Solving 3-COL with Tissue P Systems

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

Dpto. de Ciencias de la Computación e Inteligencia Artificial
E.T.S. Ingeniería Informática. Universidad de Sevilla
Avda. Reina Mercedes s/n, 41012, Sevilla, España
{sbdani, magutier, marper}@us.es

Summary. In the literature, several examples of the efficiency of cell-like P systems in order to solve **NP**-complete problems in polynomial time can be found. Recently, various new models of tissue-like P systems have received important attention from the scientific community. In this paper we present a linear-time solution to an **NP**-complete problem, the 3-COL problem, and discuss the possibilities of tissue-like P systems to solve hard problems.

1 Introduction

Membranes are involved in many reactions taking place inside various compartments of a cell, and they act as selective channels of communication between different compartments as well as between the cell and its environment [1]. Membrane Computing is a branch of Natural Computing which starts from the assumption that the processes taking place in the compartmental structure of a living cell can be interpreted as computations.

This is a new cross-disciplinary field with contributions by computer scientists, biologists, formal linguists and complexity theoreticians, enriching each others with results, open problems and promising new research lines. One of the topics in that field is the study of the computational power and efficiency of devices with multisets distributed along the regions of a membrane structure, and with rules for rewriting or moving the elements of such multisets.

This emergent branch of Natural Computing was introduced by Gh. Păun in [21]. Since then it has received important attention from the scientific community. In fact, Membrane Computing has been selected by the Institute for Scientific Information, USA, as a fast *Emerging Research Front* in Computer Science, and [20] was mentioned in [28] as a highly cited paper in October 2003.

The devices of this model are called *P systems*. Roughly speaking, a P system consists of a membrane structure, in the compartments of which one places

multisets of objects which evolve according to given rules in a synchronous nondeterministic maximally parallel manner[1].

In the last years, many different models of P systems have been proposed. If the different models are classified according to the membrane structure, the most studied variant is characterized by a *cell-like* membrane structure, where the communication channels are from a membrane to the surrounding one. In this model we have a set of nested membranes where the graph of neighborhood relation is a tree.

Different models of these cell-like P systems have been successfully used in order to design solutions to **NP**-complete problems in polynomial time (see [9] and references therein). These solutions are obtained by generating an exponential amount on workspace in polynomial time and using parallelism to check simultaneously all the candidates to solution. Inspired in living cells, cell-like P systems abstract the way of obtaining new membranes, mainly from two biological processes: *mitosis* (membrane division) and *autopoiesis*, see [13] (membrane creation). Both ways of generating new membranes have given rise to different variants of P systems: *P systems with active membranes*, where the new workspace is generated by membrane division and *P systems with membrane creation*, where the new membranes are created from objects.

Both models are universal from a computational point of view, but technically, they are pretty different. In fact, nowadays there does not exist any theoretical result which proves that these models can simulate each other in polynomial time.

Under the hypothesis $\mathbf{P}\neq\mathbf{NP}$, Zandron et al. [27] established the limitations of P systems that do not use membrane division concerning the efficient solution of **NP**-complete problems. This result was generalized by Pérez-Jiménez et al. [17] obtaining a characterization of the $\mathbf{P}\neq\mathbf{NP}$ conjecture by the polynomial time unsolvability of an **NP**-complete problem by language accepting P systems (without membrane division rules).

We shall focus here on a second type of P systems, the so-called (because of their membrane structure) *tissue P systems*. Instead of considering that membranes are hierarchically arranged, the membranes are placed in the nodes of a graph, all of them at the same level. This variant has two starting points (see [15]): 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[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 [14, 15], several research lines have been developed and other variants have arisen (see, for example, [2, 5, 6, 11, 12, 23]). One of the most interesting variants of tissue P systems was presented in [19]. This paper combines the definitions of tissue P systems and P systems with

[1] A layman-oriented introduction can be found in [22] and further bibliography at [29].
[2] This way of communication for P systems was introduced in [20].

active membranes and consider tissue P systems (with the communication done through symport/antiport rules) with cell division rules of the same form as in P systems with active membranes, but without using polarizations.

One of the main features of these tissue P systems with cell division is related to their computational efficiency. In [19], a polynomial-time solution to the **NP**-complete problem SAT is shown. In this paper we go on with the research in this variant and present a linear-time solution to another well-known **NP**-complete problem, the 3-COL problem.

The paper is organized as follows: first we recall some preliminaries and the definition of tissue P systems with cell division. Next, recognizer tissue P systems are briefly described. A linear–time solution to the 3-COL problem is presented in the following section, with a short overview of the computation and the necessary resources. Finally, the main results, some conclusions, and new open research lines are presented.

2 Preliminares

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 the string is the *length* of the string. 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 $\Sigma^\star = \cup_{n\geq 0}\sigma^n$. A *language* over Σ is a subset from $\Sigma^\star$.

Let D be a set. A *multiset* over D is a pair[3] $\langle D, f\rangle$ where $f : D \to \mathbb{N}$ is a mapping. If $\mathcal{A} = \langle A, f\rangle$ is a multiset, its *support*, $supp(\mathcal{A})$ is defined as $supp(\mathcal{A}) = \{x \in A \,|\, f(x) > 0\}$ and its *size* denoted by $|A|$ is defined as

$$|A| = \sum_{a\in A} f(a).$$

A multiset is empty (resp. finite) if its support is the empty set (resp. finite). The set of all the multisets on D is denoted by $\mathrm{M}(D)$.

If $m = (A, f)$ is a finite multiset on A, then it will be denoted as $m = \{\{a_1, \ldots, a_m\}\}$, where the elements a_i occurs $f(a_i)$ times.

A *directed graph* $\mathcal{G}$ is a pair $\mathcal{G} = (V, E)$ where V is a set and $E \subseteq V \times V$. If $(u, v) \in E$ or $(v, u) \in E$, we will say that u is *adjacent* to v. The *degree* of $v \in V$ is the number of adjacent vertices to v.

A *path* of length n from a vertex x to a vertex y in a directed graph $\mathcal{G} = (V, E)$ is a sequence $\{v_0, v_1, \ldots, v_n\}$ of vertices such that $v_0 = x$, $v_n = y$ and $\{(v_i, v_{i+1}) \,|\, i = 0, \ldots, n-1\} \subseteq E$. If there is a path from u to v in $\mathcal{G}$, we will say that v is *reachable* from u in $\mathcal{G}$ and it will be denoted by $u \rightsquigarrow_{\mathcal{G}} v$. The vertices u and v are *connected* in G if $u \rightsquigarrow_{\mathcal{G}} v$ or $v \rightsquigarrow_{\mathcal{G}} u$. The directed graph is *connected* if

[3] A detailed presentation of multisets can be found, for example, in [26].

for every pair of different vertices from V, one of them is reachable from the other one.

In what follows we assume the reader is already familiar with the basic notions and the terminology underlying P systems. For details, see [24].

3 Tissue P Systems with Cell Division

In the first works on tissue P systems the membrane structure does not change along the computation [14, 15]. Based on the cell-like model of P systems with active membranes, in [19] a new model of tissue P systems is presented, *with cells able to divide*. The biological inspiration is clear. Alive tissues are not *static* network of cells. Cell are duplicated via mitosis in a natural way. As computational model, the main features of this model are that cells are not polarized (the contrary holds in the cell-like model of P systems with active membranes, see [24]); the cells obtained by division have the same labels as the father cell and if a cell is divided, the interaction of the cell with other cells or with the environment is blocked during the mitosis process. In some sense, this means that a cell which divides, first cuts all its communication channels with the other cells and with the environment.

Formally, a *tissue P system with cell division* is a construction of the form

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

where:

1. $m \geq 1$ is the initial degree of the system. At the beginning, the system has m cells, labeled by $1, 2, \ldots, m$.
2. Γ is a finite *alphabet*, whose symbols will be called *objects*.
3. $w_1, \ldots, w_m$ are strings over Γ, describing the multisets of objects placed in the m cells of the system.
4. $E \subseteq \Gamma$ is the set of objects placed in the environment in an arbitrary large amount of copies.
5. $\mathcal{R}$ is a finite set of evolution rules of the following form:
   (a) *Communication rules*: $(i, x/y, j)$, for $i, j \in \{0, 1, 2, \ldots, m\}, i \neq j$, $x, y \in \Gamma^*$, where $1, 2, \ldots, m$ are the labels of the cells of the system (0 is the label of the environment); when the rule $(i, x/y, j)$ is applied, the objects of the multiset represented by x are sent from the region i to the region j and simultaneously, the objects of the multiset represented by y are sent from the region j to the region i.
   (b) *Division rules*: $[a]_i \rightarrow [b]_i[c]_i$, where $i \in \{1, 2, \ldots, m\}$ and $a, b, c \in \Gamma$; when an object a appears in a cell labeled by i, the cell divides into other two membranes 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.
6. i_o is the output cell.

Rules are applied 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 evolve by a rule of any form must do it, i.e., in each step we apply a maximal set of rules and no other rule can be added.

This way of applying rules has only one restriction: when a cell is divided, the division rule is the only which is applied for that cell in that step; the objects inside that cell do not evolve in the step when the cell is divided.

4 Recognizer tP Systems

Usually, **NP**-completeness has been 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 a similar way to other P system models, some special features have to be imposed for studying the efficiency of such systems in order to solve **NP**-complete problems.

In order to study the computing efficiency for solving **NP**-complete decision problems in [19] a variant of tissue P systems with membrane division is introduced: *recognizer tissue P systems with input*. The key idea of such recognizer system is the same as for recognizer P systems with cell-like structure.

Recognizer P systems were introduced in [18] and are the natural framework to study and solve decision problems, 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, recognizer 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 codifying each instance as a multiset placed in an *input membrane*. The output of the computation (`yes` or `no`) is sent to the environment. In this way, 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 recognizer tissue P system is a tuple

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

where:

- $(O, w_1, \ldots, w_m, E, \mathcal{R}, i_{in})$ is a tissue P system.
- The working alphabet O has two distinguished objects `yes` and `no`, present in at least one copy in w_1, w_2, …, w_n but not present in E.

- Σ is an (input) alphabet strictly contained in O.
- $i_{in} \in \{1, \ldots, m\}$ is the input cell.
- The output region 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.

Another important feature of P systems is the non-determinism. When designing a family of recognizer P systems, one has to be aware of that, because all possible non-deterministic computations must produce the same answer.

Definition 1. *A* P system with input *is a tuple (Π, Σ, i_Π), where: (a) Π is a P system, with working alphabet Γ, with p membranes labeled by $1, \ldots, p$, and initial multisets $w_1, \ldots, w_p$ associated with them; (b) Σ is an (input) alphabet strictly contained in Γ; the initial multisets are over $\Gamma - \Sigma$; and (c) i_Π is the label of a distinguished (input) membrane.*

The computations of the system Π start from configurations of the form $(w_1, w_2, \ldots, w_{in}w, \ldots, w_m; E)$, where $w \in \Gamma^*$ (that is, after adding the multiset w to the contents of the input cell). We say that the multiset w is recognized by Π if and only if the object `yes` is sent to the environment. 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 2. *Let $\mathcal{F}$ be a class of recognizer P systems. 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 $\mathcal{F}$, and we denote this by $X \in \mathbf{PMC}_{\mathcal{F}}$, if the following holds:*

- *The family $\mathbf{\Pi}$ is* polynomially uniform by Turing machines, *that is, there exists a deterministic Turing machine constructing $\Pi(n)$ from $n \in \mathbb{N}$ in polynomial time.*
- *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 $cod(u)$ 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 must always give the *same* answer.

The set of all decision problems which can be solved by means of recognizer tP systems with cell division in a number of steps bounded by a mapping f form the complexity[4] class $\mathbf{MC}_{TD}(f)$. We are interested in polynomial time solutions, therefore we consider the class $\mathbf{PMC}_{TD}$, obtained as the union of the classes $\mathbf{MC}_{TD}(f)$, for all polynomials f.

5 A Solution to 3-COL

Let G be a graph with $V(G)$ as set of vertices and $E(G)$ as set of edges. A k-coloring of a graph G is a function $C : V(G) \to \{1, \ldots, k\}$ such that for all $v, w \in V(G)$, if $C(v) = C(w)$, then $((v, w) \notin E(G)$. The $k - COL$ problem is to determine the number of such k-colorings for G.

This problem is related to the famous Four Color Conjecture (proved by Appel and Haken [3, 4]). It is a special case of the problem of k-colorability of a graph, in which the range of C is $\{1, \ldots, k\}$ with k being specified as part of the instance. The **NP**-completeness of the 3-COL problem was proved by Stockmeyer [25] (see [8]).

Next, we will see that the 3-COL problem can be solved by tissue P systems with cellular division in linear time.

Let us consider a graph $G = (U, V)$, where $U = \{u_i \mid 1 \leq i \leq n\}$ is the set of vertices and $V = \{v_j \mid v_j = u_{j_1}u_{j_2} \wedge 1 \leq j \leq m \wedge 1 \leq j_1 < j_2 \leq n\}$ is the set of edges.

Let us consider $(n, (A_{ij})_n)$ in order to denote a generic instance of the problem, with n the size of the graph G, i.e., the number of vertices in G and let A be the set of the edges in the graph G, with

$$(A_{ij})_n = (A_{ij} : A_iA_j \in V \wedge 1 \leq i < j \leq n).$$

We will address the resolution via a brute force algorithm, in the framework of recognizer tissue P systems with cell division, which consists in the following phases:

- *Generation Stage*: The initial cell, labeled by 2, is divided into two new cells. These cells are divided again, and so on. After several divisions we have an element R_i, B_i or G_i for each element A_i. Hence, after an appropriate number of divisions, in each cell labeled with 2 we will have a candidate solution to the problem. Simultaneously, in the membrane labeled by 1 a counter evolves to determine the moment in which the checking stage starts.

[4] More precise definitions of complexity classes in terms of membrane computing can be found in [16].

- *Checking Stage*: After the generation stage, the checking stage begins. For each cell labeled by 2, we check if there exists a pair of adjacent vertices with the same color. If this happens, an element $\flat$ is brought from the environment. Otherwise, the object $\flat$ is not brought in.
- *Output Stage*: The system sends to the environment the right answer according to the previous stage.
  1. **Answer** `Yes`: There exists a cell labeled by 2 such that it does not contain an object $\flat$. In this case, an object T is sent to the cell labeled by 1 and an object `Yes` is sent to the environment.
  2. **Answer** `No`: It is the converse case. If all the cells labeled by 2 contain an object $\flat$, then the cell labeled by 1 does not receive any object T, and an object `No` is sent to the environment.

Next, we provide a linear time solution of 3-COL by a family of recognizer tissue P systems with membrane division.

Let us consider a *size* mapping on the set of instances of the problem. Let s be the mapping $s : I_{3-\mathtt{COL}} \to \mathbb{N}$ where $I_{3-\mathtt{COL}}$ is the set of all instances $u = (n, (A_{ij})_n)$ of the 3-COL problem and $s(u) = n$. Obviously, s is linear time computable.

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

$$\Pi(n) = (\Gamma(n), \Sigma(n), \mathcal{R}(n), \mathcal{E}(n), \mathcal{M}_1(n), \mathcal{M}_2(n), i(n), o(n)),$$

where:

- $\Gamma(n) = \{A_i, R_i, T_i, B_i, G_i, \overline{R}_i, \overline{B}_i, \overline{G}_i \mid 1 \leq i \leq n\} \cup \{a_i \mid 1 \leq i \leq 2n + \lceil log_2 m \rceil + 12\} \cup \{c_i \mid 1 \leq i \leq 2n+1\} \cup \{d_i \mid 1 \leq i \leq \lceil log_2 m \rceil + 1\} \cup \{f_i \mid 2 \leq i \leq \lceil log_2 m \rceil + 7\} \cup \{P_{ij}, \overline{P}_{ij}, R_{ij}, B_{ij}, G_{ij} \mid 1 \leq i < j \leq n\} \cup \{b, D, E, e, T, S, N, \flat, \mathtt{yes}, \mathtt{no}\}$,
- $\Sigma(n) = \{A_{ij} \mid 1 \leq i < j \leq n\}$,
- $\mathcal{M}_1(n) = a_1, b, c_1, \mathtt{yes}, \mathtt{no}$,
- $\mathcal{M}_2(n) = D, A_1, \ldots, A_n$,
- $\mathcal{R}(n)$ is the set of rules:
  1. **Division rules:**
     $r_{1,i} = [A_i]_2 \to [R_i]_2[T_i]_2$ for $i = 1, \ldots, n$,
     $r_{2,i} = [T_i]_2 \to [B_i]_2[G_i]_2$ for $i = 1, \ldots, n$,
  2. **Communication rules:**
     $r_{3,i} = (1, a_i / a_{i+1}, 0)$ for $i = 1, \ldots, 2n + \lceil log_2 m \rceil + 11$,
     $r_{4,i} = (1, c_i / c_{i+1}^2, 0)$ for $i = 1 \ldots, 2n$,
     $r_5 = (1, c_{2n+1} / D, 2)$,
     $r_6 = (2, c_{2n+1} / d_1 E, 0)$,
     $r_{7,i} = (2, d_i / d_{i+1}^2, 0)$ for $i = 1, \ldots, \lceil log_2 m \rceil$,
     $r_8 = (2, E / e\, f_2, 0)$,
     $r_{9,i} = (2, f_i / f_{i+1}, 0)$ for $i = 2, \ldots, \lceil log_2 m \rceil + 6$,
     $r_{10,ij} = (2, d_{\lceil log_2 m \rceil + 1} A_{ij} / P_{ij}, 0)$ for $1 \leq i < j \leq n$,
     $r_{11,ij} = (2, P_{ij} / R_{ij} \overline{P}_{ij}, 0)$ for $1 \leq i < j \leq n$,
     $r_{12,ij} = (2, \overline{P}_{ij} / B_{ij} G_{ij}, 0)$ for $1 \leq i < j \leq n$,

$r_{13,ij} = (2, R_i R_{ij} / R_i \overline{R}_j, 0)$ for $1 \leq i < j \leq n$,
$r_{14,ij} = (2, B_i B_{ij} / B_i \overline{B}_j, 0)$ for $1 \leq i < j \leq n$,
$r_{15,ij} = (2, G_i G_{ij} / G_i \overline{G}_j, 0)$ for $1 \leq i < j \leq n$,
$r_{16,j} = (2, \overline{R}_j R_j / \flat, 0)$ for $1 \leq j \leq n$,
$r_{17,j} = (2, \overline{B}_j B_j / \flat, 0)$ for $1 \leq j \leq n$,
$r_{18,j} = (2, \overline{G}_j G_j / \flat, 0)$ for $1 \leq j \leq n$,
$r_{19} = (2, e\flat / \lambda, 0)$,
$r_{20} = (2, f_{\lceil log_2 m \rceil + 7} e / T, 0)$,
$r_{21} = (2, T / \lambda, 1)$,
$r_{22} = (1, b\, T / S, 0)$,
$r_{23} = (1, S\mathtt{yes} / \lambda, 0)$,
$r_{24} = (1, a_{2n + \lceil log_2 m \rceil + 12} b / N, 0)$,
$r_{25} = (1, N\mathtt{no} / \lambda, 0)$,

- $\mathcal{E}(n) = \Gamma - \{\mathtt{yes}, \mathtt{no}\}$,
- $i(n) = 2$ is the *input cell*,
- $o(n) = env$ is the *output cell*.

5.1 An overview of the computation

First of all we define a polynomial encoding of the 3-COL problem in the family $\mathbf{\Pi}$ constructed in the previous section. Given an instance u of the problem, $u = (n, (A_{ij})_n)$, with size $s(u) = n$ the codification of the instance will be the multiset $cod(u) = \{\{A_{ij} \mid 1 \leq i < j \leq n\}\}$.

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 do not work. In this stage we have two parallel processes.

- On the one hand, in the cell labeled by 1 we have two counters: a_i, which will be used in the answer stage and c_i, which will be multiplied until step $2n$, where 4^n copies are obtained.
- On the other hand, in the cell labeled by 2, the division rules are applied. For each object A_i (which codifies a vertex of the graph) we get three cells labeled by 2, each of them encoding one of the three colors.
  After the appropriate divisions, in the step $2n$ we get exactly 3^n cells encoding all the possible 3-colorings of the graph

In this way, in the step $2n$ the generation step is finished and the *checking stage* starts.

One of the copies of the counter c in the cell 1 is traded by each object D that appears in each cell labeled by 2. Therefore, in the cell 1, there remain $4^n - 3^n$ copies of the counter c. When the object c_{2n+1} arrives to the cell labeled by 2, the communication starts.

At the beginning of the process, we pay attention to the counters d and f. The former, d, will be multiplied until at least m copies are obtained. At this point, we

will start the work with the edges. The latter, f, will be useful in order to send an object T to the cell 1.

When the m copies of the object d are obtained, each of them are send together an object A_{ij} (which codifies an edge for each $1 \leq i < j \leq n$) to the environment and an object P_{ij} is got.

When the objects P_{ij} are obtained, they are interchanged by R_{ij}, B_{ij}, G_{ij} with the environment. In this way, all the possible colors for the edges are obtained and the proper checking stage starts.

This stage consists on checking, for each cell labeled by 2, if for each edge there exist two adjacent vertices with the same color. Let us take a color, say *red* (for *green* and *blue*, the process is similar):

- An edge R_{ij} is taken.
- If the vertex i is encoded in the cell 2 with the color *red*, then the object $\overline{R}_j$ is brought from the environment.
- If the vertex j is also of color *red*, then the objects $R_j\overline{R}_j$ are traded with the environment by $\flat$.
- If $\flat$ appears, it is sent out together e to the environment and the rule which sends an object T to the cell 1 can not be triggered.
- If the object $\flat$ does not appears in the step $\lceil log_2 m \rceil + 11$, then T is brought from the environment, and it is sent in the next step to the cell 1.

Notice that in the generation stage, the processes are parallel, but in the checking stage, when the red color is checked, the checking of the other two colors has not begun. If there do not exist two vertices with the same color when the counter f ends, then the object T is changed with the objects $f_{\lceil log_2 m \rceil + 7}$ and e. The counter d is multiplied until it reaches m copies (the number of edges).

If the object T appears in the cell 1 in the step $2n + \lceil log_2 m \rceil + 11$, then the object S is brought from the environment interchanged by the objects b and T. In the next step S and `yes` are sent to the environment and the computation halts. No more rules can be applied, since b has been sent out from the cell 1.

Otherwise, if in the step $2n + \lceil log_2 m \rceil + 12$ the object T has not appeared, the last element of the counter a, together with b are sent to the environment and N is sent into the cell. In the next step, N and `no` are sent to the environment. Therefore, the number of steps is $2n + \lceil log_2 m \rceil + 15$ if the answer is not.

5.2 Necessary Resources

The presented family of tissue P systems that solves the 3-COL problem is polynomially uniform by Turing machines. It can be observed that the definition of the family is done in a recursive manner from a given instance, in particular from the constants n and m. Furthermore the necessary resources to build an element of family are:

- size of the alphabet: $12n + 6m + 3\lceil log_2 m \rceil + 31 \in \theta(n + m)$,

- initial number of cells: $2 \in \theta(1)$,
- initial number of objects: $n + m + 6 \in \theta(n + m)$
- number of rules: $9n + 6m + 3\lceil log_2 m \rceil + 27$, the maximal length of a rule is 4, then the sum of the rules' lengths is about $\theta(n + m)$.

So a Turing machine can build $\Pi(h(u))$ in lineal time with respect to $h(u)$.

5.3 Main results

From the discussion in the previous sections and according to Section 4, we deduce the following result:

Theorem 1. *3-COL*$\in \mathbf{PMC}_{TD}$.

Although the next result is a corollary of Theorem 1, we formulate it as another theorem, in order to stress its relevance.

Theorem 2. $\mathbf{NP} \subseteq \mathbf{PMC}_{TD}$.

Proof. It suffices to make the following observations: the 3-COL problem is **NP**-complete, 3-COL $\in \mathbf{PMC}_{TD}$ and the class $\mathbf{PMC}_{TD}$ is stable under polynomial-time reduction. □

This theorem can be extended, if we notice that the class $\mathbf{PMC}_{TD}$ is closed under complement.

Theorem 3. $\mathbf{NP} \cup co - \mathbf{NP} \subseteq \mathbf{PMC}_{TD}$.

6 Conclusions and Future Work

The power and efficiency of cell-like P systems for solving **NP**-complete problems have been widely studied. Nevertheless, there are very few works studying the case of tissue-like P systems.

In this paper we propose a new solution to an **NP**-complete problem which can be used as a scheme for designing solutions to other problems from Graph Theory as the Vertex Cover Problem, Clique, etc. Moreover, the type of solution presented can be also adapted for solving numerical problems.

Recently, a new P system model based on the idea of spiking neurons has been presented (see, for example, [10]). It remains as further work to bridge tissue P systems and this new model.

Acknowledgement

This work is supported by project TIN2005-09345-C04-01 of Ministerio de Educación y Ciencia of Spain, cofinanced by FEDER funds.

References

1. B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, P. Walter: *The Molecular Biology of the Cell*. Fourth Edition. Garland Publ. Inc., London, 2002.
2. A. Alhazov, R. Freund, M. Oswald: Tissue P systems with antiport rules and small numbers of symbols and cells. In *Developments in Language Theory, 9th International Conference, DLT 2005* (C. de Felice, A. Restivo, eds.) Palermo, Italy, 2005. LNCS 3572, Springer, 2005, 100–111.
3. K. Appel, W. Haken: Every planar map is 4-colorable - 1: Discharging. *Illinois Journal of Mathematics*, 21, (1977), 429–490.
4. K. Appel, W. Haken: Every planar map is 4-colorable - 2: Reducibility. *Illinois Journal of Mathematics*, 21, (1977), 491–567.
5. F. Bernardini, M. Gheorghe: Cell communication in tissue P systems and cell division in population P systems. *Soft Computing*, 9 (2005), 640–649.
6. R. Freund, Gh. Păun, M.J. Pérez-Jiménez: Tissue P systems with channel states. *Theoretical Computer Science*, 330, (2005), 101–116.
7. P. Frisco, H.J. Hoogeboom: Simulating counter automata by P systems with symport/antiport. In *Membrane Computing. International Workshop WMC 2002, Curtea de Arges, Romania, Revised Papers* (Gh. Păun, G. Rozenberg, A. Salomaa, C. Zandron, eds.), LNCS 2597, Springer, 2003, 288–301.
8. M.R. Garey, D.S. Johnson: *Computers and Intractability A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.
9. M.A. Gutiérrez-Naranjo, M.j. Pérez-Jiménez, F.J. Romero-Campero: A linear solution for QSAT with membrane creation. In *Membrane Computing* (R. Freund, Gh. Păun, G. Rozenberg, A. Salomaa, eds.), LNCS 3850, Springer, 2006, 241–252.
10. M. Ionescu, Gh. Păun, T. Yokomori: Spiking neural P systems. *Fundamenta Informaticae*, 71, 2-3 (2006), 279–308.
11. S.N. Krishna, K. Lakshmanan, R. Rama: Tissue P systems with contextual and rewriting tules. *WMC-CdeA 2002* (Gh. Păun et al., eds.), LNCS 2597, Springer, 2003, 339–351.
12. K. Lakshmanan, R. Rama: On the power of tissue P systems with insertion and deletion rules. In *Preproceedings of the Workshop on Membrane Computing*, Tarragona, 2003 (A. Alhazov, C. Martín-Vide, Gh. Păun, eds.), Report URV, Tarragona, 2003, 304–318.
13. P.L. Luisi: The chemical implementation of autopoiesis. In *Self-Production of Supramolecular Structures* (G.R. Fleishaker et al., eds.), Kluwer, Dordrecht, 1994.
14. C. Martín-Vide, J. Pazos, Gh. Păun, A. Rodríguez Patón: A new class of symbolic abstract neural nets: Tissue P systems. In *Proc. COCOON 2002* (O.H. Ibarra, L. Zhang, eds.), LNCS 2387, Springer, 2002, 290–299.
15. C. Martín-Vide, J. Pazos, Gh. Păun, A. Rodríguez Patón: Tissue P systems. *Theoretical Computer Science*, 296 (2003), 295–326
16. M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini: *Teoría de la Complejidad en Modelos de Computación Celular con Membranas*. Editorial Kronos, Sevilla, 2002.
17. M.J. Pérez-Jiménez, A. Romero-Jim'enez, F. Sancho-Caparrini: The P versus NP problem through cellular computing with membranes. In *Aspects of Molecular Computing, Essays Dedicated to Tom Head on the Ocassion of His 70th Birthday* (N. Jonoska, Gh. Păun, G. Rozenberg, eds.), LNCS 2950, Springer, 2004, 338–352.

18. M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini: A polynomial complexity class in P systems using membrane division. In *Proceedings of the 5th Workshop on Descriptional Complexity of Formal Systems, DCFS 2003* (E. Csuhaj-Varjú, C. Kintala, D. Wotschke, Gy. Vaszyl, eds.), Budapest, 2003, 284–294.
19. Gh. Păun, M.J. Pérez-Jiménez, A. Riscos Núñez: Tissue P systems with cell division. In *Second Brainstorming Week on Membrane Computing* (Gh. Păun, A. Riscos-Núñez, A. Romero-Jiménez, F. Sancho-Caparrini, eds.), Sevilla, Spain, Feb 2-7, 2004, Dept. of Computer Sciences and Artificial Intelligence, Univ. of Sevilla Tech. Rep 01/2004, 380–386.
20. A. Păun, Gh. Păun: The power of communication: P systems with symport/antiport. *New Generation Computing*, 20, 3 (2002), 295–305.
21. Gh. Păun: Computing with membranes. *Journal of Computer and System Sciences*, 61 (2000), 108–143.
22. Gh. Păun, M.J. Pérez-Jiménez: Recent computing models inspired from biology: DNA and membrane computing. *Theoria*, 18, 46, (2003), 72–84.
23. V.J. Prakash: On the power of tissue P systems working in the maximal-one mode. In *Preproceedings of the Workshop on Membrane Computing* (A. Alhazov, C. Martín-Vide, Gh. Păun, eds.), Tarragona, July 17-22, 2003, 356-364
24. Gh. Păun: *Membrane Computing. An Introduction.* Springer, Berlin, 2002.
25. L.J. Stockmeyer: Planar 3-colorability is NP-complete. *SIGACT News*, 5, 3 (1973), 19–25.
26. A. Syropoulos: Mathematics of multisets. In *Multisets processing* (C.S. Calude et al., eds.), LNCS 2235, Springer, 2001, 347–358.
27. C. Zandron, C. Ferreti, G. Mauri: Solving NP-complete problems using P systems with active membranes. In *Unconventional Models of Computation, UMC'2K*, Springer, 2000, 289–301.
28. ISI web page: `http://esi-topics.com/erf/october2003.html`
29. P systems web page: `http://psystems.disco.unimib.it/`