Narrowing Frontiers of Efficiency with Evolutional Communication Rules and Cell Separation

David Orellana-Martín[1], Luis Valencia-Cabrera[1], Bosheng Song[2], Linqiang Pan[2,3], Mario J. Pérez-Jiménez[1]

[1]Research Group on Natural Computing,
Department of Computer Science and Artificial Intelligence,
Universidad de Sevilla
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
E-mail: {dorellana, lvalencia, marper}@us.es
Key [2]Laboratory of Image Information Processing and Intelligent Control of Education Ministry of China, School of Automation, Huazhong University of Science and Technology, Wuhan, Hubei, 430074, China
E-mail: boshengsong@163, lqpan@mail.hust.edu.cn
[3]School of Electric and Information Engineering,
Zhengzhou University of Light Industry, Zhengzhou 450002, China

Summary. In the framework of *Membrane Computing*, several efficient solutions to computationally hard problems have been given. To find new borderlines between families of P systems that can solve them and the ones that cannot is an important way to tackle the **P** versus **NP** problem. Adding syntactic and/or semantic ingredients can mean passing from non-efficiency to *presumably* efficiency. Here, we try to get narrow frontiers, setting the stage to adapt efficient solutions from a family of P systems to another one. In order to do that, a solution to the SAT problem is given by means of a family of tissue P systems with evolutional symport/antiport rules and cell separation with the restriction that both the left-hand side and the right-hand side of the rules have at most two objects.

Key words: Membrane Computing, symport/antiport rules, the **P** versus **NP** problem, SAT problem.

1 Introduction

Membrane Computing is a bio-inspired computing paradigm based on the structure and behavior of living cells. There are several classes of P systems, the computational models of this paradigm. It was first introduced in [7], defining one of the main models, cell-like P systems that abstract the hierarchical arrangement of membranes within a single cell. In [4], the idea of the interactions of networks of cells (placed in the nodes of a directed graph) between cells and between cells and their environment is used to develop tissue-like P systems, named by the ensemble of cells in living beings. Another approach with the same structure are the so-called spiking neural P systems [2], SN P systems for short, inspired by the way that neurons communicate with each other by means of short electrical impulses (spikes).

Within these models, several variants can be defined only by changing syntactic and/or semantic ingredients, such as kinds of rules possible, length of rules, parallelism permitted, number of objects and so on. Computational complexity theory in the framework of Membrane Computing uses special variants of P systems called *recognizer* membrane systems, devices that, given an initial configuration depending on an instance of a decision problem, return *yes* or *no* depending of the answer to such instance. A deep vision of complexity can be seen in [8, 9].

Tissue P systems have been widely investigated from this point of view, giving characterizations for most of their variants. For instance, in [1] and [11], the borderline of efficiency for tissue P systems with symport/antiport rules and cell division by means of the length of communication rules is given, that is, passing from 1 to 2 means passing from non-efficiency to presumably efficiency. In [5] and [10], a similar result is given for tissue P systems with symport/antiport rules and cell separation, but in this case, rules with length at most 3 are needed in order to solve efficiently computationally hard problems. Thus, three frontiers of efficiency can be found here: two described before by means of the length of the rules, and the third one when using rules with length at most 2, between separation and division rules.

In [12], a new variant of these systems is defined. Based on the chemical reactions within cells and how reactives evolve into new components, evolutional communication rules are described as a movement of components between different cells or a cell and the environment but within the reaction objects can change into something new. It is interesting to study these systems from the computational complexity theory point of view, and in [6], an efficient solution to the SAT problem is given by these systems with some restrictions about the length of their rules, but the narrowest borderline is not defined. The purpose of this paper is to tight it.

The paper is organized as follows: first, we recall some concepts that are going to be used through the work. In Section 3 the framework of tissue P systems with evolutional symport/antiport rules is introduced. After that, Sections 4 and 5 are devoted to give a solution to SAT by means of a family of P systems with evolutional symport/antiport rules with cell separation and rules with length at most $(2, 2)$ and a formal verification of a design. Finally, some conclusions and open research lines are exposed.

2 Preliminaries

In order to make this work self-contained, we introduce some notions that are going to be used through the paper.

2.1 Alphabets and sets

An *alphabet* Γ is a non-empty set and their elements are called *symbols*. A *string* u over Γ is an ordered finite sequence of symbols, that is, a mapping from a natural number $n \in \mathbb{N}$ onto Γ. The number n is called the *length* of the string u and it is denoted by $\mid u \mid$. The empty string (with length 0) is denoted by λ. The set of all strings over an alphabet Γ is denoted by Γ^*. A *language* over Γ is a subset of Γ^*.

A *multiset* over an alphabet Γ is an ordered pair (Γ, f) where f is a mapping from Γ onto the set of natural numbers $\mathbb{N}$. The *support* of a multiset $m = (\Gamma, f)$ is defined as $supp(m) = \{x \in \Gamma \mid f(x) > 0\}$. A multiset is finite (resp., empty) if its support is a finite (resp., empty) set. We denote by $\emptyset$ the empty multiset and we denote by $M(\Gamma)$ the set of all multisets over Γ.

Let $m_1 = (\Gamma, f_1)$, $m_2 = (\Gamma, f_2)$ be multisets over Γ, then the union of m_1 and m_2, denoted by $m_1 + m_2$, is the multiset (Γ, g), when $g(x) = f_1(x) + f_2(x)$ for each $x \in \Gamma$.

2.2 Decision problems

A decision problem X can be informally defined as one whose solution is either *yes* or *no*. This can be formally defined by an ordered pair (I_X, θ_X), where I_X is a language over a finite alphabet Σ_X and θ_X is a total Boolean function over I_X. The elements of I_X are called *instances* of the problem X. Each decision problem X has associated a language L_X over the alphabet Σ_X as follows: $L_X = \{u \in E_X \mid \theta_X(u) = 1\}$. Conversely, every language L over an alphabet Σ has associated a decision problem $X_L = (I_{X_L}, \theta_{X_L})$ as follows: $I_{X_L} = \Sigma^*$ and $\theta_{X_L}(u) = 1$ if and only if $u \in L$. Then, given a decision problem X we have $X_{L_X} = X$, and given a language L over an alphabet Σ we have $L_{X_L} = L$.

It is worth pointing out that any Turing machine M (with input alphabet Σ_M) has associated a *decision* problem $X_M = (I_M, \theta_M)$ defined as follows: $I_M = \Sigma_M^*$, and for every $u \in \Sigma_M^*$, $\theta_M(u) = 1$ if and only if M accepts u. Obviously, the decision problem X_M is solvable by the Turing machine.

3 Tissue P systems with evolutional communication rules

Definition 1. *A recognizer tissue P system with evolutional symport/antiport rules and cell separation of degree $q \geq 1$ is a tuple*

$$\Pi = (\Gamma, \Gamma_0, \Gamma_1, \Sigma, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}, i_{out})$$

where:

- *Γ and $\mathcal{E}$ are finite alphabets whose elements are called objects;*
- *Γ_0 and Γ_1 is a partition of Γ;*
- *$\mathcal{E} \subseteq \Gamma$;*

- *$\mathcal{M}_q, \ldots, \mathcal{M}_q$ are multisets over Γ;*
- *$\mathcal{R}$ is a finite set of rules, of the following forms:*
  1. *Evolutional communication rules:*
     a) *$[\,u\,]_i[\quad]_j \rightarrow [\quad]_i[\,u'\,]_j$, where $1 \leq i, j \leq q$, $i \neq j$, $u \in M_f^+(\Gamma)$ and $u' \in M_f(\Gamma)$ (evolutional symport rules);*
     b) *$[\,u\,]_i[\,v\,]_j \rightarrow [\,v'\,]_i[\,u'\,]_j$, where $1 \leq i, j \leq q$, $i \neq j$, $u, v \in M_f^+(\Gamma)$ and $u', v' \in M_f(\Gamma)$ (evolutional antiport rules);*
  2. *$[\,a\,]_i \rightarrow [\,\Gamma_0]_i[\,\Gamma_1]_i$, where $i \in \{1, \ldots, q\}, i \neq i_{out}$ and $a \in \Gamma$; (separation rules);*
- *$i_{out} \in \{0, 1, \ldots, q\}$.*

A recognizer tissue P system with evolutional symport/antiport rules and cell separation of degree $q \geq 1$

$$\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$$

can be viewed as a set of q cells, labelled by $1, \ldots, q$ such that (a) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ represent the multisets of objects initially placed in the q cells of the system; (b) $\mathcal{E}$ is the set of objects initially located in the environment of the system, all of them available in an arbitrary number of copies; (c) i_{out} represents a distinguished *region* which will encode the output of the system. We use the term *region* i $(0 \leq i \leq q)$ to refer to cell i in the case $1 \leq i \leq q$ and to refer to the environment in the case $i = 0$.

A *configuration* at any instant of a tissue P system with evolutional symport/antiport rules and cell separation is described by the multisets of objects in each cell and the multiset of objects over $\Gamma \setminus \mathcal{E}$ in the environment at that moment. The initial configuration of $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$ is $\mathcal{M}_1, \ldots, \mathcal{M}_q; \emptyset)$.

An evolutional symport rule $[\,u\,]_i[\quad]_j \rightarrow [\quad]_i[\,u'\,]_j$ is applicable at a configuration $\mathcal{C}_t$ at an instant t if there is a region i from $\mathcal{C}_t$ which contains multiset u. By applying an eovlutional symport rule, the multiset of objects in region i from $\mathcal{C}_t$ is consumed and the multiset of objects u' is generated in region j from $\mathcal{C}_{t+1}$.

An evolutional symport rule $[\,u\,]_i[\,v\,]_j \rightarrow [\,v'\,]_i[\,u'\,]_j$ is applicable at a configuration $\mathcal{C}_t$ at an instant t if there is a region i from $\mathcal{C}_t$ which contains multiset u and there is a region j which contains multiset v. By applying an eovlutional symport rule, the multiset of objects u in region i and multiset of objects v in region j from $\mathcal{C}_t$ are consumed and the multiset of objects u' is generated in region j and the multiset of objects v' in region i from $\mathcal{C}_{t+1}$.

A separation rule $[\,a\,]_i \rightarrow [\,\Gamma_0\,]_i[\,\Gamma_1\,]_i$ is applicable at a configuration $\mathcal{C}_t$ at an instant t if there is a cell i from $\mathcal{C}_t$ which contains object a and $i \neq i_{out}$. By applying a separation rule to such a cell i, (a) object a is consumed from such cell; (b) two new cells with label i are generated at configuration $\mathcal{C}_{t+1}$; and (c) objects from Γ_0 from the original cell are placed in one of the new cells, while objects from Γ_1 from the original cell are placed in the other one.

The rules of a tissue P system with evolutional symport/antiport rules and cell separation are applied in a maximally parallel manner, following the previous remarks, and taking into account that when a cell i is being separated at one transition step, no other rules can be applied to that cell i at that step.

A transition from a configuration $\mathcal{C}_t$ to another configuration $\mathcal{C}_{t+1}$ is obtained by applying rules in a maximally parallel manner following the previous remarks. A *computation* of the system is a (finite or infinite) sequence of transitions starting from the initial configuration, where any term of the sequence other than the first one is obtained from the previous configuration in one transition step. If the sequence is finite (called *halting computation*) then the last term of the sequence is a *halting configuration*, that is, a configuration where no rule is applicable to it. A computation gives a result only when a halting configuration is reached, and that result is encoded by the multiset of objects present in the output region i_{out}.

A natural framework to solve decision problems is to use recognizer P systems.

Definition 2. *A recognizer tissue P system with evolutional symport/antiport rules and cell separation of degree $q \geq 1$ is a tuple*

$$\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out}),$$

where

- *the tuple $(\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}, i_{out})$ is a tissue P system with evolutional symport/antiport rules of degree $q \geq 1$, where Γ strictly contains an (input) alphabet Σ and two distinguished objects* `yes` *and* `no`*, and $\mathcal{M}_i$ $(1 \leq i \leq q)$ are multisets over $\Gamma \setminus \Sigma$;*
- *$i_{in} \in \{1, \dots, 1\}$ is the input cell and i_{out} is the label of the environment;*
- *for each multiset m over the input alphabet Σ, any computation of the system Π with input m starts from the configuration of the form $(\mathcal{M}_1, \dots, \mathcal{M}_{i_{in}} + m, \dots, \mathcal{M}_q; \emptyset)$, it always halts and either object* `yes` *or object* `no` *(but not both) must appear in the environment at the last step.*

For each ordered pair of natural numbers (k_1, k_2) greater or equal to 1, the class of recognizer P systems with evolutional symport/antiport rules and cell separation with evolutional communication rules of length at most (k_1, k_2) is denoted by $\mathbf{TSEC}(k_1, k_2)$. This means that, given an evolutional communication rule $[\,u\,]_i[\,v\,]_j \to [\,v'\,]_i[\,u'\,]_j$ the LHS (resp., RHS) of any evolutional communication rule in a system from $\mathbf{TSEC}(k_1, k_2)$ involves at most $k_1 = |u| + |v|$ objects (resp., $k_2 = |u'| + |v'|$ objects).

Next, we define the concept of solving a problem in a uniform way and in polynomial time by a family of recognizer tissue P systems with evolutional symport/antiport rules and cell separation.

Definition 3. *A decision problem $X = (I_X, \theta_X)$ is solvable in a uniform way and in polynomial time by a family $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N})\}$ of recognizer tissue P systems with evolutional symport/antiport rules and cell separation if the following conditions hold:*

1. *the family $\mathbf{\Pi}$ is polynomially uniform by Turing machines; and*

2. *there exists a polynomial encoding (cod, s) of I_X in $\mathbf{\Pi}$ 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; and (c) the family $\mathbf{\Pi}$ is polynomially bounded, sound and complete with regard to (X, cod, s).*

The set of all decision problems that can be solved by recognizer tissue P systems with evolutional symport/antiport rules and cell separation with evolutional communication rules of length at most (k_1, k_2) in a uniform way and polynomial time is denoted by $\mathbf{PMC}_{\mathbf{TSEC}(k_1,k_2)}$.

4 Solution to SAT with evolutional communication rules and separation rules

In [6] an efficient solution to the SAT problem is given by means of a family of P systems from $\mathbf{TSEC}(3,2)$. A frontier of efficiency is given, but some open problems remain, as indicate Figure 1 of such work. It shows that the class of problems that can be solved by P systems from $\mathbf{TSEC}(2,k)$ with $k \geq 2$ is unknown. In this work we improve this borderline closing the previous open questions, giving an efficient solution of the SAT problems by means of a family of P systems from $\mathbf{TSEC}(2,2)$.

Let us briefly recall the description of the SAT problem: given a boolean formula in conjunctive normal form (CNF), to determine whether or not there exists an assignment to its variables, called truth assignment, on which it evaluates true.

Theorem 1. SAT $\in \mathbf{PMC}_{\mathbf{TSEC}(2,2)}$

For each $n, p \in \mathbb{N}$, we consider the recognizer P system

$$\Pi(\langle n,p\rangle) = (\Gamma, \Gamma_0, \Gamma_1, \Sigma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$$

from $\mathbf{TSEC}(2,2)$ defined as follows:

1. Working alphabet Γ:
   $\{\mathtt{yes}, \mathtt{no}, y_1, y_2, n_1, n_2, \#\} \cup$
   $\{a_{i,j} \mid 1 \leq i \leq n, 0 \leq j \leq i\} \cup$
   $\{a'_{i,j} \mid 2 \leq i \leq n, 0 \leq j \leq i-1\} \cup$
   $\{a^L_{i,j}, a^R_{i,j} \mid 2 \leq i \leq n, 1 \leq j \leq i-1\} \cup$
   $\{\alpha_j, \alpha'_j, \alpha^L_j, \alpha^R_j \mid 1 \leq j \leq p+1\} \cup$
   $\{t_i, f_i, t'_i, t''_i f''_i, t^L_i, t^R_i, f^L_i, f^R_i \mid 1 \leq i \leq n\} \cup$
   $\{\beta_{l,k}, \beta'_{l,k}, \beta^L_{l,k}, \beta^R_{l,k} \mid 0 \leq k \leq n, 1 \leq l \leq n\} \cup$
   $\{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n+j-1\} \cup$
   $\{x'_{i,j,k}, \overline{x}'_{i,j,k}, {x^*}'_{i,j,k}, x''_{i,j,k}, \overline{x}''_{i,j,k}, {x^*}''_{i,j,k}, x'''_{i,j,k}, \overline{x}'''_{i,j,k}, {x^*}'''_{i,j,k}, \mid$
   $0 \leq i \leq n, 1 \leq j \leq p, 1 \leq k \leq n\} \cup$
   $\{c_{j,k} \mid 1 \leq j \leq p, j \leq k \leq p\} \cup \{\delta_i \mid 0 \leq i \leq 4n+p+2\} \cup$
   $\{\delta'_i \mid 0 \leq i \leq 4n+p\}$.

2. $\Gamma_1 = \Gamma \setminus \Gamma_0$, $\Gamma_0 = \{a^L_{i,j} \mid 2 \le i \le n, 1 \le j \le i-1\} \cup$
   $\{\alpha^L_j \mid 1 \le j \le p+1\} \cup \{t^L_i, f^L_i \mid 1 \le i \le n\} \cup$
   $\{\beta^L_{l,k} \mid 0 \le k \le n, k+1 \le l \le n\}$
3. Input alphabet Σ: $\{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,k,0} \mid 1 \le i \le n, 1 \le j \le p\}$.
4. Environment alphabet $\mathcal{E}$: $\{\gamma\}$.
5. $\mathcal{M}_1 = \{\delta_0, \delta'_0\} \cup \{\beta^{n+p+1}_{l,0} \mid 1 \le l \le n\}$,
   $\mathcal{M}_2 = \{a_{i,0} \mid 1 \le i \le n\} \cup \{\alpha_j \mid 1 \le j \le p+1\}$.
6. The set of rules $\mathcal{R}$ consists of the following rules:

1.1 Rules for $(4k+1)$-th steps.

$$[\,a_{i,i-1}\,]_2[\,\gamma\,]_0 \to [\,a'_{i,i-1}\,t'_i\,]_2[\quad]_0 \;\text{, for } 1 \le i \le n$$

$$\left.\begin{array}{l}[\,t_i\,]_2[\,\gamma\,]_0 \to [\,t''_i\,]_2[\quad]_0\\ [\,f_i\,]_2[\,\gamma\,]_0 \to [\,f''_i\,]_2[\quad]_0\end{array}\right\}\text{for } 1 \le i \le n$$

$$[\,a_{i,j}\,]_2[\,\gamma\,]_0 \to [\,a'_{i,j}\,]_2[\quad]_0 \;\text{, for } 2 \le i \le n, 0 \le j \le i-2$$

$$[\,\alpha_j\,]_2[\,\gamma\,]_0 \to [\,\alpha'_j\,]_2[\quad]_0 \;\text{, for } 1 \le j \le p+1$$

$$[\,\beta_{l,k}\,]_1[\,\gamma\,]_0 \to [\,\beta'_{l,k}\,]_1[\quad]_0 \,\}\,\text{for}\begin{array}{c}0 \le k \le n,\\ k+1 \le l \le n\end{array}$$

$$\left.\begin{array}{l}[\,x_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,x'_{i,j,k}\,]_1[\quad]_0\\ [\,\overline{x}_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}'_{i,j,k}\,]_1[\quad]_0\\ [\,x^*_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,x^{*\prime}_{i,j,k}\,]_1[\quad]_0\end{array}\right\}\text{for}\begin{array}{c}1 \le i \le n,\\ 1 \le j \le p,\\ 0 \le k \le n-1\end{array}$$

1.2 Rules for $(4k+2)$-th steps.

$$\left.\begin{array}{l}[\,a'_{i,i-1}\,]_2[\,\gamma\,]_0 \to [\,a_{i,i}\,f^R_i\,]_2[\quad]_0\\ [\,t'_i\,]_2[\,\gamma\,]_0 \to [\,t^L_i\,]_2[\quad]_0\end{array}\right\}\text{for } 1 \le i \le n$$

$$\left.\begin{array}{l}[\,t''_i\,]_2[\,\gamma\,]_0 \to [\,t^L_i\,t^R_i\,]_2[\quad]_0\\ [\,f''_i\,]_2[\,\gamma\,]_0 \to [\,f^L_i\,f^R_i\,]_2[\quad]_0\end{array}\right\}\text{for } 1 \le i \le n$$

$$[\,a'_{i,j}\,]_2[\,\gamma\,]_0 \to [\,a^L_{i,j+1}\,a^R_{i,j+1}\,]_2[\quad]_0 \;\text{, for}\begin{array}{c}2 \le i \le n,\\ 0 \le j \le i-1\end{array}$$

$$[\,\alpha'_j\,]_2[\,\gamma\,]_0 \to [\,\alpha^L_j\,\alpha^R_j\,]_2[\quad]_0 \;\text{, for } 1 \le j \le p+1$$

$$[\,\beta'_{l,k}\,]_1[\,\gamma\,]_0 \to [\,\beta^L_{l,k+1}\,\beta^R_{l,k+1}\,]_1[\quad]_0 \;\text{, for}\begin{array}{c}0 \le k \le n,\\ k+1 \le l \le n\end{array}$$

$$\left.\begin{array}{l}[\,x'_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,x''^{2}_{i,j,k+1}\,]_1[\quad]_0\\ [\,\overline{x}'_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}''^{2}_{i,j,k+1}\,]_1[\quad]_0\\ [\,x^{*\prime}_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,x^{*\prime\prime 2}_{i,j,k+1}\,]_1[\quad]_0\end{array}\right\}\begin{array}{l}1 \le i \le n,\\ 1 \le j \le p,\\ 0 \le k \le n-1\end{array}$$

1.3 Rules for $(4k+3)$-th steps.

$$[\,a_{i,i}\,]_2 \to [\,\Gamma_0\,]_2[\,\Gamma_1\,]_2 \;\text{, for } 1 \le i \le n$$

$$\left.\begin{array}{l}[\,\beta^O_{k,k}\,]_1[\quad]_0 \to [\quad]_1[\,\beta^O_{k,k}\,]_0\\ [\,\beta^O_{l,k}\,]_1[\quad]_0 \to [\quad]_1[\,\beta_{l,k}\,]_0\end{array}\right\}\text{for}\begin{array}{c}O \in \{L,R\},\\ 1 \le k \le n,\\ k+1 \le l \le n\end{array}$$

$$\left.\begin{array}{l}[\,x''_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,x'''_{i,j,k}\,]_1[\quad]_0\\ [\,\overline{x}''_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}'''_{i,j,k}\,]_1[\quad]_0\\ [\,x^{*\prime\prime}_{i,j,k}\,]_1[\,\gamma\,]_0 \to [\,x^{*\prime\prime\prime}_{i,j,k}\,]_1[\quad]_0\end{array}\right\}\text{for}\begin{array}{c}1 \le i \le n,\\ 1 \le j \le p,\\ 1 \le k \le n\end{array}$$

1.4 Rules for $(4k)$-th steps.

$$\left.\begin{array}{l} [\,a^O_{i,j}\,]_2[\,\beta^O_{k,k}\,]_0 \rightarrow [\,a_{i,j}\,]_2[\;\;]_0 \\ [\,r^O_i\,]_2[\,\beta^O_{k,k}\,]_0 \rightarrow [\,r_i\,]_2[\;\;]_0 \end{array}\right\} \text{for} \begin{array}{c} O \in \{L,R\}, \\ r \in \{t,f\}, \\ 1 \leq i \leq n, \\ 1 \leq j \leq n, \\ 1 \leq k \leq n \end{array}$$

$$[\,\alpha^O_j\,]_2[\,\beta^O_{k,k}\,]_0 \rightarrow [\,\alpha_j\,]_2[\;\;]_0 \;,\ \text{for} \begin{array}{c} O \in \{L,R\}, \\ 1 \leq j \leq p+1, \\ 0 \leq k \leq n \end{array}$$

$$\left.\begin{array}{l} [\,x'''_{i,j,k}\,]_1[\,\gamma\,]_0 \rightarrow [\,x_{i,j,k}\,]_1[\;\;]_0 \\ [\,\overline{x}'''_{i,j,k}\,]_1[\,\gamma\,]_0 \rightarrow [\,\overline{x}_{i,j,k}\,]_1[\;\;]_0 \\ [\,x^{*'''}_{i,j,k}\,]_1[\,\gamma\,]_0 \rightarrow [\,x^*_{i,j,k}\,]_1[\;\;]_0 \end{array}\right\} \text{for} \begin{array}{c} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 0 \leq k \leq n \end{array}$$

$$[\;\;]_1[\,\beta_{l,k}\,]_0 \rightarrow [\,\beta_{l,k}\,]_1[\;\;]_0 \;,\ \text{for } 0 \leq k \leq n, k+1 \leq l \leq n$$

2.1 Rules to check satisfied clauses.

$$\left.\begin{array}{l} [\,t_i\,]_2[\,x_{i,j,n+j-1}\,]_1 \rightarrow [\,c_{j,j}\,t_i\,]_2[\;\;]_1 \\ [\,t_i\,]_2[\,\overline{x}_{i,j,n+j-1}\,]_1 \rightarrow [\,t_i\,]_2[\;\;]_1 \\ [\,t_i\,]_2[\,x^*_{i,j,n+j-1}\,]_1 \rightarrow [\,t_i\,]_2[\;\;]_1 \\ [\,f_i\,]_2[\,x_{i,j,n+j-1}\,]_1 \rightarrow [\,f_i\,]_2[\;\;]_1 \\ [\,f_i\,]_2[\,\overline{x}_{i,j,n+j-1}\,]_1 \rightarrow [\,c_{j,j}\,f_i\,]_2[\;\;]_1 \\ [\,f_i\,]_2[\,x^*_{i,j,n+j-1}\,]_1 \rightarrow [\,f_i\,]_2[\;\;]_1 \end{array}\right\} \text{for } 1 \leq i \leq n, 1 \leq j \leq p$$

$$\left.\begin{array}{l} [\,x_{i,j,n+k}\,]_1[\,\gamma\,]_0 \rightarrow [\,x_{i,j,n+k+1}\,]_1[\;\;]_0 \\ [\,\overline{x}_{i,j,n+k}\,]_1[\,\gamma\,]_0 \rightarrow [\,\overline{x}_{i,j,n+k+1}\,]_1[\;\;]_0 \\ [\,x^*_{i,j,n+k}\,]_1[\,\gamma\,]_0 \rightarrow [\,x^*_{i,j,n+k+1}\,]_1[\;\;]_0 \end{array}\right\} \text{for} \begin{array}{c} 1 \leq i \leq n, \\ 1 \leq j \leq p, \\ 0 \leq k \leq j-2 \end{array}$$

$$[\,c_{j,k}\,]_2[\,\gamma\,]_0 \rightarrow [\,c_{j,k+1}\,]_2[\;\;]_0 \;,\ \text{for } 1 \leq j \leq p, j \leq k \leq p-1$$

3.1 Rules to check if all clauses are satisfied by a truth assignment.

$$[\,\alpha_{p+1}\,]_2[\,\delta'_{4n+p}\,]_1 \rightarrow [\,\alpha'_{p+1}\,]_2[\;\;]_0$$
$$[\,\alpha_j\,c_{j,p}\,]_2[\;\;]_0 \rightarrow [\;\;]_2[\,\#\,]_0 \;,\ \text{for } 1 \leq j \leq p$$

4.1 General counters.

$$[\,\delta_i\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta_{i+1}\,]_1[\;\;]_0 \;,\ \text{for } 0 \leq i \leq 4n+p+1$$
$$[\,\delta'_{4i+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,{\delta'}^2_{4i+2}\,]_1[\;\;]_0 \;,\ \text{for } 0 \leq i \leq n-1$$
$$[\,\delta'_{4i+k}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta'_{4i+k+1}\,]_1[\;\;]_0 \;,\ \text{for } 0 \leq i \leq n-1, k \in \{0,2,3\}$$
$$[\,\delta'_{4n+i}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta'_{4n+i+1}\,]_1[\;\;]_0 \;,\ \text{for } 0 \leq i \leq p-1$$

4.2 Rules to give a negative answer.

$$[\,\alpha_j\,\alpha'_{p+1}\,]_2[\;\;]_0 \rightarrow [\;\;]_2[\,n_1\,]_0 \;,\ \text{for } 1 \leq j \leq p$$
$$[\;\;]_2[\,n_1\,]_0 \rightarrow [\,n_1\,]_2[\;\;]_0$$
$$[\,n_1\,]_2[\,\delta_{4n+p+2}\,]_1 \rightarrow [\,n_2\,]_2[\;\;]_1$$
$$[\,n_2\,]_2[\;\;]_0 \rightarrow [\;\;]_2[\,\texttt{no}\,]_0$$

4.3 Rules to give an affirmative answer.

$$[\,\alpha'_{p+1}\,]_2[\,\delta_{4n+p+2}\,]_1 \rightarrow [\,y_1\,]_2[\;\;]_1$$
$$[\,y_1\,]_2[\,\gamma\,]_0 \rightarrow [\,y_2\,]_2[\;\;]_0$$
$$[\,y_2\,]_2[\;\;]_0 \rightarrow [\;\;]_2[\,\texttt{yes}\,]_0$$

7. The input cell is the cell labelled by 1 ($i_{in} = 1$) and the output region is the environment ($i_{out} = env$).

Let $\varphi = C_1 \wedge \cdots \wedge C_p$ an instance of $\texttt{SAT}$ problem consisting of p clauses $C_j = l_{j,1} \vee \cdots \vee l_{j,r_j}$, $1 \leq j \leq p$, where $Var(\varphi) = \{x_1, \ldots, x_n\}$, and $l_{j,k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$, $1 \leq j \leq p, 1 \leq k \leq r_j$. Let us assume that the number of variables, n, and the number of clauses, p, of φ, are greater than or equal to 2.

We consider the polynomial encoding (cod, s) from $\texttt{SAT}$ in $\mathbf{\Pi}$ defined as follows: for each $\varphi \in I_{\texttt{SAT}}$ with n variables and p clauses, $s(\varphi) = \langle n, p \rangle$ and

$$cod(\varphi) = \{x_{i,j,0} \mid x_i \in C_j\} \cup \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \cup \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$$

For instance, the formula $\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$ is encoded as follows:

$$cod(\varphi) = \begin{pmatrix} x_{1,1,0} & x_{2,1,0} & \overline{x}_{3,1,0} & x^*_{4,1,0} \\ x^*_{1,2,0} & \overline{x}_{2,2,0} & x^*_{3,2,0} & x_{4,2,0} \\ x^*_{1,3,0} & \overline{x}_{2,3,0} & x_{3,3,0} & \overline{x}_{4,3,0} \end{pmatrix}$$

We define $cod_k(\varphi)$ as the set of elements of $cod(\varphi)$ when the third subscript equals k. In the same way, we define $cod'_k(\varphi)$, $cod''_k(\varphi)$ and $cod'''_k(\varphi)$ as the sets of elements of $cod(\varphi)$ when the third subscript equals k and elements are primed, double primed and triple primed, respectively. For notation convenience, we define $cod^j_k(\varphi)$ the subset of elements of $cod_k(\varphi)$ with elements of $C_j, \ldots, C_p$. For instance, $cod^2_4(\varphi)$ would be the following set:

$$cod^2_4(\varphi) = \begin{pmatrix} x^*_{1,2,4} & \overline{x}_{2,2,4} & x^*_{3,2,4} & x_{4,2,4} \\ x^*_{1,3,4} & \overline{x}_{2,3,4} & x_{3,3,4} & \overline{x}_{4,3,4} \end{pmatrix}$$

The Boolean formula φ will be processed by the system $\Pi(s(\varphi)) + cod(\varphi)$. Next, we informally describe how that system works.

The solution proposed follows a brute force algorithm in the framework of recognizer tissue P systems with separation and evolutional communication rules, and it consists of the following stages:

- *Generation stage*: Using separation rules each 4 steps, we produce 2^n membranes labelled by 2 containing each possible truths assignment. At the same time, we generate 2^n copies of $cod_n(\varphi)$. This stage spends n computation steps exactly, being n the numer of variables of φ.
- *First checking stage*: With rules from **2.1**, we can check which clauses from the input formula φ have been satisfied by a specific truth assignment. This stage takes exactly p steps.
- *Second checking stage*: With rules from **3.1**, we remove objects α_j such that they are removed from a membrane if and only if the truth assignment associated to that membrane makes true clause C_j. This stage takes exactly one step.
- *Output stage*: With rules from **4.2** and **4.3**, we can give an affirmative or a negative answer depending on if the input formula is satisfiable or not. This stage spends exactly 4 steps, regardless of whether the formula is satisfiable or not.

5 A formal verification

In this section, an exhaustive verification of the system is given.

Generation stage

At this stage, all truth assignments for the variables associated with the Boolean formula $\varphi(x_1, \ldots, x_n)$ are going to be generated, by applying separation rules from **1.2** in membranes labelled by 2. In such manner that in the $4i+2$-th step $(1 \leq i \leq n-1)$ of this stage, separation rule associated with an object $a_{i,i}$ is triggered, two new cells distributing t_i and f_i between them. In the last step of this stage, each membrane labelled by 2 will contain a truth assignment of the formula.

Proposition 1. *Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ be a computation of the system $\Pi(s(\varphi))$ with input multset $cod(\varphi)$.*

(a_0) *For each $4k$ $(0 \leq k \leq n-1)$ at configuration $\mathcal{C}_{4k}$ we have the following:*
- $\mathcal{C}_{4k}(1) = \{\delta_{4k}, {\delta'}_{4k}^{2^k}, cod_k(\varphi)^{2^k}\} \cup \{\beta_{l,k}^{2^k} \mid k+1 \leq l \leq n\}$
- *There are 2^k membranes labelled by 2 such that each of them contains*
  - *objects $a_{k+1,k}, \ldots, a_{n,k}$;*
  - *objects $r_1, \ldots, r_k$, being $r \in \{t, f\}$; and*
  - *objects $\alpha_1, \ldots, \alpha_{p+1}$.*

(a_1) *For each $4k+1$ $(0 \leq k \leq n-1)$ at configuration $\mathcal{C}_{4k+1}$ we have the following:*
- $\mathcal{C}_{4k+1}(1) = \{\delta_{4k+1}, {\delta'}_{4k+1}^{2^k}, cod'_k(\varphi)^{2^k}\} \cup \{{\beta'}_{l,k}^{2^k} \mid k+1 \leq l \leq n\}$
- *There are 2^k membranes labelled by 2 such that each of them contains*
  - *objects $a'_{k+1,k}, \ldots, a'_{n,k}$;*
  - *objects $r''_1, \ldots, r''_k$, being $r \in \{t, f\}$*
  - *an object t'_{k+1}; and*
  - *objects $\alpha'_1, \ldots, \alpha'_{p+1}$.*

(a_2) *For each $4k+2$ $(0 \leq k \leq n-1)$ at configuration $\mathcal{C}_{4k+2}$ we have the following:*
- $\mathcal{C}_{4k+2}(1) = \{\delta_{4k+2}, {\delta'}_{4k+3}^{2^{k+1}}, cod''_{k+1}(\varphi)^{2^{k+1}}\} \cup$
  $\{{\beta^O}_{l,k}^{2^k} \mid O \in \{L, R\}, k+1 \leq l \leq n\}$
- *There are 2^k membranes labelled by 2 such that each of them contains*
  - *objects $a_{k+1,k}, \ldots, a_{n,k}$;*
  - *objects $r_1, \ldots, r_k$, being $r \in \{t, f\}$; and*
  - *objects $\alpha_1, \ldots, \alpha_{p+1}$.*

(a_3) *For each $4k+3$ $(0 \leq k \leq n-1)$ at configuration $\mathcal{C}_{4k+3}$ we have the following:*
- $\mathcal{C}_{4k+3}(0) = \{{\beta^O}_{k+1,k+1}^{2^k}\} \cup \{\beta_{l,k+1}^{2^{k+1}} \mid k+2 \leq l \leq n\}$
- $\mathcal{C}_{4k+3}(1) = \{\delta_{4k+3}, {\delta'}_{4k+3}^{2^{k+1}}, cod'''_{k+1}(\varphi)^{2^{k+1}}\} \cup \{\beta_{l,k}^{2^k} \mid k+1 \leq l \leq n\}$
- *There are 2^{k+1} membranes labelled by 2 such that each of them contains*
  - *objects $a_{k+1,k}, \ldots, a_{n,k}$;*

– *objects* $r_1, \ldots, r_k$*, being* $r \in \{t, f\}$*; and*
– *objects* $\alpha_1, \ldots, \alpha_{p+1}$*.*

(*b*) $\mathcal{C}_{4n}(1) = \{\delta_{4n}, {\delta'}^{2^n}_{4n}, cod_{4n}(\varphi)^{2^n}\}$*, and there are* 2^n *membranes labelled by 2 such that each of them contains objects* $\alpha_1, \ldots, \alpha_{p+1}$*, as well as a different subset* $\{r_1, \ldots, r_n\}$*, being* $r \in \{t, f\}$*.*

Proof. (a) is going to be demonstrated by induction on k.

(a_0) The base case $k = 0$ is trivial because at the initial configuration we have: $\mathcal{C}_0(1) = \{\delta_0, \delta'_0, cod_0(\varphi)\} \cup \{\beta_{l,0} \mid 1 \le l \le n\}$ and there exists a single membrane labelled by 2 containing objects $\alpha_1, \ldots, \alpha_{p+1}$ and objects $a_{1,0}, \ldots, a_{n,0}$. Then, configuration $\mathcal{C}_0$ yields configuration $\mathcal{C}_1$ by applying the rules:

$$[\,a_{1,0}\,]_2[\,\gamma\,]_0 \rightarrow [\,a'_{1,0}\,t'_1\,]_2[\quad]_0$$
$$[\,a_{i,0}\,]_2[\,\gamma\,]_0 \rightarrow [\,a'_{i,0}\ \text{, for } 2 \le i \le n$$
$$[\,\alpha_j\,]_2[\,\gamma\,]_0 \rightarrow [\,\alpha'_j\,]_2[\quad]_0\ \text{, for } 1 \le j \le p+1$$
$$[\,\beta_{l,0}\,]_1[\,\gamma\,]_0 \rightarrow [\,\beta'_{l,0}\,]_1[\quad]_0\ \text{, for } 1 \le l \le n$$
$$\left.\begin{array}{l} [\,x_{i,j,0}\,]_1[\,\gamma\,]_0 \rightarrow [\,x'_{i,j,1}\,]_1[\quad]_0 \\ [\,\overline{x}_{i,j,0}\,]_1[\,\gamma\,]_0 \rightarrow [\,\overline{x}'_{i,j,1}\,]_1[\quad]_0 \\ [\,x^*_{i,j,0}\,]_1[\,\gamma\,]_0 \rightarrow [\,{x^*}'_{i,j,1}\,]_1[\quad]_0 \end{array}\right\} \text{for } 1 \le i \le n, 1 \le j \le p$$
$$[\,\delta_0\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta_1\,]_1[\quad]_0$$
$$[\,\delta'_0\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta'_1\,]_1[\quad]_0$$

(a_1) Thus, $\mathcal{C}_1(1) = \{\delta_1, \delta'_1, cod'_1(\varphi)\} \cup \{\beta'_{l,0} \mid 1 \le l \le n\}$ and in $\mathcal{C}_1$ there exists one membrane labelled by 2 such that its contents is the set of objects $\{a'_{1,0}, \ldots, a'_{n,0}\}$, the object t'_1 and objects $\alpha'_1, \ldots, \alpha'_{p+1}$. Then, configuration $\mathcal{C}_1$ yields configuration $\mathcal{C}_2$ by applying the rules:

$$[\,a'_{1,0}\,]_2[\,\gamma\,]_0 \rightarrow [\,a_{1,1}\,f^R_1\,]_2[\quad]_0$$
$$[\,t'_1\,]_2[\,\gamma\,]_0 \rightarrow [\,t^L_1\,]_2[\quad]_0$$
$$[\,a'_{i,0}\,]_2[\,\gamma\,]_0 \rightarrow [\,a^L_{i,1}\,a^R_{i,1}\,]_2[\quad]_0\ \text{, for} 2 \le i \le n$$
$$[\,\alpha'_j\,]_2[\,\gamma\,]_0 \rightarrow [\,\alpha^L_j\,\alpha^R_j\,]_2[\quad]_0\ \text{, for } 1 \le j \le p+1$$
$$[\,\beta'_{l,k}\,]_1[\,\gamma\,]_0 \rightarrow [\,\beta^L_{l,k+1}\,\beta^R_{l,k+1}\,]_1[\quad]_0\ \text{, for } k+1 \le l \le n$$
$$\left.\begin{array}{l} [\,x'_{i,j,0}\,]_1[\,\gamma\,]_0 \rightarrow [\,{x''}^2_{i,j,1}\,]_1[\quad]_0 \\ [\,\overline{x}'_{i,j,0}\,]_1[\,\gamma\,]_0 \rightarrow [\,{\overline{x}''}^2_{i,j,0+1}\,]_1[\quad]_0 \\ [\,{x^*}'_{i,j,0}\,]_1[\,\gamma\,]_0 \rightarrow [\,{x^*}''^2_{i,j,0+1}\,]_1[\quad]_0 \end{array}\right\} \text{for } \begin{array}{l} 1 \le i \le n, \\ 1 \le j \le p \end{array}$$
$$[\,\delta_1\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta_2\,]_1[\quad]_0$$
$$[\,\delta'_1\,]_1[\,\gamma\,]_0 \rightarrow [\,{\delta'}^2_2\,]_1[\quad]_0$$

(a_2)Thus, $\mathcal{C}_2(1) = \{\delta_2, {\delta'}^2_2, cod''_1(\varphi)\} \cup \{\beta^O_{l,1} \mid O \in \{L, R\}, 1 \le l \le n\}$ and in $\mathcal{C}_2$ there exists one membrane labelled by 2 such that its contents is the set of objects $\{a_{1,1}, \ldots, a_{n,1}\}$, objects t^L_1 and f^R_1 and objects $\alpha^O_1, \ldots, \alpha^O_{p+1}$, for $O \in \{L, R\}$. Then, configuration $\mathcal{C}_2$ yields configuration $\mathcal{C}_3$ by applying the rules:

$$[\,a_{1,1}\,]_2 \rightarrow [\,\Gamma_0\,]_2[\,\Gamma_1\,]_2\ \text{, for } 1 \le i \le n$$
$$\left.\begin{array}{l} [\,\beta^O_{1,1}\,]_1[\quad]_0 \rightarrow [\quad]_1[\,\beta^O_{1,1}\,]_0 \\ [\,\beta^O_{l,1}\,]_1[\quad]_0 \rightarrow [\quad]_1[\,\beta_{l,1}\,]_0 \end{array}\right\} \text{for } \begin{array}{l} O \in \{L, R\}, \\ k+1 \le l \le n \end{array}$$

$$\left.\begin{array}{l}[\,x''_{i,j,0}\,]_1[\,\gamma\,]_0 \to [\,x'''_{i,j,0}\,]_1[\quad]_0\\ [\,\overline{x}''_{i,j,0}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}'''_{i,j,0}\,]_1[\quad]_0\\ [\,x^{*''}_{i,j,0}\,]_1[\,\gamma\,]_0 \to [\,x^{*'''}_{i,j,0}\,]_1[\quad]_0\end{array}\right\}\text{for}\begin{array}{l}1\le i\le n,\\ 1\le j\le p,\end{array}$$
$$[\,\delta_2\,]_1[\,\gamma\,]_0 \to [\,\delta_3\,]_1[\quad]_0$$
$$[\,\delta'_2\,]_1[\,\gamma\,]_0 \to [\,\delta'_3\,]_1[\quad]_0$$

(a_3) Thus, $\mathcal{C}_3(1) = \{\delta_3, {\delta'}^2_3, cod'''_1(\varphi)\}$, at the environment there is the multiset $\{\beta^O_{1,1} \mid O \in \{L,R\}\} \cup \{\beta^2_{l,1} \mid 2 \le l \le n\}$ and in $\mathcal{C}_2$ there exists two membranes labelled by 2 such that its contents is the set of objects $\{a^O_{2,1}, \dots, a^O_{n,1}\}$ with $O = L$ (resp., $O = R$), object t^L_1 (resp., f^R_1) and objects $\alpha^O_1, \dots, \alpha^O_{p+1}$, for $O = L$ (resp., $O = R$). Hence, the result holds for $k = 0$

- Supposing that, by induction, result is true for k $(1 \le k \le n-1)$; that is,
  - (a_0) For each $4k$ $(0 \le k \le n-1)$ at configuration $\mathcal{C}_{4k}$ we have the following:
    - $\mathcal{C}_{4k}(1) = \{\delta_{4k}, {\delta'}^{2^k}_{4k}, cod_k(\varphi)^{2^k}\} \cup \{\beta^{2^k}_{l,k} \mid k+1 \le l \le n\}$
    - There are 2^k membranes labelled by 2 such that each of them contains
      - · objects $a_{k+1,k}, \dots, a_{n,k}$;
      - · objects $r_1, \dots, r_k$, being $r \in \{t, f\}$; and
      - · objects $\alpha_1, \dots, \alpha_{p+1}$.
  - (a_1) For each $4k+1$ $(0 \le k \le n-1)$ at configuration $\mathcal{C}_{4k+1}$ we have the following:
    - $\mathcal{C}_{4k+1}(1) = \{\delta_{4k+1}, {\delta'}^{2^k}_{4k+1}, cod'_k(\varphi)^{2^k}\} \cup \{{\beta'}^{2^k}_{l,k} \mid k+1 \le l \le n\}$
    - There are 2^k membranes labelled by 2 such that each of them contains
      - · objects $a'_{k+1,k}, \dots, a'_{n,k}$;
      - · objects $r''_1, \dots, r''_k$, being $r \in \{t, f\}$
      - · an object t'_{k+1}; and
      - · objects $\alpha'_1, \dots, \alpha'_{p+1}$.
  - (a_2) For each $4k+2$ $(0 \le k \le n-1)$ at configuration $\mathcal{C}_{4k+2}$ we have the following:
    - $\mathcal{C}_{4k+2}(1) = \{\delta_{4k+2}, , {\delta'}^{2^{k+1}}_{4k+2} cod''_{k+1}(\varphi)^{2^{k+1}}\} \cup \{{\beta^O}^{2^k}_{l,k} \mid O \in \{L,R\}, k+1 \le l \le n\}$
    - There are 2^k membranes labelled by 2 such that each of them contains
      - · objects $a_{k+1,k}, \dots, a_{n,k}$;
      - · objects $r_1, \dots, r_k$, being $r \in \{t, f\}$; and
      - · objects $\alpha_1, \dots, \alpha_{p+1}$.
  - (a_3) For each $4k+3$ $(0 \le k \le n-1)$ at configuration $\mathcal{C}_{4k+3}$ we have the following:
    - $\mathcal{C}_{4k+3}(0) = \{{\beta^O}^{2^k}_{k+1,k+1}\} \cup \{\beta^{2^{k+1}}_{l,k+1} \mid k+2 \le l \le n\}$
    - $\mathcal{C}_{4k+3}(1) = \{\delta_{4k+3}, {\delta'}^{2^{k+1}}_{4k+3}, cod'''_{k+1}(\varphi)^{2^{k+1}}\} \cup \{\beta^{2^k}_{l,k} \mid k+1 \le l \le n\}$
    - There are 2^{k+1} membranes labelled by 2 such that each of them contains
      - · objects $a_{k+1,k}, \dots, a_{n,k}$;
      - · objects $r_1, \dots, r_k$, being $r \in \{t, f\}$; and
      - · objects $\alpha_1, \dots, \alpha_{p+1}$.
- Then, by the induction hypothesis, we want to prove the result for $k+1$.

(a_0) Then, configuration $\mathcal{C}_{4k+3}$ yields configuration $\mathcal{C}_{4(k+1)}$ by applying the rules:

$$\left.\begin{array}{l}[\,a^{O}_{i,j}\,]_2[\,\beta^{O}_{k+1,k+1}\,]_0 \to [\,a_{i,j}\,]_2[\quad]_0\\ [\,r^{O}_{i}\,]_2[\,\beta^{O}_{k+1,k+1}\,]_0 \to [\,r_i\,]_2[\quad]_0\end{array}\right\}\text{for}\begin{array}{l}O\in\{L,R\},\\ r\in\{t,f\},\\ 1\le i\le n,\\ 1\le j\le n\end{array}$$

$$[\,\alpha^{O}_{j}\,]_2[\,\beta^{O}_{k+1,k+1}\,]_0 \to [\,\alpha_j\,]_2[\quad]_0 \ \text{, for } O\in\{L,R\}, 1\le j\le p+1$$

$$\left.\begin{array}{l}[\,x'''_{i,j,k+1}\,]_1[\,\gamma\,]_0 \to [\,x_{i,j,k+1}\,]_1[\quad]_0\\ [\,\overline{x}'''_{i,j,k+1}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}_{i,j,k+1}\,]_1[\quad]_0\\ [\,x^{*'''}_{i,j,k+1}\,]_1[\,\gamma\,]_0 \to [\,x^{*}_{i,j,k+1}\,]_1[\quad]_0\end{array}\right\}\text{for } 1\le i\le n, 1\le j\le p$$

$$[\quad]_1[\,\beta_{l,k+1}\,]_0 \to [\,\beta_{l,k+1}\,]_1[\quad]_0 \ \text{, for } k+2\le l\le n$$

$$[\,\delta_{4k+3}\,]_1[\,\gamma\,]_0 \to [\,\delta_{4(k+1)}\,]_1[\quad]_0$$

$$[\,\delta'_{4k+3}\,]_1[\,\gamma\,]_0 \to [\,\delta'_{4(k+1)}\,]_1[\quad]_0$$

Therefore, the following holds:

- $\mathcal{C}_{4(k+1)}(1) = \{\delta_{4(k+1)}, {\delta'}^{2^{k+1}}_{4(k+1)}, cod_{k+1}(\varphi)^{2^{k+1}}\} \cup \{\beta^{2^{k+1}}_{l,k+1} \mid k+2\le l\le n\}$
- There are 2^{k+1} membranes labelled by 2 such that each of them contains
  - objects $a_{k+2,k+1},\dots,a_{n,k+1}$;
  - objects $r_1,\dots,r_{k+1}$, being $r\in\{t,f\}$; and
  - objects $\alpha_1,\dots,\alpha_{p+1}$.

(a_1) Then, configuration $\mathcal{C}_{4(k+1)}$ yields configuration $\mathcal{C}_{4(k+1)+1}$ by applying the rules:

$$[\,a_{k+1,k}\,]_2[\,\gamma\,]_0 \to [\,a'_{k+1,k}\,t'_{k+1}\,]_2[\quad]_0$$

$$\left.\begin{array}{l}[\,t_i\,]_2[\,\gamma\,]_0 \to [\,t''_i\,]_2[\quad]_0\\ [\,f_i\,]_2[\,\gamma\,]_0 \to [\,f''_i\,]_2[\quad]_0\end{array}\right\}\text{for } 1\le i\le k$$

$$[\,a_{i,k+1}\,]_2[\,\gamma\,]_0 \to [\,a'_{i,k+1}\,]_2[\quad]_0 \ \text{, for } 2\le i\le n$$

$$[\,\alpha_j\,]_2[\,\gamma\,]_0 \to [\,\alpha'_j\,]_2[\quad]_0 \ \text{, for } 1\le j\le p+1$$

$$[\,\beta_{l,k+1}\,]_1[\,\gamma\,]_0 \to [\,\beta'_{l,k+1}\,]_1[\quad]_0\,\}\text{ for } k+2\le l\le n$$

$$\left.\begin{array}{l}[\,x_{i,j,k+1}\,]_1[\,\gamma\,]_0 \to [\,x'_{i,j,k+1}\,]_1[\quad]_0\\ [\,\overline{x}_{i,j,k+1}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}'_{i,j,k+1}\,]_1[\quad]_0\\ [\,x^{*}_{i,j,k+1}\,]_1[\,\gamma\,]_0 \to [\,x^{*'}_{i,j,k+1}\,]_1[\quad]_0\end{array}\right\}\text{for } 1\le i\le n, 1\le j\le p$$

$$[\,\delta_{4(k+1)}\,]_1[\,\gamma\,]_0 \to [\,\delta_{4(k+1)+1}\,]_1[\quad]_0$$

$$[\,\delta'_{4(k+1)}\,]_1[\,\gamma\,]_0 \to [\,\delta'_{4(k+1)+1}\,]_1[\quad]_0$$

Therefore, the folowing holds:

- $\mathcal{C}_{4(k+1)+1}(1) = \{\delta_{4(k+1)+1}, {\delta'}^{2^{k}}_{4(k+1)+1}, cod'_{k+1}(\varphi)^{2^{k+1}}\} \cup \{{\beta'}^{2^{k}}_{l,k} \mid k+1 \le l\le n\}$
- There are 2^{k+1} membranes labelled by 2 such that each of them contains
  - objects $a'_{k+2,k+1},\dots,a'_{n,k+1}$;
  - objects $r''_1,\dots,r''_{k+1}$, being $r\in\{t,f\}$
  - an object t'_{k+2}; and
  - objects $\alpha'_1,\dots,\alpha'_{p+1}$.

(a_2) Then, configuration $\mathcal{C}_{4(k+1)+1}$ yields configuration $\mathcal{C}_{4(k+1)+2}$ by applying the rules:

$[\,a'_{k+1,k}\,]_2[\,\gamma\,]_0 \rightarrow [\,a_{k+1,k+1}\, f^R_{k+1}\,]_2[\quad]_0$
$[\,t'_{k+1}\,]_2[\,\gamma\,]_0 \rightarrow [\,t^L_{k+1}\,]_2[\quad]_0$
$\left.\begin{array}{l}[\,t''_i\,]_2[\,\gamma\,]_0 \rightarrow [\,t^L_i\, t^R_i\,]_2[\quad]_0\\ [\,f''_i\,]_2[\,\gamma\,]_0 \rightarrow [\,f^L_i\, f^R_i\,]_2[\quad]_0\end{array}\right\}$ for $1 \leq i \leq k$
$[\,a'_{i,k+1}\,]_2[\,\gamma\,]_0 \rightarrow [\,a^L_{i,k+2}\, a^R_{i,k+2}\,]_2[\quad]_0$, for $2 \leq i \leq n$
$[\,\alpha'_j\,]_2[\,\gamma\,]_0 \rightarrow [\,\alpha^L_j\, \alpha^R_j\,]_2[\quad]_0$, for $1 \leq j \leq p+1$
$[\,\beta'_{l,k+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,\beta^L_{l,k+2}\, \beta^R_{l,k+2}\,]_1[\quad]_0$, for $k+2 \leq l \leq n$
$\left.\begin{array}{l}[\,x'_{i,j,k+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,x''^2_{i,j,k+2}\,]_1[\quad]_0\\ [\,\overline{x}'_{i,j,k+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,\overline{x}''^2_{i,j,k+2}\,]_1[\quad]_0\\ [\,x^{*\prime}_{i,j,k+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,x^{*\prime\prime 2}_{i,j,k+2}\,]_1[\quad]_0\end{array}\right\}$ $\begin{array}{l}1 \leq i \leq n,\\ 1 \leq j \leq p\end{array}$
$[\,\delta_{4(k+1)+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta_{4(k+1)+2}\,]_1[\quad]_0$
$[\,\delta'_{4(k+1)+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta'^2_{4(k+1)+2}\,]_1[\quad]_0$

Therefore, the following holds:

– $\mathcal{C}_{4(k+1)+2}(1) = \{\delta_{4(k+1)+2}, \delta'^{2^{k+2}}_{4(k+1)+2}, cod''_{k+2}(\varphi)^{2^{k+2}}\} \cup \{\beta^{2^{k+1}}_{l,k} \mid k+1 \leq l \leq n\}$
– There are 2^{k+1} membranes labelled by 2 such that each of them contains
  · objects $a_{k+2,k+1}, \ldots, a_{n,k+1}$;
  · objects $r_1, \ldots, r_{k+1}$, being $r \in \{t, f\}$; and
  · objects $\alpha_1, \ldots, \alpha_{p+1}$.

(a_3) Then, configuration $\mathcal{C}_{4(k+1)+2}$ yields configuration $\mathcal{C}_{4(k+1)+3}$ by applying the rules:

$[\,a_{k+1,k+1}\,]_2 \rightarrow [\,\Gamma_0\,]_2[\,\Gamma_1\,]_2$, for $1 \leq i \leq n$
$\left.\begin{array}{l}[\,\beta^O_{k+1,k+1}\,]_1[\quad]_0 \rightarrow [\quad]_1[\,\beta^O_{k+1,k+1}\,]_0\\ [\,\beta^O_{l,k+1}\,]_1[\quad]_0 \rightarrow [\quad]_1[\,\beta_{l,k+1}\,]_0\end{array}\right\}$ for $O \in \{L, R\}, k+2 \leq l \leq n$
$\left.\begin{array}{l}[\,x''_{i,j,k+2}\,]_1[\,\gamma\,]_0 \rightarrow [\,x'''_{i,j,k+2}\,]_1[\quad]_0\\ [\,\overline{x}''_{i,j,k+2}\,]_1[\,\gamma\,]_0 \rightarrow [\,\overline{x}'''_{i,j,k+2}\,]_1[\quad]_0\\ [\,x^{*\prime\prime}_{i,j,k+2}\,]_1[\,\gamma\,]_0 \rightarrow [\,x^{*\prime\prime\prime}_{i,j,k+2}\,]_1[\quad]_0\end{array}\right\}$ for $\begin{array}{l}1 \leq i \leq n,\\ 1 \leq j \leq p\end{array}$
$[\,\delta_{4(k+1)+2}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta_{4(k+1)+3}\,]_1[\quad]_0$
$[\,\delta'_{4(k+1)+2}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta'_{4(k+1)+3}\,]_1[\quad]_0$

Therefore, the following holds:

– $\mathcal{C}_{4(k+1)+3}(0) = \{\beta O^{2^{k+1}}_{k+2,k+2}\} \cup \{\beta^{2^{k+2}}_{l,k+2} \mid k+3 \leq l \leq n\}$
– $\mathcal{C}_{4(k+1)+3}(1) = \{\delta_{4(k+1)+3}, \delta'^{2^{k+2}}_{4(k+1)+3}, cod'''_{k+2}(\varphi)^{2^{k+2}}\} \cup \{\beta^{2^{k+1}}_{l,k+1} \mid k+2 \leq l \leq n\}$
– There are 2^{k+2} membranes labelled by 2 such that each of them contains
  · objects $a_{k+2,k+1}, \ldots, a_{n,k+1}$;
  · objects $r_1, \ldots, r_{k+1}$, being $r \in \{t, f\}$; and
  · objects $\alpha_1, \ldots, \alpha_{p+1}$.

• In order to prove (b) it is enough to notice that, on the one hand, from (a_3) configuration $\mathcal{C}_{4n-1}$[1] holds:

[1] Here, $4n-1 = 4k+3$ for $k = n-1$.

- $\mathcal{C}_{4n-1}(1) = \{\delta_{4n-1}, {\delta'}^{2^n}_{4n-1}, cod'''_n(\varphi)\}$.
- There are 2^n membranes labelled by 2 such that each of them contains
  - a different subset $\{r_1^O, \ldots, r_n^O\}$, being $r \in \{t, f\}$ and $O \in \{L, R\}$; and
  - objects $\alpha^O, \ldots, \alpha^O_{p+1}$, for $O \in \{L, R\}$.

- On the other hand, configuration $\mathcal{C}_{4n-1}$ yields configuration $\mathcal{C}_{4n}$ by applying the rules:

$$[\, r_i^O \,]_2[\, \beta^O_{n,n} \,]_0 \to [\, r_i \,]_2[\quad]_0 \text{ , for } \begin{array}{l} O \in \{L,R\}, \\ r \in \{t,f\}, \\ 1 \le i \le n \end{array}$$

$$[\, \alpha_j^O \,]_2[\, \beta^O_{n,n} \,]_0 \to [\, \alpha_j \,]_2[\quad]_0 \text{ , for } O \in \{L,R\}, 1 \le j \le p+1$$

$$\left.\begin{array}{l} [\, x'''_{i,j,n} \,]_1[\, \gamma \,]_0 \to [\, x_{i,j,n} \,]_1[\quad]_0 \\ [\, \overline{x}'''_{i,j,n} \,]_1[\, \gamma \,]_0 \to [\, \overline{x}_{i,j,n} \,]_1[\quad]_0 \\ [\, {x^*}'''_{i,j,n} \,]_1[\, \gamma \,]_0 \to [\, x^*_{i,j,n} \,]_1[\quad]_0 \end{array}\right\} \text{for } 1 \le i \le n, 1 \le j \le p$$

$$[\, \delta_{4n-1} \,]_1[\, \gamma \,]_0 \to [\, \delta_{4n} \,]_1[\quad]_0$$

$$[\, \delta'_{4n-1} \,]_1[\, \gamma \,]_0 \to [\, \delta'_{4n} \,]_1[\quad]_0$$

- Then, we have $\mathcal{C}_{4n}(1) = \{\delta_{4n}, {\delta'}^{2^n}_{4n}, cod_{4n}(\varphi)^{2^n}\}$, and there are 2^n membranes labelled by 2 such that each of them contains objects $\alpha_1, \ldots, \alpha_{p+1}$, as well as a different subset $\{r_1, \ldots, r_n\}$, being $r \in \{t, f\}$. □

First checking stage

Following the generation stage comes the first checking stage, where objects $c_{j,k}$ are created in order to know if clause C_j has been satisfied by the truth assignment encoded in membranes labelled by 2. In each step, we fire rules for a single clause, therefore in p steps we can obtain objects $c_{j,k}$ if this clause is satisfied. This can be because of two reasons:

- Literal x_i appears in clause C_j, and the the valoration of variable x_i in a truth assignment is `True`. Then, we can say that such truth assignment satisfies this clause; or
- Literal $\neg x_i$ appears in clause C_j, and the the valoration of variable x_i in a truth assignment is `False`. Then, we can say that such truth assignment satisfies this clause.

In any other way, variable x_i has nothing to do with clause C_j. At the final step of this stage, membranes labelled by 2 will have objects $c_{j,p}$ where C_j are clauses satisfied by such truth assignment. We obtain an object α'_{p+1} to use it in the next stage.

Proposition 2. *Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_q)$ be a compuation of th system $\Pi(s(\varphi))$ with input multiset $cod(\varphi)$.*

(a) For each k $(0 \le k \le p-1)$ at configuration $\mathcal{C}_{4n+k}$ we have the following:
- $\mathcal{C}_{4n+k}(1) = \{\delta_{4n+k}, {\delta'}^{2^n}_{4n+k}, cod_n^k(\varphi)^{2^n}\}$
- *There are 2^n membranes labelled by 2 such that each of them contains*

   - *objects $r_1, \ldots, r_n$, being $r \in \{t, f\}$;*
   - *objects $\alpha_1, \ldots, \alpha_{p+1}$; and*
   - *objects $c_{1,k}, \ldots, c_{k,k}$, where $c_{j,k}$ represents that clause C_j has been satisfied by the truth formula encoded in such membrane.*

(*b*) *$\mathcal{C}_{4n+p}(1) = \{\delta_{4n+p}, {\delta'}^{2^n}_{4n+p}\}$, and there are 2^n membranes labelled by 2 such that each of them contains objects $\alpha_1, \ldots, \alpha_{p+1}$, a different subset $\{r_1, \ldots, r_n\}$ and objects c_j when clause C_j is satisfied in that membrane.*

Proof. (a) is going to be demonstrated by induction on k.

(a) The base case $k = 0$ is trivial because at the initial configuration we have: $\mathcal{C}_{4n}(1) = \{\delta_{4n}, {\delta'}^{2^n}_{4n}, cod_{4n}(\varphi)\}$ and there exist 2^n membranes labelled by 2 containing objects $\alpha_1, \ldots, \alpha_{p+1}$ and a different subset $\{r_1, \ldots, r_n\}$, being $r \in \{t, f\}$. Then, configuration $\mathcal{C}_{4n}$ yields configuration $\mathcal{C}_{4n+1}$ by applying the rules:

$$\left.\begin{array}{l}
[\,t_i\,]_2[\,x_{i,1,n}\,]_1 \to [\,c_{1,1}\,t_i\,]_2[\quad]_1 \\
[\,t_i\,]_2[\,\overline{x}_{i,1,n}\,]_1 \to [\,t_i\,]_2[\quad]_1 \\
[\,t_i\,]_2[\,x^*_{i,1,n}\,]_1 \to [\,t_i\,]_2[\quad]_1 \\
[\,f_i\,]_2[\,x_{i,1,n}\,]_1 \to [\,f_i\,]_2[\quad]_1 \\
[\,f_i\,]_2[\,\overline{x}_{i,1,n}\,]_1 \to [\,c_{1,1}\,f_i\,]_2[\quad]_1 \\
[\,f_i\,]_2[\,x^*_{i,1,n}\,]_1 \to [\,f_i\,]_2[\quad]_1
\end{array}\right\} \text{for } 1 \le i \le n, 1 \le j \le p$$

$$\left.\begin{array}{l}
[\,x_{i,j,n+k}\,]_1[\,\gamma\,]_0 \to [\,x_{i,j,n+k+1}\,]_1[\quad]_0 \\
[\,\overline{x}_{i,j,n+k}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}_{i,j,n+k+1}\,]_1[\quad]_0 \\
[\,x^*_{i,j,n+k}\,]_1[\,\gamma\,]_0 \to [\,x^*_{i,j,n+k+1}\,]_1[\quad]_0
\end{array}\right\} \text{for } 1 \le i \le n, 2 \le j \le p$$

$$\begin{array}{l}
[\,\delta_{4n}\,]_1[\,\gamma\,]_0 \to [\,\delta_{4n+1}\,]_1[\quad]_0 \\
[\,\delta'_{4n}\,]_1[\,\gamma\,]_0 \to [\,\delta'_{4n+1}\,]_1[\quad]_0
\end{array}$$

Thus, $\mathcal{C}_{4n+1}(1) = \{\delta_{4n+1}, {\delta'}^{2^n}_{4n+1}, cod^2_{4n+1}(\varphi)^{2^n}\}$ and in $\mathcal{C}_{4n+1}$ there exist 2^n membranes labelled by 2 such that their contents are objects $\alpha_1, \ldots, \alpha_{p+1}$, a different subset $\{r_1, \ldots, r_n\}$, being $r \in \{t, f\}$ and objects $c_{1,1}$ if some literal present in C_j satisfies it[2]. Hence, the result holds for $k = 1$.

Supposing that, by induction, result is true for k $(0 \le k \le p - 1)$; that is,

- $\mathcal{C}_{4n+k}(1) = \{\delta_{4n+k}, {\delta'}^{2^n}_{4n+k}, cod^{k+1}_n(\varphi)^{2^k}\}$
- There are 2^n membranes labelled by 2 such that each of them contains
  - objects $r_1, \ldots, r_n$, being $r \in \{t, f\}$;
  - objects $\alpha_1, \ldots, \alpha_{p+1}$; and
  - objects $c_{1,k}, \ldots, c_{k,k}$, where $c_{j,k}$ represents that clause C_j has been satisfied by the truth formula encoded in such membrane.

Then, configuration $\mathcal{C}_{4n+k}$ yields configuration $\mathcal{C}_{4n+k+1}$ by applying the rules:

[2] Here, objects # are created, but they are not used anymore, so they are not going to be noted here.

$$\left.\begin{array}{l}
[\,t_i\,]_2[\,x_{i,k+1,n+k}\,]_1 \to [\,c_{k+1,k+1}\,t_i\,]_2[\quad]_1\\
[\,t_i\,]_2[\,\overline{x}_{i,k+1,n+k}\,]_1 \to [\,t_i\,]_2[\quad]_1\\
[\,t_i\,]_2[\,x^*_{i,k+1,n+k}\,]_1 \to [\,t_i\,]_2[\quad]_1\\
[\,f_i\,]_2[\,x_{i,k+1,n+k}\,]_1 \to [\,f_i\,]_2[\quad]_1\\
[\,f_i\,]_2[\,\overline{x}_{i,k+1,n+k}\,]_1 \to [\,c_{k+1,k+1}\,f_i\,]_2[\quad]_1\\
[\,f_i\,]_2[\,x^*_{i,k+1,n+k}\,]_1 \to [\,f_i\,]_2[\quad]_1
\end{array}\right\}\text{for } 1 \le i \le n, 1 \le j \le p$$

$$\left.\begin{array}{l}
[\,x_{i,j,n+k}\,]_1[\,\gamma\,]_0 \to [\,x_{i,j,n+k+1}\,]_1[\quad]_0\\
[\,\overline{x}_{i,j,n+k}\,]_1[\,\gamma\,]_0 \to [\,\overline{x}_{i,j,n+k+1}\,]_1[\quad]_0\\
[\,x^*_{i,j,n+k}\,]_1[\,\gamma\,]_0 \to [\,x^*_{i,j,n+k+1}\,]_1[\quad]_0
\end{array}\right\}\text{for } 1 \le i \le n, k+2 \le j \le p$$

$$[\,c_{j,k}\,]_2[\,\gamma\,]_0 \to [\,c_{j,k+1}\,]_2[\quad]_0\ \text{, for } 1 \le j \le p, k \le k \le p-1$$
$$[\,\delta_{4n+k}\,]_1[\,\gamma\,]_0 \to [\,\delta_{4n+k+1}\,]_1[\quad]_0$$
$$[\,\delta'_{4n+k}\,]_1[\,\gamma\,]_0 \to [\,\delta'_{4n+k+1}\,]_1[\quad]_0$$

Thus, $\mathcal{C}_{4n+k+1}(1) = \{\delta_{4n+k+1}, {\delta'}^{2^n}_{4n+k+1}, cod^{k+2}_{4n+k+1}(\varphi)^{2^n}\}$ and in $\mathcal{C}_{4n+k+1}$ there exist 2^n membranes labelled by 2 such that their contents are objects $\alpha_1, \dots, \alpha_{p+1}$, a different subset $\{r_1, \dots, r_n\}$, being $r \in \{t, f\}$ and objects $c_{1,k}, \dots, c_{k,k}$ if some literal present in C_j satisfies them.

In order to demonstrate (b) it is enough to notice that, on the one hand, from (a) configuration $\mathcal{C}_{4n+p-1}$ holds:

- $\mathcal{C}_{4n+p-1}(1) = \{\delta_{4n+p-1}, {\delta'}^{2^n}_{4n+p-1}, cod^p_n(\varphi)^{2^n}\}$
- There are 2^n membranes labelled by 2 such that each of them contains
  - objects $r_1, \dots, r_n$, being $r \in \{t, f\}$;
  - objects $\alpha_1, \dots, \alpha_{p+1}$; and
  - objects $c_{1,p-1}, \dots, c_{p-1,p-1}$, where $c_{j,p-1}$ represents that clause C_j has been satisfied by the truth formula encoded in such membrane.

On the other hand, configuration $\mathcal{C}_{4n+p-1}$ yields configuration $\mathcal{C}_{4n+p}$ by applying the rules:

$$\left.\begin{array}{l}
[\,t_i\,]_2[\,x_{i,p,n+p-1}\,]_1 \to [\,c_{p,p}\,t_i\,]_2[\quad]_1\\
[\,t_i\,]_2[\,\overline{x}_{i,p,n+p-1}\,]_1 \to [\,t_i\,]_2[\quad]_1\\
[\,t_i\,]_2[\,x^*_{i,p,n+p-1}\,]_1 \to [\,t_i\,]_2[\quad]_1\\
[\,f_i\,]_2[\,x_{i,p,n+p-1}\,]_1 \to [\,f_i\,]_2[\quad]_1\\
[\,f_i\,]_2[\,\overline{x}_{i,p,n+p-1}\,]_1 \to [\,c_{p,p}\,f_i\,]_2[\quad]_1\\
[\,f_i\,]_2[\,x^*_{i,p,n+p-1}\,]_1 \to [\,f_i\,]_2[\quad]_1
\end{array}\right\}\text{for } 1 \le i \le n, 1 \le j \le p$$

$$[\,c_{j,p-1}\,]_2[\,\gamma\,]_0 \to [\,c_{j,p}\,]_2[\quad]_0\ \text{, for } 1 \le j \le p-1$$
$$[\,\delta_{4n+p-1}\,]_1[\,\gamma\,]_0 \to [\,\delta_{4n+p}\,]_1[\quad]_0$$
$$[\,\delta'_{4n+p-1}\,]_1[\,\gamma\,]_0 \to [\,\delta'_{4n+p}\,]_1[\quad]_0$$

Then, we have $\mathcal{C}_{4n+p}(1) = \{\delta_{4n+p}, {\delta'}^{2^n}_{4n+p}\}$, and in $\mathcal{C}_{4n+p}$ there are 2^n membranes labelled by 2 such that each of them contains a different subset $\{r_1, \dots, r_n\}$, being $r \in \{t, f\}$[3], objects $\alpha_1, \dots, \alpha_{p+1}$ and objects $c_{j,p}$ when clause C_j has been satisfied by the truth assignment encoded in such membrane. □

[3] This subset is not used anymore, so it will not be noted from now on.

Second checking stage

Here, when rules from **3.1** are fired at the $(4n+p+1)$-th step, objects α_j within a membrane labelled by 2 are removed if and only if the truth assignment associated to that membrane makes true clause C_j, that is, if there is at least one object c_j in such membrane. At configuration $\mathcal{C}_{4n+p}$ we have $\mathcal{C}_{4n+p}(1) = \{\delta_{4n+p}, {\delta'}^{2^n}_{4n+p}\}$ and each membrane labelled by 2 contains objects $\alpha_1, \dots, \alpha_p$ and objects c_j such that the corresponding truth assignment satisfies the clause C_j. By applying rules from **3.1** and rule $[\,\delta_{4n+p}\,]_1[\,\gamma\,]_0 \to [\,\delta_{4n+p+1}\,]_1[\quad]_0$, object δ_{4n+p} evolves into δ_{4n+p+1} within the membrane labelled by 1, and in each membrane labelled by 2, objects α_j such that their corresponding object $c_{j,p}$ are "removed" from the system, and let the next stage to check whether or not they are present, besides the object α_{p+1}, that is prepared, evolving to α'_{p+1}, to react with the remaining objects α_j. This stage takes exactly one step.

Output stage

The output phase starts at the $(4n+p+2)$-th step, and takes exactly four steps, regardless of whether the input formula φ is satisfied or not by some truth assignment.

- *Affirmative answer*: If the input formula φ of `SAT` problem is satisfiable then at least one of the truth assignments from a membrane with label 2 has satisfied all clauses. Then, there will be a membrane labelled by 2 such that all objects α_j, with $1 \le j \le p$ have dissapeared in the previous step. At configuration $\mathcal{C}_{4n+p+1}$, we have $\mathcal{C}_{4n+p+1}(1) = \{\delta_{4n+p+1}\}$ and in each membrane labelled by 2 there remain objects α_j if the corresponding truth assignment does **not** make true clause C_j and one object α'_{p+1}. In this step, only rule $[\,\delta_{4n+p+1}\,]_1[\,\gamma\,]_0 \to [\,\delta_{4n+p+2}\,]_1[\quad]_0$ will be fired and rules $[\,\alpha_j\,\alpha'_{p+1}\,]_2[\quad]_0 \to [\quad]_2[\,n_1\,]_0$ will be fired in membranes labelled by 2 such that at least one clause is not satisfied by the corresponding truth assignment. Then, at configuration $\mathcal{C}_{4n+p+2}$, we have $\mathcal{C}_{4n+p+2}(1) = \{\delta_{4n+p+2}, n_1^t\}$, being t the number of truth assignments that have at least one clause **not** satisfied by the corresponding truth assignment, and membranes labelled by 2 contains an object α'_{p+1} if and only if the corresponding truth assignment makes true all clauses from φ, and can contain objects α_j, $1 \le j \le p$, if clause C_j is not satisfied by the corresponding truth assignment.
  In the next step, applying rules $[\quad]_2[\,n_1\,]_0 \to [\,n_1\,]_2[\quad]_0$ and $[\,\alpha'_{p+1}\,]_2[\,\delta_{4n+p+2}\,]_1 \to [\,y_1\,]_2[\quad]_1$, we obtain an object y_1 in a membrane labelled by 2 if and only if the corresponding truth assignment makes true the input formula. Let us remark that more than one membrane labelled by 2 can contain a truth assignment that makes true φ, but in this case, we as we want to know if *at least* one truth assignment makes true the input formula φ, we only want one object y_1. Then, at configuration $\mathcal{C}_{4n+p+3}$ we have that $\mathcal{C}_{4n+p+3}(1) = \emptyset$ and in membranes la-

belled by 2, we can have objects n_1[4], adding up to t in all membranes labelled by 2, being t the number of truth assignments that do not make true the input formula, an object α'_{p+1} if the corresponding truth assignment makes true all clauses, excepting one membrane labelled by 2 which corresponding truth assignment makes true the input formula that will contain an object y_1, and can contain objects α_j, $1 \leq j \leq p$, if clause C_j is not satisfied by the corresponding truth assignment. In the next step the only rule that can be fired is $[\,y_1\,]_2[\,\gamma\,]_0 \rightarrow [\,y_2\,]_2[\quad]_0$, that will be useful to synchronize the affirmative and the negative answer. Let us note that rule $[\,n_1\,]_2[\,\delta_{4n+p+2}\,]_1 \rightarrow [\,n_2\,]_2[\quad]_1$ cannot be fired because object δ_{4n+3} has been consumed in the previous step by an object α'_{p+1}. Then, at configuration $\mathcal{C}_{4n+p+4}$, we have that $\mathcal{C}_{4n+p+4}(1) = \emptyset$ and in membranes labelled by 2, we can have objects n_1, adding up to t in all membranes labelled by 2, being t the number of truth assignments that do not make true the input formula, an object α'_{p+1} if the corresponding truth assignment makes true all clauses, excepting one membrane labelled by 2 which corresponding truth assignment makes true the input formula that will contain an object y_2, and can contain objects α_j, $1 \leq j \leq p$, if clause C_j is not satisfied by the corresponding truth assignment. At the last step of the computation, rule $[\,y_2\,]_2[\quad]_0 \rightarrow [\quad]_2[\,\texttt{yes}\,]_0$ is fired, sending an object `yes` to the environment. Then, at configuration $\mathcal{C}_{4n+p+5}$, we have that $\mathcal{C}_{4n+p+5}(1) = \emptyset$ and in membranes labelled by 2, we can have objects n_1, adding up to t in all membranes labelled by 2, being t the number of truth assignments that do not make true the input formula, an object α'_{p+1} if the corresponding truth assignment makes true all clauses, excepting one membrane labelled by 2 which corresponding truth assignment makes true the input formula, and can contain objects α_j, $1 \leq j \leq p$, if clause C_j is not satisfied by the corresponding truth assignment, and there will be an object `yes` in the environment. Here, the computation halts and returns an affirmative answer.

- *Negative answer*: If the input formula φ of `SAT` problem is not satisfiable then none of the truth assignments encoded by a membrane labelled by 2 makes the formula φ true. Thus, some object α_j $(1 \leq j \leq p)$ will be within all membranes labelled by 2 will not remain in such membranes. At configuration $\mathcal{C}_{4n+p+1}$, we have $\mathcal{C}_{4n+p+1}(1) = \{\delta_{4n+p+1}\}$ and in each membrane labelled by 2 there remain objects α_j if the corresponding truth assignment does **not** make true clause C_j. In this step, only rules $[\,\alpha_j\,\alpha'_{p+1}\,]_2[\quad]_0 \rightarrow [\quad]_2[\,n_1\,]_0$, for $1 \leq j \leq p$ and rule $[\,\delta_{4n+p+1}\,]_1[\,\gamma\,]_0 \rightarrow [\,\delta_{4n+p+2}\,]_1[\quad]_0$ will be fired. Then, at configuration $\mathcal{C}_{4n+p+2}$ we have in the environmet 2^n copies of object n_1, $\mathcal{C}_{4n+p+2}(1) = \{\delta_{4n+p+2}\}$ and membranes labelled by 2 will contain objects α_j $(1 \leq j \leq p)$ when clauses C_j are not satisfied by the corresponding truth assignment. In the $(4n + p + 3)$-th step, rule $[\quad]_2[\,n_1\,]_0 \rightarrow [\,n_1\,]_2[\quad]_0$ will be fired. Here, objects n_1 will be sent to a membrane labelled by 2. Then,

[4] Let us note that a membrane containing an object n_1 does not say that the corresponding truth assignment does not makes true the input formula. In fact, we can have more than one object n_1 within a single membrane labelled by 2.

at configuration $\mathcal{C}_{4n+p+3}$ we have $\mathcal{C}_{4n+p+3}(1) = \{\delta_{4n+p+2}\}$ and membranes labelled by 2 contain objects α_j $(1 \leq j \leq p)$ if clause C_j is not satisfied by the corresponding truth assignment, and can contain t objects n_1 $(0 \leq t \leq 2^n)$. At the $(4n+p+4)$-th step rule $[\, n_1 \,]_2[\, \delta_{4n+p+2} \,]_1 \rightarrow [\, n_2 \,]_2[\quad]_1$ is fired, since object δ_{4n+3} has not been consumed by any rule from **4.3**, creating an object n_2 in a membrane labelled by 2. Then, at configuration $\mathcal{C}_{4n+p+4}$ we have $\mathcal{C}_{4n+p+4}(1) = \emptyset$ and membranes labelled by 2 contain objects α_j $(1 \leq j \leq p)$ if clause C_j is not satisfied by the corresponding truth assignment, and can contain t objects n_1 $(0 \leq t \leq 2^n)$, and one of them contains an object n_2. At the last step of the computation, rule $[\, n_2 \,]_2[\quad]_0 \rightarrow [\quad]_2[\, \texttt{no} \,]_0$ is fired, sending an object no to the environment. Then, at configuration $\mathcal{C}_{4n+p+5}$ we have that $\mathcal{C}_{4n+p+5}(1) = \emptyset$ and membranes labelled by 2 contain objects α_j $(1 \leq j \leq p)$ if clause C_j is not satisfied by the corresponding truth assignment, and can contain t objects n_1 $(0 \leq t \leq 2^n)$, and there will be an object no in the environment. Here, the computation halts and returns a negative answer.

Result

Proof. The family of P systems previously constructed verifies the following:

- Every system of the family $\mathbf{\Pi}$ is a recognizer P systems from $\mathbf{TSEC}(2,2)$.
- The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines because for each $n, p \in \mathbb{N}$, the rules of $\Pi(\langle n,p\rangle)$ of the family are recursively defined from $n, p \in \mathbb{N}$, and the amount of resources needed to build an element of the family is of a polynomial order in n and p, as shown below:
  - Size of the alphabet: $9n^2p + 6n^2 + \frac{3np^2}{2} - 3np + 22n + \frac{p^2}{2} + \frac{13p}{2} + 14 \in \Theta(max\{n^2p, np^2\})$.
  - Initial number of cells: $2 \in \Theta(1)$.
  - Initial number of objects in cells: $n^2 + n(p+1) + p + 3 \in \Theta(n^2)$.
  - Number of rules: $8n^3 + \frac{27n^2p}{2} + 4n^1 + \frac{19np}{2} + 23n + \frac{p^2}{2} + \frac{17p}{2} + 11 \in \Theta(n^3)$.
  - Maximal number of objects involved in any rule: $4 \in \Theta(1)$.
- The pair (cod, s) of polynomial-time computable functions defined fulfill the following: for each input formula φ of SAT problem, $s(\varphi)$ is a natural number, $cod(\varphi)$ is an input multiset of the system $\Pi(s(\varphi))$, and for each $n \in \mathbb{N}$, $s^{-1}(n)$ is a finite set.
- The family $\mathbf{\Pi}$ is polynomially bounded: indeed for each input formula φ of SAT problem, the deterministic P system $\Pi(s(\varphi)) + cod(\varphi)$ takes exactly $4n+p+5$ steps, being n the number of variables of φ and p the number of clauses.
- The family $\mathbf{\Pi}$ is sound with regard to (X, cod, s): indeed, for each formula φ, if the computation of $\Pi(s(\varphi)) + cod(\varphi)$ is an accepting computation, then φ is satisfiable.
- The family $\mathbf{\Pi}$ is complete with regard to (X, cod, s): indeed, for each input formula φ such that it is satisfiable, the computation of $\Pi(s(\varphi)) + cod(\varphi)$ is an accepting computation. □

Corollary 1. $\mathbf{NP} \cup \mathbf{co} - \mathbf{NP} \subseteq \mathbf{PMC}_{\mathbf{TSEC}(2,2)}$.

Proof. It suffices to notice that `SAT` problem is a **NP**-complete problem, `SAT` $\in \mathbf{PMC}_{\mathbf{TSEC}(2,2)}$, and the complexity class $\mathbf{PMC}_{\mathbf{TSEC}(2,2)}$ is closed under polynomial-time reduction and under complement. □

6 Conclusions and future work

In [6] a tight frontier of efficiency in the framework of tissue P systems with evolutional symport/antiport rules and cell separation is defined by the length of the RHS, that is, passing from 1 to 2 is enough to pass from non-efficiency to presumably efficiency while the length of the LHS is at least 3. This result is demonstrated giving a solution of the `SAT` problem by means of a family of P system from $\mathbf{TSEC}(3,2)$. But an open problem remains open here: what happens with P systems from $\mathbf{TSEC}(k,2)$ $(k \geq 2)$? Can we solve computationally hard problems restricting the length of the LHS to 2?

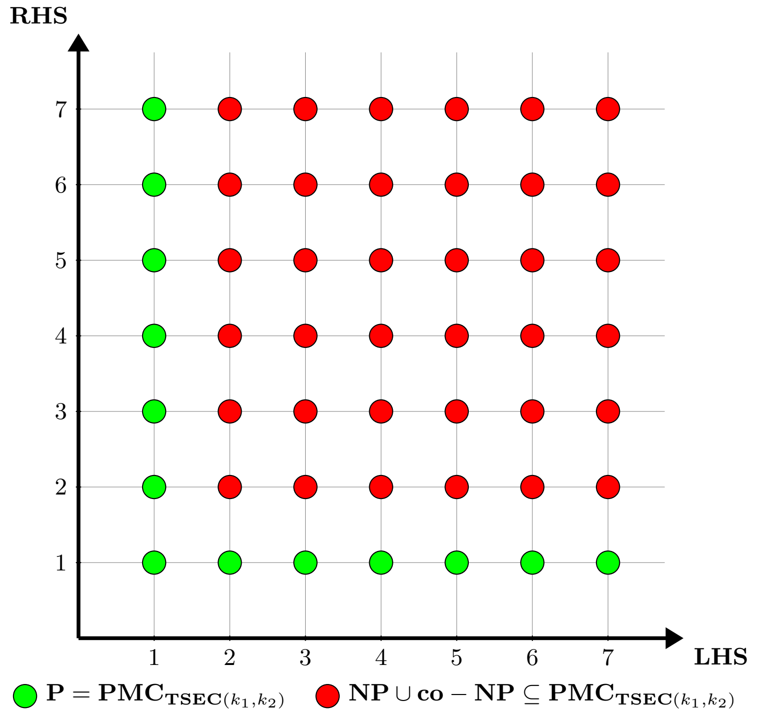
In this paper, an efficient solution to the **SAT** problem is given by means of a family of P systems from $\mathbf{TSEC}(2,2)$, so the previous problem is solved. Then, we can conclude here with a similar figure to the presented in [6] but with the new results included.

Of course, after this work we can define several clear research lines to continue investigating these kinds of P systems.

– What happens when the environment "dissapear"?
– Do the structure matter? By this we mean using cell-like structure with this kind of rules.
– In [12] another definition of length is given. Let k be the length of the rule defined as follows: if $r \equiv [\,u\,]_i[\,v\,]_j \rightarrow [\,v'\,]_i[\,u'\,]_j$, $k = |u| + |v| + |u'| + |v'|$. Then the complexity class of tissue P systems with evolutional communication rules with at most length k and cell separation is denoted by $\mathbf{PMC}_{\mathbf{TSEC}(k)}$. What are the borderline here?
– What is the upper bound of these systems? In [3] a characterization of tissue P systems with symport/antiport rules and both cell division and separation is given matching their efficiency to the class $\mathbf{P}^{\#\mathbf{P}}$, and it seems that this class of P system can reach the same complexity class.

Acknoledgements

This work was supported by Project TIN2017-89842-P of the Ministerio de Economía y Competitividad of Spain and by Grants No 61320106005 of the National Natural Science Foundation of China.



References

1. R. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, M. Rius–Font. Characterizing tractability by tissue–like P systems. In R. Gutiérrez–Escudero, M.A. Gutiérrez–Naranjo, Gh. Păun, I. Pérez–Hurtado and A. Riscos Núñez (eds.) *Proceedings of the Seventh Brainstorming Week on Membrane Computing*, Fénix Editora, Seville, 2009, pp. 169–180.
2. M. Ionescu, Gh. Păun, T. Yokomori. Spiking Neural P Systems. *Fundamenta Informaticae*, **71**, 2,3 (2006), 279–308.
3. A. Leporati, L. Manzoni, A.E. Porreca, C. Zandron. Characterising the complexity of tissue P systems with fission rules. *Journal of Computer and System Sciences*, **90** (2017), 115–128.
4. C. Martín-Vide, Gh. Păun, J. Pazos, A. Rodríguez-Patón. Tissue P systems. *Theoretical Computer Science*, **296**, Issue 2, 2003, 295–326.
5. L. Pan, M.J. Pérez-Jiménez, A. Riscos-Núñez, M. Rius-Font. New frontiers of the efficiency in tissue P systems. L. Pan, Gh. Păun, T. Song (eds.) *Pre-proceedings of Asian Conference on Membrane Computing (ACMC 2012)*, Huazhong University of Science and Technology, Wuhan, China, October 15-18, 2012, pp. 61-73.
6. L. Pan, B. Song, L. Valencia-Cabrera, M.J. Pérez-Jiménez. The computational complexity of tissue P systems with evolutional symport/antiport rules. *Complexity*, 21 pages, 2018.
7. Gh. Păun. Computing with membranes. *Journal of Computer and System Sciences*, **61**, 1 (2000), 108–143, and *Turku Center for CS-TUCS Report* No. 208, 1998

8. M.J. Pérez-Jiménez. An approach to computational complexity in Membrane Computing. In G. Mauri, Gh. Păun, M.J. Pérez-Jiménez, G. Rozenberg, A. Salomaa (eds.). *Membrane Computing, International Workshop, WMC5, Milano, Italy, 2004, Selected Papers, Lecture Notes in Computer Science*, **3365** (2005), 85-109.
9. M.J. Pérez-Jiménez, Á. Romero-Jiménez, F. Sancho-Caparrini. Complexity classes in models of cellular computing with membranes. *Natural Computing*, **2**, 3 (2003), 265–285.
10. M.J. Pérez-Jiménez, P. Sosík. Improving the efficiency of tissue P systems with cell separation. In M. García-Quismondo, L.F. Macías-Ramos, Gh. Păun, I. Pérez- Hurtado, L. Valencia-Cabrera (eds.) *Proceedings of the Tenth Brainstorming Week on Membrane Computing*, Volume II, Seville, Spain, January 30 - February 3, 2012, Report RGNC 01/2012, Fénix Editora, 2012, pp. 105-140.
11. A.E. Porreca, N. Murphy, M.J. Pérez-Jiménez. An Optimal Frontier of the Efficiency of Tissue P Systems with Cell Division. In M.Á. Martínez-del-Amor, Gh. Păun, I. Pérez–Hurtado and F.J. Romero-Campero (eds.) *Proceedings of the Tenth Brainstorming Week on Membrane Computing*, Volume II, Seville, Spain, January 30 - February 3, 2012, Report RGNC 01/2012, Fénix Editora, 2012, 141–166.
12. B. Song, C. Zhang, L. Pan. Tissue-like P systems with evolutional symport/antiport rules. *Information Sciences*, **378** (2017), 177-.193.