E \wedge 1 \le i < j \le n\}$. Then, the recognizing tissue P system $\Pi(s(u) = \Pi(g(n,m))$ with input $cod(u)$ processes the instance G via a brute force algorithm, which consists in the following stages:

- *Generation Stage*: The initial cell labelled by 2 is divided into two new cells; and the divisions are iterated until a cell has been produced for each possible candidate solution. Simultaneously, in the cell labelled by 1 there is a counter that will determine the moment in which the checking stage starts.
- *Pre–checking Stage*: After obtaining all possible 3–colorings encoded in cells labelled by 2, this stage provides objects R_{ij}, G_{ij}, B_{ij} in such cells, for every edge A_{ij}.
- *Checking Stage*: Objects R_{ij}, G_{ij}, B_{ij} will be used in cells labelled by 2 to check if there exists a pair of adjacent vertices with the same color in the corresponding candidate solution.
- *Output Stage*: The system sends to the environment the right answer according to the results of the previous stage.

6 A Look Inside Tissue Simulator

The application has been developed using Java and C♯, which are portable and powerful object-oriented languages. Java has been used to parse the data introduced using a grammar generated with ANTLR (one of the most known tools for this purpose, [20]). C♯ has been used to develop the kernel of the application and the graphical interface. The interconnection between both languages is transparent to the user, that just have to move between windows by clicking on buttons. The data are stored in a XML format after serializing the objects that contain the information

The rules of the tissue P system can be rewritten in a way that is very similar to the one used in papers from computational theory. Because computers can not understand directly those kinds of expressions, it has been necessary to develop a *code generator* that writes and compiles during runtime the source code in C♯ simulating the behavior of the rules of the system.

The software follows the Model-View-Controller (MVC) [21], an architecture model of software development used in interactive systems. Three different parts or layers can be distinguished: data handling layer, algorithmic or business logic layer, and user interface or graphical layer. With this, it is easier to do maintenance of the code.

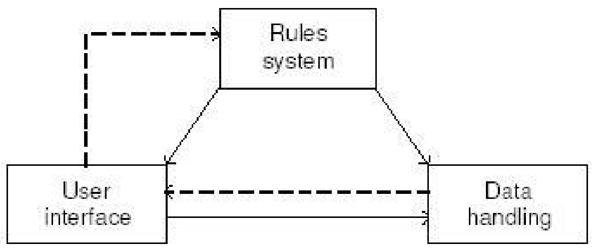


Figure 1: Model-View-Controller architecture software

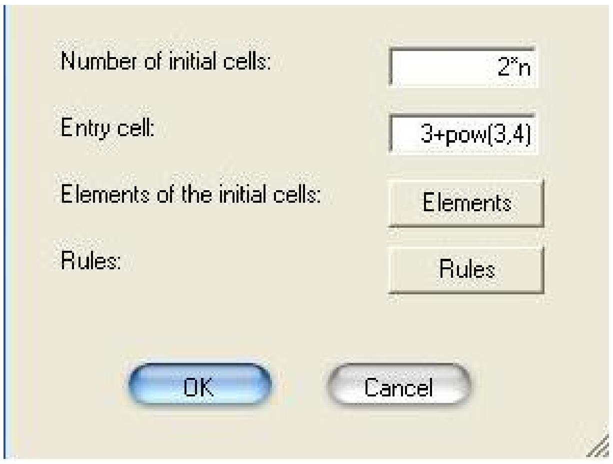
Algorithmic layer implements a recognizing tissue P system with cell division. All rules are applied in a non deterministic and maximal parallel way. The design of the tissue P system machine for solving the problem is non deterministic until $2n$ step. Every computation ways reach the same configuration in $2n$ step. From this moment, the machine is deterministic. By this way, we have chosen only a computation way in order to implement the tissue P system in the software. The computation selected to make the software simulation is the one determined by the lexicographical order of the rule. Other solutions can be easily added. Graphical layer allows the user a visual and friendly interaction with the application. At the end of the simulation, the result of the problem and the trace of the execution is displayed, and for each step the rules applied and the configurations of the tissue are shown.

7 A User Overview of the Application

This software tool allows to follow step by step the performance of tissue P system with cell division solutions of several problems. The pictures presented in this paper have been captured when a solution of the 3-coloring problem (whose files of codification can be found in the folder of application) is provided to the tool.

7.1 Running a simulation

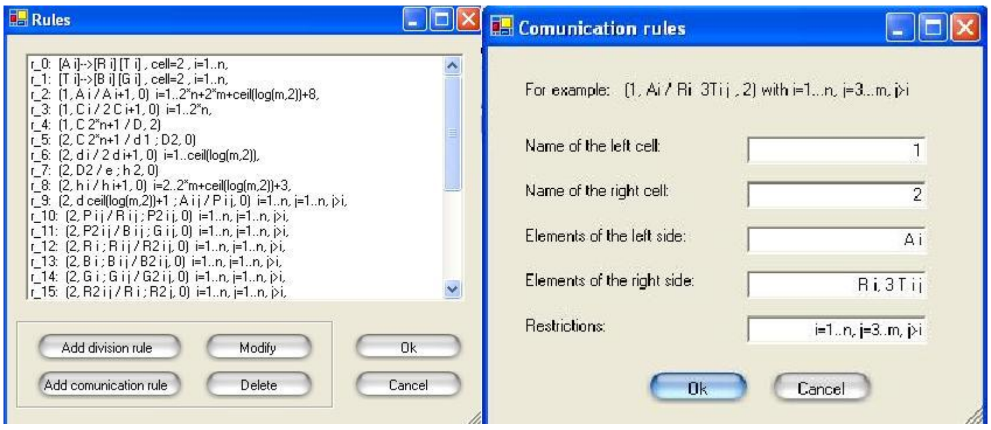
When it is wanted to introduce a concrete tissue P system it has to be selected in tissue menu. Several options are presented and it must be selected new system button. The corresponding window can be depicted in Figure 2.



Number of initial cells:	2*n
Entry cell:	3+pow(3,4)
Elements of the initial cells:	Elements
Rules:	Rules

Figure 2: Definition of system window

Then the rules (Figure 3) and the initial multisets can be incorporated in the window.



Rules

```
r_0:  [A i]-->[R i] [T i] , cell=2 , i=1..n,
r_1:  [T i]-->[B i] [G i] , cell=2 , i=1..n,
r_2:  (1, A i / A i+1, 0)  i=1..2*n+2*m+ceil(log(m,2))+8,
r_3:  (1, C i / 2 C i+1, 0)  i=1..2*n,
r_4:  (1, C 2*n+1 / D, 2)
r_5:  (2, C 2*n+1 / d 1 ; D2, 0)
r_6:  (2, d i / 2 d i+1, 0)  i=1..ceil(log(m,2)),
r_7:  (2, D2 / e ; h 2, 0)
r_8:  (2, h i / h i+1, 0)  i=2..2*m+ceil(log(m,2))+3,
r_9:  (2, d ceil(log(m,2))+1 ; A i j / P i j, 0)  i=1..n, j=1..n, j>i,
r_10:  (2, P i j / R i j ; P2 i j, 0)  i=1..n, j=1..n, j>i,
r_11:  (2, P2 i j / B i j ; G i j, 0)  i=1..n, j=1..n, j>i,
r_12:  (2, R i ; R i j / R2 i j, 0)  i=1..n, j=1..n, j>i,
r_13:  (2, B i ; B i j / B2 i j, 0)  i=1..n, j=1..n, j>i,
r_14:  (2, G i ; G i j / G2 i j, 0)  i=1..n, j=1..n, j>i,
r_15:  (2, R2 i j / R i ; R2 j, 0)  i=1..n, j=1..n, j>i,
```

Comunication rules

For example: (1, Ai / Ri 3Ti j , 2) with i=1...n, j=3...m, j>i

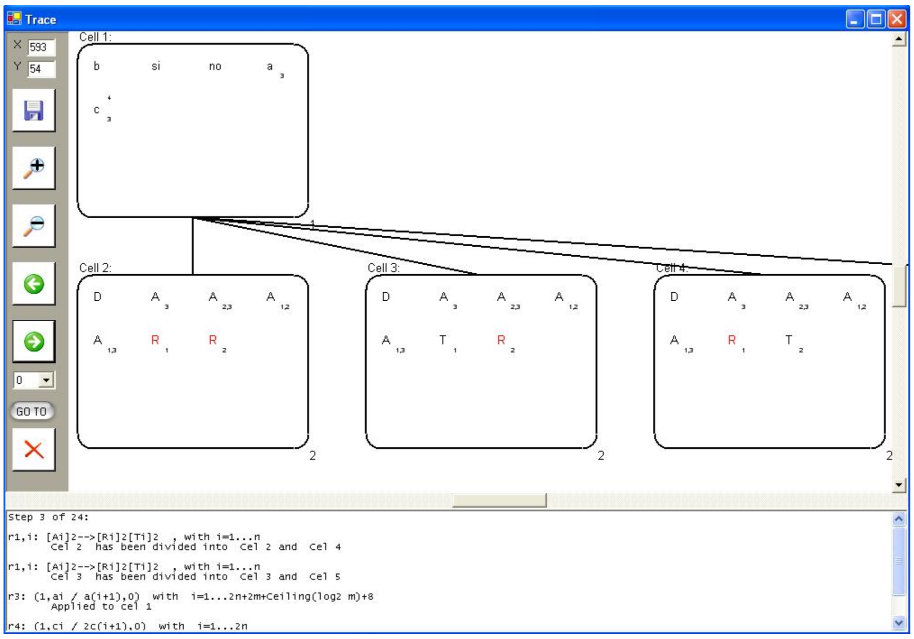
Name of the left cell:	1
Name of the right cell:	2
Elements of the left side:	A i
Elements of the right side:	R i, 3 T i j
Restrictions:	i=1..n, j=3..m, j>i

Figure 3: The set of rules for a solution of 3-coloring problem

Next, a concrete input multiset, that depends on the specific instance of the

3-coloring problem, is supplied. For this, the *new instance* option in tissue menu is selected.

If the user needs run a simulation, it is just needed to select the option *Run* in tissue menu. Then the system ask the path of two .xml files, one for the model of the system, and another one for a concrete instance of the problem. After that, it calculates all the steps of the computation, shows the answer, and open the trace window of the system, as can be depicted in Figure 4.



```
Step 3 of 24:

r1,i: [Ai]2-->[Ri]2[Ti]2  , with i=1...n
      Cel 2  has been divided into  Cel 2 and  Cel 4

r1,i: [Ai]2-->[Ri]2[Ti]2  , with i=1...n
      Cel 3  has been divided into  Cel 3 and  Cel 5

r3: (1,ai / a(i+1),0)  with  i=1...2n+2m+Ceiling(log2 m)+8
      Applied to cel 1

r4: (1,ci / 2c(i+1),0)  with  i=1...2n
```

Figure 4: Trace window for a solution of 3-coloring problem

For each step, a situation can be seen in a graphical way, and at the bottom of the screen appear the different rules applied in that step. The picture can be easily handled at any moment, and can be saved in different images format. Moving between the steps can be done in two ways: first, by pressing the buttons with arrows to move from the i-th step to the $(i-1)$-th or the $(i+1)$-th one. Second, choosing the "GO TO" option to go directly to one step of the system.

Software, manuals, examples, a mailing list, useful links, and other resources of *tissue simulator* can be found at: `http://www.tissuesimulator.es.kz`.

8 Future Work

In this paper, a simulator of recognizing tissue P systems has been presented. The simulator has been developed using Java and C♯, which are portable and powerful object-oriented languages. Future works will be focused on different improvements of this software application like introducing a semantic parser, contemplating others kind of rules, such as membrane creation, etc.

One interesting future tasks would be to expand the tool so it can works with others systems like spiking neural P systems.

Acknowledgement

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

References

[1] A. Alhazov, R. Freund, M. Oswald. Tissue P Systems with Antiport Rules ans Small Numbers of Symbols and Cells. In volume 3572 of *Lecture Notes in Computer Science*, pages 100–111. 2005.

[2] F. Bernardini, M. Gheorghe. Cell Communication in Tissue P Systems and Cell Division in Population P Systems. *Soft Computing*, 9(9):640–649, 2005.

[3] R. Borrego–Ropero, D. Díaz–Pernil, J.A. Nepomuceno–Chamorro. VisualTissue: a friendly tool to study tissue P systems solutions for graph problems. In M.A. Gutiérrez Naranjo, Gh. Paun, A. Romero–Jiménez, A. Riscos–Núñez, editors, *Proceedings of the Fifth Brainstorming Week on Membrane Computing*, pages 87–96. Fénix Editora, 2007.

[4] D. Díaz–Pernil, M.A. Gutiérrez–Naranjo, M.J. Pérez–Jiménez, A. Riscos–Núñez. A uniform family of tissue P system with cell division solving 3-COL in a linear time. *Theoretical Computer Science*, in press.

[5] D. Díaz–Pernil, M.A. Gutiérrez–Naranjo, M.J. Pérez Jiménez M.J., A. Riscos–Núñez. A Linear Solution for Subset Sum Problem with Tissue P Systems with Cell Division. In M.A. Gutiérrez Naranjo, Gh. Paun, A. Romero–Jiménez, A. Riscos–Núñez, editors, *Proceedings of the Fifth Brainstorming Week on Membrane Computing*, pages 113–130. Fénix Editora, 2007.

[6] R. Freund, Gh. Păun, M.J. Pérez-Jiménez. Tissue P Systems with channel states. *Theoretical Computer Science*, 330:101–116, 2005.

[7] C. Martín–Vide, J. Pazos, Gh. Păun, A. Rodríguez–Patón. A New Class of Symbolic Abstract Neural Nets: Tissue P Systems. In volume 2387 of *Lecture Notes in Computer Science*, pages 290–299, 2002.

[8] C. Martín–Vide, J. Pazos, Gh. Păun, A. Rodríguez–Patón. Tissue P systems. *Theoretical Computer Science*, 296:295–326, 2003.

[9] I.A. Nepomuceno–Chamorro. A Java Simulator for Membrane Computing. *Journal of Universal Computer Science*, 10(5):620–629, 2004.

[10] Gh. Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143, 2000.

[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):295–305, 2002.

[13] Gh. Păun, M.J. Pérez–Jiménez, A. Riscos-Núñez, A. Tissue P System with cell division. In Gh. Păun, A. Riscos-Núñez, A. Romero-Jiménez, F. Sancho–Caparrini, editors, *Second Brainstorming Week on Membrane Computing*, pages 380–386. Sevilla, Report RGNC 01/2004, 2004.

[14] M.J. Pérez–Jiménez. An approach to computational complexity in Membrane Computing. In volume 3365 of *Lecture Notes in Computer Science*, pages 85–109. 2005.

[15] M.J. Pérez–Jiménez, F.J. Romero. P systems, a new computational modelling tool for Systems Biology. In *Transactions on Computational Systems Biology VI.*, volume 4220 of *Lecture Notes in Bioinformatics*, pages 176–197, 2006.

[16] 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. Vaszil, editors, *Proceedings of the 5th Workshop on Descriptional Complexity of Formal Systems, DCFS 2003*, pages 284–294. 2003.

[17] 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, editors, *Preproceedings of the Workshop on Membrane Computing*, pages 356–364. Report RGML 28/03, Tarragona, 2003.

[18] D. Ramírez–Martínez,M.A. Gutiérrez–Naranjo. A Software Tool for Dealing with Spiking Neural P Systems. In M.A. Gutiérrez Naranjo, Gh. Paun, A. Romero–Jiménez, A. Riscos–Núñez, editors, *Proceedings of the Fifth Brainstorming Week on Membrane Computing*, pages 299–313. Fénix Editora, 2007.

[19] P systems web page `http://psystems.disco.unimib.it/`

[20] web site `http://www.antlr.org`

[21] web site `http://en.wikipedia.org/wiki/Model-view-controller`