Improving the Efficiency of Tissue P Systems with Cell Separation

Mario J. Pérez-Jiménez[1], Petr Sosík[2,3]

[1] Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla, 41012 Sevilla, Spain
marper@us.es
[2] Departamento de Inteligencia Artificial, Facultad de Informática,
Universidad Politécnica de Madrid, Campus de Montegancedo s/n,
Boadilla del Monte, 28660 Madrid, Spain
[3] Research Institute of the IT4Innovations Centre of Excellence,
Faculty of Philosophy and Science, Silesian University in Opava,
74601 Opava, Czech Republic
psosik@fi.upm.es

Summary. Cell fission process consists of the division of a cell into two new cells such that the contents of the initial cell is distributed between the newly created cells. This process is modelled by a new kind of *cell separation rules* in the framework of *Membrane Computing*. Specifically, in tissue-like membrane systems, cell separation rules have been considered joint with communication rules of the form symport/antiport. These models are able to create an exponential workspace, expressed in terms of the number of cells, in linear time. On the one hand, an efficient and uniform solution to the `SAT` problem by using cell separation and communication rules with length at most 8 has been recently given. On the other hand, only tractable problems can be efficiently solved by using cell separation and communication rules with length at most 1. Thus, in the framework of tissue P systems with cell separation, and assuming that $\mathbf{P} \neq \mathbf{NP}$, a first frontier between efficiency and non-efficiency is obtained when passing from communication rules with length 1 to communication rules with length at most 8.

In this paper we improve the previous result by showing that the `SAT` problem can be solved by a family of tissue P systems with cell separation in linear time, by using communication rules with length at most 3. Hence, we provide a new tractability borderline: passing from 1 to 3 amounts to passing from non–efficiency to efficiency, assuming that $\mathbf{P} \neq \mathbf{NP}$.

1 Introduction

Membrane Computing is a young branch of *Natural Computing* initiated by Gh. Păun in the end of 1998 [16]. It is inspired by the structure and functioning of living cell, as well as from the organization of cells in tissues, organs, and other higher order structures. The devices of this paradigm, called *P systems*, provide models for distributed, parallel and non-deterministic computing.

Membrane Computing has received an important attention from the scientific community since then, and many applications have been reported ([3], [21]). It was selected by the Institute for Scientific Information, USA, as a fast *Emerging Research Front in Computer Science*, and [19] was mentioned in [25] as a highly cited paper in October 2003.

Roughly speaking, the main ingredient of a membrane system is a cell-like *membrane structure* (a rooted tree), in the *compartments* of which one places *multisets* of *symbol-objects.* The objects evolve in a synchronous maximally parallel manner according to given *evolution rules*, also associated with the membranes (for introduction see [18] and for further bibliography see [26]).

Several different models of cell-like P systems have been successfully used to solve computationally hard problems efficiently, by trading space for time: an exponential workspace is created in polynomial time by using some kind of rules, and then massive parallelism is used to simultaneously check all the candidate solutions. Inspired by living cell, several ways for obtaining exponential workspace in polynomial time were proposed: membrane division (*mitosis*) [17], membrane creation (*autopoiesis*) [9], and membrane separation (*membrane fission*) [14]. These three ways have given rise to the following models: *P systems with active membranes*, *P systems with membrane creation*, and *P systems with membranes separation.*

A new type of P systems, the so-called *tissue P systems*, was considered in [12]. Instead of considering a hierarchical arrangement, membranes/cells are placed in the nodes of a virtual graph. This variant has two biological justifications (see [13]): 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, which were introduced to P systems in [19]. 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 definitions of tissue P systems [12, 13], several research lines have been developed and other variants have arisen (see, for example, [1, 2, 6, 10, 11, 24]). One of the most interesting variants of tissue P systems was presented in [20], where the definition of tissue P systems is combined with the one of P systems with active membranes, yielding *tissue P systems with cell division.* In this kind of models [20], there exists cell replication, that is, the two new cells generated by a division rule have exactly the same objects except for at most a pair of different objects.

In the biological phenomenon of fission, the contents of the two new cells evolved from a cell can be significantly different, and membrane separation inspired by this biological phenomenon in the framework of cell-like P systems was proved to be an efficient way to obtain exponential workspace in polynomial time

[14]. In [15], a new class of tissue P systems based on cell fission, called *tissue P systems with cell separation*, was presented. Its computational efficiency was investigated, and two important results were obtained: (a) only tractable problems can be efficiently solved by using cell separation and communication rules with length at most 1, and (b) an efficient (uniform) solution to the SAT problem by using cell separation and communication rules with length at most 8 was presented. Hence, in the framework of recognizer tissue P systems with cell separation, the length of the communication rules provide a borderline between efficiency and non-efficiency, that is, a frontier is there when we pass from length 1 to length 6, assuming that $\mathbf{P} \neq \mathbf{NP}$.

In this paper we present an improvement of the previous borderline of the tractability. Specifically, we propose a (uniform) family of tissue P systems with cell separation and communication rules with length at most 3 which solves the SAT problem in linear time. Hence, a new borderline is provided in this paper: passing from 1 to 3 amounts to passing from non–efficiency to efficiency, assuming that $\mathbf{P} \neq \mathbf{NP}$.

The paper is organized as follows: first, we recall some preliminaries, and then, the definition of tissue P systems with cell separation is given. Next, recognizer tissue P systems and computational complexity classes in this framework, are briefly described. In Section 5, an efficient (uniform) solution to the SAT problem by using cell separation and communication rules with length at most 3 is shown. Section 6 is devoted to present a detailed formal verification of the main result. Finally, conclusions and further works are presented.

2 Preliminaries

An *alphabet*, Σ, is a non–empty set whose elements are called *symbols*. An ordered finite sequence of symbols is a *string* o *word*. If u and v are strings over Σ, then so is their *concatenation* uv, obtained by juxtaposition, that is, writing u and v after one another. 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 all strings over an alphabet Σ is denoted by Σ^*. In algebraic terms, Σ^* is the free monoid generated by Σ under the operation of concatenation. Subsets, finite or infinite, of Σ^* are referred to as *languages* over Σ.

The *Parikh vector* associated with a string $u \in \Sigma^*$ with respect to the alphabet $\Sigma = \{a_1, \dots, a_r\}$ is $\Psi_\Sigma(u) = (|u|_{a_1}, \dots, |u|_{a_r})$, where $|u|_{a_i}$ denotes the number of ocurrences of the symbol a_i in the string u. This is called the *Parikh mapping* associated with Σ. Notice that in this definition the ordering of the symbols from Σ is relevant. If $\Sigma_1 = \{a_{i_1}, \dots, a_{i_s}\} \subseteq \Sigma$ then we define $\Psi_{\Sigma_1}(u) = (|u|_{a_{i_1}}, \dots, |u|_{a_{i_s}})$, for each $u \in \Sigma^*$.

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 $supp(m) = \{x \in A \mid f(x) > 0\}$. A multiset is empty (resp. finite) if its support is the empty set (resp. a finite set). If

$m = (A, f)$ is a finite multiset over A, and $supp(m) = \{a_1, \dots, a_k\}$ then it will be denoted as $m = \{a_1^{f(a_1)}, \dots, a_k^{f(a_k)}\}$. That is, superscripts indicate the multiplicity of each element, and if $f(x) = 0$ for $x \in A$, then the element x is omitted. A finite multiset $m = \{a_1^{f(a_1)}, \dots, a_k^{f(a_k)}\}$ can also be represented by the string $a_1^{f(a_1)} \dots a_k^{f(a_k)}$ over the alphabet $\{a_1, \dots, a_k\}$. Nevertheless, all permutations of this string precisely identify the same multiset m. Throughout this paper, we speak about "the finite multiset m" where m is a string, and meaning "the finite multiset represented by the string m".

If $m_1 = (A, f_1)$, $m_2 = (A, f_2)$ are multisets over A, then we define the union of m_1 and m_2 as $m_1 + m_2 = (A, g)$, where $g = f_1 + f_2$.

For any sets A and B the *relative complement* $A \setminus B$ of B in A is defined as follows:

$$A \setminus B = \{x \in A \mid \ x \notin B\}$$

In what follows, we assume the reader is already familiar with the basic notions and the terminology of P systems. For details, see [18].

3 Tissue P Systems with Cell Separation

Let us recall that the model of *tissue P systems with cell separation* is based on the cell-like model of P systems with membranes separation [14]. The biological inspiration is the following: alive tissues are not *static* network of cells, since new cells are generated by membrane fission in a natural way. In these models, the cells are not polarized; the two cells obtained by separation have the same labels as the original cell, and if a cell is separated, its interaction with other cells or with the environment is blocked during the separation process. In some sense, this means that while a cell is separating it closes its communication channels.

Definition 3.1 *A tissue P system with cell separation of degree $q \geq 1$ is a tuple*

$$\Pi = (\Gamma, \Gamma_1, \Gamma_2, \mathcal{E}, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}, i_{out}),$$

where:

1. *Γ is a finite* alphabet *whose elements are called* objects*;*
2. *$\{\Gamma_1, \Gamma_2\}$ is a partition of Γ, that is, $\Gamma = \Gamma_1 \cup \Gamma_2$, $\Gamma_1, \Gamma_2 \neq \emptyset$, $\Gamma_1 \cap \Gamma_2 = \emptyset$;*
3. *$\mathcal{E} \subseteq \Gamma$ is a finite alphabet representing the set of objects initially in the environment of the system, and 0 is the label of the environment (the environment is not properly a cell of the system); let us assume that objects in the environment appear in arbitrary copies each;*
4. *$\mathcal{M}_1, \dots, \mathcal{M}_q$ are strings over Γ, representing the finite multisets of objects placed in the q cells of the system at the beginning of the computation; $1, 2, \cdots, q$ are labels which identify the cells of the system;*
5. *$\mathcal{R}$ is a finite set of rules of the following forms:*

   (a) Communication rules*: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \ldots, q\}, i \neq j$, $u, v \in \Gamma^*$, $|uv| > 0$. When applying a rule $(i, u/v, j)$, the objects of the multiset represented by u are sent from region i to region j and, simultaneously, the objects of the multiset v are sent from region j to region i;*
   (b) Separation rules*: $[a]_i \to [\Gamma_1]_i[\Gamma_2]_i$, where $i \in \{1, 2, \ldots, q\}$ and $a \in \Gamma$, and $i \neq i_{out}$. In reaction with an object a, the cell i is separated into two cells with the same label; at the same time, object a is consumed; the objects from Γ_1 are placed in the first cell, those from Γ_2 are placed in the second cell; the output cell i_{out} cannot be separated;*
6. *$i_{out} \in \{0, 1, 2, \ldots, q\}$ is the output cell.*

A communication rule $(i, u/v, j)$ is called a *symport rule* if $u = \lambda$ or $v = \lambda$. A symport rule $(i, u/\lambda, j)$, with $i \neq 0, j \neq 0$, provides a virtual arc from cell i to cell j. A communication rule $(i, u/v, j)$ is called an *antiport rule* if $u \neq \lambda$ and $v \neq \lambda$. An antiport rule $(i, u/v, j)$, with $i \neq 0, j \neq 0$, provides two arcs: one from cell i to cell j and another one from cell j to cell i. Thus, every tissue P systems has an underlying directed graph whose nodes are the cells of the system and the arcs are obtained from communication rules. In this context, the environment can be considered as a virtual node of the graph such that their connections are defined by the communication rules of the form $(i, u/v, j)$, with $i = 0$ or $j = 0$.

The length of the communication rule $(i, u/v, j)$ is defined as $|u| + |v|$.

The rules of a system like the above one are used in the non-deterministic maximally parallel manner as customary in Membrane Computing. At each step, all cells which can evolve must evolve in a maximally parallel way (at each step we apply a multiset of rules which is maximal, no further rule can be added being applicable). This way of applying rules has only one restriction: when a cell is separated, the separation rule is the only one which is applied for that cell at that step; thus, the objects inside that cell do not evolve by means of communication rules. The new cells resulting from separation could participate in the interaction with other cells or the environment by means of communication rules at the next step – providing that they are not separated once again. The label of a cell precisely identify the rules which can be applied to it.

An *instanstaneous description* or a *configuration* at any instant of a tissue P system with cell separation is described by all multisets of objects over Γ associated with all the cells present in the system, and the multiset of objects over $\Gamma - \mathcal{E}$ associated with the environment at that moment. Bearing in mind the objects from $\mathcal{E}$ have infinite copies in the environment, they are not properly changed along the computation. The *initial configuration* is $(\mathcal{M}_1, \cdots, \mathcal{M}_q; \emptyset)$. A configuration is a *halting configuration* if no rule of the system is applicable to it.

Let us fix a tissue P system with cell separation Π. We say that configuration C_1 yields configuration C_2 in one *transition step*, denoted $C_1 \Rightarrow_\Pi C_2$, if we can pass from C_1 to C_2 by applying the rules from $\mathcal{R}$ following the previous remarks. A *computation* of Π is a (finite or infinite) sequence of configurations such that:

1. the first term of the sequence is the initial configuration of the system;

2. each non-initial configuration of the sequence is obtained from the previous configuration by aplying rules of the system in a maximally parallel manner with the restrictions previously mentioned; and
3. if the sequence is finite (called *halting computation*) then the last term of the sequence is a halting configuration.

All computations start from an initial configuration and proceed as stated above; only halting computations give a result, which is encoded by the objects present in the output cell i_{out} in the halting configuration.

We denote by $\mathbf{Comp}(\Pi)$ the set of computations of the tissue P system Π. If $\mathcal{C} = \{\mathcal{C}_i\}_{i<r+1}$ of Π $(r \in \mathbf{N})$ is a halting computation, then the length of $\mathcal{C}$ is r, that is, the number of non-initial configurations which appear in the finite sequence $\mathcal{C}$. We denote it by $|\mathcal{C}|$. We also denote by $\mathcal{C}_i(j)$ the contents of cell j at the configuration $\mathcal{C}_i$.

In the framework of tissue P systems with symport/antiport rules, it is interesting to highlight some differences between a division rule of the type $[a]_i \rightarrow [b]_i\ [c]_i$, and a separation rule of the type $[a]_i \rightarrow [\Gamma_1]_i\ [\Gamma_2]_i$:

1. The object a triggers both rules and it is consumed. Nevertheless,
   - ⋆ *Division rule*: Produces an object (b or c) in each new cell.
   - ⋆ *Separation rule*: Does not produce any new object in new cells.
2. The remaining objects in cell i:
   - ⋆ *Division rule*: Are <u>replicated</u> in each new cell.
   - ⋆ *Separation rule*: Are <u>distributed</u> between the new cells, according to sets Γ_1 and Γ_2.
3. If there is n objects in the cell i where the rule is applied:
   - ⋆ *Division rule*: The total number of objects in the cells created is $2n$, each of them contains n objects.
   - ⋆ *Separation rule*: The total number of objects in the cells created is $n - 1$.
4. If the rules are consecutively applied during k transtition steps in a cell i which contains n objects:
   - ⋆ *Division rule*: 2^k new cells are created, and the total number of objects is $n \cdot 2^k$.
   - ⋆ *Separation rule*: $2 \cdot k$ new cells are created, and the total number of objects is $n - k$.

Hence, division and separation rules have the ability to produce an exponential number of new cells in linear time, but only division rules are able to simultaneously produce an exponential number of objects.

3.1 Recognizer Tissue P Systems with Cell Separation

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. Many abstract problems are not decision problems, for example, in *combinatorial optimization problems* some value must be optimized (minimized or maximized). In order to deal with such problems, they can be transformed into roughly equivalent decision problems by supplying a target/threshold value for the quantity to be optimized, and then asking whether this value can be attained.

A natural correspondence between decision problems and languages over a finite alphabet, can be established as follows. Given a decision problem $X = (I_X, \theta_X)$, its associated language is $L_X = \{w \in I_X : \theta_X(w) = 1\}$. Conversely, given a language L over an alphabet Σ, its associated decision problem is $X_L = (I_{X_L}, \theta_{X_L})$, where $I_{X_L} = \Sigma^*$, and $\theta_{X_L} = \{(x,1) : x \in L\} \cup \{(x,0) : x \notin L\}$. The solvability of decision problems is defined through the recognition of the languages associated with them, by using languages recognizer devices.

In order to study the computational efficiency of membrane systems, the notions from classical *computational complexity theory* are adapted for Membrane Computing, and a special class of cell-like P systems is introduced in [23]: *recognizer P systems* (called *accepting P systems* in a previous paper [22]). For tissue P systems, with the same idea as recognizer cell-like P systems, *recognizer tissue P systems* is introduced in [20].

Definition 3.2 *A* recognizer tissue P system with cell separation *of degree* $q \geq 1$ *is a tuple*

$$\Pi = (\Gamma, \Gamma_1, \Gamma_2, \Sigma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$$

where:

1. $(\Gamma, \Gamma_1, \Gamma_2, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$ *is a tissue P system with cell separation of degree* $q \geq 1$ *(as defined in the previous section).*
2. *The working alphabet* Γ *has two distinguished objects* `yes` *and* `no` *being, at least, one copy of them present in some initial multisets* $\mathcal{M}_1, \ldots, \mathcal{M}_q$, *but none of them are present in* $\mathcal{E}$.
3. Σ *is an (input) alphabet strictly contained in* Γ, *and* $\mathcal{E} \subseteq \Gamma \setminus \Sigma$.
4. $\mathcal{M}_1, \ldots, \mathcal{M}_q$ *are strings over* $\Gamma \setminus \Sigma$;
5. $i_{in} \in \{1, \ldots, q\}$ *is the input cell.*
6. *The output region* i_{out} *is the environment.*
7. *All computations halt.*
8. *If* $\mathcal{C}$ *is a computation of* Π, *then either object* `yes` *or object* `no` *(but not both) must have been released into the environment, and only at the last step of the computation.*

For each $w \in \Sigma^*$, the *computation of the system* Π *with input* $w \in \Sigma^*$ starts from the configuration of the form $(\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_{i_{in}} + w, \ldots, \mathcal{M}_q; \emptyset)$, that is, the input multiset w has been added to the contents of the input cell i_{in}. Therefore, we have an initial configuration associated with each input multiset w (over the input alphabet Σ) in this kind of systems.

Given a recognizer tissue P system with cell division, and a halting computation $\mathcal{C} = \{C_i\}_{i<r+1}$ of Π $(r \in \mathbf{N})$, we define the result of $\mathcal{C}$ as follows:

$$Output(\mathcal{C}) = \begin{cases} \texttt{yes}, & \text{if } \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{r,0}) = (1,0) \wedge \\ & \quad \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{i,0}) = (0,0) \text{ for } i = 0, \dots, r-1 \\ \texttt{no}, & \text{if } \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{r,0}) = (0,1) \wedge \\ & \quad \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{i,0}) = (0,0) \text{ for } i = 0, \dots, r-1 \end{cases}$$

where Ψ is the Parikh function, and $M_{i,0}$ is the multiset over $\Gamma \setminus \mathcal{E}$ associated with the environment at configuration C_i, in particular, $M_{r,0}$ is the multiset over $\Gamma \setminus \mathcal{E}$ associated with the environment at the halting configuration C_r.

We say that a computation $\mathcal{C}$ is an *accepting computation* (respectively, *rejecting computation*) if $Output(\mathcal{C}) = \texttt{yes}$ (respectively, $Output(\mathcal{C}) = \texttt{no}$), that is, if object `yes` (respectively, object `no`) appears in the environment associated with the corresponding halting configuration of $\mathcal{C}$, and neither object `yes` nor `no` appears in the environment associated with any non–halting configuration of $\mathcal{C}$.

For each natural number $k \geq 1$, we denote by $\mathbf{TSC}(k)$ the class of recognizer tissue P systems with cell separation and communication rules of length at most k. We denote by **TSC** the class of recognizer tissue P systems with cell separation and without restriction on the length of communication rules. Obviously, $\mathbf{TSC}(k) \subseteq \mathbf{TSC}$ for all $k \geq 1$.

3.2 Polynomial Complexity Classes of Tissue P systems with Cell Separation

Next, we define what means solving a decision problem in the framework of tissue P systems efficiently and in a uniform way. Bearing in mind that they provide devices with a finite description, a numerable family of tissue P systems will be necessary in order to solve a decision problem.

Definition 1. *We say that a decision problem $X = (I_X, \theta_X)$ is solvable in a uniform way and polynomial time by a family $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ of recognizer tissue P systems with cell separation if the following holds:*

1. *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 $n \in \mathbb{N}$.*
2. *There exists a pair (cod, s) of polynomial-time computable functions over I_X such that:*
   (a) *for each instance $u \in I_X$, $s(u)$ is a natural number and $cod(u)$ is an input multiset of the system $\Pi(s(u))$;*
   (b) *for each $n \in \mathbb{N}$, $s^{-1}(n)$ is a finite set;*
   (c) *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 it performs at most $p(|u|)$ steps;
   (d) *the family $\mathbf{\Pi}$ is* sound *with regard to (X, cod, s), that is, for each $u \in I_X$, if* <u>*there exists*</u> *an accepting computation of $\Pi(s(u))$ with input $cod(u)$, then $\theta_X(u) = 1$;*

(e) *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.*

From the soundness and completeness conditions above we deduce that every P system $\Pi(n)$ is *confluent*, in the following sense: every computation of a system with the *same* input multiset must always give the *same* answer.

Let $\mathbf{R}$ be a class of recognizer tissue P systems. We denote by $\mathbf{PMC_R}$ the set of all decision problems which can be solved in a uniform way and polynomial time by means of families of systems from $\mathbf{R}$.

4 Computational Efficiency of Tissue P Systems with Cell Separation

It is well known that tissue P systems with cell division are able to solve computationally hard problems efficiently. Specifically, **NP**–complete problems have been solved in linear time [5] by using families of tissue P systems with cell division and communication rules of length at most 3.

In [15] two important results related to the computational efficiency of tissue P systems with cell separation were obtained. On the one hand, only tractable problems can be efficiently solved by using families of tissue P systems with cell separation and communication rules of length 1, that is, $\mathbf{P} = \mathbf{PMC}_{TSC(1)}$. On the other hand, an efficient solution to the SAT problem has been given by means of a uniform family of tissue P systems with cell separation and communication rules of length at most 8, that is, $\mathtt{SAT} \in \mathbf{PMC}_{TSC(8)}$, hence $\mathbf{NP} \ \cup \ \mathbf{co-NP} \subseteq \mathbf{PMC}_{TSC(8)}$. Therefore, passing the maximum length of communication rules of the systems from 1 to 6 amounts to passing from non–efficiency to efficiency, assuming that $\mathbf{P} \neq \mathbf{NP}$. An interesting challenge is to refine that efficiency borderline, that is, to provide new efficient solutions to computationally hard problems by means of tissue P systems with cell separation by using communication with length under 6.

In the next Section, we improve the result from [15] by giving a family of tissue P systems with cell separation and communication rules of length at most 3 which solves the SAT problem in linear time.

5 Solving the SAT Problem by using TSC(3)

Let us recall that the SAT problem is the following: *given a boolean formula in conjunctive normal form (CNF), to determine whether or not there exists an assignment to its variables on which it evaluates true.* This is a well known **NP**-complete problem [7].

In this Section, we propose a solution following a brute force algorithm implemented in the framework of recognizer tissue P systems with cell separation. The solution consists of the following stages:

- *Generation Stage*: All truth assignments associated with the input formula are produced by using cell separation in an adequate way.
- *Checking Stage*: In each cell, it is checked whether or not the formula is satisfiable by the truth assignment encoded by that cell.
- *Output Stage*: The system sends to the environment the right answer according to the results of the previous stage.

Let us consider the polynomial–time computable function (the *pair function*)

$$\langle m,n\rangle = ((m+n)(m+n+1)/2)+m$$

which is also a primitive recursive and bijective function from $\mathbb{N}\times\mathbb{N}$ to $\mathbb{N}$.

Next, we define a family $\mathbf{\Pi} = \{\Pi(t) :\ t\in\mathbb{N}\}$ of recognizer tissue P system with cell separation from $\mathbf{TSC}(3)$, such that each system $\Pi(t)$ will process all instances φ of `SAT` with n variables and m clauses, where $t=\langle m,n\rangle$, provided that the appropriate input multiset $cod(\varphi)$ is supplied to the system.

For each $(m,n)\in\mathbb{N}\times\mathbb{N}$, we consider the recognizer tissue P system with cell separation from $\mathbf{TSC}(3)$,

$$\Pi(\langle m,n\rangle) = (\Gamma,\Gamma_1,\Gamma_2,\Sigma,\mathcal{E},\mathcal{M}_1,\mathcal{M}_2,\mathcal{M}_3,\mathcal{R},i_{in},i_{out})$$

defined as follows:

- The input alphabet is

$$\Sigma = \{x_{i,j},\overline{x}_{i,j} :\ 1\le i\le n\,,\ 1\le j\le m\}$$

- The working alphabet is $\Gamma = \Sigma\ \cup\ \Gamma_1\ \cup\Gamma_2$, where:

$$\begin{aligned}\Gamma_1 = \{&A_i,B_i :\ 1\le i\le n+1\}\ \cup\ \{a_i,b_i,T_i,F_i,y_i,v_i,w_i :\ 1\le i\le n\}\ \cup\\ &\{c_i,t_i,f_i,s_i,z_i :\ 1\le i\le n-1\}\ \cup\ \{E_j :\ 1\le j\le m+1\}\ \cup\\ &\{\alpha_i :\ 0\le i\le 3n+2m+1\}\ \cup\ \{\beta_i :\ 0\le i\le 3n+2m+2\}\ \cup\\ &\{q_{i,j},r_{i,j},u_{i,j} :\ 1\le i,j\le n-1\}\ \cup\\ &\{x_{i,j},\overline{x}_{i,j},e_{i,j},\overline{e}_{i,j} :\ 1\le i\le n\,,\ 1\le j\le m\}\ \cup\\ &\{d_{i,j,k},\overline{d}_{i,j,k} :\ 1\le i\le n\,,\ 1\le j\le m\,,\ \ 1\le k\le n\}\ \cup\ \{q_0,S,\mathtt{yes},\mathtt{no}\}\end{aligned}$$

$$\Gamma_2 = \{A'_i,B'_i :\ 1\le i\le n+1\}\cup\{a'_i,b'_i,T'_i,F'_i :\ 1\le i\le n\}$$

- The alphabet of the environment is:

$$\begin{aligned}\mathcal{E} = \{S\}\ \cup\ &\{A_i,B_i,A'_i,B'_i :\ 2\le i\le n+1\}\ \cup\ \{T_i,F_i,F'_i,y_i,w_i :\ 1\le i\le n\}\ \cup\\ &\{a_i,a'_i,b_i,b'_i,v_i :\ 2\le i\le n\}\ \cup\ \{T'_i,c_i,t_i,f_i,s_i,z_i :\ 1\le i\le n-1\}\ \cup\\ &\{E_j :\ 1\le j\le m+1\}\ \cup\ \{\alpha_i :\ 1\le i\le 3n+2m+1\}\ \cup\\ &\{\beta_i :\ 1\le i\le 3n+2m+2\}\ \cup\\ &\{q_{i,j},r_{i,j},u_{i,j} :\ 1\le i\le n-1\,,\ 2\le j\le n-1\}\ \cup\\ &\{e_{i,j},\overline{e}_{i,j} :\ 1\le i\le n\,,\ 1\le j\le m\}\ \cup\\ &\{d_{i,j,k},\overline{d}_{i,j,k} :\ 1\le i,k\le n\,,\ 1\le j\le m\}\end{aligned}$$

- Initial multisets:

$\mathcal{M}_1 = A_1\, B_1$
$\mathcal{M}_2 = a_1\, a'_1\, b_1\, b'_1\, v_1\, q_{1,1}\, \alpha_0\, \texttt{yes}\, \texttt{no}$
$\mathcal{M}_3 = \beta_0$

- The set R of rules consists of the following rules:

(1) $(1\,,\, A_i \,/\, a_i\, a'_i\,,\, 2)$, for $1 \le i \le n$, and $(1\,,\, A_{n+1} \,/\, E_1\,,\, 2)$.
(2) $(1\,,\, A'_i \,/\, a_i\, a'_i\,,\, 2)$, for $1 \le i \le n$, and $(1\,,\, A'_{n+1} \,/\, E_1\,,\, 2)$.
(3) $(1\,,\, B_i \,/\, b_i\, b'_i\,,\, 2)$, for $1 \le i \le n$.
(4) $(1\,,\, B'_i \,/\, b_i\, b'_i\,,\, 2)$, for $1 \le i \le n$.
(5) $(1\,,\, T_i \,/\, t_i\,,\, 2)$, for $1 \le i \le n-1$.
(6) $(1\,,\, T'_i \,/\, t_i\,,\, 2)$, for $1 \le i \le n-1$.
(7) $(1\,,\, F_i \,/\, f_i\,,\, 2)$, for $1 \le i \le n-1$.
(8) $(1\,,\, F'_i \,/\, f_i\,,\, 2)$, for $1 \le i \le n-1$.
(9) $(1\,,\, t_i \,/\, T_i\, T'_i,\, 0)$, for $1 \le i \le n-1$.
(10) $(1\,,\, f_i \,/\, F_i\, F'_i,\, 0)$, for $1 \le i \le n-1$.
(11) $(1\,,\, b_i \,/\, B_{i+1}\, S\,,\, 0)$, for $1 \le i \le n$, and $(1\,,\, B_{n+1} \,/\, \lambda\,,\, 0)$.
(12) $(1\,,\, b'_i \,/\, B'_{i+1}\,,\, 0)$, for $1 \le i \le n$, and $(1\,,\, B'_{n+1} \,/\, \lambda\,,\, 0)$.
(13) $(1\,,\, a_i \,/\, T_i\, A_{i+1}\,,\, 0)$, for $1 \le i \le n$.
(14) $(1\,,\, a'_i \,/\, F'_i\, A'_{i+1}\,,\, 0)$, for $1 \le i \le n$.
(15) $(2\,,\, A_i \,/\, c_i\,,\, 0)$, for $1 \le i \le n-1$, and $(2\,,\, A_i \,/\, \lambda\,,\, 0)$, for $n \le i \le n+1$.
(16) $(2\,,\, A'_i \,/\, c_i\,,\, 0)$, for $1 \le i \le n-1$, and $(2\,,\, A'_i \,/\, \lambda\,,\, 0)$, for $n \le i \le n+1$.
(17) $(2\,,\, B_i \,/\, c_i\,,\, 0)$, for $1 \le i \le n-1$, and $(2\,,\, B_n \,/\, \lambda\,,\, 0)$.
(18) $(2\,,\, B'_i \,/\, c_i\,,\, 0)$, for $1 \le i \le n-1$, and $(2\,,\, B'_n \,/\, \lambda\,,\, 0)$.
(19) $(2\,,\, c_i \,/\, b_{i+1}\, b'_{i+1}\,,\, 0)$, for $1 \le i \le n-1$.
(20) $(2\,,\, v_i \,/\, y_i^2\,,\, 0)$, for $1 \le i \le n$.
(21) $(2\,,\, y_i \,/\, z_i\, w_i\,,\, 0)$, for $1 \le i \le n-1$, and $(2\,,\, y_n \,/\, w_n\,,\, 0)$.
(22) $(2\,,\, z_i \,/\, v_{i+1}\,,\, 0)$, for $1 \le i \le n-1$.
(23) $(2\,,\, w_i \,/\, a_{i+1}\, a'_{i+1}\,,\, 0)$, for $1 \le i \le n-1$, and $(2\,,\, w_n \,/\, E_1\,,\, 0)$.
(24) $(2\,,\, q_{1,1} \,/\, r_{1,1}\,,\, 0)$.
(25) $(2\,,\, q_{i,j} \,/\, r_{i,j}^2\,,\, 0)$, for $1 \le i \le n-1,\ 2 \le j \le n-1$.
(26) $(2\,,\, r_{i,j} \,/\, s_i\, u_{i,j}\,,\, 0)$, for $1 \le i, j \le n-1$.
(27) $(2\,,\, s_i \,/\, t_i\, f_i\,,\, 0)$, for $1 \le i \le n-1$.
(28) $(2\,,\, u_{1,j} \,/\, q_{1,j+1}\, q_{2,j+1}\,,\, 0)$, for $1 \le j \le n-2$.
(29) $(2\,,\, u_{i,j} \,/\, q_{i+1,j+1}\,,\, 0)$, for $2 \le i, j \le n-2$.
(30) $(2\,,\, u_{i,n-1} \,/\, \lambda\,,\, 0)$, for $1 \le i \le n-1$.
(31) $(2\,,\, T_i \,/\, \lambda\,,\, 0)$, for $1 \le i \le n-1$.
(32) $(2\,,\, T'_i \,/\, \lambda\,,\, 0)$, for $1 \le i \le n-1$.
(33) $(2\,,\, F_i \,/\, \lambda\,,\, 0)$, for $1 \le i \le n-1$.
(34) $(2\,,\, F'_i \,/\, \lambda\,,\, 0)$, for $1 \le i \le n-1$.
(35) $[\, S\,]_1 \longrightarrow [\, \varGamma_1\,]_1\, [\, \varGamma_2\,]_1$
(36) $(2\,,\, \alpha_i \,/\, \alpha_{i+1}\,,\, 0)$, for $0 \le i \le 3n+2m$.
(37) $(3\,,\, \beta_i \,/\, \beta_{i+1}\,,\, 0)$, for $0 \le i \le 3n+2m+1$.
(38) $(3\,,\, x_{i,j} \,/\, d_{i,j,1}^2\,,\, 0)$, $(3\,,\, \bar{x}_{i,j} \,/\, \bar{d}_{i,j,1}^2\,,\, 0)$, for $1 \le i \le n$, $1 \le j \le m$

(39) $(3\,,\, d_{i,j,k}\,/\,d^2_{i,j,k+1}\,,\,0)$, $(3\,,\,\bar{d}_{i,j,k}\,/\,\bar{d}^2_{i,j,k+1}\,,\,0)$, for $1 \le i \le n$, $1 \le j \le m$, $1 \le k \le n-1$.
(40) $(3\,,\,d_{i,j,n}\,/\,e_{i,j},0)$, $(3\,,\,\bar{d}_{i,j,n}\,/\,\bar{e}_{i,j}\,,\,0)$, for $1 \le i \le n$, $1 \le j \le m$.
(41) $(1\,,\,T_iE_j\,/\,e_{i,j}\,,\,3)$, $(1\,,\,F_iE_j\,/\,\bar{e}_{i,j}\,,\,3)$, $(1\,,\,T'_iE_j\,/\,e_{i,j}\,,\,3)$, $(1\,,\,F'_iE_j\,/\,\bar{e}_{i,j}\,,\,3)$, for $1 \le i \le n$, $1 \le j \le m$.
(42) $(1\,,\,e_{i,j}\,/\,T_iE_{j+1}\,,\,0)$, $(1\,,\,\bar{e}_{i,j}\,/\,F_iE_{j+1}\,,\,0)$, for $1 \le i \le n$, $1 \le j \le m-1$.
(43) $(1\,,\,e_{i,m}\,/\,E_{m+1}\,,\,0)$, $(1\,,\,\bar{e}_{i,m}\,/\,E_{m+1}\,,\,0)$, for $1 \le i \le n$.
(44) $(3, T_i\,/\,\lambda\,,0)$, $(3\,,\,F_i\,/\,\lambda\,,\,0)$, $(3\,,\,T'_i\,/\,\lambda\,,\,0)$, $(3\,,\,F'_i\,/\,\lambda\,,\,0)$, for $1 \le i \le n$.
(45) $(3\,,\,E_j\,/\,\lambda\,,\,0)$, for $1 \le j \le m$.
(46) $(1\,,\,E_{m+1}\,/\,\texttt{yes}\;\alpha_{3n+1+2m}\,,\,2)$.
(47) $(1\,,\,\texttt{yes}\,/\,\beta_{3n+1+2m+1}\,,\,3)$.
(48) $(2\,,\,\alpha_{3n+1+2m}\,/\,\beta_{3n+1+2m+1}\,,\,3)$.
(49) $(2\,,\,\texttt{no}\;\beta_{3n+1+2m+1}\,/\,\lambda\,,\,0)$.
(50) $(3\,,\,\texttt{yes}\,/\,\lambda\,,\,0)$.

- The input cell is $i_{in} = 3$.
- The output cell is the environment, $i_{out} = 0$.

5.1 An Overview of the Computation

A family of recognizer tissue P systems with cell separation is constructed above. For an instance of the SAT problem $\varphi = C_1 \wedge \cdots \wedge C_m$, consisting of m clauses $C_j = l_{j,1} \vee \cdots \vee l_{j,r_j}$, $1 \le j \le m$, where $Var(\varphi) = \{x_1, \cdots, x_n\}$, $l_{j,k} \in \{x_i, \neg x_i \mid 1 \le i \le n\}$, $1 \le j \le m, 1 \le k \le r_j$. Let us assume that the number of variables, n, and the number of clauses, m, of the input formula φ, are greater or equal to 2.

The *size* mapping on the set of instances is defined as $s(\varphi) = \langle m, n \rangle$, and the encoding of the instance is the multiset

$$cod(\varphi) = \{x_{i,j} :\ x_i \in C_j\} \ \cup\ \{\overline{x}_{i,j} :\ \neg x_i \in C_j\}$$

That is, $x_{i,j}$ (respectively, $\overline{x}_{i,j}$) denotes variable x_i (respectively, $\neg x_i$) belongs to clause C_j. Then the formula φ will be processed by the system $\Pi(s(\varphi))$ with input multiset $cod(\varphi)$.

Next, we informally describe how system $\Pi(s(\varphi))$ with input multiset $cod(\varphi)$ works, in order to process the instance φ of the SAT problem.

At the initial configuration we have objects A_1, B_1 in cell 1, objects $a_1, a'_1, b_1, b'_1, v_1, q_{1,1}, \alpha_0, \texttt{yes}, \texttt{no}$ in cell 2, and $cod(\varphi), \beta_0$ in cell 3.

Let us start with the **generation stage**. This stage spends $3n+1$ steps and has, basically, two parallel processes. On the one hand, n loops are executed, each loop spends 3 steps involving cells 1 and 2. After the loops are finished, an additional step goes on. On the other hand, in cell 3 there is a counter β that evolves from β_0 to β_{3n+1} by applying rules of the type (37), and $cod(\varphi)$ produces $((cod(\varphi))_e^{2^n}$ after the $3n+1$ steps at this stage.

At the first step of the i–th loop ($0 \le i \le n$) involving cells 1 and 2, objects

$$A_{i+1}, A'_{i+1}, B_{i+1}, B'_{i+1}, T_j, T'_j, F_j, F'_j$$

in cell 1 exchange objects

$$a_{i+1}a'_{i+1}, a_{i+1}a'_{i+1}, b_{i+1}b'_{i+1}, b_{i+1}b'_{i+1}, t_j, t_j, f_j, f_j$$

with cell 2, where also v_{i+1} produces y_{i+1}^2, and $q_{1,i+1}, \dots q_{i+1,i+1}$ ($q_{1,1}$ at step 1) produce objects $r_{1,i+1}^2, \dots r_{i+1,i+1}^2$ ($r_{1,1}$ at step 1).

At the second step of the i–th loop ($0 \leq i \leq n$), objects

$$a_{i+1}, a'_{i+1}, b_{i+1}, b'_{i+1}, t_j, f_j$$

in cells 1 produce objects

$$T_{i+1}A_{i+2}, F'_{i+1}A'_{i+2}, B_{i+2}S, B'_{i+2}, T_jT'_j, F_jF'_j$$

according to the rules (9), (10), (11), (12), (13), (14). Simultaneously, at this step objects

$$A_{i+1}, A'_{i+1}, B_{i+1}, B'_{i+1}, T_j, T'_j, F_j, F'_j, y_{i+1}, r_{1,i+1}, \dots r_{i+1,i+1}$$

in cell 2 produce objects

$$c_{i+1}, c_{i+1}, c_{i+1}, c_{i+1}, \lambda, \lambda, \lambda, \lambda, \lambda, z_{i+1}w_{i+1}, s_1u_{1,i+1} \dots s_{i+1}u_{i+1,i+1}$$

respectively, according to the rules (15), (16), (17), (18), (21), (26), (31), (32), (33), (34).

At the third step of the i–th loop ($1 \leq i \leq n-1$), object S triggers the separation of objects of cells 1 in two new cells 1 by applying the separation rule (35), according to Γ_1 (objects without primes) and Γ_2 (objects with primes). At this step, objects

$$c_{i+1}, z_{i+1}, w_{i+1}, s_1, \dots, s_{i+1}, u_{1,i+1}, \dots, u_{i+1,i+1}$$

in cell 2 produce objects

$$b_{i+2}b'_{i+2}, v_{i+2}, a_{i+2}a'_{i+2}, f_1t_1, \dots, f_{i+1}t_{i+1}, q_{1,i+2} \dots q_{i+1,i+2}, q_{i+2,i+2}$$

according to the rules (19), (22), (23), (27), (29), respectively.

After $3(n-1)$ transition steps, we have

(a) 2^{n-1} cells 1 such that 2^{n-2} cells contain objects T_{n-1}, A_n, B_n and a different truth assignment of $\sigma_{n-2,j}$ of the set $\{x_1, \dots, x_{n-2}\}$, and 2^{n-2} cells contain objects F'_{n-1}, A'_n, B'_n and a different truth assignment of $\tau_{n-2,j}$ of the set $\{x_1, \dots, x_{n-2}\}$.

(b) A cell 2 that contains objects

$$a_n^{2^{n-1}}, a_n'^{2^{n-1}}, b_n^{2^{n-1}}, b_n'^{2^{n-1}}, v_n^{2^{n-1}}, f_1^{2^{n-2}}, t_1^{2^{n-2}}, \dots, f_{n-1}^{2^{n-2}}, t_{n-1}^{2^{n-2}}$$

(c) A cell 3 which contains object $\beta_{3(n-1)}$ and $(cod(\varphi))_e^{2^n}$.

By applying rules (1), (2), (3), (4), (5), (6), (7), (8), (20), (36), and (37) at step $3n-2$, and rules (9), (10), (11), (12), (13), (14), (15), (16), (17, (18), (31), (32), (33), (34),(36), and (37) at step $3n-1$, and rules $(1, B_{n+1}/\lambda, 0)$, $(1, B'_{n+1}/\lambda, 0)$ $(2, w_n/E_1, 0)$, (35), (36), and (37) at step $3n$, we reach the following configuration $\mathcal{C}_{3n+1}$:

- There are 2^n cells 1 which contain object E_1 and each of them encodes a different truth assignment of the set $\{x_1, \ldots, x_n\}$.
- There is a cell 2 which contains objects $A_{n+1}^{2^{n+1}}$, $A'^{2^{n+1}}_{n+1}$, α_{3n+1} , $\texttt{yes}$, $\texttt{no}$.
- There is a cell 3 which contains object β_{3n+1} and $(cod(\varphi))_e^{2^n}$.

In this way, after the $(3n+1)$–th step the generation stage finishes and the **checking stage** starts. This stage spends $2m$ steps and consists of m loops each of them spending 2 steps.

At the first step of the j–th loop ($1 \leq j \leq m$), objects $e_{i,j}$ and $\overline{e}_{i,j}$ from cell 3 are traded for objects E_j from cell 1, in the case that cell 1 encodes a truth assignment making clauses $C_1, \ldots, C_j$ true. Simultaneously, in cell 2 counter α continue evolving and objects yes and no remain unchanged. In cell 3, counter β continue evolving, and object E_j appears k_j times, where k_j is the number of cells labelled by 2 encoding a truth assignment making clauses $C_1, \ldots, C_j$ true.

At the second step of the j–th loop ($1 \leq j \leq m$), rules (41) produce objects T_i, E_{j+1} in each cell 1 encoding a truth assignment making clauses $C_1, \ldots, C_j$ true. Simultaneously, in cell 2 counter α continue evolving and objects yes and no remain unchanged. In cell 3, counter β, and objects E_{j+1} are removed by applying rule (5).

At the end of the checking stage, there are 2^n cells labelled by 1 at configuration $\mathcal{C}_{(3n+1)+2m}$, and the formula φ is satisfiable if and only if there is, at least, one of such cell which contains object E_{m+1}. Also, there is a cell labelled by 2 which contains objects $\texttt{yes}, \texttt{no}, \alpha_{(3n+1)+2m}$, and a cell labelled by 3 which contains object $\beta_{(3n+1)+2m}$ and some irrelevant objects of the type $e_{i,j'}, \overline{e}_{i,j'}$ with $1 \leq j' \leq m$. Irrelevant objects are those which remain unchanged at the following computation steps and do not take part in the application of any rule of the system.

The **output stage** starts at the $((3n+1)+2m+1)$-th step, and spends 3 steps.

– *Affirmative answer* : If a truth assignment encoded by a cell 1 makes the formula φ true, then an object E_{m+1} appears in that cell. By applying rule (46) one (and only one) object E_{m+1} is replaced by objects yes and $\alpha_{3n+1+2m}$ from cell 2. At the next step, object yes from cell 1 is exchanged for object $\beta_{3n+1+2m+1}$ from cell 2. Finally, at step $3n+1+2m+3$ object yes from cell 3 is sent out to the environment by applying rule (50), and the computation halts.

– *Negative answer* : If none of the truth assignments encoded by a cell 1 makes the formula φ true, then object E_{m+1} does not appear at any cell labelled by 1. Thus, rule (46) is not applicable at configuration $\mathcal{C}_{(3n+1)+2m}$, and only rule

(37) is applicable and produces object $\beta_{3n+1+2m+1}$ in cell 3. Then, only rule (48) is applicable at configuration $\mathcal{C}_{(3n+1)+2m+1}$ and replaces object $\alpha_{3n+1+2m}$ from cell 2 by object $\beta_{3n+1+2m+1}$ from cell 3. Finally, at step $3n+1+2m+3$ objects `no` and $\beta_{3n+1+2m+1}$ from cell 2 are sent out to the environment by applying rule (49), and the computation halts.

6 A Formal Verification

The aim of this section is to present a formal proof that the family of recognizer tissue P systems with cell separation constructed in the previous section solves in a uniform way and polynomial time the `SAT` problem, according to Definition 1.

6.1 Polynomial Uniformity of the Family

In this subsection, we shall show that the family

$$\mathbf{\Pi} = \{\Pi(\langle m,n\rangle) \mid m,n \in \mathbb{N}\}$$

defined above is polynomially uniform by Turing machines. To this aim we prove that $\Pi(\langle m,n\rangle)$ is built in polynomial time with respect to the size parameter m and n of instances of the `SAT` problem.

It is easy to check that the rules of a system $\Pi(\langle m,n\rangle)$ of the family are recursively defined from the values m and n. The amount of resources to build an element of the family is of a polynomial order in the number n of the variables and the number m of clauses, as shown below:

1. Size of the alphabet: $2mn^2+5mn+3n^2+5m+27n+12 \in \Theta(mn^2)$.
2. Initial number of cells: $3 \in \Theta(1)$.
3. Initial number of objects: $12 \in \Theta(1)$.
4. Number of rules: $mn^2+3mn+3n^2+5m+30n+12 \in \Theta(mn^2)$.
5. Maximal length of a rule: $3 \in \Theta(1)$.

Therefore, there exists a deterministic Turing machine that builds the system $\Pi(\langle m,n\rangle)$ in a polynomial time with respect to m and n.

6.2 Soundness and Completeness of the Family

Let us start by fixing some notations that will allow us to describe the invariants, appearing in the computation, in a simpler way.

Let $\{x_1,\dots,x_i\}$ a set of propositional variables. A truth assignment of $\{x_1,\dots,x_i\}$ will be indistinctly denoted by:

- $\sigma_i = (\alpha_1,\dots,\alpha_i)$, where $\alpha_j \in \{T,F\}$.
- $\tau_i = (\beta_1,\dots,\beta_i)$, where $\beta_j \in \{T',F'\}$.
- $\epsilon_i = (\gamma_1,\dots,\gamma_i)$, where $\gamma_j \in \{t,f\}$.

The 2^i truth assignment of the set $\{x_1, \dots, x_i\}$ will be indistinctly denoted by $\{\sigma_{i,1}, \dots, \sigma_{i,2^i}\}$, $\{\tau_{i,1}, \dots, \tau_{i,2^i}\}$, or $\{\epsilon_{i,1}, \dots, \epsilon_{i,2^i}\}$, respectively. Notice that given a truth assignment $\sigma_{i,j}$ $(1 \le j \le 2^i)$ of $\{x_1, \dots, x_i\}$, we can briefly write the same truth assignment with primes as $\tau_{i,j}$, or in lowercase as $\epsilon_{i,j}$.

Let $\varphi = C_1 \wedge \dots \wedge C_m$, where $C_j = l_{j,1} \vee \dots \vee l_{j,r_j}$, $1 \le j \le m$, and each $l_{j,k}$ is an element of the set $\mathrm{Var}(\varphi) = \{x_i, \neg x_i \mid 1 \le i \le n\}$. We denote

$$\begin{aligned}
cod(\varphi) &= \{x_{i,j} : \ x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m\} \cup \\
&\quad \{\overline{x}_{i,j} : \ \neg x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m\} \\
(cod(\varphi))_e &= \{e_{i,j} : \ x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m\} \cup \\
&\quad \{\overline{e}_{i,j} : \ \neg x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m\} \\
(cod(\varphi))_e^t &= \{e^t_{i,j} : \ x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m\} \cup \\
&\quad \{\overline{e}^t_{i,j} : \ \neg x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m\}
\end{aligned}$$

For each k $(1 \le k \le n)$ we denote

$$\begin{aligned}
(cod(\varphi))_{e,>k} &= (cod(\varphi))_e - (\{e_{i,j} : \ x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m, 1 \le j \le k\} \cup \\
&\qquad \{\overline{e}_{i,j} : \ \neg x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m, 1 \le j \le k\}) \\
(cod(\varphi))^t_{e,>k} &= (cod(\varphi))^t_e - (\{e^t_{i,j} : \ x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m, 1 \le j \le k\} \cup \\
&\qquad \{\overline{e}^t_{i,j} : \ \neg x_i \in C_j, \ 1 \le i \le n, 1 \le j \le m, 1 \le j \le k\})
\end{aligned}$$

For each i, j, k $(1 \le i, k \le n, 1 \le j \le m)$ we denote

$$\begin{aligned}
(cod(\varphi))_{d_{i,j,k}} &= \{d_{i,j,k} : \ x_i \in C_j\} \ \cup \ \{\overline{d}_{i,j,k} : \ \neg x_i \in C_j\} \\
(cod(\varphi))^t_{d_{i,j,k}} &= \{d^t_{i,j,k} : \ x_i \in C_j\} \ \cup \ \{\overline{d}^t_{i,j,k} : \ \neg x_i \in C_j\}
\end{aligned}$$

The 2^n cells labelled by 1 generated by the system will be enumerated by $(1,1), (1,2), \dots, (1, 2^{n-1}), (1, 2^{n-1}+1), \dots, (1, 2^n)$, in such a way that cells labelled by $(1,1), (1,2), \dots, (1, 2^{n-1})$ contain T_n and the values of the truth assignment *without primes* $\sigma_{n-1,1}, \dots, \sigma_{n-1,2^{n-1}}$ of the set $\{x_1, \dots, x_{n-1}\}$, and cells labelled by $(1, 2^{n-1}+1), \dots, (1, 2^n)$ contain F'_n and the values of the truth assignment *with primes* $\tau_{n-1,1}, \dots, \tau_{n-1,2^{n-1}}$ of the set $\{x_1, \dots, x_{n-1}\}$. If $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ is a computation of the tissue P system $\Pi(\langle m, n \rangle)$ and l is the label of a cell, then we denote by $\mathcal{C}_i(l)$ the contents of cell l at configuration $\mathcal{C}_i$.

Theorem 6.1 *Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ be a computation of the tissue P system $\Pi(\langle m, n \rangle)$. For every i $(1 \le i \le n-1)$, we have the following:*

(1) At configuration $\mathcal{C}_{3i}$:

(a) There are 2^i cells labelled by 1 from which:

- $\star$ 2^{i-1} *cells contain objects T_i, , A_{i+1} , B_{i+1}. Moreover, each of them contains a different truth assignment $\sigma_{i-1,j}$ of the set $\{x_1, \dots, x_{i-1}\}$.*
- $\star$ 2^{i-1} *cells contain objects F'_i, , A'_{i+1} , B'_{i+1}. Moreover, each of them contains a different truth assignment $\tau_{i-1,j}$ of the set $\{x_1, \dots, x_{i-1}\}$.*

(b) There is a cell labelled by 2. This cell contains objects α_{3i}, `yes`, `no`*, and*

- ⋆ *If $i < n-1$ then it contains objects*

$$a_{i+1}^{2^i},\ {a'}_{i+1}^{2^i},\ b_{i+1}^{2^i},\ {b'}_{i+1}^{2^i},\ v_{i+1}^{2^i},\ t_1^{2^{i-1}}\ f_1^{2^{i-1}},\ \dots,\ t_i^{2^{i-1}}\ f_i^{2^{i-1}},$$
$$q_{1,i+1}^{2^{i-1}},\ \dots,\ q_{i+1,i+1}^{2^{i-1}}$$

- ⋆ *If $i = n-1$ then it contains objects*

$$a_{i+1}^{2^i},\ {a'}_{i+1}^{2^i},\ b_{i+1}^{2^i},\ {b'}_{i+1}^{2^i},\ v_{i+1}^{2^i},\ t_1^{2^{i-1}}\ f_1^{2^{i-1}},\ \dots,\ t_i^{2^{i-1}}\ f_i^{2^{i-1}}$$

(c) There is a cell labelled by 3. This cell contains object β_{3i}, and

- ⋆ *If $3i \le n$ then it also contains $(cod(\varphi))_{d_{i,j,3i}}^{2^{3i}}$*
- ⋆ *If $3i > n$ then it also contains $(cod(\varphi))_e^{2^n}$*

(2) At configuration $\mathcal{C}_{3i+1}$:

(a) There are 2^i cells labelled by 1.

- ⋆ *Each of them contains objects a_{i+1}, ${a'}_{i+1}$, b_{i+1}, ${b'}_{i+1}$.*
- ⋆ *Each of them contains a different truth assignment $\epsilon_{i,j}$ of the set $\{x_1, \dots, x_i\}$.*

(b) There is a cell labelled by 2. This cell contains objects

$$A_{i+1}^{2^{i-1}},\ {A'}_{i+1}^{2^{i-1}},\ B_{i+1}^{2^{i-1}},\ {B'}_{i+1}^{2^{i-1}},\ y_{i+1}^{2^{i+1}},\ \alpha_{3i+1},\ \texttt{yes},\ \texttt{no}$$

$$T_i^{2^{i-1}}\ \sigma_{i-1,1}\ \dots\ \sigma_{i-1,2^{i-1}}\quad {F'}_i^{2^{i-1}}\ \tau_{i-1,1}\ \dots\ \tau_{i-1,2^{i-1}}$$

Moreover, if $i < n-1$ then it also contains objects

$$r_{1,i+1}^{2^i},\ \dots,\ r_{i+1,i+1}^{2^i}$$

(c) There is a cell labelled by 3. This cell contains object β_{3i+1}, and

- ⋆ *If $3i+1 \le n$ then it also contains $(cod(\varphi))_{d_{i,j,3i+1}}^{2^{3i+1}}$*
- ⋆ *If $3i+1 > n$ then it also contains $(cod(\varphi))_e^{2^n}$*

(3) At configuration $\mathcal{C}_{3i+2}$:

(a) There are 2^i cells labelled by 1.

- ⋆ *Each of them contains objects*

$$A_{i+2},\ {A'}_{i+2},\ B_{i+2},\ {B'}_{i+2},\ S,\ T_{i+1},\ {F'}_{i+1}$$

- ⋆ *Each of them contains a different truth assignment $\sigma_{i,j}$ of the set $\{x_1, \dots, x_i\}$, as well as an identical copy, $\tau_{i,j}$, but for primes.*

(b) There is a cell labelled by 2. This cell contains objects $\alpha_{3i+2}, \texttt{yes}, \texttt{no}$, and such that:

- ⋆ *If $i < n-1$ then it also contains objects*

$$c_{i+1}^{2^{i+1}},\ z_{i+1}^{2^{i+1}},\ w_{i+1}^{2^{i+1}},\ ,\ s_1^{2^i},\ \dots,\ s_{i+1}^{2^i},\ u_{1,i+1}^{2^i},\ \dots,\ u_{i+1,i+1}^{2^i}$$

⋆ *If* $i = n-1$ *then it also contains objects* $w_{i+1}^{2^{i+1}}$.

(c) There is a cell labelled by 3. This cell contains object β_{3i+2}*, and such that:*

⋆ *If* $3i+2 \le n$ *then it also contains* $(cod(\varphi))^{2^{3i+2}}_{d_{i,j,3i+2}}$

⋆ *If* $3i+2 > n$ *then it also contains* $(cod(\varphi))^{2^n}_e$

Proof: By induction on i. Let us start analyzing the basic case $i = 1$.

At the initial configuration we have:

$$\begin{cases} \mathcal{C}_0(1) = \{A_1, B_1\} \\ \mathcal{C}_0(2) = \{a_1, a'_1, b_1, b'_1, v_1, q_{1,1}, \alpha_0, \texttt{yes}, \texttt{no}\} \\ \mathcal{C}_0(3) = \{\beta_0\} \,\cup\, cod(\varphi) \end{cases}$$

Then, rules (1) and (3) allow to exchange objects A_1, B_1 from cell 1 for objects a_1, a'_1, b_1, b'_1 from cell 2. Simultaneously, the application of rules (20), (24) and (36) produce objects $y_1^2, r_{1,1}, \alpha_1$ in cell 2. Rule (37) produces object β_1 in cell 3, and rule (38) produce objects $d^2_{i,j,1}$ if $x_{i,j} \in cod(\varphi)$, and objects $\overline{d}^2_{i,j,1}$ if $\overline{x}_{i,j} \in cod(\varphi)$, in cell 3. Therefore,

$$\begin{cases} \mathcal{C}_1(1) = \{a_1, a'_1, b_1, b'_1\} \\ \mathcal{C}_1(2) = \{A_1, B_1, y_1^2, r_{1,1}, \alpha_1, \texttt{yes}, \texttt{no}\} \\ \mathcal{C}_1(3) = \{\beta_1\} \,\cup\, \{d^2_{i,j,1} :\ x_{i,j} \in cod(\varphi)\} \,\cup\, \{\overline{d}^2_{i,j,1} :\ \overline{x}_{i,j} \in cod(\varphi)\} \end{cases}$$

At configuration $\mathcal{C}_1$:

(a) Rules (11), (12), (13) and (14) produce objects $B_2S, B'_2, T_1A_2, F'_1A'_2$ in cell 1.

(b) Rules (15), (17), (21), (26) and (36) produce objects $c_1, c_1, z_1^2w_1^2, s_1u_{1,1}, \alpha_2$ in cell 2.

(c) Rules (37) and (39) produce objects $\beta_2, d^{2^2}_{i,j,2}$ with $x_{i,j} \in cod(\varphi)$, and $\overline{d}^{2^2}_{i,j,2}$ with $\overline{x}_{i,j} \in cod(\varphi)$ in cell 3.

That is,

$$\begin{cases} \mathcal{C}_2(1) = \{T_1, A_2, F'_1, A'_2, B_2, S, B'_2\} \\ \mathcal{C}_2(2) = \{c_1^2, z_1^2, w_1^2, s_1, u_{1,1}, \alpha_2,\ \texttt{yes}, \texttt{no}\} \\ \mathcal{C}_2(3) = \{\beta_2\} \,\cup\, \{d^{2^2}_{i,j,2} :\ x_{i,j} \in cod(\varphi)\} \,\cup\, \{\overline{d}^{2^2}_{i,j,2} :\ \overline{x}_{i,j} \in cod(\varphi)\} \end{cases}$$

At configuration $\mathcal{C}_2$:

(a) Object S triggers separation rule (35) creating two new cells 1, one of them (1,1) containing $\{A_2, B_2, T_1\}$, and the other one (1,2) containing $\{A'_2, B'_2, F'_1\}$.

(b) If $1 = i = n-1$ (that is, $n = 2$) rules (19), (22), (23), (27), and (36) produce objects

$$b_2b'_2, v_2^2a_2^2a_2'^2, t_1f_1, \alpha_3$$

in cell 2. Rule (30) remove object $u_{1,1}$.

⋆ If $1 = i < n-1$ (that is, $n > 2$) rules (19), (22), (23), (27), (28) and (36) produce objects

$$b_2 b'_2, v_2^2, a_2^2 a'^2_2, t_1 f_1, q_{1,2} q_{2,2}, \alpha_3$$

in cell 2.

(c) If $3 = 3i \leq n$ (that is, $n > 2$) rules (37), and (39) produce objects $\beta_3, d_{i,j,3}^{2^3}$ with $x_{i,j} \in cod(\varphi)$, and $\overline{d}_{i,j,3}^{2^3}$ with $\overline{x}_{i,j} \in cod(\varphi)$ in cell 3.

⋆ If $3 = 3i > n$ (that is, $n = 2$) rules (37), and (40) produce objects $\beta_3, e_{i,j}^{2^2}$ with $x_{i,j} \in cod(\varphi)$, and $\overline{e}_{i,j}^{2^2}$ with $\overline{x}_{i,j} \in cod(\varphi)$ in cell 3.

That is,

$$\begin{cases} \mathcal{C}_3(1,1) = \{A_2, B_2, T_1\} \\ \mathcal{C}_3(1,2) = \{A'_2, B'_2, F'_1\} \\ \mathcal{C}_3(2) \quad = \{a_2^2, {a'}_2^2, b_2^2, {b'}_2^2, v_2^2, t_1, f_1, \alpha_3, \ \texttt{yes}, \texttt{no}\}, \text{ if } n = 2 \\ \mathcal{C}_3(2) \quad = \{a_2^2, {a'}_2^2, b_2^2, {b'}_2^2, v_2^2, t_1, f_1, q_{1,2}, q_{2,2}, \alpha_3, \ \texttt{yes}, \texttt{no}\}, \text{ if } n > 2 \\ \mathcal{C}_3(3) \quad = \{\beta_3\} \ \cup \ \{e_{i,j}^{2^2} : \ x_{i,j} \in cod(\varphi)\} \ \cup \ \{\overline{e}_{i,j}^{2^2} : \ \overline{x}_{i,j} \in cod(\varphi)\}, \text{ if } n = 2 \\ \mathcal{C}_3(3) \quad = \{\beta_3\} \ \cup \ \{d_{i,j,3}^{2^3} : \ x_{i,j} \in cod(\varphi)\} \ \cup \ \{\overline{d}_{i,j,3}^{2^3} : \ \overline{x}_{i,j} \in cod(\varphi)\}, \text{ if } n > 2 \end{cases}$$

At configuration $\mathcal{C}_3$:

(a) Rules (1), (2), (3), (4), (5), and (8) replace objects

$$A_2, A'_2, B_2, B'_2, T_1, F'_1$$

from cell 1 by objects

$$a_2, a'_2, a_2, a'_2, b_2, b'_2, b_2, b'_2, t_1, f_1$$

from cell 2.

(b) Rules (20) and (36) produce objects $y_2^{2^2}, \alpha_4$ in cell 2. Moreover, if $1 = i < n-1$ (that is, $n > 2$) then rule (25) produce objects $r_{1,2}^2, r_{2,2}^2$.

(c) Rule (37) produces object β_4 in cell 3. Moreover, if $4 = 3i + 1 \leq n$ (that is, $3i \leq n$) then rule (39) produce objects $d_{i,j,4}^{2^4}$ with $x_{i,j} \in cod(\varphi)$, and $\overline{d}_{i,j,4}^{2^4}$ with $\overline{x}_{i,j} \in cod(\varphi)$ in cell 3.

⋆ If $4 = 3i + 1 > n$ and $n = 2$, then objects $e_{i,j}^{2^2}$, $\overline{e}_{i,j}^{2^2}$ in cell 3 do not evolve.

⋆ If $4 = 3i + 1 > n$ and $n = 3$, then rule (40) produce objects $e_{i,j}^{2^3}$ with $x_{i,j} \in cod(\varphi)$, and $\overline{e}_{i,j}^{2^3}$ with $\overline{x}_{i,j} \in cod(\varphi)$ in cell 3.

That is,

$$\begin{cases} \mathcal{C}_4(1,1) = \{a_2, a'_2, b_2, b'_2, t_1\} \\ \mathcal{C}_4(1,2) = \{a_2, a'_2, b_2, b'_2, f_1\} \\ \mathcal{C}_4(2) \quad = \{A_2, A'_2, B_2, B'_2, T_1, F'_1, y_2^{2^2}, \alpha_4, \ \texttt{yes}, \texttt{no}\}, \text{ if } n = 2 \\ \mathcal{C}_4(2) \quad = \{A_2, A'_2, B_2, B'_2, T_1, F'_1, y_2^{2^2}, \alpha_4, r_{1,2}^2, r_{2,2}^2, \ \texttt{yes}, \texttt{no}\}, \text{ if } n > 2 \\ \mathcal{C}_4(3) \quad = \{\beta_4\} \ \cup \ \{e_{i,j}^{2^n} : \ x_{i,j} \in cod(\varphi)\} \ \cup \ \{\overline{e}_{i,j}^{2^n} : \ \overline{x}_{i,j} \in cod(\varphi)\}, \text{ if } n = 2, 3 \\ \mathcal{C}_4(3) \quad = \{\beta_4\} \ \cup \ \{d_{i,j,4}^{2^4} : \ x_{i,j} \in cod(\varphi)\} \ \cup \ \{\overline{d}_{i,j,4}^{2^4} : \ \overline{x}_{i,j} \in cod(\varphi)\}, \text{ if } n \geq 4 \end{cases}$$

At configuration $\mathcal{C}_4$:

(a) Rules (9), (11), (12), (13), and (14) produce objects

$$T_1T_1', B_3S, B_3', T_2A_3, F_2', A_3'$$

in cell (1,1), and rules (10), (11), (12), and (13) produce objects

$$F_1F_1', B_3S, B_3', T_2A_3, F_2', A_3'$$

in cell (1,2).

(b) Rule (36) produces object α_5 in cell 2. Moreover, if $1 = i < n-1$ (that is, $n > 2$) then rules (15), (16), (17), (18), (21), and (26) produce objects

$$c_2, c_2, c_2, c_2, z_2^{2^2} w_2^{2^2}, s_1^2 u_{1,2}^2, s_2^2 u_{2,2}^2$$

in cell 2.

⋆ If $i = 1 = n-1$ (that is, $n = 2$), then rules (15), (16), (17), and (18) remove objects A_2, A_2', B_2, B_2' from cell 2, and rule (21) produce objects w_2^2. Rules (31) and (34) remove objects T_1, F_1' from cell 2.

(c) Rule (37) produces object β_5 in cell 3. Moreover,

⋆ if $5 = 3i+2 \le n$ (thus $3i+1 < n$) then rule (39) produce objects $d_{i,j,5}^{2^5}$ with $x_{i,j} \in cod(\varphi)$, and $\overline{d}_{i,j,5}^{2^5}$ with $\overline{x}_{i,j} \in cod(\varphi)$ in cell 3.

⋆ If $5 = 3i+2 > n$ and $n = 2, 3$, then objects $e_{i,j}^{2^n}$, $\overline{e}_{i,j}^{2^n}$ in cell 3 do not evolve.

⋆ If $5 = 3i+2 > n$ and $n = 4$, then rule (40) produce objects $e_{i,j}^{2^4}$ with $x_{i,j} \in cod(\varphi)$, and $\overline{e}_{i,j}^{2^4}$ with $\overline{x}_{i,j} \in cod(\varphi)$ in cell 3.

That is,

$$\begin{cases} \mathcal{C}_5(1,1) = \{A_3, A_3', B_3, B_3', S, T_1, T_1', T_2, F_2'\} \\ \mathcal{C}_5(1,2) = \{A_3, A_3', B_3, B_3', S, F_1, F_1', T_2, F_2'\} \\ \mathcal{C}_5(2) \quad = \{w_2^{2^2}, \alpha_5, \texttt{ yes}, \texttt{no}\}, \text{ if } n = 2 \\ \mathcal{C}_5(2) \quad = \{c_2^{2^2}, z_2^{2^2}, s_1^2, u_{1,2}^2, s_2^2, u_{2,2}^2, w_2^{2^2}, \alpha_5, \texttt{ yes}, \texttt{no}\}, \text{ if } n > 2 \\ \mathcal{C}_5(3) \quad = \{\beta_5\} \cup \{e_{i,j}^{2^n} : x_{i,j} \in cod(\varphi)\} \cup \{\overline{e}_{i,j}^{2^n} : \overline{x}_{i,j} \in cod(\varphi)\}, \text{ if } n < 5 \\ \mathcal{C}_5(3) \quad = \{\beta_5\} \cup \{d_{i,j,5}^{2^5} : x_{i,j} \in cod(\varphi)\} \cup \{\overline{d}_{i,j,5}^{2^5} : \overline{x}_{i,j} \in cod(\varphi)\}, \text{ if } n \ge 5 \end{cases}$$

Thus, the result of the theorem hold for $i = 1$.

By induction hypothesis, let i be such that $1 \le i < n-1$ and let us suppose (1), (2), and (3) hold for i. Let us see that (1), (2), and (3) also hold for $i+1$.

Then we assume that:

$$\begin{cases} \mathcal{C}_{3i+2}(1,1) \;\; = \{A_{i+2}, A_{i+2}', B_{i+2}, B_{i+2}', S, T_{i+1}, F_{i+1}', \sigma_{i,1}, \tau_{i,1}\} \\ \cdots\cdots\cdots \quad \cdots\cdots\cdots\cdots\cdots\cdots \\ \mathcal{C}_{3i+2}(1,2^i) = \{A_{i+2}, A_{i+2}', B_{i+2}, B_{i+2}', S, T_{i+1}, F_{i+1}', \sigma_{i,2^i}, \tau_{i,2^i}\} \\ \mathcal{C}_{3i+2}(2) \quad\;\; = \{c_{i+1}^{2^{i+1}}, z_{i+1}^{2^{i+1}}, w_{i+1}^{2^{i+1}}, s_1^{2^i}, \ldots, s_{i+1}^{2^i}, u_{1,i+1}^{2^i}, \ldots u_{i+1,i+1}^{2^i}, \alpha_{3i+2}, \texttt{yes}, \texttt{no}\} \\ \mathcal{C}_{3i+2}(3) \quad\;\; = \{\beta_{3i+2}\} \cup (cod(\varphi))_{d_{i,j,3i+2}}^{2^{3i+2}}, \text{ if } 3i+2 \le n \\ \mathcal{C}_{3i+2}(3) \quad\;\; = \{\beta_{3i+2}\} \cup (cod(\varphi))_e^{2^n}, \text{ if } 3i+2 > n \end{cases}$$

At configuration $\mathcal{C}_{3i+2}$:

(a) Object S triggers separation rule (35), creating 2^i new cells 1 having a total of 2^{i+1} cells labelled by 1 from which:
- 2^i cells 1 contain objects $A_{i+2}, B_{i+2}, T_{i+1}$. Moreover, each of them contains a different truth assignment $\sigma_{i,j}$ of the set $\{x_1, \ldots, x_i\}$.
- 2^i cells 1 contain objects $A'_{i+2}, B'_{i+2}, F'_{i+1}$. Moreover, each of them contains a different truth assignment $\tau_{i,j}$ of the set $\{x_1, \ldots, x_i\}$.

(b) Rule (36) produces object α_{3i+3}. Objects `yes` and `no` do not evolve at this transition step.

⋆ If $i+1 < n-1$ (that is, $n > i+2$) rules (19), (22), (23), (27), and (25) produce objects

$$b_{i+2}^{2^{i+1}}, {b'}_{i+2}^{2^{i+1}}, v_{i+2}^{2^{i+1}}, a_{i+2}^{2^{i+1}}, {a'}_{i+2}^{2^{i+1}}, t_1^{2^i}, f_1^{2^i}, \ldots, t_{i+1}^{2^i}, f_{i+1}^{2^i}, q_{1,i+2}^{2^i}, \ldots, q_{i+2,i+2}^{2^i}$$

in cell 2.

⋆ If $i+1 = n-1$ (that is, $n = i+2$) rules (19), (22), (23), and (27) produce objects

$$b_{i+2}^{2^{i+1}}, {b'}_{i+2}^{2^{i+1}}, v_{i+2}^{2^{i+1}}, a_{i+2}^{2^{i+1}}, {a'}_{i+2}^{2^{i+1}}, t_1^{2^i}, f_1^{2^i}, \ldots, t_{i+1}^{2^i}, f_{i+1}^{2^i}$$

in cell 2. Rule (30) erases objects $u_{1,i+1}^{2^i}, \ldots, u_{i+1,i+1}^{2^i}$ from cell 2.

(c) Rule (37) produces object β_{3i+3}. Moreover,

⋆ If $3i+3 \leq n$ (that is, $n > 3i+2$) rule (39) produces $(cod(\varphi))_{d_{i,j,3i+3}}^{2^{3i+3}}$ in cell 3.

⋆ If $n < 3i+2$, then objects from $(cod(\varphi))_e^{2^n}$ do not evolve.

⋆ If $n = 3i+2$, then rule (40) produces $(cod(\varphi))_e^{2^n}$ from $(cod(\varphi))_{d_{i,j,n}}^{2^n}$ in cell labelled by 3.

Thus, the result holds for configuration $\mathcal{C}_{3(i+1)}$.

At configuration $\mathcal{C}_{3(i+1)}$:

(a) Rules (1), (2), (3), (4), (5), (6), (7), and (8) trade objects

$$A_{i+2}, A'_{i+2}, B_{i+2}, B'_{i+2}, T_1, \ldots, T_{i+1}, T'_1, \ldots, T'_{i+1}, F_1, \ldots, F_{i+1}, F'_1, \ldots, F'_{i+1}$$

from cell 1 for objects

$$a_{i+2}, a'_{i+2}, b_{i+2}, b'_{i+2}, f_1, t_1, \ldots, f_{i+1}, t_{i+1}$$

from cell 2. Then, we have 2^{i+1} cells labelled by 1 such that each of them contains objects $a_{i+2}, a'_{i+2}, b_{i+2}, b'_{i+2}$ and also contain a different truth assignment $\epsilon_{i+1,j}$ of the set $\{x_1, \ldots, x_{i+1}\}$.

(b) Rule (36) produces object α_{3i+4}. Objects `yes` and `no` do not evolve at this transition step. After the interchange of objects with cell 1, cell 2 contains objects

$$A_{i+2}^{2^i}, {A'}_{i+2}^{2^i}, B_{i+2}^{2^i}, {B'}_{i+2}^{2^i}, T_{i+1}^{2^i}, {F'}_{i+1}^{2^i}, \sigma_{i,1}, \ldots, \sigma_{i,2^i}, \tau_{i,1}, \ldots, \tau_{i,2^i}$$

- ⋆ If $i+1 < n-1$ (that is, $n > i+2$) then rules (20) and (25) produce objects $y_{i+2}^{2^{i+2}}, r_{1,i+2}^{2^{i+1}}, \dots, r_{i+2,i+2}^{2^{i+1}}$ in cell 2.
- ⋆ If $i+1 = n-1$ (that is, $n = i+2$) rule (20) produces objects $y_{i+2}^{2^{i+2}}$ in cell 2.

(c) Rule (37) produces object $\beta_{3(i+1)+1}$. Moreover,
- ⋆ If $n > 3(i+1)$, then rule (39) produces $(cod(\varphi))_{d_{i,j,3(i+1)+1}}^{2^{3(i+1)+1}}$ in cell 3.
- ⋆ If $n < 3(i+1)$, then objects from $(cod(\varphi))_e^{2^n}$ do not evolve.
- ⋆ If $n = 3(i+1)$, then rule (40) produces $(cod(\varphi))_e^{2^n}$ from $(cod(\varphi))_{d_{i,j,n}}^{2^n}$ in cell 3.

Hence, the result holds for the configuration $\mathcal{C}_{3(i+1)+1}$.

At configuration $\mathcal{C}_{3(i+1)+1}$:

(a) Rules (9), (10), (11), and (12) produce objects

$$T_{i+2}A_{i+3}, F'_{i+2}A'_{i+3}, F_1F'_1, T_1T'_1, \dots, F_{i+1}F'_{i+1}, T_{i+1}T'_{i+1}, B_{i+3}, S, B'_{i+3}$$

in cell 1. Specifically, there are 2^{i+1} cells labelled by 1 such that each of them contains objects $A_{i+3}, A'_{i+3}, B_{i+3}, S, B'_{i+3}, T_{i+2}, F'_{i+2}$, and also contains a different truth assignment $\sigma_{i+1,j}$ of the set $\{x_1, \dots x_{i+1}\}$, as well as an identical copy $\tau_{i+1,j}$ of the set $\{x_1, \dots x_{i+1}\}$ but for primes.

(b) Rule (36) produces object α_{3i+5}. Objects `yes` and `no` do not evolve at this transition step. Moreover,
- ⋆ If $i+1 < n-1$ (that is, $n > i+2$) then rules (15), (16), (17) and (18) produce objects $c_{i+2}^{2^i}, c_{i+2}^{2^i}, c_{i+2}^{2^i}, c_{i+2}^{2^i}$ (that is, $c_{i+2}^{2^{i+2}}$) in cell 2. Also, rules (31), (32), (33) and (34) erase objects T_i, T'_i, F_i, F'_i from cell 2. Rules (21) and (26) produce objects
$$z_{i+2}^{2^{i+2}}, w_{i+2}^{2^{i+2}}, s_1^{2^{i+1}}, u_{1,i+2}^{2^{i+1}}, \dots, s_{i+2}^{2^{i+1}}, u_{i+2,i+2}^{2^{i+1}}$$
- ⋆ If $i+1 = n-1$ (that is, $n = i+2$) rules (15), (16), (17), (18), (31), (32), (33) and (34) erase objects
$$A_{i+2}, A'_{i+2}, B_{i+2}, B'_{i+2}, T_i, T'_i, F_i, F'_i$$
from cell 2. Also rule (21) produces object $w_{i+2}^{2^{i+2}} \equiv w_n^{2^n}$.

(c) Rule (37) produces object $\beta_{3(i+1)+2}$ in cell 3. Moreover,
- ⋆ If $n > 3(i+1)+1$, then rule (39) produces $(cod(\varphi))_{d_{i,j,3(i+1)+2}}^{2^{3(i+1)+2}}$ in cell 3.
- ⋆ If $n < 3(i+1)+1$, then objects from $(cod(\varphi))_e^{2^n}$ do not evolve.
- ⋆ If $n = 3(i+1)+1$, then rule (40) produces $(cod(\varphi))_e^{2^n}$ from $(cod(\varphi))_{d_{i,j,n}}^{2^n}$ in cell 3.

Hence, the result holds for configuration $\mathcal{C}_{3(i+1)+2}$.

Then the proof of the theorem completes. □

Theorem 6.2 *Let* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ *be a computation of the tissue P system* $\Pi(\langle m, n\rangle)$. **At configuration** $\mathcal{C}_{3n}$, *we have the following:*

(a) There are 2^n *cells labelled by 1 from which:*

* $\star$ 2^{n-1} *cells contain objects* T_n, , A_{n+1}. *Moreover, each of them also contains a different truth assignment* $\sigma_{n-1,j}$ *of the set* $\{x_1, \dots, x_{n-1}\}$.
* $\star$ 2^{n-1} *cells contain objects* F'_n, , A'_{n+1}. *Moreover, each of them also contains a different truth assignment* $\tau_{n-1,j}$, *of the set* $\{x_1, \dots, x_{n-1}\}$.

(b) There is a cell labelled by 2. This cell contains objects $E_1^{2^n}$ α_{3n} `yes` `no`.
(c) There is a cell labelled by 3. This cell contains objects β_{3n}, *and* $(cod(\varphi))_e^{2^n}$.

Proof: From Theorem 6.1 for $i = n-1$ we deduce that at configuration $\mathcal{C}_{3(n-1)+2} = \mathcal{C}_{3n-1}$ we have:

- There are 2^{n-1} cells labelled by 1 such that:
  (a) Each of them contains objects $A_{n+1}, A'_{n+1}, B_{n+1}, B'_{n+1}, S, T_n, F'_n$.
  (b) Each of them contains a different truth assignment $\sigma_{n-1,j}$ of the set $\{x_1, \dots, x_{n-1}\}$ as well as an identical copy $\tau_{n-1,j}$ but for primes.
- There is a cell labelled by 2 which contains objects $w_n^{2^n}, \alpha_{3(n-1)+2} = \alpha_{3n-1}$, `yes`, `no`.
- There is a cell labelled by 3 which contains object $\beta_{3(n-1)+2} = \beta_{3n-1}$, and objects from $(cod(\varphi))_e^{2^n}$ (because $3(n-1)+2 > n$)).

By applying rules $(1, B_{n+1}/\lambda, 0)$ and $(1, B'_{n+1}/\lambda, 0)$, objects B_{n+1} and B'_{n+1} are removed from cell 1. By applying separation rule (35), each cell 1 creates two new cells labelled by 1: one of them containing objects with primes, and the other containing objects without primes. That is, at configuration $\mathcal{C}_{3n}$ we have 2^n cell 1 such that:

(a) 2^{n-1} cells contain objects T_n, A_{n+1}. Moreover, each of them contains a different truth assignment $\sigma_{n-1,j}$ of the set $\{x_1, \dots, x_{n-1}\}$.
(b) 2^{n-1} cells contain objects F'_n, A'_{n+1}. Moreover, each of them contains a different truth assignment $\tau_{n-1,j}$ of the set $\{x_1, \dots, x_{n-1}\}$.

Rule (36) produces object α_{3n} in cell 2. Rule $(2, w_n/E_1, 0)$ produces objects $E_1^{2^n}$ in cell 2. Neither objects `yes` or `no` evolve at this transition step. That is,

$$\mathcal{C}_{3n}(2) = \{E_1^{2^n}, \alpha_{3n}, \texttt{yes}, \texttt{no}\}$$

Rule (37) produces object β_{3n} in cell 2. Objects from $(cod(\varphi))_e^{2^n}$ do not evolve at this transition step. That is,

$$\mathcal{C}_{3n}(3) = \{\beta_{3n}\} \cup (cod(\varphi))_e^{2^n}$$

$\square$

Theorem 6.3 *Let* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ *be a computation of the tissue P system* $\Pi(\langle m, nrangle)$. **At configuration** $\mathcal{C}_{3n+1}$, *we have the following:*

(a) There are 2^n *cells labelled by 1 which contain object* E_1. *Besides,*

- ⋆ 2^{n-1} *of those cells, enumerated by* $(1,1), \dots (1, 2^{n-1})$*, contain object* T_n*, and each of them contains a different truth assignment* $\sigma_{n-1,j}$ *of the set* $\{x_1, \dots, x_{n-1}\}$.
- ⋆ 2^{n-1} *of those cells, enumerated by* $(1, 2^{n-1}+1), \dots (1, 2^n)$*, contain object* F'_n*, and each of them contains a different truth assignment* $\tau_{n-1,j}$ *of the set* $\{x_1, \dots, x_{n-1}\}$.

(b) There is a cell labelled by 2. This cell contains objects $\alpha_{3n+1}\,\texttt{yes}\,\texttt{no}\,A_{n+1}^{2^{n-1}}\,{A'}_{n+1}^{2^{n-1}}$.

(c) There is a cell labelled by 3. This cell contains objects β_{3n+1}*, and* $(cod(\varphi))_e^{2^n}$.

Proof: At configuration $\mathcal{C}_{3n}$:

(a) Rules $(1, A_{n+1}/E_1, 2)$ and $(1, A'_{n+1}/E_1, 2)$ exchange objects A_{n+1}, A'_{n+1} from cell 1 for objects E_1 from cell 2. Hence, there are 2^n cells labelled by 2 each of them containing object E_1. Besides:

- ⋆ 2^{n-1} of those cells, enumerated by $(1,1), \dots (1, 2^{n-1})$, contain object T_n, and each of them contains a different truth assignment $\sigma_{n-1,j}$ of the set $\{x_1, \dots, x_{n-1}\}$.
- ⋆ 2^{n-1} of those cells, enumerated by $(1, 2^{n-1}+1), \dots (1, 2^n)$, contain object F'_n, and each of them contains a different truth assignment $\tau_{n-1,j}$ of the set $\{x_1, \dots, x_{n-1}\}$.

(b) Rule (36) produces object α_{3n+1} in cell 2. Objects `yes` and `no` do not evolve at this transition step. That is,

$$\mathcal{C}_{3n+1}(2) = \{A_{n+1}^{2^{n-1}}, {A'}_{n+1}^{2^{n-1}}, \alpha_{3n+1}, \texttt{yes}, \texttt{no}\}$$

(c) Rule (37) produces object β_{3n+1} in cell 2. Objects from $(cod(\varphi))_e^{2^n}$ do not evolve at this transition step. That is,

$$\mathcal{C}_{3n+1}(3) = \{\beta_{3n+1}\} \cup (cod(\varphi))_e^{2^n}$$

□

In this way, the **generating stage** finishes at step $3n+1$ and the **checking stage** would start at the next step.

Theorem 6.4 *Let* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ *be a computation of the tissue P system* $\Pi(\langle m, n\rangle)$. **At configuration** $\mathcal{C}_{(3n+1)+1}$*, the following holds:*

(a) There are 2^n *cells labelled by 1. Besides,*

- ⋆ *If the truth assignment* $\sigma_{n,s}$ *associated with a cell* $(1,t)$*, where* $1 \le t \le 2^n$*, makes the clause* C_1 *<u>true</u>, then*
  - *If* $1 \le t \le 2^{n-1}$ *then it contains* $e_{i,1} + (\sigma_{n,s} - \{T_i\})$*, for some* i *such that* $x_i \in C_1$*, or it contains* $\overline{e}_{i,1} + (\sigma_{n,s} - \{F_i\})$*, for some* i *such that* $\neg x_i \in C_1$.

– *If* $2^{n-1}+1 \leq t \leq 2^n$ *then it contains* $e_{i,1} + (\tau_{n,s} - \{T'_i\})$*, for some* i *such that* $x_i \in C_1$*, or it contains* $\overline{e}_{i,1} + (\tau_{n,s} - \{F'_i\})$*, for some* i *such that* $\neg x_i \in C_1$*.*

⋆ *If the truth assignment* $\sigma_{n,s}$ *associated with a cell* $(1,t)$*, where* $1 \leq t \leq 2^n$*, makes clause* C_1 *false , then their contents coincide with the corresponding contents in the previous configuration* $\mathcal{C}_{3n+1}$*. In particular, that cell does not contain any object* $e_{i,1}$ *nor* $\overline{e}_{i,1}$*.*

(b) There is a cell labelled by 2. This cell contains objects $\alpha_{(3n+1)+1}, \texttt{yes}, \texttt{no}$*.*

(c) There is a cell labelled by 3. This cell contains:

⋆ k_1 *copies of object* E_1*, being* k_1 *the number of truth assignments making clause* C_1 *of* φ *true.*

⋆ $(cod(\varphi))^{2^n}_{e,>1}$ *representing* 2^n *copies of the objects* $e_{i,j}$ *and* $\overline{e}_{i,j}$ *such that* $j > 1$ *and* $x_i \in C_j$ *in the first case, and* $\neg x_i \in C_j$ *in the second one.*

⋆ *Object* $\beta_{(3n+1)+1}$*.*

⋆ *Some irrelevant objects of the type* T_i, T'_i, F_i, F'_i *that will dissappear at the next step.*

⋆ *Some irrelevant objects of the type* $e_{i,1}, \overline{e}_{i,1}$ *that will not be considered anymore.*

Proof: At configuration $\mathcal{C}_{3n+1}$:

(a) Rules of type (41) are applied to cells labelled by 1 trading objects

$$E_1, T_i, T'_i, F_i, F'_i$$

from cell 1 for objects $e_{i,1}, \overline{e}_{i,1}$ from cell 3 according to the following conditions: if a cell 1 encodes a truth assignment making clause C_1 true, then it replaces objects E_1T_i or $E_1T'_i$ (respectively, objects E_1F_i or $E_1F'_i$) by objects $e_{i,1}$ (respectively, objects $\overline{e}_{i,1}$), if $x_i \in C_1$ (respectively, if $\neg x_i \in C_1$). This transition step is non-deterministic because object E_1 can choose different truth values T, T', F or F' from cells labelled by 1 making clause C_1 true.

Let us suppose that the truth assignment $\sigma_{n,s}$ associated with a cell $(1,t)$ $(1 \leq t \leq 2^n)$ makes the clause C_1 true (on the contrary, rule (41) is not applicable to configuration $\mathcal{C}_{3n+1}$, so $\mathcal{C}_{(3n+1)+1}(1,t) = \mathcal{C}_{(3n+1)}(1,t)$).

⋆ Case 1: $1 \leq t \leq 2^{n-1}$.
If $x_i \in C_1$ then objects E_1T_i from cell $(1,t)$ are replaced by object $e_{i,1}$ from cell 3. So, the contents of cell $(1,t)$ is $e_{i,1} + (\sigma_{n,s} - \{T_i\})$.
If $\neg x_i \in C_1$ then objects E_1F_i from cell $(1,t)$ are replaced by object $\overline{e}_{i,1}$ from cell 3. So, the contents of cell $(1,t)$ is $\overline{e}_{i,1} + (\sigma_{n,s} - \{F_i\})$.

⋆ Case 2: $2^{n-1}+1 \leq t \leq 2^n$.
If $x_i \in C_1$ then objects $E_1T'_i$ from cell $(1,t)$ are exchanged for object $e_{i,1}$ from cell 3. So, the contents of cell $(1,t)$ is $e_{i,1} + (\tau_{n,s} - \{T'_i\})$.
If $\neg x_i \in C_1$ then objects $E_1F'_i$ from cell $(1,t)$ are exchanged for object $\overline{e}_{i,1}$ from cell 3. So, the contents of cell $(1,t)$ is $\overline{e}_{i,1} + (\tau_{n,s} - \{F'_i\})$.

(b) Rules (15) and (16) remove objects $A^{2^{n-1}}_{n+1}, A'^{2^{n-1}}_{n+1}$ from cell 2. Rule (36) produces object α_{3n+2}. Hence, $\mathcal{C}_{3n+2}(2) = \{\alpha_{3n+2}, \texttt{yes}, \texttt{no}\}$.

(c) Rule (37) produces object β_{3n+2} in cell 3 which also contains:
- ⋆ A number k_1 of copies of object E_1 equal to the number of truth assignment making clause C_1 true.
- ⋆ $(cod(\varphi))^{2^n}_{e,>1}$.
- ⋆ Garbagge objects T_i, T'_i, F_i, F'_i which will be removed at the next step.
- ⋆ Garbagge objects $e_{i,1}, \overline{e}_{i,1}$ which will not be considered anymore.

□

Theorem 6.5 *Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ be a computation of the tissue P system $\Pi(\langle m,n\rangle)$. For every j $(1 \leq j \leq m-1)$ we have:*

(1) At configuration $\mathcal{C}_{(3n+1)+2j}$, *the following holds:*

(a) There are 2^n cells labelled by 1. Besides,
- ⋆ *If the truth assignment $\sigma_{n,s}$ associated with a cell $(1,t)$, where $1 \leq t \leq 2^n$, makes $C_1 \wedge \dots \wedge C_j$ true, then it contains object E_{j+1}. Moreover,*
  - – *If $1 \leq t \leq 2^{n-1}$ then it contains object T_i, for some i such that $x_i \in C_j$, or it contains object F_i, for some i such that $\neg x_i \in C_j$. Besides, objects T_i and F_i of that cell 1 at configuration $\mathcal{C}_{(3n+1)+2j-1}$ remain at configuration $\mathcal{C}_{(3n+1)+2j}$.*
  - – *If $2^{n-1}+1 \leq t \leq 2^n$ then it contains T_i, for some i such that $x_i \in C_j$, or it contains F_i, for some i such that $\neg x_i \in C_j$. Besides, objects T'_i and F'_i of that cell 1 at configuration $\mathcal{C}_{(3n+1)+2j-1}$ remain at configuration $\mathcal{C}_{(3n+1)+2j}$.*
- ⋆ *If the truth assignment $\sigma_{n,s}$ associated with a cell $(1,t)$, where $1 \leq t \leq 2^n$, makes $C_1 \wedge \dots \wedge C_j$ false, then their contents coincide with the corresponding contents in the previous configuration $\mathcal{C}_{(3n+1)+2j-1}$. In particular, that cell does not contain object E_{j+1}.*

(b) There is a cell labelled by 2. This cell contains objects $\alpha_{(3n+1)+2j}$, `yes`, `no`.

(c) There is a cell labelled by 3. This cell contains:
- ⋆ *$(cod(\varphi))^{2^n}_{e,>j}$ representing 2^n copies of the objects $e_{i,j'}$ and $\overline{e}_{i,j'}$ such that $j' > j$ and $x_i \in C_{j'}$ in the first case, and $\neg x_i \in C_{j'}$ in the second one.*
- ⋆ *Object $\beta_{(3n+1)+2j}$.*
- ⋆ *Some irrelevant objects of the type $e_{i,j'}, \overline{e}_{i,j'}$, with $1 \leq j' \leq j$ that will not be considered anymore.*

(2) At configuration $\mathcal{C}_{(3n+1)+2j+1}$, *the following holds:*

(a) There are 2^n cells labelled by 1. Besides,
- ⋆ *If the truth assignment $\sigma_{n,s}$ associated with a cell $(1,t)$, where $1 \leq t \leq 2^n$, makes $C_1 \wedge \dots \wedge C_{j+1}$ true, then*
  - – *If $1 \leq t \leq 2^{n-1}$ then it contains $e_{i,j+1} + (\sigma_{n,s} - \{T_i\})$, for some i such that $x_i \in C_{j+1}$, or it contains $\overline{e}_{i,j+1} + (\sigma_{n,s} - \{F_i\})$, for some i such that $\neg x_i \in C_{j+1}$.*
  - – *If $2^{n-1}+1 \leq t \leq 2^n$ then it contains $e_{i,j+1} + (\tau_{n,s} - \{T'_i\})$, for some i such that $x_i \in C_{j+1}$, or it contains $\overline{e}_{i,j+1} + (\tau_{n,s} - \{F'_i\})$, for some i such that $\neg x_i \in C_{j+1}$.*

- ⋆ *If the truth assignment* $\sigma_{n,s}$ *associated with a cell* $(1,t)$*, where* $1 \leq t \leq 2^n$*, makes* $C_1 \wedge \cdots \wedge C_{j+1}$ *false, then their contents coincide with the corresponding contents in the previous configuration* $\mathcal{C}_{(3n+1)+2j}$*. In particular, that cell does not contain any object* $e_{i,j+1}$ *nor* $\overline{e}_{i,j+1}$*.*

(b) There is a cell labelled by 2. This cell contains objects $\alpha_{(3n+1)+2j+1}, \mathtt{yes}, \mathtt{no}$*.*

(c) There is a cell labelled by 3. This cell contains:

- ⋆ k_{j+1} *copies of object* E_{j+1}*, being* k_{j+1} *the number of truth assignment making clauses* $C_1, \ldots, C_{j+1}$ *of* φ *true.*
- ⋆ $(cod(\varphi))^{2^n}_{e,>(j+1)}$ *representing* 2^n *copies of the objects* $e_{i,j'}$ *and* $\overline{e}_{i,j'}$ *such that* $j' > j+1$ *and* $x_i \in C_{j'}$ *in the first case, and* $\neg x_i \in C_{j'}$ *in the second one.*
- ⋆ *Object* $\beta_{(3n+1)+2j+1}$*.*
- ⋆ *Some irrelevant objects of the type* T_i, T'_i, F_i, F'_i *that will dissappear at the next step.*
- ⋆ *Some irrelevant objects of the type* $e_{i,j'}, \overline{e}_{i,j'}$ *with* $1 \leq j' \leq j+1$ *that will not be considered anymore.*

Proof: By induction on j. Let us start analyzing the basic case $j = 1$.

At **configuration** $\mathcal{C}_{(3n+1)+1}$:

(a) Rule (42) produces objects T_iE_2 in a cell 1 which contains object $e_{i,1}$, and produces objects F_iE_2 in a cell 1 which contains object $\overline{e}_{i,1}$. So, there are 2^n cells labelled by 1 such that:

- ⋆ If the truth assignment associated with a cell $(1,t)$ makes clause C_1 true, then it contains objects E_2. Moreover, it contains object T_i for some i such that $x_i \in C_1$, or object F_i for some i such that $\overline{x}_i \in C_1$. Besides, the remaining objects at configuration $\mathcal{C}_{3n+2}$ stay unchanged at this transition step.
- ⋆ If the truth assignment associated with a cell $(1,t)$ makes clause C_1 false, then their contents coincide with the corresponding contents of the previous configuration $\mathcal{C}_{(3n+1)+1}$.

(b) Only rule (36) is applicable to cell 2 at configuration $\mathcal{C}_{3n+2}$. So,

$$\mathcal{C}_{3n+3}(2) = \{\alpha_{(3n+1)+2}, \mathtt{yes}, \mathtt{no}\}$$

(c) Rule (37) produces object β_{3n+3} in cell 3. Rules (44) and (45) remove objects $E_1, T_i, T'_i, F_i, F'_i$ from cell 3.

At **configuration** $\mathcal{C}_{(3n+1)+2}$:

(a) If the truth assignment $\sigma_{n,s}$ associated with a cell $(1,t)$ makes clause C_2 true, then

- ⋆ If $1 \leq t \leq 2^{n-1}$, rules (41) replace objects T_iE_2 from cell 1 by objects $e_{i,2}$ from cell 3, for some i such that $x_i \in C_2$, or objects F_iE_2 from cell 1 by objects $\overline{e}_{i,2}$ from cell 3, for some i such that $\overline{x}_i \in C_2$. Hence, such a cell 1 contains

$$\begin{cases} e_{i,2} + (\sigma_{n,s} - \{T_i\}), \text{ if objects } T_iE_2 \text{ have been exchanged} \\ e_{i,2} + (\sigma_{n,s} - \{F_i\}), \text{ if objects } F_iE_2 \text{ have been exchanged} \end{cases}$$

- ⋆ If $2^{n-1} + 1 \leq t \leq 2^n$, rule (41) either replaces objects T_iE_2 or objects T'_iE_2 by objects $e_{i,2}$ from cell 3, for some i such that $x_i \in C_2$, either objects F_iE_2 or objects F'_iE_2 by objects $\overline{e}_{i,2}$ from cell 3, for some i such that $\overline{x}_i \in C_2$. Hence, such a cell 1 contains

$$\begin{cases} e_{i,2} + (\tau_{n,s} - \{T_i\}), \text{ if objects } T_iE_2 \text{ have been exchanged} \\ e_{i,2} + (\tau_{n,s} - \{T'_i\}), \text{ if objects } T'_iE_2 \text{ have been exchanged} \\ e_{i,2} + (\tau_{n,s} - \{F_i\}), \text{ if objects } F_iE_2 \text{ have been exchanged} \\ e_{i,2} + (\tau_{n,s} - \{F'_i\}), \text{ if objects } F'_iE_2 \text{ have been exchanged} \end{cases}$$

(b) Only rule (36) is applicable to cell 2 at configuration $\mathcal{C}_{(3n+1)+2}$. So,

$$\mathcal{C}_{3n+4}(2) = \{\alpha_{(3n+1)+3}, \texttt{yes}, \texttt{no}\}$$

(c) Also rule (37) is applicable to cell 3 producing object β_{3n+4}. Then, cell 3 contains:
- – k_2 copies of object E_2, being k_2 the number of truth assignment making clauses C_1, C_2 of φ true.
- – $(cod(\varphi))^{2^n}_{e,>2}$ representing 2^n copies of the objects $e_{i,j'}$ and $\overline{e}_{i,j'}$ such that $j' > 2$ and $x_i \in C_{j'}$ in the first case, and $\neg x_i \in C_{j'}$ in the second one.
- – Object $\beta_{(3n+1)+3}$.
- – Garbagge objects of the type T_i, T'_i, F_i, F'_i that will dissappear at the next step.
- – Garbagge objects of the type $e_{i,j'}, \overline{e}_{i,j'}$ with $1 \leq j' \leq j+1$ that will not be considered anymore.

By induction hypothesis, let j such that $1 \leq j < m-1$ and let us the result holds for j. Let us see that the result also holds for $j+1$.

At **configuration** $\mathcal{C}_{(3n+1)+2j+1}$:

(a) Rule (42) produces objects T_iE_{j+2} in a cell 1 which contains object $e_{i,j}$, and produces objects F_iE_{j+2} in a cell 1 which contains object $\overline{e}_{i,j}$. So, there are 2^n cells labelled by 1 such that:
- ⋆ If the truth assignment associated with a cell $(1,t)$ makes $C_1 \wedge \cdots \wedge C_{j+2}$ true, then it contains objects E_{j+2}. Moreover, it contains object T_i for some i such that $x_i \in C_{j+2}$, or object F_i for some i such that $\overline{x}_i \in C_{j+2}$. Besides, the remaining objects at configuration $\mathcal{C}_{(3n+1)+2j+1}$ stay unchanged at this transition step.
- ⋆ If the truth assignment associated with a cell $(1,t)$ makes $C_1 \wedge \cdots \wedge C_{j+2}$ false, then their contents coincide with the corresponding contents of the previous configuration $\mathcal{C}_{(3n+1)+2j+1}$.

(b) Only rule (36) is applicable to cell 2 at configuration $\mathcal{C}_{(3n+1)+2j+1}$. So,

$$\mathcal{C}_{(3n+1)+2j+2}(2) = \{\alpha_{(3n+1)+2j+2}, \texttt{yes}, \texttt{no}\}$$

(c) Rule (37) produces object $\beta_{(3n+1)+2j+2}$ in cell 3. Rules (44) and (45) remove objects $E_1, T_i, T'_i, F_i, F'_i$ from cell 3.

At **configuration** $\mathcal{C}_{(3n+1)+2j+2}$:

(a) If the truth assignment $\sigma_{n,s}$ associated with a cell $(1,t)$ makes $C_1 \wedge \cdots \wedge C_{j+2}$ true, then

⋆ If $1 \le t \le 2^{n-1}$, rules (41) replace objects $T_i E_{j+2}$ from cell 1 by objects $e_{i,j+2}$ from cell 3, for some i such that $x_i \in C_{j+2}$, or objects $F_i E_{j+2}$ from cell 1 by objects $\overline{e}_{i,j+2}$ from cell 3, for some i such that $\overline{x}_i \in C_{j+2}$. Hence, such a cell 1 contains

$$\begin{cases} e_{i,j+2} + (\sigma_{n,s} - \{T_i\}) \text{ , if objects } T_i E_{j+2} \text{ have been exchanged} \\ e_{i,j+2} + (\sigma_{n,s} - \{F_i\}) \text{ , if objects } F_i E_{j+2} \text{ have been exchanged} \end{cases}$$

⋆ If $2^{n-1} + 1 \le t \le 2^n$, rules (41) either replace objects $T_i E_{j+2}$ or objects $T'_i E_{j+2}$ from cell 1 by objects $e_{i,j+2}$ from cell 3, for some i such that $x_i \in C_{j+2}$, either objects $F_i E_{j+2}$ or objects $F'_i E_{j+2}$ from cell 1 by objects $\overline{e}_{i,j+2}$ from cell 3, for some i such that $\overline{x}_i \in C_{j+2}$. Hence, such a cell 1 contains

$$\begin{cases} e_{i,j+2} + (\tau_{n,s} - \{T_i\}), \text{ if objects } T_i E_{j+2} \text{ have been exchanged} \\ e_{i,j+2} + (\tau_{n,s} - \{T'_i\}), \text{ if objects } T'_i E_{j+2} \text{ have been exchanged} \\ e_{i,j+2} + (\tau_{n,s} - \{F_i\}), \text{ if objects } F_i E_{j+2} \text{ have been exchanged} \\ e_{i,j+2} + (\tau_{n,s} - \{F'_i\}), \text{ if objects } F'_i E_{j+2} \text{ have been exchanged} \end{cases}$$

(b) Only rule (36) is applicable to cell 2 at configuration $\mathcal{C}_{(3n+1)+2j+2}$. So,

$$\mathcal{C}_{(3n+1)+2j+3}(2) = \{\alpha_{(3n+1)+2j+3}, \texttt{yes}, \texttt{no}\}$$

(c) Also rule (37) is applicable to cell 3 producing object $\beta_{(3n+1)+2j+3}$. Then, cell 3 contains:

- k_{j+2} copies of object E_{j+2}, being k_{j+2} the number of truth assignment making $C_1 \wedge \cdots \wedge C_{j+2}$ true.
- $(cod(\varphi))^{2^n}_{e,>j+2}$ representing 2^n copies of the objects $e_{i,j'}$ and $\overline{e}_{i,j'}$ such that $j' > j+2$ and $x_i \in C_{j'}$ in the first case, and $\neg x_i \in C_{j'}$ in the second one.
- Object $\beta_{(3n+1)+2j+3}$.
- Garbagge objects of the type T_i, T'_i, F_i, F'_i that will dissappear at the next step.
- Garbagge objects of the type $e_{i,j'}, \overline{e}_{i,j'}$ with $1 \le j' \le j+2$ that will not be considered anymore.

Hence, the result is also true for $j+1$. Then the proof of the theorem completes. □

Theorem 6.6 *Let* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ *be a computation of the tissue P system* $\Pi(\langle m, n\rangle)$. **At configuration** $\mathcal{C}_{(3n+1)+2m}$, *the following holds:*

(a) There are 2^n cells labelled by 1, and the formula φ is satisfiable if and only if there is, at least, one of such cell which contains object E_{m+1}.
(b) There is a cell labelled by 2. This cell contains objects $\alpha_{(3n+1)+2m}$, $\texttt{yes}, \texttt{no}$.
(c) There is a cell labelled by 3. This cell contains object $\beta_{(3n+1)+2m}$, and some irrelevant objects of the type $e_{i,j'}, \overline{e}_{i,j'}$ with $1 \le j' \le m$ that will not be considered anymore.

Proof: From Theorem 6.5, at configuration $\mathcal{C}_{(3n+1)+2(m-1)+1}$ we have:
(a) There are 2^n cells labelled by 1 each such that:
- ⋆ Let $\sigma_{n,s}$ a truth assignment associated with a cell $(1,t)$, where $1 \le t \le 2^n$, making $C_1 \wedge \cdots \wedge C_m$ <u>true</u>. Then
  - If $1 \le t \le 2^{n-1}$ then it contains $e_{i,m} + (\sigma_{n,s} - \{T_i\})$, for some i such that $x_i \in C_m$, or $\overline{e}_{i,m} + (\sigma_{n,s} - \{F_i\})$, for some i such that $\neg x_i \in C_m$.
  - If $2^{n-1}+1 \le t \le 2^n$ then it contains $e_{i,m} + (\tau_{n,s} - \{T'_i\})$, for some i such that $x_i \in C_m$, or $\overline{e}_{i,m} + (\tau_{n,s} - \{F'_i\})$, for some i such that $\neg x_i \in C_m$.
- ⋆ Let $\sigma_{n,s}$ a truth assignment associated with a cell $(1,t)$, where $1 \le t \le 2^n$, making $C_1 \wedge \cdots \wedge C_m$ <u>false</u>. Then their contents coincide with the corresponding contents in the previous configuration $\mathcal{C}_{(3n+1)+2(m-1)}$. In particular, that cell does not contain any object $e_{i,m}$ nor $\overline{e}_{i,m}$.

(b) There is a cell labelled by 2 which contains objects $\alpha_{(3n+1)+2(m-1)+1}$, $\texttt{yes}, \texttt{no}$.
(c) There is a cell labelled by 3 which contains object $\beta_{(3n+1)+2(m-1)+1}$, and:
- k_m copies of object E_m, being k_m the number of truth assignments making clauses $C_1, \dots, C_m$ true, that is, k_m is the number of truth assignment making true the formula φ.
- Some irrelevant objects of the type T_i, T'_i, F_i, F'_i that will dissappear at the next step.
- Some irrelevant objects of the type $e_{i,j'}, \overline{e}_{i,j'}$ with $1 \le j' \le m$ that will not be considered anymore.

Then

(a) Rule (43) produces objects E_{m+1} in every cell 1 which encodes a truth assignment making the formula φ true. Moreover, if a cell labelled by 1 encodes a truth assignment making the formula φ false, then it does not contain object E_{m+1}.
(b) Rule (36) produces object $\alpha_{(3n+1)+2m}$ in cell 2. Thus,
$$\mathcal{C}_{(3n+1)+2m}(2) = \{\alpha_{(3n+1)+2m}, \texttt{yes}, \texttt{no}\}$$
(c) Rules (44) and (45) remove objects $E_{m+1}, T_i, T'_i, F_i, F'_i$ from cell 3. In addition, rule (37) is applicable to cell 3 producing object $\beta_{(3n+1)+2m}$. Cell 3 also contains irrelevant objects of the type $e_{i,j'}, \overline{e}_{i,j'}$, with $1 \le j' \le m$, that appear at the previous configuration. □

Theorem 6.7 *Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ be a computation of the tissue P system $\Pi(\langle m, n\rangle)$.* **At configuration** $\mathcal{C}_{(3n+1)+2m+1}$*, the following holds:*

(a) There are 2^n cells labelled by 1. Besides,

- ⋆ *If the formula φ is satisfiable, then there is one (and only one) cell labelled by 1 which contains objects* $\alpha_{(3n+1)+2m}, \mathtt{yes}$.
- ⋆ *If the formula φ is not satisfiable, then their contents coincide with the contents in the previous configuration* $\mathcal{C}_{(3n+1)+2m}$.

(b) There is a cell labelled by 2. Besides,
- ⋆ *If the formula φ is satisfiable, then it contains objects* $E_{m+1}, \mathtt{no}$.
- ⋆ *If the formula φ is not satisfiable, then it contains objects* $\alpha_{(3n+1)+2m}, \mathtt{yes}, \mathtt{no}$.

(c) There is a cell labelled by 3. The contents of this cell is the same that in the previous configuration $\mathcal{C}_{(3n+1)+2m}$*, except object* $\beta_{(3n+1)+2m}$ *that evolves to* $\beta_{(3n+1)+2m+1}$.

Proof: At configuration $\mathcal{C}_{(3n+1)+2m+1}$:

(a) There are 2^n cells labelled by 1, and
- ⋆ If the formula φ is satisfiable, then there are cells labelled by 1 which contain objects E_{m+1}. Then, one (and only one) of these objects can be used to apply rule (46), allowing its trade for objects $\alpha_{(3n+1)+2m}, \mathtt{yes}$ from cell 2.
- ⋆ If the formula φ is not satisfiable, then their contents coincide with the contents in the previous configuration $\mathcal{C}_{(3n+1)+2m}$. In particular, rule (46) can not be applied to any cell labelled by 1, because any such cell encodes a truth assignment making the formula φ true.

(b) There is a cell labelled by 2 such that
- ⋆ If the formula φ is satisfiable, then

$$\mathcal{C}_{(3n+1)+2m+1}(2) = \{E_{m+1}, \mathtt{no}\}$$

- ⋆ If the formula φ is not satisfiable, then no rule of the system is applicable to that cell 2. Therefore,

$$\mathcal{C}_{(3n+1)+2m+1} = \{\alpha_{(3n+1)+2m}, \mathtt{yes}, \mathtt{no}\}$$

(c) There is a cell labelled by 3. Only rule (37) is applicable at this cell and produces object $\beta_{(3n+1)+2m+1}$.

□

Theorem 6.8 *Let* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ *be a computation of the tissue P system* $\Pi(\langle m, n\rangle)$. **At configuration** $\mathcal{C}_{(3n+1)+2m+2}$*, the following holds:*

(a) There are 2^n cells labelled by 1. Besides,
- ⋆ *If the formula φ is satisfiable, then there is one (and only one) cell labelled by 1 which contains objects* $\alpha_{(3n+1)+2m}$ *and* $\beta_{(3n+1)+2m+1}$.
- ⋆ *If the formula φ is not satisfiable, then their contents coincide with the contents in the previous configuration* $\mathcal{C}_{(3n+1)+2m+1}$.

(b) There is a cell labelled by 2. Besides,

- *If the formula φ is satisfiable, their contents coincide with the contents in the previous configuration $\mathcal{C}_{(3n+1)+2m+1}$.*
- *If the formula φ is not satisfiable, then it contains objects* yes, no, $\beta_{(3n+1)+2m+1}$.

(c) There is a cell labelled by 3. Besides,

- *If the formula φ is satisfiable, then it contains object* yes.
- *If the formula φ is not satisfiable, then it contains object $\alpha_{(3n+1)+2m}$.*

Proof: At configuration $\mathcal{C}_{(3n+1)+2m+1}$:

(a) There are 2^n cells labelled by 1, and

- If the formula φ is satisfiable, there is one (and only one) such cell 1 which contains objects $\alpha_{(3n+1)+2m}$, yes. By applying rule 47, object yes from such cell is traded for object $\beta_{(3n+1)+2m+1}$ from cell 3. Thus, there is one (and only one) cell 1 which contains objects $\alpha_{(3n+1)+2m}$ and $\beta_{(3n+1)+2m+1}$.
- If the formula φ is not satisfiable, then their contents coincide with the contents at the previous configuration $\mathcal{C}_{(3n+1)+2m+1}$. In particular, rule (47) cannot be applied to any cell labelled by 1.

(b) There is a cell labelled by 2. This cell verifies:

- If the formula φ is satisfiable, then any rule is applicable to such cell. Therefore,
$$\mathcal{C}_{(3n+1)+2m+2}(2) = \{E_{m+1}, \texttt{no}\}$$
- If the formula φ is not satisfiable, then rule (48) is applicable allowing the exchange of object $\alpha_{(3n+1)+2m}$ from cell 2 for object $\beta_{(3n+1)+2m+1}$ from cell 3. Hence,
$$\mathcal{C}_{(3n+1)+2m+2}(2) = \{\beta_{(3n+1)+2m+1}, \texttt{yes}, \texttt{no}\}$$

(c) There is a cell labelled by 3. This cell verifies:

- If the formula φ is satisfiable, then rule (47) produces object yes in this cell.
- If the formula φ is not satisfiable, then rule (48) produces object $\alpha_{(3n+1)+2m}$ in this cell.

□

Theorem 6.9 *Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots)$ be a computation of the tissue P system $\Pi(\langle m, n\rangle)$.* **At configuration** $\mathcal{C}_{(3n+1)+2m+3}$*, the following holds:*

(a) If the formula φ is satisfiable, then yes $\in \mathcal{C}_{(3n+1)+2m+3}(0)$.
(b) If the formula φ is not satisfiable, then no $\in \mathcal{C}_{(3n+1)+2m+3}(0)$.
(c) The configuration $\mathcal{C}_{(3n+1)+2m+3}$ is a halting configuration.

Proof:

(a) Let us suppose that formula φ is satisfiable. Then no rule is applicable to any cell labelled by 1 at configuration $\mathcal{C}_{(3n+1)+2m+2}$. Bearing in mind that $\mathcal{C}_{(3n+1)+2m+2}(2) = \{E_{m+1}, \texttt{no}\}$, and yes $\in \mathcal{C}_{(3n+1)+2m+2}(3)$, only rule (50) is applicable to configuration $\mathcal{C}_{(3n+1)+2m+2}$. Hence, yes $\in \mathcal{C}_{(3n+1)+2m+3}(0)$.

(b) Let us suppose that formula φ is not satisfiable. Then no rule is applicable to any cell labelled by 1 at configuration $\mathcal{C}_{(3n+1)+2m+2}$. Bearing in mind that $\mathcal{C}_{(3n+1)+2m+2}(2) = \{\beta_{3n+1+2m+1}, \mathtt{yes}, \mathtt{no}\}$, and $\alpha_{3n+1+2m} \in \mathcal{C}_{(3n+1)+2m+2}(3)$, only rule (49) is applicable to configuration $\mathcal{C}_{(3n+1)+2m+2}$. Hence, $\mathtt{no} \in \mathcal{C}_{(3n+1)+2m+3}(0)$.

(c) From (a) and (b), it is easy to check that no rule of the system is applicable to configuration $\mathcal{C}_{(3n+1)+2m+3}$.

□

Corollary 6.10 *The family* $\mathbf{\Pi}$ *is polynomially bounded.*

Proof: From Theorem 6.9 we deduce that any computation $\mathcal{C}$ of the tissue P system $\Pi(\langle m, n\rangle)$ spends $(3n+1)+2m+3 = 3n+2m+4$ transition steps exactly.
□

6.3 Computational Efficiency of TSC(3)

The family of tissue P systems with cell separation constructed in Section 5 verifies the following:

(a) The defined family $\mathbf{\Pi}$ is *consistent*, in the sense that all systems of the family are recognizer tissue P systems with cell separation: (1) the working alphabet Γ has two distinguished objects `yes` and `no`, at least one copy of them present in some initial multisets but none of them are present in $\mathcal{E}$; (2) the output region i_{out} is the environment; (3) all computations halt; and (4) if $\mathcal{C}$ is a computation of a system, then either object `yes` or object `no` (but not both) has been released into the environment, and only at the last step of the computation. Besides, these systems use communication rules with length at most 3.
(b) The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines (Subsection 5.1).
(c) (cod, s) is a pair of polynomial–time computable functions.
(d) The family $\mathbf{\Pi}$ is polynomially bounded with regard to $(\mathtt{SAT}, cod, s)$ (Corollary 5.10).
(e) The family $\mathbf{\Pi}$ is sound and complete with regard to $(\mathtt{SAT}, cod, s)$ (Subsection 5.2).

Therefore, according to Definition 1, the uniform family $\mathbf{\Pi}$ of tissue P systems constructed in Section 5 solve the `SAT` problem in polynomial time with respect to the number of variables and the number of clauses.

Hence, we have the following result:

Theorem 6.11 $\mathtt{SAT} \in \mathbf{PMC}_{TSC(3)}$.

Corollary 6.12 $\mathbf{NP} \cup \mathbf{co-NP} \subseteq \mathbf{PMC}_{TSC(3)}$.

Proof: It suffices to notice that the `SAT` problem is **NP**-complete, $\mathtt{SAT} \in \mathbf{PMC}_{TSC(3)}$, and this complexity class is closed under polynomial-time reduction and under complement.

□

7 Conclusions and Future Work

The space-time tradeoff method is used to efficiently solve computationally hard problems in the framework of *Membrane Computing*. The efficiency of tissue P systems with cell division for solving **NP**-complete problems has been previously studied [4, 5, 20]. Cell division rules allow the duplication of all objects in the new created cells except the object that activate the cell division operation. Therefore, the cell division can be used to generate an exponential workspace, expressed in terms of the number of cells and the number of objects, in linear time.

In the framework of tissue P systems with cell division, the length of communication rules provide a frontier for the tractability of decision problems. In [8] the limitation on the efficiency of tissue P systems with cell division and communication rules of length 1 it has been established that only tractable problems can be solved efficiently in that framework. Nevertheless, in [5] a linear time solution to `Vertex Cover` problem by using a family of tissue P systems with cell division and communication rules of length at most 3 has been provided. Hence, in tissue P systems with cell division, passing from communication rules of length 1 to communication rules of length at most 3 amounts to passing from non–efficiency to efficiency, assuming that $\mathbf{P} \neq \mathbf{NP}$.

Recently [15], cell separation rules have been introduced into tissue P systems, inspired by the cellular fission, and its computational efficiency was investigated. This kind of rules allows the creation of two new cells from one cell although there is no replication of objects between the new cells, that is, the contents of the cell is distributed between the new created cells, except the object triggering the rule which is consumed. Therefore, by using cell separation it is possible to construct an exponential workspace, expressed only in terms of the number of cells, in linear time. In [15] two important results were obtained in that framework: (a) only tractable problems can be efficiently solved by using cell separation and communication rules with length at most 1, and (b) a uniform and linear time solution to the `SAT` problem by using cell separation and communication rules with length <u>at most 8</u> was presented.

In this paper, the previous result has been improved by showing a family of tissue P systems with cell separation and communication rules with length <u>at most 3</u>, solving the `SAT` problem in a uniform way and linear time. Hence, with regard to tissue P systems with cell separation, a similar result concerning the frontier of tractability can be formulated in the new framework: by using families of tissue P systems with cell separation, passing from communication rules of length 1 to communication rules of length at most 3, amounts to passing from non–efficiency to efficiency, assuming that $\mathbf{P} \neq \mathbf{NP}$. It is worth to highlight that separation rules seem weaker than division rules from the point of view of computational complexity.

Next, we propose several open problems related to the efficiency of tissue P systems:

(a) What is the computational efficiency of tissue P systems with cell separation or with cell division, and communication rules with length at most 2 are allowed?
(b) What happens if only symport (respectively, only antiport) rules are allowed in tissue P systems with cell division or cell separation?
(c) In [4] tissue P systems with cell division and without environment were introduced, that is, tissue P systems where the alphabet $\mathcal{E}$ of the environment is empty. In this kind of P systems there are no objects appearing in the system in arbitrary copies each. What is the relationship between the polynomial complexity classes of tissue P systems with cell division (respectively, with cell separation) and the corresponding tissue P systems without environment?

Acknowledgements

The work of the first author was supported by Project TIN2009-13192 of the Ministerio de Ciencia e Innovación of Spain and Project of Excellence with *Investigador de Reconocida Valía*, from Junta de Andalucía, grant P08 – TIC 04200. The work of the second author was supported by the Silesian University in Opava under the Student Funding Scheme, project no SGS/7/2011.

This work was also supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070).

References

1. Alhazov, A., Freund, R. and Oswald, M. Tissue P Systems with Antiport Rules ans Small Numbers of Symbols and Cells. *Lecture Notes in Computer Science* **3572**, (2005), 100–111.
2. Bernardini, F. and Gheorghe, M. Cell Communication in Tissue P Systems and Cell Division in Population P Systems. *Soft Computing* **9**, 9, (2005), 640–649.
3. Ciobanu, G, Păun, Gh. and Pérez-Jiménez, M.J. *Applications of Membrane Computing*, Natural Computing Series, Springer, 2006.
4. Christinal, H.A., Díaz-Pernil, D., Gutiérrez-Naranjo, M.A. and Pérez-Jiménez, M.J. Tissue-like P systems without environment. In M.A. Martínez-del-Amor, Gh. Păun, I. Pérez-Hurtado, A. Riscos-Núñez (eds.) *Proceedings of the Eight Brainstorming Week on Membrane Computing*, Sevilla, Spain, February 1-5, 2010, Fénix Editora, Report RGNC 01/2010, pp. 53–64.
5. Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez , A. and Romero–Campero, F.J. Computational efficiency of cellular division in tissue-like P systems. *Romanian Journal of Information Science and Technology* **11**, 3, (2008), 229–241.
6. Freund, R., Păun, Gh. and Pérez-Jiménez, M.J. Tissue P Systems with channel states. *Theoretical Computer Science* **330**, (2005), 101–116.
7. Garey, M.R. and Johnson, D.S. *Computers and Intractability A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, (1979).
8. Gutiérrez-Escudero, R., Pérez-Jiménez, M.J. and Rius–Font, M. Characterizing tractability by tissue-like P systems. *Lecture Notes in Computer Science* **5957**, (2010), 289–300.

9. Ito, M., Martín Vide, C. and Păun, Gh. A characterization of Parikh sets of ET0L laguages in terms of P systems. In M. Ito, Gh. Păun, S. Yu (eds.) *Words, Semigroups and Transducers*, World Scientific, Singapore, 2001, 239-254.
10. Krishna, S.N., Lakshmanan K. and Rama, R. Tissue P Systems with Contextual and Rewriting Rules. *Lecture Notes in Computer Science* **2597**, (2003), 339–351.
11. Lakshmanan K. and Rama, R. On the Power of Tissue P Systems with Insertion and Deletion Rules. In A. Alhazov, C. Martín-Vide and Gh. Păun (eds.) *Preproceedings of the Workshop on Membrane Computing*, Tarragona, Report RGML 28/03, (2003), pp. 304–318.
12. Martín Vide, C. Pazos, J. Păun, Gh. and Rodríguez Patón, A. A New Class of Symbolic Abstract Neural Nets: Tissue P Systems. *Lecture Notes in Computer Science* **2387**, (2002), 290–299.
13. Martín Vide, C. Pazos, J. Păun, Gh. and Rodríguez Patón, A. Tissue P systems. *Theoretical Computer Science*, **296**, (2003), 295–326.
14. Pan, L. and Ishdorj, T.-O. P systems with active membranes and separation rules. *Journal of Universal Computer Science*, **10**, 5, (2004), 630–649.
15. Pan, L. and Pérez-Jiménez, M.J. Computational complexity of tissue–like P systems. *Journal of Complexity*, **26**, 3 (2010), 296–315.
16. Păun, Gh. Computing with membranes. *Journal of Computer and System Sciences*, **61**, 1, (2000), 108–143. Also in Turku Center for Computer Science–TUCS, Report 208, November 1998.
17. Păun, Gh. Attacking **NP**-complete problems. In *Unconventional Models of Computation, UMC'2K* (I. Antoniou, C. Calude, M. J. Dinneen, eds.), Springer-Verlag, 2000, pp. 94-115.
18. Păun, Gh. *Membrane Computing. An Introduction*. Springer–Verlag, Berlin, (2002).
19. Păun, A. and Păun, Gh. The power of communication: P systems with symport/antiport. *New Generation Computing*, **20**, 3, (2002), 295–305.
20. Păun, Gh., Pérez-Jiménez, M.J. and Riscos-Núñez, A. Tissue P System with cell division. *In. J. of Computers, Communications and Control*, **3**, 3, (2008), 295–303.
21. Gh. Păun, G. Rozenberg and A. Salomaa. *The Oxford Handbook of Membrane Computing*, Oxford University Press, 2009.
22. Pérez-Jiménez, M.J., Romero-Jiménez, A. and Sancho-Caparrini, F. Complexity classes in models of cellular computing with membranes. *Natural Computing*, **2**, 3 (2003), 265–285.
23. Pérez-Jiménez, M.J., Romero-Jiménez, A. and Sancho-Caparrini, F. A polynomial complexity class in P systems using membrane division. *Journal of Automata, Languages and Combinatorics*, **11**, 4, (2006), 423-434.
24. Prakash, V.J. On the Power of Tissue P Systems Working in the Maximal-One Mode. In A. Alhazov, C. Martín-Vide and Gh. Păun (eds.). *Preproceedings of the Workshop on Membrane Computing*, Tarragona, Report RGML 28/03, (2003), pp. 356–364.
25. ISI web page `http://esi-topics.com/erf/october2003.html`
26. P systems web page `http://ppage.psystems.eu/`