Tissue P Systems with Protein on Cells

Bosheng Song, Linqiang Pan*
Key Laboratory of Image Information Processing and Intelligent Control
School of Automation
Huazhong University of Science and Technology, Wuhan 430074, Hubei, China
boshengsong@163.com, lqpan@mail.hust.edu.cn

Mario J. Pérez-Jiménez
Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla, Avda. Reina Mercedes s/n,41012 Sevilla, Spain
marper@us.es

Abstract. Tissue P systems are a class of distributed parallel computing devices inspired by biochemical interactions between cells in a tissue-like arrangement, where objects can be exchanged by means of communication channels. In this work, inspired by the biological facts that the movement of most objects through communication channels is controlled by proteins and proteins can move through lipid bilayers between cells (if these cells are fused), we present a new class of variant tissue P systems, called tissue P systems with protein on cells, where multisets of objects (maybe empty), together with proteins between cells are exchanged. The computational power of such P systems is studied. Specifically, an efficient (uniform) solution to the SAT problem by using such P systems with cell division is presented. We also prove that any Turing computable set of numbers can be generated by a tissue P system with protein on cells. Both of these two results are obtained by such P systems with communication rules of length at most 4 (the length of a communication rule is the total number of objects and proteins involved in that rule).

Keywords: Bio-inspired computing, Membrane computing, Tissue P system, Cell protein, Cell division, Universality

*Address for correspondence: Key Laboratory of Image Information Processing and Intelligent Control, School of Automation, Huazhong University of Science and Technology, Wuhan 430074, Hubei, China.

1. Introduction

Membrane computing is a nature inspired computational paradigm initiated by Gh. Păun at the end of 1998 [1]. The aim is to abstract computing ideas from the role of membranes in compartmentalization of living cells and the behavior of some cellular processes. The obtained models are distributed and parallel computing devices, usually called *P systems*. An essential ingredient of a P system is its membrane structure, which can be a hierarchical arrangement of membranes, as in a cell [1], or a net of membranes (placed in the nodes of a graph), as in a tissue [2] or a spiking neural network [3] and its variants [4, 5, 6, 7, 8, 9]. For main information of P systems on both the general level and technical level, please refer to [10]; for up-to-date information of this area, one may view the P systems website `http://ppage.psystems.eu` for details. The present work deals with a class of tissue-like P systems.

Tissue-like P systems have the membrane structures that are described by directed graphs, where membranes (also called *cells*) are placed in nodes of a graph. An arc between two nodes corresponds to a communication channel between cells placed in these nodes. Objects can communicate between two cells or between a cell and the environment if the communication channels exist. The communication of tissue P systems is based on symport/antiport rules [11]. Symport rules move objects across a cell together in one direction, whereas in the case of antiport rules, objects residing at both sides of the cell cross it simultaneously but in opposite directions.

Many variants of tissue-like P systems have been developed (see, e.g., [12, 13, 14]). An interesting variant of tissue P systems, called *tissue P systems with cell division*, was proposed in [15], where the biological inspiration of such computing model is obvious: alive tissues are not static network of cells, since cells are duplicated via mitosis in a natural way. Cell division is an efficient approach for obtaining exponential workspace in polynomial time by trading space for time, therefore, it is natural to investigate the computational efficiency of tissue P systems with cell division. The first attempt in this topic was done in [15], a polynomial time uniform solution to the `SAT` problem was presented. In a series of subsequent works, tissue P systems with cell division were also considered to solve other **NP**-complete problems, such as vertex cover problem [16], subset sum problem [17], 3-coloring problem [18], independent set problem [19].

From biological point of view, most of the reactions (permeate certain types of molecules through communication channel, cell division, etc.) taking place in cells are controlled by proteins, which have two main types with respect to the way they are associated to the lipid bilayer: *peripheral* proteins, placed on one side of a membrane, internal or external, and *integral* proteins (also called *transmembrane proteins*), which have parts of the molecule on both sides of the membrane. Another biological fact is that some of proteins on cells are not static, they can move through lipid bilayers between cells if these cells are fused. Furthermore, in cell biology, membrane proteins perform a variety of functions vital to the survival of organisms [20] and the proteins constitute about half of the mass of the membranes in the animal cells [21]. Thus, it is rather natural to consider the role of proteins in tissue P systems.

In this work, with the above mentioned biological facts, proteins are introduced into tissue P systems, and we present a class of tissue P systems with protein on cells, where there is one and only one copy of protein placed on each cell at the beginning of computation (in fact, during the process of computation, each cell also contains one and only one protein). If a communication rule between two cells is applied, then the multisets of objects together with the proteins are exchanged (in this case, multisets of objects can be empty, that is, it allows that only the proteins are exchanged between two cells); if a communication rule between a cell and the environment is applied, then only multisets of objects between the cell and the environment are exchanged (in this case, at least one of multisets of objects is non-empty). The computational power of this kind of P systems is studied. Specifically, we present an efficient (uniform) solution to the SAT problem by using such P systems with cell division. We also prove that any Turing computable set of numbers can be generated by a tissue P system with protein on cells. Both of these two results are obtained by such P systems with communication rules of length at most 4 (the length of a communication rule is the total number of objects and proteins involved in that rule).

2. Preliminaries

In this section, we only introduce a few basic notions and notations from formal languages theory. Readers can refer to [22] for details.

An *alphabet* Σ is a non–empty set and its elements are called *symbols*. An ordered finite sequence of symbols forms a *string* or *word*. 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^* and by $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$ we denote the set of non-empty strings. A *language* over Σ is a subset of Σ^*.

A *multiset* m over an alphabet Σ is a pair (Σ, f) where $f : \Sigma \to \mathbb{N}$ is a mapping from Σ to the set of non-negative number $\mathbb{N}$. Let $m_1 = (\Sigma, f_1)$, $m_2 = (\Sigma, f_2)$ are multisets over Σ, then we define the union of m_1 and m_2 as $m_1 + m_2 = (\Sigma, g)$, where $g(x) = f_1(x) + f_2(x)$. The *relative complement* of m_2 in m_1, denoted by $m_1 \setminus m_2$ is the multiset (Σ, g), where $g(x) = f_1(x) - f_2(x)$ if $f_1(x) \geq f_2(x)$, and $g(x) = 0$ otherwise.

In what follows, we introduce the notion of *register machines*, which are used to the characterization of NRE (the family of sets of numbers which are Turing computable).

Definition 2.1. A register machine is a tuple $M = (m, H, l_0, l_h, I)$, where:

- m is the number of registers;
- H is a set of labels;
- $l_0, l_h \in H$ are distinguished labels, where l_0 is the initial, and l_h is the halting one;
- I is a set of labelled program instructions of the following forms:
  - $l_i : (\texttt{ADD}(r), l_j, l_k)$ (add 1 to register r and continue with one of the instructions with labels l_j, l_k, non-deterministically chosen);
  - $l_i : (\texttt{SUB}(r), l_j, l_k)$ (if register r is non-zero, then subtract 1 from it, and go to the instruction with label l_j; otherwise, go to the instruction with label l_k);
  - $l_h : \texttt{HALT}$.

A register machine M generates a set $N(M)$ of numbers in the following way: the machine starts with all registers being empty (i.e., storing the number zero); the machine applies the instruction with label l_0 and continues to apply instructions as indicated by the labels (and made possible by the contents of registers); if it reaches the halt instruction, then the number n presented in the specified register 1 at that time is said to be generated by M. If the computation does not halt, then no number is generated. It is known that register machines generate all sets of numbers which are Turing computable, hence they characterize NRE [23].

We use the following convention. When comparing the power of two number computing devices, the number zero is ignored. Thus, when we say that a set Q is in NRE, we do not care whether or not $0 \in Q$ (this corresponds to the usual practice of ignoring the empty string when comparing the power of two grammars or automata).

3. Tissue P Systems with Protein on Cells and Cell Division

The model of tissue P systems with protein on cells is based on the model of P systems with proteins on membranes [24]. These two kinds of models have some differences. In our model, proteins on cells cannot change, but they can move together with multisets of objects; however, in the model of P systems with proteins on membranes, proteins on cells can be changed, but they cannot move between membranes. Both of these models allow membrane division rules.

Definition 3.1. A tissue P system with protein on cells of degree $q \geq 1$ is a tuple $\Pi = (\Gamma, P, \mathcal{E}, \mathcal{M}_1/p_1, \ldots, \mathcal{M}_q/p_q, \mathcal{R}, i_{out})$, where:

- Γ and P are finite non-empty alphabets such that $\Gamma \cap P = \emptyset$;
- $\mathcal{E}$ is a finite alphabet such that $\mathcal{E} \subseteq \Gamma$;
- $\mathcal{M}_i$, $1 \leq i \leq q$, are finite multisets over Γ;
- p_i, $1 \leq i \leq q$, are elements in P;
- $\mathcal{R}$ is a finite set of communication rules of the following forms:

  (a) $(i, (p_i, u)/(p_j, v), j)$, for $i, j \in \{1, \ldots, q\}$, $i \neq j$, $p_i, p_j \in P$, $u, v \in \Gamma^*$.
  (b) $(i, (p_i, u)/v, 0)$, for $i \in \{1, \ldots, q\}$, $p_i \in P$, $u, v \in \Gamma^*$, $|uv| > 0$.

- $i_{out} \in \{0, 1, \ldots, q\}$.

Definition 3.2. A tissue P system with protein on cells and cell division of degree $q \geq 1$ is a tuple $\Pi = (\Gamma, P, \mathcal{E}, \mathcal{M}_1/p_1, \ldots, \mathcal{M}_q/p_q, \mathcal{R}, i_{out})$, where all components are as in a tissue P system with protein on cells, and $\mathcal{R}$ is a finite set of rules, which contains communication rules of the forms (a), (b) as mentioned in Definite 3.1, and division rules of the form:

(c) $[\, p_i \mid a \,]_i \rightarrow [\, p'_i \mid b \,]_i [\, p''_i \mid c \,]_i$, for $i \in \{1, 2, \ldots, q\}$, $p_i, p'_i, p''_i \in P$, $a, b, c \in \Gamma$, $i \neq i_{out}$.

A tissue P system with protein on cells (and cell division) of degree $q \geq 1$ 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 finite multisets of objects (symbols of the alphabet Γ) initially placed in the q cells of the system; (b) $p_1, \ldots, p_q$ represent one and only one copy of protein (symbols of the alphabet P) initially placed on the q cells of the system; (c) $\mathcal{E}$ is the set of objects initially located in the environment of the system, all of them available in an arbitrary number of copies; and (d) i_{out} represents a distinguished *zone* which will encode the output of the system. We use the term *zone* i ($0 \leq i \leq q$) to refer to cell i in the case of $1 \leq i \leq q$ and to refer to the environment in the case of $i = 0$. The length of a communication rule is the total number of objects and proteins involved in that rule.

A *configuration* of a tissue P system with protein on cells (and cell division) at any instant is described by all multisets of objects over Γ associated with all the cells present in the system, all the proteins presented on all cells, and the multiset of objects over $\Gamma \setminus \mathcal{E}$ 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 $(\mathcal{M}_1/p_1, \ldots, \mathcal{M}_q/p_q; \emptyset)$.

A communication rule of type $(i, (p_i, u)/(p_j, v), j)$ is applicable to a configuration at an instant if cell i contains the protein p_i and the multiset u of objects, cell j contains the protein p_j and the multiset v of objects (multisets u, v may be empty). When applying such a rule, under the control of the proteins p_i on cell i and p_j on cell j, both the protein p_i and the multiset u of objects are sent from region i to region j, and simultaneously, the protein p_j and the multiset v of objects are sent from region j to region i; a particular case is $(i, (p_i, \lambda)/(p_j, \lambda), j)$, where only proteins change their places. A communication rule of type $(i, (p_i, u)/v, 0)$ is applicable to a configuration at an instant if cell i contains the protein p_i and the multiset u of objects, the environment contains the multiset v of objects (at least one of multisets u, v is non-empty). When applying such a rule, under the control of the protein p_i on cell i, the multiset u of objects are sent from region i to the environment, and simultaneously, the multiset v of objects are sent from the environment to region i.

A division rule $[\, p_i \mid a \,]_i \rightarrow [\, p'_i \mid b \,]_i [\, p''_i \mid c \,]_i$ is applicable to a configuration at an instant if cell i contains the protein p_i and the object a. When applying such a rule, under the influence of protein p_i on cell i and the object a in cell i, the cell is divided into two cells with the same label; in the first copy of the cell the protein p_i is replaced by p'_i and the object a is replaced by b, in the second copy of the cell the protein p_i is replaced by p''_i and the object a is replaced by c; all the remaining objects in the original cell are replicated and distributed in each of the new cells.

Rules of a system like the above one are used in a maximally parallel way: 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 divided, the division rule is the only one which is applied to that cell at that step. In other words, division rule for that cell interrupts all its communication channels with the other cells and with the environment. The new cells resulting from division 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 divided once again. The label of a cell precisely identifies the rules which can be applied to it.

Let us fix a tissue P system with protein on cells (and cell division) Π, we by $\mathcal{C}_1 \Rightarrow_\Pi \mathcal{C}_2$ denote that configuration $\mathcal{C}_1$ yields configuration $\mathcal{C}_2$ in one transition step by a maximally parallel application of rules as described above. A configuration is a *halting configuration* if no rule of the system is applicable to it. A *computation* 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-first term of the sequence is obtained from the previous configuration by applying 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 the computations start from an initial configuration and proceed as stated above; only a halting computation gives a result, which is encoded by the objects present in the output zone i_{out} associated with the halting configuration.

By collecting the results of all possible computations in Π we obtain the set of natural number generated by Π, denoted by $N(\Pi)$. The families of all sets of numbers computed by tissue P systems with protein on cells with at most m membranes and communication rules of length at most k are denoted by $NOP_m(commu_k)$.

3.1. Recognizer Tissue P Systems with Protein on Cells and Cell Division

In order to study the computing efficiency, the notions from classical *computational complexity theory* are adapted for membrane computing. A class of cell-like P systems, recognizer P systems, is introduced in [25]. With the same idea as for recognizer cell-like P systems, recognizer tissue P systems are introduced in [15].

Definition 3.3. A recognizer tissue P system with protein on cells and cell division of degree $q \geq 1$ is a tuple $\Pi = (\Gamma, P, \Sigma, \mathcal{E}, \mathcal{M}_1/p_1, \dots, \mathcal{M}_q/p_q, \mathcal{R}, i_{in}, i_{out})$, where:

- the tuple $(\Gamma, P, \mathcal{E}, \mathcal{M}_1/p_1, \dots, \mathcal{M}_q/p_q, \mathcal{R}, i_{out})$ is a tissue P system with protein on cells and cell division of degree $q \geq 1$;
- the working alphabet Γ has two distinguished objects `yes` and `no`, with at least one copy of them presents in some initial multisets $\mathcal{M}_1, \dots, \mathcal{M}_q$, but none of them present in $\mathcal{E}$;
- Σ is an (input) alphabet strictly contained in Γ, and such that $\mathcal{E} \subseteq \Gamma \setminus \Sigma$;
- $\mathcal{M}_1, \dots, \mathcal{M}_q$ are finite multisets over $\Gamma \setminus \Sigma$;
- $i_{in} \in \{1, \dots, q\}$ is the input cell;
- the output zone i_{out} is the environment;
- all computations halt;
- 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 multiset w over Σ, the *computation of the system* Π *with input* w starts from the configuration of the form $(\mathcal{M}_1/p_1, \dots, (\mathcal{M}_{i_{in}} + w)/p_{i_{in}}, \dots, \mathcal{M}_q/p_q, \emptyset)$, 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.

We denote by **TPDC**(k) the class of recognizer tissue P systems with protein on cells and cell division with communication rules of length at most k.

3.2. Polynomial Complexity Classes of Recognizer Tissue P Systems with Protein on Cells and Cell Division

NP-completeness has been usually studied in the framework of decision problems. Let us recall that a decision problem is a pair (I_X, θ_X) where I_X is a language over a finite alphabet (whose elements are called *instances*) and θ_X is a total boolean function (that is, a predicate) over I_X.

Definition 3.4. A decision problem $X = (I_X, \theta_X)$ is solvable in polynomial time by a family $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ of recognizer tissue P systems with protein on cells and cell division in a uniform way if the following conditions hold:

- the family $\mathbf{\Pi}$ is polynomially uniform by Turing machines, that is, there exists a deterministic Turing machine working in polynomial time which constructs the system $\Pi(n)$ from $n \in \mathbb{N}$;
- there exists a pair (cod, s) of polynomial-time computable functions over I_X such that:
  - for each instance $u \in I_X$, $s(u)$ is a natural number and $cod(u)$ is an input multiset of the system $\Pi(s(u))$;
  - for each $n \in \mathbb{N}$, $s^{-1}(n)$ is a finite set;
  - 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;
  - the family $\mathbf{\Pi}$ is sound with regard to (X, cod, s), that is, for each $u \in I_X$, if there exists an accepting computation of $\Pi(s(u))$ with input $cod(u)$, then $\theta_X(u) = 1$;
  - the family $\mathbf{\Pi}$ is complete with regard to (X, cod, s), that is, for each $u \in I_X$, if $\theta_X(u) = 1$, then every computation of $\Pi(s(u))$ with input $cod(u)$ is an accepting one.

Each recognizer tissue P system with protein on cells and cell division in Definition 3.4 is *confluent* in the sense that all possible computations associated with the same input multiset must give the same answer for a given instance.

We denote by $\mathbf{PMC}_{\mathbf{TPDC}(k)}$ the set of all decision problems which can be solved by means of recognizer tissue P systems $\mathbf{TPDC}(k)$ according to the previous definition.

4. Solving the SAT Problem by Using TPDC(4)

In this section, we show how to efficiently solve the SAT problem by tissue P systems with protein on cells and cell division with communication rules of length at most 4.

The SAT problem is defined as follows: given a Boolean formula in conjunctive normal form (CNF), determine whether or not there exists an assignment to its variables such that the formula is evaluated to be true. This is a well known **NP**-complete problem [26].

The solution proposed follows a brute force algorithm in the framework of recognizer tissue P systems with protein on cells and cell division. The solution consists of the following phases:

- *Generation phase*: all truth assignments for the n variables are produced by using cell division in an adequate way.
- *Checking phase*: it is checked whether or not there is a truth assignment that makes the Boolean formula evaluate to be true.
- *Output phase*: the system sends to the environment the right answer according to the results of the previous phase.

Let us consider the polynomial-time computable function $\langle m, n\rangle = ((m+n)(m+n+1)/2) + m$ (the pair function), which is a primitive recursive and bijective function from $\mathbb{N}^2$ to $\mathbb{N}$.

We construct a family $\mathbf{\Pi} = \{\Pi(t) \mid t \in \mathbb{N}\}$ such that each system $\Pi(t)$ will process all instances of the SAT problem 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 $m, n \in \mathbb{N}$, we consider the recognizer tissue P system with protein on cells and cell division from **TPDC**(4),

$$\Pi(\langle m, n\rangle) = (\Gamma, P, \Sigma, \mathcal{E}, \mathcal{M}_1/p_1, \mathcal{M}_2/q_1, \mathcal{M}_3/r, \mathcal{M}_4/s, \mathcal{R}, i_{in}, i_{out}),$$

with the following components:

$$\begin{aligned}
\Gamma &= \Sigma \cup \{a_i \mid 1 \leq i \leq n\} \cup \{b_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq m+1\} \\
&\cup \{c_i, d_{i,0}, d_{i,1} \mid 1 \leq i \leq m\} \cup \{g_i \mid 1 \leq i \leq mn + 3n + 4m\} \\
&\cup \{a_{n+1}, d_{m+1,0}, h, \texttt{yes}, \texttt{no}\}, \\
\Sigma &= \{x_{i,j}, \bar{x}_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq m\}, \\
P &= \{p_i, q_i \mid 1 \leq i \leq n+1\} \cup \{\bar{p}_i \mid 2 \leq i \leq n+1\} \cup \{r, s\}, \\
\mathcal{E} &= \{c_i, d_{i,0}, d_{i,1} \mid 1 \leq i \leq m\} \cup \{b_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq m+1\} \\
&\cup \{g_i \mid 1 \leq i \leq mn + 3n + 4m\}, \\
\mathcal{M}_1 &= \{a_1, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}\}, \mathcal{M}_2 = \{b_{1,1}\}, \mathcal{M}_3 = \{\texttt{yes}, \texttt{no}\}, \mathcal{M}_4 = \{g_1\}, \\
i_{in} &= 1 \text{ is the input cell}, \\
i_{out} &= 0 \text{ is the output zone},
\end{aligned}$$

and the set $\mathcal{R}$ of rules consists of the following rules:

$r_{1,i} \equiv [\, p_i \mid a_i \,]_1 \to [\, p_{i+1} \mid h \,]_1 [\, \bar{p}_{i+1} \mid h \,]_1$, $1 \leq i \leq n$;

$r_{2,i} \equiv [\, \bar{p}_i \mid a_i \,]_1 \to [\, p_{i+1} \mid h \,]_1 [\, \bar{p}_{i+1} \mid h \,]_1$, $2 \leq i \leq n$;

$r_{3,i,j} \equiv (1, (p_{i+1}, x_{i,j})/c_j, 0)$, $1 \leq i \leq n, 1 \leq j \leq m$;

$r_{4,i,j} \equiv (1, (\bar{p}_{i+1}, \bar{x}_{i,j})/c_j, 0)$, $1 \leq i \leq n, 1 \leq j \leq m$;

$r_{5,i,j} \equiv (2, (q_i, b_{i,j})/b_{i,j+1}, 0)$, $1 \leq i \leq n, 1 \leq j \leq m$;

$r_{6,i} \equiv [\, q_i \mid b_{i,m+1} \,]_2 \to [\, q_{i+1} \mid a_{i+1} \,]_2 [\, q_{i+1} \mid a_{i+1} \,]_2$, $1 \leq i \leq n$;

$r_{7,i} \equiv (1, (p_i, b_{i,1})/(q_i, a_i), 2)$, $2 \leq i \leq n$;

$r_{8,i} \equiv (1, (\bar{p}_i, b_{i,1})/(q_i, a_i), 2)$, $2 \leq i \leq n$;

$r_{9,i} \equiv (1, (q_i, \lambda)/(p_i, \lambda), 2)$, $2 \leq i \leq n$;

$r_{10,i} \equiv (1, (q_i, \lambda)/(\bar{p}_i, \lambda), 2)$, $2 \leq i \leq n$;

$r_{11,j} \equiv (1, (p_{n+1}, c_j d_{j,0})/(q_{n+1}, \lambda), 2)$, $1 \leq j \leq m$;

$r_{12,j} \equiv (1, (\bar{p}_{n+1}, c_j d_{j,0})/(q_{n+1}, \lambda), 2)$, $1 \leq j \leq m$;

$r_{13,j} \equiv (2, (p_{n+1}, d_{j,0})/d_{j,1}, 0)$, $1 \leq j \leq m$;

$r_{14,j} \equiv (2, (\bar{p}_{n+1}, d_{j,0})/d_{j,1}, 0)$, $1 \leq j \leq m$;

$r_{15,j} \equiv (1, (q_{n+1}, \lambda)/(p_{n+1}, d_{j,1}), 2)$, $1 \leq j \leq m$;

$r_{16,j} \equiv (1, (q_{n+1}, \lambda)/(\bar{p}_{n+1}, d_{j,1}), 2)$, $1 \leq j \leq m$;

$r_{17,j} \equiv (1, (p_{n+1}, d_{j,1})/d_{j+1,0}, 0)$, $1 \leq j \leq m$;

$r_{18,j} \equiv (1, (\bar{p}_{n+1}, d_{j,1})/d_{j+1,0}, 0)$, $1 \leq j \leq m$;

$r_{19} \equiv (1, (p_{n+1}, d_{m+1,0})/(r, \texttt{yes}), 3)$;

$r_{20} \equiv (1, (\bar{p}_{n+1}, d_{m+1,0})/(r, \texttt{yes}), 3)$;

$r_{21} \equiv (1, (r, \texttt{yes})/\lambda, 0)$;

$r_{22,i} \equiv (4, (s, g_i)/g_{i+1}, 0)$, $1 \leq i \leq mn + 3n + 4m - 1$;

$r_{23} \equiv (4, (s, g_{mn+3n+4m})/(r, \lambda), 3)$;

$r_{24} \equiv (3, (s, g_{mn+3n+4m}\texttt{no})/\lambda, 0)$.

4.1. An Overview of the Computation

A family of recognizer tissue P systems with protein on cells and cell division is constructed as above. We describe an arbitrary instance φ of the SAT problem as $\varphi = C_1 \wedge \cdots \wedge C_m$, with $C_j = l_{j,1} \vee \cdots \vee l_{j,r_j}$, $1 \leq j \leq m$, where $Var(\varphi) = \{x_1, \ldots, x_n\}$, $l_{j,k} \in \{x_i, \neg x_i \mid 1 \leq i \leq n\}$, $1 \leq j \leq m$, $1 \leq k \leq r_j$.

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 $cod(\varphi) = \{x_{i,j} : x_i \in C_j\} \cup \{\bar{x}_{i,j} : \neg x_i \in C_j\}$, that is, $x_{i,j}$ (resp., $\bar{x}_{i,j}$) denotes variable x_i (resp., $\neg x_i$) belongs to clause C_j. Hence, the formula φ will be processed by the system $\Pi(s(\varphi))$ with input multiset $cod(\varphi)$.

In what follows, we informally describe how system $\Pi(s(\varphi))$ with input multiset $cod(\varphi)$ works.

At the initial configuration, we have objects $a_1, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi)$ in cell 1 and the protein p_1 on cell 1, object $b_{1,1}$ in cell 2 and the protein q_1 on cell 2, objects yes, no in cell 3 and the protein r on cell 3, object g_1 in cell 4 and the protein s on cell 4.

Let us start with the generation phase. The system takes $m+3$ steps to assign truth assignment of each variable x_i, and look for the clauses satisfied by the truth assignment of each variable x_i ($1 \leq i \leq n-1$), and it takes $m + 1$ steps to assign truth assignment of variable x_n, and look for the clauses satisfied by the truth assignment of each variable x_n. Hence, the generation phase takes $(m + 3)(n - 1) + m + 1 = mn + 3n - 2$ steps.

At the first $m + 1$ steps of the i-th loop ($1 \leq i \leq n$), we have three parallel processes, which are described in several items.

– The object a_i in cell 1 corresponds to variable x_i, $1 \leq i \leq n$. Under the influences of protein p_i (resp., $\bar{p}_i$) on cells 1 and the object a_i in cells 1, all cells with label 1 are divided. By using the rules $r_{1,i}$ ($1 \leq i \leq n$) and $r_{2,i}$ ($2 \leq i \leq n$) at the same step (at step 1, only rule $r_{1,1}$ is used), a cell with label 1 is divided into two copies of cell with label 1, one copy of cell 1 contains the protein p_{i+1} (corresponding to the value *true*) and the object h, the other copy of cell 1 contains the protein $\bar{p}_{i+1}$ (corresponding to the value *false*) and the object h. All the objects different from a_i in cell 1 are replicated and distributed in each of the new cells. In the next m steps, under the control of protein p_{i+1} (resp., $\bar{p}_{i+1}$) on cells 1, rules $r_{3,i,j}$ (resp., $r_{4,i,j}$) are used, the objects $x_{i,j}$ (resp., $\bar{x}_{i,j}$) from cells 1 are exchanged with the objects c_j from the environment, which correspond to the process of looking for the clauses satisfied by the truth assignment *true* (resp., *false*) of variable x_i. Note that in a cell with label 1, only one copy of object $x_{i,j}$ (resp., $\bar{x}_{i,j}$) is exchanged with the object c_j from the environment in one step, since there is only one copy of protein p_{i+1} (resp., $\bar{p}_{i+1}$) on each cell 1.

– In cells with label 2, under the influence of protein q_i, the counter object $b_{i,j}$ grows its subscript j from 1 to $m+1$ by using the rules $r_{5,i,j}$. With the protein q_i on all cells 2 and the object $b_{i,m+1}$ in all cells 2, rule $r_{6,i}$ is applied, one copy of cell 2 is divided into two copies of cell 2, the protein q_i is replaced by q_{i+1} on each cell 2, and the object $b_{i,m+1}$ is replaced by a_{i+1} in each of cell 2.

– In parallel with the above process, the counter object g_i in the cell with label 4 grows its subscript by using the rules $r_{22,i}$.

At the $m+2$ step of the i-th loop ($1 \leq i \leq n-1$), the protein p_i (resp., $\bar{p}_i$) and the object $b_{i,1}$ in all cells 1 are exchanged with the protein q_i and the object a_i in all cells 2 (there are 2^{i-2} copies of cell with label 1 that contain the protein p_i and the object $b_{i,1}$, 2^{i-2} copies of cell with label 1 that contain the protein $\bar{p}_i$ and the object $b_{i,1}$ and 2^{i-1} copies of cell with label 2 that contain the protein q_i and the object a_i; each protein p_i or $\bar{p}_i$ and object $b_{i,1}$ in cells 1 are exchanged with the protein q_i and object a_i in cells 2 by using the rules $r_{7,i}$ and $r_{8,i}$ in a maximally parallel manner). In cell with label 4, by using the rule $r_{22,i}$, the counter object g_i grows its subscript by one.

At the $m+3$ step of the i-th loop ($1 \leq i \leq n-1$), each protein q_i on cells 1 is exchanged with the protein p_i or $\bar{p}_i$ on cells 2 by using the rules $r_{9,i}$ and $r_{10,i}$ in a maximally parallel manner. Simultaneously, by using the rule $r_{22,i}$, the counter object g_i in cell 4 increases its subscript by one.

In this way, after $mn+3n-2$ steps, the generation phase finishes and checking phase starts. At this moment, we have

– 2^{n-1} copies of cell 1 which contain the protein p_{n+1} and 2^{n-1} copies of cell 1 which contain the protein $\bar{p}_{n+1}$, each of them contains an object $d_{1,0}$ and some objects from the set $\{c_1, c_2, \ldots, c_m\}$ whose elements denote the corresponding clauses satisfied by the truth assignments of the variables. Each cell 1 also contains n copies of object h, which is the "garbage" object and remains idle for the follow-up computation;

– 2^n copies of cell 2, each of them contains the protein q_{n+1} and the object a_{n+1};

– a cell 3 which contains the protein r and the objects `yes`,`no`;

– a cell 4 which contains the protein s and the object $g_{mn+3n-1}$.

The checking phase takes $4m$ steps and consists of m loops (each loop takes 4 steps). In parallel with checking whether there is a truth assignment that makes the boolean formula evaluate to true, the counter object g_i in cell 4 also grows its subscript by one for each step.

At the first step of the j-th loop ($1 \leq j \leq m$) of checking phase, the protein p_{n+1} (resp., $\bar{p}_{n+1}$) and the objects $c_j, d_{j,0}$ in cells with label 1 are exchanged with the protein q_{n+1} in cells with label 2 by using the rule $r_{11,j}$ (resp., $r_{12,j}$). If a cell with label 1 has no object c_j, then rule $r_{11,j}$ or $r_{12,j}$ cannot be used in that cell. Note that rule $r_{11,j}$ (resp., $r_{12,j}$) can be used to a cell 1 only when such cell contains all the objects $c_1, c_2, \ldots, c_{j-1}$.

At the second step of the j-th loop ($1 \leq j \leq m$) of checking phase, under the influence of protein p_{n+1} (resp., $\bar{p}_{n+1}$) on cell 2, the object $d_{j,0}$ from cell 2 is exchanged with the object $d_{j,1}$ from the environment by using the rule $r_{13,j}$ (resp., $r_{14,j}$).

At the third step of the j-th loop ($1 \leq j \leq m$) of checking phase, by applying the rule $r_{15,j}$ (resp., $r_{16,j}$), the protein q_{n+1} in cell 1 is exchanged with the protein p_{n+1} (resp., $\bar{p}_{n+1}$) and the object $d_{j,1}$ in cell 2. Note that the cells with label 1 involved in the rule $r_{15,j}$ (resp., $r_{16,j}$) contain the object c_j, otherwise, the protein placed on that cell with label 1 is p_{n+1} or $\bar{p}_{n+1}$.

At the fourth step of the j-th loop ($1 \leq j \leq m$) of checking phase, under the control of protein p_{n+1} (resp., $\bar{p}_{n+1}$) on cell 1, the object $d_{j,1}$ from cell 1 is exchanged with the object $d_{j+1,0}$ from the environment by using the rule $r_{17,j}$ (resp., $r_{18,j}$).

By using the rules $r_{11,j} - r_{18,j}$ in cells with label 1, we check whether or not all clauses are satisfied by the corresponding truth assignment. For each clause which is satisfied, the subscript j of $d_{j,0}$ is increased by one; hence the object $d_{m+1,0}$ appears in a cell with label 1 if and only if that cell contains all the objects $c_1, c_2, \ldots, c_m$ (all clauses are satisfied by that truth assignment).

The output phase starts at the $(mn + 3n + 4m - 1)$-th step, and takes 3 steps.

- *Affirmative answer*: if one of the truth assignments from a cell with label 1 has satisfied all clauses, then in that cell there is an object $d_{m+1,0}$ as described above. By using the rule r_{19} or r_{20}, the protein p_{n+1} or $\bar{p}_{n+1}$ and the object $d_{m+1,0}$ in cell 1 are exchanged with the protein r and the object `yes` in cell 3. Simultaneously, by using the rule $r_{22,mn+3n+4m-1}$, the counter object $g_{mn+3n+4m}$ will appear in cell with label 4. In the next step, the object `yes` leaves the system by using the rule r_{21}, signaling the fact that the formula is satisfiable. The computation halts at step $mn + 3n + 4m$.

- *Negative answer*: if none of the truth assignments encoded by a cell with label 1 makes the formula φ true, then object $d_{m+1,0}$ does not appear in any cell labelled by 1. Thus, at step $mn+3n+4m-1$, rules r_{19}, r_{20} cannot be applied, only rule $r_{22,mn+3n+4m-1}$ is applicable and the object $g_{mn+3n+4m}$ will appear in cell with label 4. In the next step, the protein s and the object $g_{mn+3n+4m}$ in cell 4 are exchanged with the protein r in cell 3. Finally, at step $mn + 3n + 4m + 1$, under the control of protein s on cell 3, the objects $g_{mn+3n+4m}$ and `no` are sent to the environment, signaling that the formula is not satisfiable, and the computation halts.

4.2. Formal Verification

In this subsection, we prove that the family of recognizer tissue P systems with protein on cells and cell division constructed above solves the `SAT` problem in polynomial time according to Definition 3.4.

4.2.1. Polynomial Uniformity of the Family

We will 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, it will be proved that $\Pi(\langle m, n\rangle)$ is built in polynomial time with respect to the size parameters 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 defined recursively from the values m and n. The necessary resources for building an element of the family are of a polynomial order, as shown below:

- size of the set Γ: $4mn + 7m + 5n + 5 \in O(mn)$;
- size of the set P: $3n + 4 \in O(n)$;
- initial number of cells: $4 \in O(1)$;
- initial number of objects: $n + 5 \in O(n)$;
- initial number of proteins: $4 \in O(1)$;
- number of rules: $4mn + 10n + 12m - 1 \in O(mn)$;
- maximum length of a rule: $4 \in O(1)$.

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.

4.2.2. Soundness and Completeness of the Family

In order to prove the soundness and completeness of the family $\mathbf{\Pi}$ with respect to $(\mathtt{SAT}, cod, s)$, we shall prove that for a given instance φ of the SAT problem, the system $\Pi(s(\varphi))$ with input $cod(\varphi)$ sends out an object `yes` if and only if the answer to the problem for the instance is affirmative and the object `no` is sent out otherwise. In both cases the answer will be sent to the environment in the last step of the computation.

Let $\{x_1, \dots, x_i\}$ be a set of propositional variables. A truth assignment of $\{x_1, \dots, x_i\}$ will be indistinctly denoted by $\sigma = (\alpha_2, \dots, \alpha_{i+1})$, where $\alpha_j \in \{p_j, \bar{p}_j\}$, $2 \le j \le i+1$. The 2^i truth assignment of the set $\{x_1, \dots, x_i\}$ will be indistinctly denoted by $\{\sigma_{i,1}, \dots, \sigma_{i,2^i}\}$. For each i ($1 \le i \le n$), we denote $\tau_i = \Sigma_{1 \le j \le m} |cod(\varphi)|_{x_{i,j}}$, $\bar{\tau}_i = \Sigma_{1 \le j \le m} |cod(\varphi)|_{\bar{x}_{i,j}}$, $\rho_i = cod(\varphi) \cap \{x_{i,j} \mid 1 \le j \le m\}$, $\bar{\rho}_i = cod(\varphi) \cap \{\bar{x}_{i,j} \mid 1 \le j \le m\}$. $cod(\varphi)_i^k$ the set of elements ρ_i, where the number of elements is k; $cod(\varphi)_i^{\bar{k}}$ the set of elements $\bar{\rho}_i$, where the number of elements is k. We also denote

$$\delta_{i,j} = \bigcup_{1 \le k \le i} \{\{\rho_k \mid t_k \in \sigma_{i,j}\} \cup \{\bar{\rho}_k \mid f_k \in \sigma_{i,j}\}\},\ 1 \le i \le n, 1 \le j \le 2^i;$$

$$\delta'_{i,j} = cod(\varphi) \setminus \delta_{i,j};\ \xi_{i,j} = \delta'_{i,j} \cup \rho_i;\ \bar{\xi}_{i,j} = \delta'_{i,j} \cup \bar{\rho}_i,\ \eta_k = \bigcup_{1 \le j \le k} \{c_j\} \cup \{d_{1,0}\}.$$

2^i cells with label 1 generated by the system will be denoted by $1_{(i,1)}, 1_{(i,2)}, \dots, 1_{(i,2^i)}$.

Given a computation $\mathcal{C}$ we denote the configuration at the i-th step as $\mathcal{C}_i$. Moreover, $\mathcal{C}_i(l)$ will denote the multiset associated with cell l in such a configuration. The protein associated with cell l at configuration i is denoted by $\mathcal{P}_i(l)$.

Lemma 4.1. Let $\mathcal{C}$ be an arbitrary computation of the system, then for every i ($1 \le i \le n-1$), we have the following:

(1) At configuration $\mathcal{C}_{(m+3)(i-1)+1}$, we have

(a) 2^i copies of cell with label 1 from which:

- 2^{i-1} copies of cell with label 1 contain the protein p_{i+1}, the cell with label $1_{(i,j)}$ ($1 \le j \le 2^i$) contains the objects $b_{i+1,1}, \ldots, b_{n,1}$, $d_{1,0}, \xi_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j} \setminus \{p_{i+1}\}$;
- 2^{i-1} copies of cell with label 1 contain the protein $\bar{p}_{i+1}$, the cell with label $1_{(i,j)}$ ($1 \le j \le 2^i$) contains the objects $b_{i+1,1}, \ldots, b_{n,1}$, $d_{1,0}, \bar{\xi}_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j} \setminus \{\bar{p}_{i+1}\}$;

(b) 2^{i-1} copies of cell with label 2, each of them contains the protein q_i and an object $b_{i,2}$;

(c) a cell 3 that contains the protein r and the objects `yes`, `no`;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(i-1)+2}$.

(2) At configuration $\mathcal{C}_{(m+3)(i-1)+1+k}$ ($1 \le k < m$), we have

(a) 2^i copies of cell with label 1 from which:

- 2^{i-1} copies of cell with label 1 contain the protein p_{i+1}. If $k < \tau_i$, then the cell with label $1_{(i,j)}$ ($1 \le j \le 2^i$) contains $\mathcal{C}_{(m+3)(i-1)+1}\ (1_{(i,j)}) \setminus cod(\varphi)_i^k$ and some objects c_j, where an object c_j corresponds to the object $x_{i,j}$ that has been sent out in previous k steps. If $\tau_i \le k < m$, then the cell with label $1_{(i,j)}$ contains the objects $b_{i+1,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j}$;
- 2^{i-1} copies of cell with label 1 contain the protein $\bar{p}_{i+1}$. If $k < \bar{\tau}_i$, then the cell with label $1_{(i,j)}$ ($1 \le j \le 2^i$) contains $\mathcal{C}_{(m+3)(i-1)+1}\ (1_{(i,j)}) \setminus cod(\varphi)_i^{\bar{k}}$ and some objects c_j, where an object c_j corresponds to the object $\bar{x}_{i,j}$ that has been sent out in previous k steps. If $\bar{\tau}_i \le k < m$, then the cell with label $1_{(i,j)}$ contains the objects $b_{i+1,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j}$;

(b) 2^{i-1} copies of cell with label 2, each of them contains the protein q_i and an object $b_{i,k+2}$;

(c) a cell 3 that contains the protein r and the objects `yes`, `no`;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(i-1)+2+k}$.

(3) At configuration $\mathcal{C}_{(m+3)(i-1)+m+1}$, we have

(a) 2^{i-1} copies of cell with label 1 contain the protein p_{i+1}, 2^{i-1} copies of cell with label 1 contain the protein $\bar{p}_{i+1}$. The cell with label $1_{(i,j)}$ ($1 \le j \le 2^i$) contains the objects $b_{i+1,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j}$;

(b) 2^i copies of cell with label 2, each of them contains the protein q_{i+1} and an object a_{i+1};

(c) a cell 3 that contains the protein r and the objects $\mathtt{yes}, \mathtt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(i-1)+m+2}$.

(4) At configuration $\mathcal{C}_{(m+3)(i-1)+m+2}$, we have

(a) 2^i copies of cell with label 1, each of them contains the protein q_{i+1}, the objects $a_{i+1}, b_{i+2,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j}$;

(b) 2^{i-1} copies of cell with label 2 contain the protein p_{i+1}, 2^{i-1} copies of cell with label 2 contain the protein $\bar{p}_{i+1}$, each of them contains an object $b_{i+1,1}$;

(c) a cell 3 that contains the protein r and the objects $\mathtt{yes}, \mathtt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(i-1)+m+3}$.

(5) At configuration $\mathcal{C}_{(m+3)(i-1)+m+3}$, we have

(a) 2^{i-1} copies of cell with label 1 contain the protein p_{i+1}, 2^{i-1} copies of cell with label 1 contain the protein $\bar{p}_{i+1}$. The cell with label $1_{(i,j)}$ $(1 \le j \le 2^i)$ contains the objects $a_{i+1}, b_{i+2,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j}$;

(b) 2^i copies of cell with label 2, each of them contains the protein q_{i+1} and an object $b_{i+1,1}$;

(c) a cell 3 that contains the protein r and the objects $\mathtt{yes}, \mathtt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(i-1)+m+4}$.

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

At the initial configuration, we have:
$\mathcal{C}_0(1) = \{a_1, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi)\}$, $\mathcal{P}_0(1) = \{p_1\}$;
$\mathcal{C}_0(2) = \{b_{1,1}\}$, $\mathcal{P}_0(2) = \{q_1\}$;
$\mathcal{C}_0(3) = \{\mathtt{yes}, \mathtt{no}\}$, $\mathcal{P}_0(3) = \{r\}$;
$\mathcal{C}_0(4) = \{g_1\}$, $\mathcal{P}_0(4) = \{s\}$.

At step 1, rule $r_{1,1}$ is used, the cell with label 1 is divided into two copies of cell 1, one cell with label 1 contains the protein p_2 (representing the true value *true*) and the object h, the other cell with label 1 contains the protein $\bar{p}_2$ (representing the true value *false*) and the object h. Simultaneously, rule $r_{5,1,1}$ is applied, object $b_{1,2}$ appears in cell 2. The counter object g_2 presents in cell 4 by using the rule $r_{22,1}$. Therefore,
$\mathcal{C}_1(1_{(1,1)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi)\}$, $\mathcal{P}_1(1_{(1,1)}) = \{p_2\}$;
$\mathcal{C}_1(1_{(1,2)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi)\}$, $\mathcal{P}_1(1_{(1,2)}) = \{\bar{p}_2\}$;
$\mathcal{C}_1(2) = \{b_{1,2}\}$, $\mathcal{P}_1(2) = \{q_1\}$;
$\mathcal{C}_1(3) = \{\mathtt{yes}, \mathtt{no}\}$, $\mathcal{P}_1(3) = \{r\}$;
$\mathcal{C}_1(4) = \{g_2\}$, $\mathcal{P}_1(4) = \{s\}$.

At step 2, if input multiset $cod(\varphi)$ contains the object $x_{1,j}$ (resp., $\bar{x}_{1,j}$), with the protein p_2 (resp., $\bar{p}_2$) on cell 1, rule $r_{3,1,j}$ (resp., $r_{4,1,j}$) is used, an object $x_{i,j}$ (resp., $\bar{x}_{i,j}$) from cell 1 is exchanged with c_j from the environment. Note that if $\tau_1 > 1$ (resp., $\bar{\tau}_1 > 1$), then $x_{1,j}$ and $x_{1,k}$ (resp., $\bar{x}_{1,j}$ and $\bar{x}_{1,k}$) are chosen to use non-deterministically, and only one of them can be used in one step. Simultaneously, object $b_{1,3}$ presents in cell 2 by applying the rule $r_{5,1,2}$ and the object g_3 appears in cell 4 by using the rule $r_{22,2}$.

$\mathcal{C}_2(1_{(1,1)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi) \setminus cod(\varphi)^1_1, c_j\}$, $\mathcal{P}_2(1_{(1,1)}) = \{p_2\}$;

$\mathcal{C}_2(1_{(1,2)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi) \setminus cod(\varphi)^{\bar{1}}_1, c_j\}$, $\mathcal{P}_2(1_{(1,2)}) = \{\bar{p}_2\}$;

$\mathcal{C}_2(2) = \{b_{1,3}\}$, $\mathcal{P}_2(2) = \{q_1\}$;

$\mathcal{C}_2(3) = \{\texttt{yes}, \texttt{no}\}$, $\mathcal{P}_2(3) = \{r\}$;

$\mathcal{C}_2(4) = \{g_3\}$, $\mathcal{P}_2(4) = \{s\}$.

Clearly, at step $j+1$ ($2 \leq j < m$), we have the following.

If $j < \tau_1$, then $\mathcal{C}_{j+1}(1_{(1,1)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi) \setminus cod(\varphi)^j_1\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the objects $x_{1,k}$ that have been sent out in previous j steps, $\mathcal{P}_{j+1}(1_{(1,1)}) = \{p_2\}$; if $\tau_1 \leq j < m$, $\mathcal{C}_{j+1}(1_{(1,1)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,1}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,1}$, $\mathcal{P}_{j+1}(1_{(1,1)}) = \{p_2\}$.

If $j < \bar{\tau}_1$, then $\mathcal{C}_{j+1}(1_{(1,2)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, cod(\varphi) \setminus cod(\varphi)^{\bar{j}}_1\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the objects $\bar{x}_{1,k}$ that have been sent out in previous j steps, $\mathcal{P}_{j+1}(1_{(1,2)}) = \{p_2\}$; if $\bar{\tau}_1 \leq j < m$, $\mathcal{C}_{j+1}(1_{(1,2)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,2}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,2}$, $\mathcal{P}_{j+1}(1_{(1,2)}) = \{p_2\}$.

$\mathcal{C}_{j+1}(2) = b_{1,j+2}$, $\mathcal{P}_2(2) = \{q_1\}$;

$\mathcal{C}_2(3) = \{\texttt{yes}, \texttt{no}\}$, $\mathcal{P}_2(3) = \{r\}$;

$\mathcal{C}_2(4) = \{g_{j+2}\}$, $\mathcal{P}_2(4) = \{s\}$.

At configuration $\mathcal{C}_m$, rules $r_{6,1}, r_{22,m+1}$ and $r_{3,1,j}$ (if all the objects $x_{1,1}, \ldots, x_{1,m}$ appear in the cell $1_{(1,1)}$) or $r_{4,1,j}$ (if all the objects $\bar{x}_{1,1}, \ldots, \bar{x}_{1,m}$ appear in the cell $1_{(1,2)}$) are applied, the object c_j will enter the corresponding cell and the cell with label 2 is divided into two copies of cell with label 2, each of them contains the protein q_2 and the object a_2. In cell 4, the object g_{m+1} is exchanged with the object g_{m+2} from the environment. That is,

$\mathcal{C}_{m+1}(1_{(1,1)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,1}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,1}$, $\mathcal{P}_{m+1}(1_{(1,1)}) = \{p_2\}$;

$\mathcal{C}_{m+1}(1_{(1,2)}) = \{h, b_{2,1}, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,2}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,2}$, $\mathcal{P}_{m+1}(1_{(1,2)}) = \{\bar{p}_2\}$;

$\mathcal{C}_{m+1}(2) = \{a_2\}$, $\mathcal{P}_{m+1}(2) = \{q_2\}$ (there are two copies of cell 2 with the same protein and object in each cell);

$\mathcal{C}_{m+1}(3) = \{\texttt{yes}, \texttt{no}\}$, $\mathcal{P}_{m+1}(3) = \{r\}$;

$\mathcal{C}_{m+1}(4) = \{g_{m+2}\}$, $\mathcal{P}_{m+1}(4) = \{s\}$.

At configuration $\mathcal{C}_{m+1}$, the protein p_2 (resp., $\bar{p}_2$) and object $b_{2,1}$ in cell $1_{(1,1)}$ (resp., $1_{(1,2)}$) are exchanged with the protein q_2 and the object a_2 in cell 2 by using rule $r_{7,2}$ (resp., $r_{8,2}$). The counter object g_i increases its subscript by one in cell 4. That is,

$\mathcal{C}_{m+2}(1_{(1,1)}) = \{h, a_2, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,1}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,1}$, $\mathcal{P}_{m+2}(1_{(1,1)}) = \{q_2\}$;

$\mathcal{C}_{m+2}(1_{(1,2)}) = \{h, a_2, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,2}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,2}$, $\mathcal{P}_{m+2}(1_{(1,2)}) = \{q_2\}$;

$\mathcal{C}_{m+2}(2) = \{b_{2,1}\}$ (there are two copies of cell 2, one contains the protein p_2, and the other contains the protein $\bar{p}_2$);

$\mathcal{C}_{m+2}(3) = \{\mathtt{yes}, \mathtt{no}\}$, $\mathcal{P}_{m+2}(3) = \{r\}$;

$\mathcal{C}_{m+2}(4) = \{g_{m+3}\}$, $\mathcal{P}_{m+2}(4) = \{s\}$.

At configuration $\mathcal{C}_{m+2}$, the protein q_2 on cell 1 is exchanged with the protein p_2 (resp., $\bar{p}_2$) by using the rule $r_{9,2}$ (resp., $r_{10,2}$). The object g_{m+3} from cell 4 is exchanged with the object g_{m+4} from the environment. That is,

$\mathcal{C}_{m+3}(1_{(1,1)}) = \{h, a_2, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,1}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,1}$;

$\mathcal{C}_{m+3}(1_{(1,2)}) = \{h, a_2, b_{3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{1,2}\}$, it also contains some objects from set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{1,2}$;

$\mathcal{P}_{m+3}(1_{(1,1)}) = \{p_2\}$, $\mathcal{P}_{m+3}(1_{(1,2)}) = \{\bar{p}_2\}$; or

$\mathcal{P}_{m+3}(1_{(1,1)}) = \{\bar{p}_2\}$, $\mathcal{P}_{m+3}(1_{(1,2)}) = \{p_2\}$;

$\mathcal{C}_{m+3}(2) = \{b_{2,1}\}$, $\mathcal{P}_{m+3}(2) = \{q_2\}$ (there are two copies of cell 2 with the same protein and object in each cell);

$\mathcal{C}_{m+3}(3) = \{\mathtt{yes}, \mathtt{no}\}$, $\mathcal{P}_{m+3}(3) = \{r\}$;

$\mathcal{C}_{m+3}(4) = \{g_{m+4}\}$, $\mathcal{P}_{m+3}(4) = \{s\}$.

Thus, the results of the Lemma hold for $i = 1$.

By induction hypothesis, suppose (1), (2), (3), (4) and (5) hold for i ($1 \leq i < n-1$). Let us see that (1), (2), (3), (4) and (5) also hold for $i+1$.

We assume that after $(m+3)i$ steps, we have

(a) 2^{i-1} copies of cell with label 1 contain the protein p_{i+1}, 2^{i-1} copies of cell with label 1 contain the protein $\bar{p}_{i+1}$. The cell with label $1_{(i,j)}$ contains the objects $a_{i+1}, b_{i+2,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i,j}$;

(b) 2^i copies of cell with label 2, each of them contains the protein q_{i+1} and an object $b_{i+1,1}$;

(c) a cell 3 that contains the protein r and the objects $\mathtt{yes}$, $\mathtt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)i+1}$.

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

(a) by applying the rule $r_{1,i+1}$ (resp., $r_{2,i+1}$), a cell with label 1 contained the protein p_{i+1} (resp., $\bar{p}_{i+1}$) and the object a_{i+1} is divided into two copies of cell with label 1. Thus, we have 2^{i+1} copies of cell with label 1 from which:

   – 2^i copies of cell with label 1 contain the protein p_{i+2}, the cell with label $1_{(i+1,j)}$ contains the objects $b_{i+2,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$ (obviously, $\xi_{i+1,j} = \delta'_{i,j}$), and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i+1,j} \setminus \{p_{i+2}\}$;
   – 2^i copies of cell with label 1 contain the protein $\bar{p}_{i+2}$, the cell with label $1_{(i+1,j)}$ contains the objects $b_{i+2,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i,j}, h$ (obviously, $\bar{\xi}_{i+1,j} = \delta'_{i,j}$), and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i+1,j} \setminus \{\bar{p}_{i+2}\}$;

(b) object $b_{i+1,1}$ from each cell 2 is exchanged with $b_{i+1,2}$ from the environment by using the rule $r_{5,i+1,1}$. Thus, there are 2^i copies of cell with label 2, each of them contains the protein q_{i+1} and an object $b_{i+1,2}$;

(c) a cell 3 that contains the protein r and the objects yes, no;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)i+2}$ (the object $g_{(m+3)i+1}$ from cell 4 is exchanged with the object $g_{(m+3)i+2}$ from the environment by using the rule $r_{22,(m+3)i+2}$).

Hence, the result holds for configuration $\mathcal{C}_{(m+3)i+1}$.
At configuration $\mathcal{C}_{(m+3)i+k}$ $(1 \leq k < m)$, we have:

(a) 2^{i+1} copies of cell with label 1 from which:

- 2^i copies of cell with label 1 contain the protein p_{i+2}. In each step, an object $x_{i+1,j}$ (if possible) from cell 1 is exchanged with the object c_j from the environment by applying the rule $r_{3,i+1,j}$. If $k < \tau_{i+1}$, then the cell with label $1_{(i+1,j)}$ contains $\mathcal{C}_{(m+3)i+1}(1_{(i+1,j)}) \setminus cod(\varphi)_{i+1}^{k}$ and some objects c_j, where an object c_j corresponds to the object $x_{i+1,j}$ that has been sent out in previous k steps. If $\tau_{i+1} \leq k < m$, then the cell with label $1_{(i+1,j)}$ contains the objects $b_{i+2,1}, \dots, b_{n,1}, d_{1,0}, \delta'_{i+1,j}, h$, and some objects from the set $\{c_1, \dots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i+1,j}$;
- 2^i copies of cell with label 1 contain the protein $\bar{p}_{i+2}$. In each step, an object $\bar{x}_{i+1,j}$ (if possible) from cell 1 is exchanged with the object c_j from the environment by applying the rule $r_{4,i+1,j}$. If $k < \bar{\tau}_{i+1}$, then the cell with label $1_{(i+1,j)}$ contains $\mathcal{C}_{(m+3)i+1}(1_{(i+1,j)}) \setminus cod(\varphi)_{i+1}^{\bar{k}}$ and some objects c_j, where an object c_j corresponds to the object $\bar{x}_{i+1,j}$ that has been sent out in previous k steps. If $\bar{\tau}_{i+1} \leq k < m$, then the cell with label $1_{(i+1,j)}$ contains the objects $b_{i+2,1}, \dots, b_{n,1}, d_{1,0}, \delta'_{i+1,j}, h$, and some objects from the set $\{c_1, \dots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i+1,j}$;

(b) 2^i copies of cell with label 2, each of them contains the protein q_{i+1} and an object $b_{i+1,k+2}$;

(c) a cell 3 that contains the protein r and the objects yes, no;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)i+2+k}$ (by applying the rule $r_{22,(m+3)i+k}$).

Hence, the result holds for configuration $\mathcal{C}_{(m+3)i+k+1}$ $(1 \leq k < m)$.
At configuration $\mathcal{C}_{(m+3)i+m}$:

(a) 2^i copies of cell with label 1 contain the protein p_{i+2}, 2^i copies of cell with label 1 contain the protein $\bar{p}_{i+2}$. If the cell with label 1 contains all the objects $x_{i+1,1}, \dots, x_{i+1,m}$ (resp., $\bar{x}_{i+1,1}, \dots, \bar{x}_{i+1,m}$), then one rule $r_{3,i+1,j}$ (resp., $r_{4,i+1,j}$) is applied, the object $x_{i+1,j}$ (resp., $\bar{x}_{i+1,j}$) from the cell 1 is exchanged with the object c_j from the environment. Thus, the cell with label $1_{(i+1,j)}$ contains the objects $b_{i+2,1}, \dots, b_{n,1}, d_{1,0}, \delta'_{i+1,j}, h$, and some objects from the set $\{c_1, \dots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i+1,j}$;

(b) rule $r_{6,i+1}$ is used, one copy of cell with label 2 is divided into two copies of cell with label 2, each cell contains the same protein and object. Thus, we obtain 2^{i+1} copies of cell with label 2, each of them contains the protein q_{i+2} and an object a_{i+2};

(c) a cell 3 that contains the protein r and the objects yes, no;

(d) rule $r_{22,(m+3)i+m+1}$ is used, and the object $g_{(m+3)i+m+2}$ enters the cell 4. Thus, a cell 4 that contains the protein s and the object $g_{(m+3)i+m+2}$.

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

(a) the protein p_{i+2} (resp., $\bar{p}_{i+2}$) and the object $b_{i+2,1}$ in all cells with label 1 are exchanged with the protein q_{i+2} and the object a_{i+2} (there are 2^{i+1} copies of cell with label 1 and 2^{i+1} copies of cell with label 2, a protein q_{i+2} and an object a_{i+2} enter a cell 1 by using the rules $r_{7,i+2}$ and $r_{8,i+2}$ in a maximally parallel manner). Thus, there are 2^{i+1} copies of cell with label 1, each of them contains the protein q_{i+2}, the objects $a_{i+2}, b_{i+3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i+1,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i+1,j}$;

(b) 2^i copies of cell with label 2 contain the protein p_{i+2}, 2^i copies of cell with label 2 contain the protein $\bar{p}_{i+2}$, each of them contains an object $b_{i+2,1}$;

(c) a cell 3 that contains the protein r and the objects yes, no;

(d) by using the rule $r_{22,(m+3)i+m+2}$, a cell 4 that contains the protein s and the object $g_{(m+3)i+m+3}$.

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

(a) the proteins q_{i+2} in all cells with label 1 are exchanged with the proteins p_{i+2} and $\bar{p}_{i+2}$ (there are 2^{i+1} copies of cell with label 1 and 2^{i+1} copies of cell with label 2, a protein q_{i+2} in cell 1 is exchanged with a protein p_{i+2} or $\bar{p}_{i+2}$ in cell 2 by using the rules $r_{9,i+2}$ and $r_{10,i+2}$ in a maximally parallel manner). Thus, we obtain 2^i copies of cell with label 1 contain the protein p_{i+2}, 2^i copies of cell with label 1 contain the protein $\bar{p}_{i+2}$. The cell with label $1_{(i+1,j)}$ contains the objects $a_{i+2}, b_{i+3,1}, \ldots, b_{n,1}, d_{1,0}, \delta'_{i+1,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{i+1,j}$;

(b) 2^{i+1} copies of cell with label 2, each of them contains the protein q_{i+2} and an object $b_{i+2,1}$;

(c) a cell 3 that contains the protein r and the objects yes, no;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)i+m+4}$.

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

Lemma 4.2. Let $\mathcal{C}$ be an arbitrary computation of the system, at configuration $C_{(m+3)(n-1)+1}$, we have the following:

(a) 2^n copies of cell with label 1 from which:

- 2^{n-1} copies of cell with label 1 contain the protein p_{n+1}, the cell with label $1_{(n,j)}$ ($1 \leq j \leq 2^n$) contains the objects $d_{1,0}, \xi_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j} \setminus \{p_{n+1}\}$;

– 2^{n-1} copies of cell with label 1 contain the protein $\bar{p}_{n+1}$, the cell with label $1_{(n,j)}$ ($1 \leq j \leq 2^n$) contains the objects $d_{1,0}, \bar{\xi}_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j} \setminus \{\bar{p}_{n+1}\}$;

(b) 2^{n-1} copies of cell with label 2, each of them contains the protein q_n and an object $b_{n,2}$;

(c) a cell 3 that contains the protein r and the objects `yes`, `no`;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(n-1)+2}$.

Proof:
From Lemma 4.1 for $i = n - 1$, we know that at configuration $\mathcal{C}_{(m+3)(n-1)}$, we have:

(a) 2^{n-2} copies of cell with label 1 contain the protein p_n, 2^{n-2} copies of cell with label 1 contain the protein $\bar{p}_n$. The cell with label $1_{(n-1,j)}$ ($1 \leq j \leq 2^{n-1}$) contains the objects $a_n, d_{1,0}, \delta'_{n-1,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n-1,j}$;

(b) 2^{n-1} copies of cell with label 2, each of them contains the protein q_n and an object $b_{n,1}$;

(c) a cell 3 that contains the protein r and the objects `yes`, `no`;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(n-1)+1}$.

By applying the rules $r_{1,n}$ and $r_{2,n}$, each cell with label 1 contained the protein p_n (resp., $\bar{p}_n$) and the object a_n is divided into two copies of cell with label 1, one of them contains the protein p_{n+1} and the object t_n, the other contains the protein $\bar{p}_{n+1}$ and the object f_n. Thus, we have 2^n copies of cell with label 1 from which:

– 2^{n-1} copies of cell with label 1 contain the protein p_{n+1}, the cell with label $1_{(n,j)}$ ($1 \leq j \leq 2^n$) contains the objects $d_{1,0}, \xi_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j} \setminus \{p_{n+1}\}$;

– 2^{n-1} copies of cell with label 1 contain the protein $\bar{p}_{n+1}$, the cell with label $1_{(n,j)}$ ($1 \leq j \leq 2^n$) contains the objects $d_{1,0}, \bar{\xi}_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j} \setminus \{\bar{p}_{n+1}\}$;

There are 2^{n-1} copies of cell with label 2. In each cell with label 2, the object $b_{n,1}$ from cell 2 is exchanged with the object $b_{n,2}$ from the environment by using the rule $r_{5,n,1}$.

In cell 3, there is no rule used, and it contains the protein r and the objects `yes`, `no`.

By applying the rule $r_{22,(m+3)(n-1)+1}$, object $g_{(m+3)(n-1)+1}$ from cell 4 is exchanged with the object object $g_{(m+3)(n-1)+2}$ from the environment. Thus, the cell with label 4 contains the protein s and the object $g_{(m+3)(n-1)+2}$. □

Lemma 4.3. Let $\mathcal{C}$ be an arbitrary computation of the system, at configuration $\mathcal{C}_{(m+3)(n-1)+1+k}$ ($1 \leq k < m$), we have the following:

(a) 2^n copies of cell with label 1 from which:

– 2^{n-1} copies of cell with label 1 contain the protein p_{n+1}. If $k < \tau_n$, then the cell with label $1_{(n,j)}$ $(1 \leq j \leq 2^n)$ contains $\mathcal{C}_{(m+3)(n-1)+1}\ (1_{(n,j)}) \setminus cod(\varphi)_n^k$ and some objects c_j, where an object c_j corresponds to the object $x_{n,j}$ that has been sent out in previous k steps. If $\tau_n \leq k < m$, then the cell with label $1_{(n,j)}$ contains the objects $d_{1,0}, \delta'_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j}$;

– 2^{n-1} copies of cell with label 1 contain the protein $\bar{p}_{n+1}$. If $k < \bar{\tau}_n$, then the cell with label $1_{(n,j)}$ $(1 \leq j \leq 2^n)$ contains $\mathcal{C}_{(m+3)(n-1)+1}\ (1_{(n,j)}) \setminus cod(\varphi)_n^{\bar{k}}$ and some objects c_j, where an object c_j corresponds to the object $\bar{x}_{i,j}$ that has been sent out in previous k steps. If $\bar{\tau}_i \leq k < m$, then the cell with label $1_{(n,j)}$ contains the objects $d_{1,0}, \delta'_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j}$;

(b) 2^{n-1} copies of cell with label 2, each of them contains the protein q_n and an object $b_{n,k+2}$;

(c) a cell 3 that contains the protein r and the objects yes, no;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(n-1)+2+k}$.

Proof:
From Lemma 4.2, at configuration $\mathcal{C}_{(m+3)(n-1)+1}$, there are 2^n copies of cell with label 1 from which: 2^{n-1} copies of cell with label 1 contain the protein p_{n+1}, 2^{n-1} copies of cell with label 1 contain the protein $\bar{p}_{n+1}$. If the input multiset contains some of the objects from the set $\{x_{n,1}, \ldots, x_{n,m}\}$, then one object $x_{n,j}$ (resp., $\bar{x}_{n,j}$) from a cell 1 is exchanged with the object c_j from the environment in one step by applying the rules $r_{3,n,j}$ (resp., $r_{4,n,j}$), that is, if $k < \tau_n$ (resp., $k < \bar{\tau}_n$), then the cell with label $1_{(n,j)}$ $(1 \leq j \leq 2^n)$ contains $\mathcal{C}_{(m+3)(n-1)+1}\ (1_{(n,j)}) \setminus cod(\varphi)_n^k$ (resp., $\mathcal{C}_{(m+3)(n-1)+1}\ (1_{(n,j)}) \setminus cod(\varphi)_n^{\bar{k}}$) and some objects c_j, where an object c_j corresponds to the object $x_{n,j}$ that has been sent out in previous k steps. If $\tau_n \leq k < m$ (resp., $\bar{\tau}_n \leq k < m$), then the cell with label $1_{(n,j)}$ contains the objects $d_{1,0}, \delta'_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j}$.

In all cells with label 2, the subscript j of object $b_{n,j}$ increases one in one step by using the rule $r_{5,n,j}$, thus, at configuration $\mathcal{C}_{(m+3)(n-1)+1+k}$ there are 2^{n-1} copies of cell with label 2, each of them contains the protein q_n and an object $b_{n,k+2}$.

The cell with label 3 is not evolved, and it contains the protein r and the objects yes, no.

In cell with label 4, the subscript j of object g_j increases one in one step by using the rule $r_{22,j}$, thus, at configuration $\mathcal{C}_{(m+3)(n-1)+1+k}$, the cell 4 contains the protein s and the object $g_{(m+3)(n-1)+2+k}$. □

Lemma 4.4. Let $\mathcal{C}$ be an arbitrary computation of the system, at configuration $\mathcal{C}_{(m+3)(n-1)+1+m}$, we have the following:

(a) 2^{n-1} copies of cell with label 1 contain the protein p_{n+1}, 2^{n-1} copies of cell with label 1 contain the protein $\bar{p}_{n+1}$. The cell with label $1_{(n,j)}$ $(1 \leq j \leq 2^n)$ contains the objects $d_{1,0}, \delta'_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j}$;

(b) 2^n copies of cell with label 2, each of them contains the protein q_{n+1} and an object a_{n+1};

(c) a cell 3 that contains the protein r and the objects $\texttt{yes}, \texttt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{(m+3)(n-1)+m+2}$.

Proof:
At configuration $\mathcal{C}_{(m+3)(n-1)+m}$, we have 2^{n-1} copies of cell with label 1 contain the protein p_{n+1} (resp., $\bar{p}_{n+1}$), if the input multiset contains all the object $x_{n,1}, \ldots, x_{n,m}$ (resp., $\bar{x}_{n,1}, \ldots, \bar{x}_{n,m}$), then rule $r_{3,n,j}$ (resp., $r_{4,n,j}$) is used; otherwise, there is no rule used in this step. Thus, the cell with label $1_{(n,j)}$ ($1 \le j \le 2^n$) contains the objects $d_{1,0}, \delta'_{n,j}, h$, and some objects from the set $\{c_1, \ldots, c_m\}$, which correspond to the clauses satisfied by the truth assignment $\sigma_{n,j}$.

In step $(m+3)(n-1)+1+m$, each cell with label 2 is divided into two copies of cell with label 2 by using the rule $r_{6,n}$, thus, we have 2^n copies of cell with label 2, and each of them contains the protein q_{n+1} and an object a_{n+1}.

There is no rule used in this step in cell 3, thus, it contains the protein r and the objects $\texttt{yes}, \texttt{no}$.

By using the rule $r_{22,(m+3)(n-1)+m+1}$, the object $g_{(m+3)(n-1)+m+2}$ will appear in cell 4. □

Lemma 4.5. Let $\mathcal{C}$ be an arbitrary computation of the system, for every k ($1 \le k \le m$), we have the following:

(1) At configuration $\mathcal{C}_{mn+3n-2+4(k-1)+1}$, we have:

(a) 2^n copies of cell with label 1. If a cell $1_{n,j}$ ($1 \le j \le 2^n$) has the objects c_k and $d_{k,0}$, then such cell contains the protein q_{n+1} and the objects $\mathcal{C}_{mn+3n-2+4(k-1)+1}(1_{n,j}) = \mathcal{C}_{mn+3n-2}(1_{n,j}) \setminus \eta_k$;

(b) 2^n copies of cell with label 2. If a cell with label 2 contains the protein p_{n+1} or $\bar{p}_{n+1}$, then such cell contains the objects $a_{n+1}, c_k, d_{k,0}$ and possible objects from the set $\{c_1, \ldots, c_{k-1}\}$;

(c) a cell 3 that contains the protein r and the objects $\texttt{yes}, \texttt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{mn+3n-2+4(k-1)+2}$.

(2) At configuration $\mathcal{C}_{mn+3n-2+4(k-1)+2}$, we have:

(a) 2^n copies of cell with label 1.
$\mathcal{C}_{mn+3n-2+4(k-1)+2}(1_{n,j}) = \mathcal{C}_{mn+3n-2+4(k-1)+1}(1_{n,j})$,
$\mathcal{P}_{mn+3n-2+4(k-1)+2}(1_{n,j}) = \mathcal{P}_{mn+3n-2+4(k-1)+1}(1_{n,j})$;

(b) 2^n copies of cell with label 2. If a cell with label 2 contains the protein p_{n+1} or $\bar{p}_{n+1}$, then such cell contains the objects $a_{n+1}, c_k, d_{k,1}$ and possible objects from the set $\{c_1, \ldots, c_{k-1}\}$;

(c) a cell 3 that contains the protein r and the objects $\texttt{yes}, \texttt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{mn+3n-2+4(k-1)+3}$.

(3) At configuration $\mathcal{C}_{mn+3n-2+4(k-1)+3}$, we have:

(a) 2^n copies of cell with label 1. The cells with label 1 which have the objects $c_k, d_{k,0}$ contain the protein p_{n+1} or $\bar{p}_{n+1}$, the objects $d_{k,1}$, and possible objects from the set $\{c_1, \ldots, c_m\}$;

(b) 2^n copies of cell with label 2. Each of them contains the protein q_{n+1}, the object a_{n+1} and the possible objects from the set $\{c_1, \ldots, c_k\}$;

(c) a cell 3 that contains the protein r and the objects $\mathtt{yes}$, $\mathtt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{mn+3n-2+4(k-1)+4}$.

(4) At configuration $\mathcal{C}_{mn+3n-2+4(k-1)+4}$, we have:

(a) 2^n copies of cell with label 1. The cells with label 1 which have the objects $c_k, d_{k,0}$ contain the protein p_{n+1} or $\bar{p}_{n+1}$, the objects $d_{k+1,0}$, and possible objects from the set $\{c_1, \ldots, c_m\}$;

(b) 2^n copies of cell with label 2. Each of them contains the protein q_{n+1}, the object a_{n+1} and the possible objects from the set $\{c_1, \ldots, c_k\}$;

(c) a cell 3 that contains the protein r and the objects $\mathtt{yes}$, $\mathtt{no}$;

(d) a cell 4 that contains the protein s and the object $g_{mn+3n-2+4(k-1)+5}$.

Proof:

By induction on k. Let us start analyzing the basic case $k = 1$.

At configuration $\mathcal{C}_{mn+3n-2}$, rules $r_{11,1}$ and $r_{12,1}$ are enabled in cells with label 1 which contain the objects $c_1, d_{1,0}$. By using the rule $r_{11,1}$ (resp., $r_{12,1}$), the protein p_{n+1} (resp., $\bar{p}_{n+1}$) and the objects $c_1, d_{1,0}$ in cell 1 are exchanged with the protein q_{n+1} in cell 2. Thus, the cells with label 1 which contain the objects $c_1, d_{1,0}$ have the protein q_{n+1} and the objects $\mathcal{C}_{mn+3n-2}(1_{n,j}) \setminus \eta_1$. The cells with label 2 which contain the protein p_{n+1} or $\bar{p}_{n+1}$ have the objects $a_{n+1}, c_1, d_{1,0}$. In cell 3, no rule is used at this configuration, thus, it contains the protein r and the objects $\mathtt{yes}$, $\mathtt{no}$. Rule $r_{22,mn+3n-1}$ is used, and the object g_{mn+3n} appears in cell 4.

At configuration $\mathcal{C}_{mn+3n-1}$, there is no rule can be used in all cells with label 1 and in cell with label 3, thus, the proteins and the objects in all cell 1 and in cell 3 are not changed. With the protein p_{n+1} or $\bar{p}_{n+1}$ on cell 2, the object $d_{1,0}$ from cell 2 is exchanged with the object $d_{1,1}$ by using the rule $r_{13,1}$ or $r_{14,1}$, thus, if the cells with label 2 contain the protein p_{n+1} or $\bar{p}_{n+1}$, then they have the objects $a_{n+1}, c_1, d_{1,1}$. The count object g_i in cell 4 increases its subscript by one, thus, it contains the protein s and the object $g_{mn+3n+1}$.

At configuration $\mathcal{C}_{mn+3n}$, the protein q_{n+1} on cell 1 is exchanged with the protein p_{n+1} (resp., $\bar{p}_{n+1}$) on cell 2 by applying the rule $r_{15,1}$ (resp., $r_{16,1}$). Thus, the cells with label 1 which have the objects $c_1, d_{1,0}$ contain the protein p_{n+1} or $\bar{p}_{n+1}$, the objects $d_{1,1}$, and possible objects from the set $\{c_1, \ldots, c_m\}$; the cells with label 2 contain the protein q_{n+1}, the object a_{n+1} and the possible object c_1. In cell 4, object $g_{mn+3n+2}$ will be generated by using the rule $r_{22,mn+3n+1}$.

At configuration $\mathcal{C}_{mn+3n+1}$, under the control of the protein p_{n+1} (resp., $\bar{p}_{n+1}$), the object $d_{1,1}$ from cell 1 is exchanged with the object $d_{2,0}$ from the environment by applying the rule $r_{17,1}$ (resp., $r_{18,1}$). Thus, the cells with label 1 which have the objects $c_1, d_{1,0}$ contain the protein p_{n+1} or $\bar{p}_{n+1}$, the objects $d_{2,0}$, and possible objects from the set $\{c_1, \ldots, c_m\}$; there is no rule can be used in cell 3 and all cells 2, so the proteins and the objects in these cells are not changed; rule $r_{22,mn+3n+2}$ is used in cell 4, thus, the object $g_{mn+3n+3}$ presents in this cell.

Hence, the results of Lemma hold for $k = 1$.

By induction hypothesis, suppose (1), (2), (3) and (4) hold for k ($1 \leq k < m$). Let us see that (1), (2), (3) and (4) also hold for $k + 1$.

At configuration $\mathcal{C}_{mn+3n+4k-2}$, if a cell $1_{n,j}$ has the objects c_{k+1} and $d_{k+1,0}$, then rule $r_{11,k+1}$ (resp., $r_{12,k+1}$) is used, the protein p_{n+1} (resp., $\bar{p}_{n+1}$) and the objects $c_{k+1}, d_{k+1,0}$ are exchanged with the protein q_{n+1}. Thus, the cells with label 1 which contain the objects $c_{k+1}, d_{k+1,0}$ have the protein q_{n+1} and the objects $\mathcal{C}_{mn+3n-2}(1_{n,j}) \setminus \eta_{k+1}$.

Hence, the result holds for configuration $\mathcal{C}_{mn+3n+4k-1}$.

At configuration $\mathcal{C}_{mn+3n+4k-1}$, if the protein on cells with label 2 is p_{n+1} (resp., $\bar{p}_{n+1}$), then rule $r_{13,k+1}$ (resp., $r_{14,k+1}$) is applied, the object $d_{k+1,0}$ from cell 2 is exchanged with the object $d_{k+1,1}$ from the environment. Thus, the cells with label 2 which have the protein p_{n+1} or $\bar{p}_{n+1}$ contain the objects $a_{n+1}, c_{k+1}, d_{k+1,1}$, and possible objects from the set $\{c_1, \ldots, c_k\}$; the proteins and objects in cells 1 and in cell 3 are not changed at this configuration; in cell 4, rule $r_{22,mn+3n+4k}$ is used, and the object $g_{mn+3n+4k+1}$ will appear in this cell.

Hence, the result holds for configuration $\mathcal{C}_{mn+3n+4k}$.

At configuration $\mathcal{C}_{mn+3n+4k}$, the protein q_{n+1} in cell 1 is exchanged with the protein p_{n+1} (resp., $\bar{p}_{n+1}$) in cell 2 by applying the rule $r_{15,k+1}$ (resp., $r_{16,k+1}$). Thus, the cells with label 1 which have the objects $c_{k+1}, d_{k+1,0}$ contain the protein p_{n+1} or $\bar{p}_{n+1}$, the objects $d_{k+1,1}$, and possible objects from the set $\{c_1, \ldots, c_m\}$; the cells with label 2 contain the protein q_{n+1}, the object a_{n+1} and the possible objects from the set $\{c_1, \ldots, c_{k+1}\}$; the object $g_{mn+3n+4k+1}$ from cell 4 is exchanged with the object $g_{mn+3n+4k+2}$ from the environment by using the rule $r_{22,mn+3n+4k+1}$.

Hence, the result holds for configuration $\mathcal{C}_{mn+3n+4k+1}$.

At configuration $\mathcal{C}_{mn+3n+4k+1}$, if a cell with label 1 has the object $d_{k+1,1}$, then rule $r_{17,k+1}$ (resp., $r_{18,k+1}$) is applied, the object $d_{k+2,0}$ will present in such cell. Thus, the cells with label 1 which have the objects $c_{k+1}, d_{k+1,0}$ contain the protein p_{n+1} or $\bar{p}_{n+1}$, the object $d_{k+2,0}$, and possible objects from the set $\{c_1, \ldots, c_m\}$; the cells with label 2 contain the protein q_{n+1}, the object a_{n+1} and the possible objects from the set $\{c_1, \ldots, c_{k+1}\}$; the object $g_{mn+3n+4k+3}$ will appear in cell 4 by using the rule $r_{22,mn+3n+4k+2}$.

Hence, the result holds for configuration $\mathcal{C}_{mn+3n+4k+2}$. □

Lemma 4.6. Let $\mathcal{C}$ be an arbitrary computation of the system, at configuration $\mathcal{C}_{mn+3n+4m-1}$, we have the following:

(a) there are 2^n copies of cell with label 1. Besides,

- if the formula φ is satisfiable, then there is one and only one cell with label 1, which contains the protein r and the object yes;
- if the formula φ is not satisfiable, then the contents of the cells with label 1 coincide with the contents in the previous configuration $\mathcal{C}_{mn+3n+4m-2}$;

(b) there are 2^n copies of cell with label 2, each of them contains the protein q_{n+1}, the object a_{n+1} and possible objects from the set $\{c_1, \ldots, c_m\}$;

(c) there is a cell with label 3. Besides,

- if the formula φ is satisfiable, then the cell 3 contains the protein p_{n+1} or $\bar{p}_{n+1}$ and the object $d_{m+1,0}$,no;
- if the formula φ is not satisfiable, then the contents of the cells with label 3 coincide with the contents in the previous configuration $\mathcal{C}_{mn+3n+4m-2}$;

(d) there is a cell with label 4 that contains the protein s and the object $g_{mn+3n+4m}$.

Proof:
At configuration $\mathcal{C}_{mn+3n+4m-1}$, there are 2^n copies of cell with label 1. If the formula φ is satisfiable, then there is at least one copy of cell with label 1 which contains the object $d_{m+1,0}$. Because there is only one copy of cell with label 3 which contains the object yes, so one and only one of the protein p_{n+1} (resp., $\bar{p}_{n+1}$) and the object $d_{m+1,0}$ in cell 1 are exchanged with the protein r and the object yes in cell 3 by using the rule r_{19} (resp., r_{20}). If the formula φ is not satisfiable, there is no rule can be used in cells 1, thus, the contents of the cells with label 1 coincide with the contents in the previous configuration $\mathcal{C}_{mn+3n+4m-2}$.

There are 2^n copies of cell with label 2, and no rule can be used in these cells, the contents of the cells with label 2 coincide with the contents in the previous configuration $\mathcal{C}_{mn+3n+4m-2}$, that is, each of them contains the protein q_{n+1}, the object a_{n+1} and possible objects from the set $\{c_1, \dots, c_m\}$.

There is a cell with label 3. If the formula φ is satisfiable, rule r_{19} (resp., r_{20}) can be used, and the cell 3 contains the protein p_{n+1} or $\bar{p}_{n+1}$ and the objects $d_{m+1,0}$,no; if the formula φ is not satisfiable, no rule can be applied in cell 3, thus the contents of the cell with label 3 coincide with the contents in the previous configuration $\mathcal{C}_{mn+3n+4m-2}$.

By applying the rule $r_{mn+3n+4m-1}$, object $g_{mn+3n+4m}$ will present in cell 4. □

Lemma 4.7. Let $\mathcal{C}$ be an arbitrary computation of the system, at configuration $\mathcal{C}_{mn+3n+4m}$, we have the following:

(a) if the formula φ is satisfiable, then the object yes appears in $\mathcal{C}_{mn+3n+4m}(0)$, and the configuration $\mathcal{C}_{mn+3n+4m}$ is a halting configuration;

(b) if the formula φ is not satisfiable, then the cell with label 3 contains the protein s and the object $g_{mn+3n+4m}$, the cell with label 4 contains the protein r.

Proof:
At configuration $\mathcal{C}_{mn+3n+4m-1}$.

If the formula φ is satisfiable, rule r_{21} is applied, object yes is sent to the environment, thus, the object yes appears in $\mathcal{C}_{mn+3n+4m}(0)$. It is easy to check that no rule of the system is applicable to configuration $\mathcal{C}_{mn+3n+4m}$.

If the formula φ is not satisfiable, only rule r_{23} can be used, the protein s and the object $g_{mn+3n+4m}$ in cell 4 are exchanged with the protein r in cell 3. Thus, the cell with label 3 contains the protein s and the object $g_{mn+3n+4m}$, the cell with label 4 contains the protein r. □

Lemma 4.8. Let $\mathcal{C}$ be an arbitrary computation of the system, if the formula φ is not satisfiable, then the object no appears in $\mathcal{C}_{mn+3n+4m+1}(0)$, and the configuration $\mathcal{C}_{mn+3n+4m+1}$ is a halting configuration.

Proof:
At configuration $\mathcal{C}_{mn+3n+4m}$, if the formula φ is not satisfiable, only rule r_{24} can be applied, the objects $g_{mn+3n+4m}$, no are sent to the environment. Thus, the object no appears in $\mathcal{C}_{mn+3n+4m+1}(0)$. It is easy to check that no rule of the system is applicable to configuration $\mathcal{C}_{mn+3n+4m+1}$. □

4.2.3. Polynomial Bound of the Family

From Lemma 4.7 and Lemma 4.8, we deduce that any computation $\mathcal{C}$ of tissue P systems with protein on cells and cell division, if the formula φ is satisfiable, the system takes $mn + 3n + 4m$ steps; if the formula φ is not satisfiable, the system takes $mn + 3n + 4m + 1$ steps.

Therefore, there exists a polynomial bound (with respect to m and n) on the number of steps of the computation.

4.3. Computational Efficiency of $\mathbf{TPDC(4)}$

The system constructed for the solution to the `SAT` problem in section 4 has communication rules with length at most 4. According to the Definition 3.4 and from the discussion in the previous subsections, we have the following result:

Theorem 4.9. $\mathtt{SAT} \in \mathbf{PMC_{TPDC(4)}}$.

Corollary 4.10. $\mathbf{NP} \cup \mathbf{co-NP} \subseteq \mathbf{PMC_{TPDC(4)}}$.

Proof:
It suffices to make the following observations: the `SAT` problem is **NP**-complete, $\mathtt{SAT} \in \mathbf{PMC_{TPDC(4)}}$, and this complexity class is closed under polynomial-time reduction and under complement. □

5. Universality of Tissue P Systems with Protein on Cells

In this section, we prove that tissue P systems with protein on cells are universal by simulating register machines.

Theorem 5.1. $NOP_2(commu_4) = NRE$.

Proof:
We only have to prove the inclusion $NRE \subseteq NOP_2(commu_4)$; the converse inclusion is straightforward from the Church-Turing thesis.

Let $M = (m, H, l_0, l_h, I)$ be a register machine that has the properties specified in Section 2. We construct the P system Π to simulate register machine M.

$$\Pi = (\Gamma, P, \mathcal{E}, \mathcal{M}_1/p_1, \mathcal{M}_2/p_2, \mathcal{R}, i_{out}),$$

where:

- $\Gamma = \{a_r \mid 1 \leq r \leq m\} \cup \{l, l', l'', l''', l^{iv}, l^v, \bar{l} \mid l \in H\}$;
- $P = \{p_1, p_2\}$;
- $\mathcal{E} = \{a_r \mid 1 \leq r \leq m\} \cup \{l, l', l'', l''', l^{iv}, l^v, \bar{l} \mid l \in H\}$;
- $\mathcal{M}_1 = \{l_0\}, \mathcal{M}_2 = \emptyset$;
- $i_{out} = 1$;

and the set R of rules constructed as follows:

- For each ADD instruction $l_i : (\texttt{ADD}(r), l_j, l_k)$, we introduce in R the rules

$$r_1 \equiv (1, (p_1, l_i)/l_j a_r, 0);$$
$$r_2 \equiv (1, (p_1, l_i)/l_k a_r, 0).$$

The simulation of the ADD instruction is obvious. By using the rule r_1 or r_2 non-deterministically, under the control of protein p_1 on cell 1, one copy of object a_r together with the object l_j or l_k are introduced into cell 1 from the environment, simultaneously, the object l_i in cell 1 is sent to the environment. Thus, the value of the register r has been increased by 1 and the system starts to simulate an instruction with label l_j or l_k.

- For each SUB instruction $l_i : (\texttt{SUB}(r), l_j, l_k)$, we introduce in R the rules

$$r_3 \equiv (1, (p_1, l_i)/l_i' l_i'', 0);$$
$$r_4 \equiv (1, (p_1, l_i')/(p_2, \lambda), 2);$$
$$r_5 \equiv (1, (p_2, l_i'' a_r)/l_i''', 0);$$
$$r_6 \equiv (2, (p_1, l_i')/l_i^{iv}, 0);$$
$$r_7 \equiv (1, (p_2, l_i'')/(p_1, l_i^{iv}), 2);$$
$$r_8 \equiv (1, (p_2, l_i''')/(p_1, l_i^{iv}), 2);$$
$$r_9 \equiv (1, (p_1, l_i^{iv})/l_i^{v}, 0);$$
$$r_{10} \equiv (2, (p_2, l_i''')/\bar{l}_j, 0);$$
$$r_{11} \equiv (2, (p_2, l_i'')/\bar{l}_k, 0);$$
$$r_{12} \equiv (1, (p_1, l_i^{v})/(p_2, \bar{l}_j), 2);$$
$$r_{13} \equiv (1, (p_1, l_i^{v})/(p_2, \bar{l}_k), 2);$$
$$r_{14} \equiv (1, (p_2, \lambda)/(p_1, l_i^{v}), 2);$$
$$r_{15} \equiv (1, (p_1, l_i^{v}\bar{l}_j)/l_j, 0);$$
$$r_{16} \equiv (1, (p_1, l_i^{v}\bar{l}_k)/l_k, 0).$$

A SUB instruction l_i is simulated in system Π in the following way (each SUB instruction is simulated in eight steps). Without loss of generality, we suppose that a SUB instruction $l_i : (\texttt{SUB}(r), l_j, l_k)$ starts to be simulated at step t. Hence at step t, there are protein p_1 on cell 1 and object $l_i z$ ($z \in \{a_1, \ldots, a_m\}^*$) in cell 1, protein p_2 on cell 2. At step $t+1$, rule r_3 is used, under the control of protein p_1 on cell 1, object l_i in cell 1 is exchanged with the objects $l_i' l_i''$ from the environment. In the next step, the protein p_1 and the object l_i' in cell 1 are exchanged with the protein p_2 on cell 2 by using the rule r_4. With the appearance of protein p_2 on cell 1, we have the following two cases.

- There is at least one copy of object a_r in cell 1. In this case, at step $t+3$, the rules r_5 and r_6 can be used. By applying the rule r_5, under the control of protein p_2 on cell 1, the objects $l_i'' a_r$ in cell 1 are exchanged with the object l_i''' from the environment; by applying the rule r_6, under the control of protein p_1 on cell 2, the object l_i' in cell 2 is exchanged with the object l_i^{iv} from the environment. In the next step, rule r_8 is used, the protein p_2 and object l_i''' in cell 1 are exchanged with the protein p_1 and object l_i^{iv} in cell 2. At step $t+5$, by using the rule r_9, the object l_i^{iv} in cell 1 is exchanged with the object l_i^{v} from the environment. In cell 2, under the control of protein p_2, the object l_i''' is exchanged with the object $\bar{l}_j$ from the environment by using the rule r_{10}. At step $t+6$, by using the rule r_{12}, the protein p_1 and object l_i^v in cell 1 are exchanged with the protein p_2 and object $\bar{l}_j$ in cell 2. In the next step, the protein p_2 on cell 1 is exchanged with the protein p_1 and object l_i^v in cell 2 by using the rule r_{14}. At step $t+8$, under the control of protein p_1 on cell 1, the objects $l_i^v\bar{l}_j$ in cell 1 are exchanged with the object l_j from the environment by using the rule r_{15}. In this case, one copy of object a_r is consumed in cell 1, and the system starts to simulate the instruction l_j (see Table 1).

Table 1. The simulation of a SUB instruction $l_i : (\mathtt{SUB}(r), l_j, l_k)$, where there is at least one copy of object a_r in cell 1. Let $z \in \{a_1, \ldots, a_m\}^*$ and $z = a_r z'$.

Step	Rules	Cell 1		Cell 2	
		Protein	Objects	Protein	Objects
0	–	p_1	$l_i z$	p_2	–
1	r_3	p_1	$l_i' l_i'' z$	p_2	–
2	r_4	p_2	$l_i'' z$	p_1	l_i'
3	r_5, r_6	p_2	$l_i''' z'$	p_1	l_i^{iv}
4	r_8	p_1	$l_i^{iv} z'$	p_2	l_i'''
5	r_9, r_{10}	p_1	$l_i^v z'$	p_2	$\bar{l}_j$
6	r_{12}	p_2	$\bar{l}_j z'$	p_1	l_i^v
7	r_{14}	p_1	$l_i^v \bar{l}_j z'$	p_2	–
8	r_{15}	p_1	$l_j z'$	p_2	–

- There is no copy of object a_r in cell 1. In this case, at step $t+3$, only rule r_6 is used, object l_i^{iv} is sent to cell 2. In the next step, by using the rule r_7, the protein p_1 and object l^{iv} present in cell 1, and in cell 2, there are the protein p_2 and object l_i''. At step $t+5$, rules r_9 and r_{11} are used, object l_i^v is sent into cell 1 and object $\bar{l}_k$ is sent into cell 2. In the following three steps, the rules r_{13}, r_{14}, r_{16} are applied one by one, the object l_k will present in cell 1 at step $t+8$. Hence, the system starts to simulate the instruction l_k (see Table 2).

Therefore, the SUB instruction is correctly simulated by Π.

When the object l_h appears in cell 1, the computation stops. The number of copies of a_1 in cell 1 clearly corresponds to the value of register 1 of M, hence $N(M) = N(\Pi)$, this concludes the proof. □

6. Conclusions and Remarks

In this work, inspired by the facts that the movement of most objects through communication channels is controlled by proteins and some of proteins on cells are not static, they can move through lipid bilayers

Table 2. The simulation of a SUB instruction $l_i : (\texttt{SUB}(r), l_j, l_k)$, where there is no copy of object a_r in cell 1. Let $z \in \{a_1, \ldots, a_m\}^*$ and $a_r \notin z$.

Step	Rules	Cell 1		Cell 2	
		Protein	Objects	Protein	Objects
0	–	p_1	$l_i z$	p_2	–
1	r_3	p_1	$l_i' l_i'' z$	p_2	–
2	r_4	p_2	$l_i'' z$	p_1	l_i'
3	r_6	p_2	$l_i'' z$	p_1	l_i^{iv}
4	r_7	p_1	$l_i^{iv} z$	p_2	l_i''
5	r_9, r_{11}	p_1	$l_i^v z$	p_2	$\bar{l}_k$
6	r_{13}	p_2	$\bar{l}_k z$	p_1	l_i^v
7	r_{14}	p_1	$l_i^v \bar{l}_k z$	p_2	–
8	r_{16}	p_1	$l_k z$	p_2	–

between cells if they are fused, we present a class of tissue P systems with protein on cells. The computational power of such P systems has been studied. Specifically, we have given an efficient (uniform) solution to the SAT problem by using such P systems with cell division. We also proved that any Turing computable set of numbers can be generated by a tissue P system with protein on cells.

The solution to the SAT problem given in section 4 has the communication rules of length at most 4. It is deserved to investigate whether tissue P systems with protein on cells and cell division with communication rules of length 2 or 3 are efficient (note that the minimal length of a communication rule is 2).

In section 5, the universality result is obtained by a tissue P system with protein on cells with two cells and communication rules of length at most 4. It remains open whether tissue P systems with protein on cells are universal by using only one cell or communication rules of length 2 and 3.

The tissue P systems considered in this work have division rules, where cell division is inspired by both protein and object, and the newly generated cells can have different proteins and objects with their parent cell. It remains open what happens if we consider division rules that are inspired only by proteins, and the newly generated cells can have different proteins with their parent cell or division rules that are inspired by both proteins and objects, but the newly generated cells have the same protein as parent cell.

Tissue P systems with cell division and without environment were introduced in [27], that is, the alphabet of the environment of such P systems is empty. It would be interesting to consider the computational efficiency of tissue P systems with protein on cells and cell division without environment.

Recently, various P systems have been used to solve **NP**-complete problems in a time-free manner in the sense that the correctness of the solution does not depend on the precise timing of the involved rules [28, 29, 30, 31, 32]. It remains open whether we can construct tissue P systems with protein on cells and cell division to solve **NP**-complete problems in the context of time-freeness.

P systems with minimal parallelism were investigated in [33], where each membrane which can evolve in a given step should do it by using at least one rule. Recently, a new strategy of using rules, called flat maximal parallelism was considered in [34], where in each step, in each membrane, a maximal set of applicable rules is chosen and each rule in the set is applied exactly once. It is of interest to investigate the computational power of tissue P systems with protein on cells by using rules in a minimally parallel way or in a flat maximally parallel way.

Tissue P systems with cell separation are a variant of tissue P system [35]. In such P systems, cells do not have the duplication function, that is, when a cell is separated, the objects in the cell are divided and placed in the newly generated cells instead of replicating the objects and distributing them in each of the newly generated cells. It is interesting to investigate the computational power of tissue P systems with protein on cells and cell separation.

Acknowledgements

The work of B. Song and L. Pan was supported by National Natural Science Foundation of China (61033003, 61320106005 and 61472154), Ph.D. Programs Foundation of Ministry of Education of China (2012014213008), Natural Science Foundation of Hubei Province (2011CDA027), and the Innovation Scientists and Technicians Troop Construction Projects of Henan Province (154200510012). The work of M.J. Pérez-Jiménez was supported by "Ministerio de Economía y Competitividad" of Spanish government (TIN2012-37434), cofunded by FEDER funds.

References

[1] Păun G. Computing with membranes. Journal of Computer and System Sciences. 2000;61(1):108–143. doi:10.1006/jcss.1999.1693.

[2] Martín-Vide C, Pazos J, Păun G, Rodriguez-Paton A. Tissue P systems. Theoretical Computer Science. 2003;296(2):295–326. doi:10.1016/S0304-3975(02)00659-X.

[3] Ionescu M, Păun G, Yokomori T. Spiking neural P systems. Fundamenta Informaticae. 2006;71(2):279–308.

[4] Jiang K, Pan L. Spiking neural P systems with anti-spikes working in sequential mode induced by maximum spike number. Neurocomputing. 2016;171:1674–1683.

[5] Song T, Pan L. Spiking neural P systems with rules on synapses working in maximum spikes consumption strategy. IEEE Transactions on NanoBioscience. 2015;14(1):38–44. doi: 10.1109/TNB.2014.2367506.

[6] Song T, Pan L. Spiking neural P systems with rules on synapses working in maximum spiking strategy. IEEE Transactions on NanoBioscience. 2015;14(4):465–477. doi: 10.1109/TNB.2015.2402311.

[7] Song T, Pan L, Păun G. Spiking neural P systems with rules on synapses. Theoretical Computer Science. 2014;529:82–95. doi:10.1016/j.tcs.2014.01.001.

[8] Zeng X, Zhang X, Song T, Pan L. Spiking neural P systems with thresholds. Neural computation. 2014; 26(7):1340–1361. doi:10.1162/NECO_a_00605.

[9] Zhang X, Pan L, Păun A. On the universality of axon P systems. IEEE Transactions on Neural Networks and Learning Systems. 2015;26(11):2816–2829. doi: 10.1109/TNNLS.2015.2396940.

[10] Păun G, Rozenberg G, Salomaa A. The Oxford Handbook of Membrane Computing. Oxford University Press; 2010. ISBN: 0199556679, 9780199556670.

[11] Păun A, Păun G. The power of communication: P systems with symport/antiport. New Generation Computing. 2002;20(3):295–305. doi:10.1007/BF03037362.

[12] Alhazov A, Freund R, Leporati A, Oswald M, Zandron C. (Tissue) P systems with unit rules and energy assigned to membranes. Fundamenta Informaticae. 2006;74(4):391–408.

[13] Alhazov A, Freund R, Oswald M. Tissue P systems with antiport rules and small numbers of symbols and cells. In: Developments in Language Theory. vol. 3572. Springer, LNCS; 2005. p. 100–111. doi:10.1007/11505877_9.

[14] Freund R, Păun G, Pérez-Jiménez MJ. Tissue P systems with channel states. Theoretical Computer Science. 2005;330(1):101–116. doi:10.1016/j.tcs.2004.09.013.

[15] Păun G, Pérez-Jiménez MJ, Riscos-Núñez A. Tissue P systems with cell division. International Journal of Computers, Communications and Control. 2008;3(3):295–303.

[16] Díaz-Pernil D, Pérez-Jiménez M, Riscos-Núñez A, Romero-Jiménez A. Computational efficiency of cellular division in tissue-like membrane systems. Romanian Journal of Information Science and Technology. 2008;11(3):229–241. Available from: http://www.imt.ro/romjist/Volum11/Number11_3/02-Diaz-Pernil.htm.

[17] Díaz-Pernil D, Gutiérrez-Naranjo MA, Pérez-Jiménez MJ, Riscos-Núñez A. Solving subset sum in linear time by using tissue P systems with cell division. In: Proceedings of 2th International Work-Conference on the Interplay Between Natural and Artificial Computation. Springer; 2007. p. 170–179. doi:10.1007/978-3-540-73053-8_17.

[18] Díaz-Pernil D, Gutiérrez-Naranjo MA, Pérez-Jiménez MJ, Riscos-Núñez A. A uniform family of tissue P systems with cell division solving 3-COL in a linear time. Theoretical Computer Science. 2008;404(1):76–87. doi:10.1016/j.tcs.2008.04.005.

[19] Díaz-Pernil D, Gutiérrez-Naranjo MA, Pérez-Jiménez MJ, Riscos-Núñez A. Solving the independent set problem by using tissue-like P systems with cell division. In: Proceedings of 3th International Work-Conference on the Interplay Between Natural and Artificial Computation. vol. 5601. Springer; 2009. p. 213–222. doi:10.1007/978-3-642-02264-7_23.

[20] Almén MS, Nordström KJ, Fredriksson R, Schiöth HB. Mapping the human membrane proteome: a majority of the human membrane proteins can be classified according to function and evolutionary origin. BMC biology. 2009;7(50). doi:10.1186/1741-7007-7-50.

[21] Alberts B, Johnson A, Lewis J, Raff M, Roberts K, Walter P. Molecular Biology of the Cell. Garland Science; 2002. ISBN- 10:0-8153-3218-1, 10:0-8153-4072-9.

[22] Rozenberg G, Salomaa A. Handbook of Formal Languages. vol. 1–3. Springer Science & Business Media; 1997.

[23] Minsky ML. Computation: Finite and Infinite Machines. Prentice-Hall, Inc., Upper Saddle River, NJ, USA. 1967. ISBN-13:978-0131655638, 10:0131655639.

[24] Păun A, Popa B. P systems with proteins on membranes. Fundamenta Informaticae. 2006;72(4):467–483.

[25] Pérez-Jiménez MJ, Romero-Jiménez A, Sancho-Caparrini F. A polynomial complexity class in P systems using membrane division. Journal of Automata, Languages and Combinatorics. 2006;11(4):423–434.

[26] Garey MR, Johnson DS. Computers and Intractability: A Guide to the Theory of NP-completeness. San Francisco, LA: Freeman; 1979. ISBN:0716710447.

[27] Christinal HA, Díaz-Pernil D, Gutiérrez-Naranjo MA, Pérez-Jiménez MJ. Tissue-like P systems without environment. In: Proceedings of the Eight Brainstorming Week on Membrane Computing, Sevilla, Spain. Citeseer; 2010. p. 53–64. Available from: http://www.gcn.us.es/8BWMC/volume/04DiazPernilEnvironment.pdf.

[28] Song T, Macías-Ramos LF, Pan L, Pérez-Jiménez MJ. Time-free solution to SAT problem using P systems with active membranes. Theoretical Computer Science. 2014;529:61–68. doi:10.1016/j.tcs.2013.11.014.

[29] Song B, Pan L. Computational efficiency and universality of timed P systems with active membranes. Theoretical Computer Science. 2015;567:74–86. doi: 10.1016/j.tcs.2014.10.051.

[30] Song B, Pérez-Jiménez MJ, Pan L. Computational efficiency and universality of timed P systems with membrane creation. Soft Computing. 2015;19(11):3043–3053. doi:10.1007/s00500-015-1732-3.

[31] Song B, Song T, Pan L. Time-free solution to SAT problem by P systems with active membranes and standard cell division rules. Natural Computing. 2015;14(4):673–681. doi: 10.1007/s11047-014-9471-4.

[32] Song B, Song T, Pan L. A time-free uniform solution to subset sum problem by tissue P systems with cell division. Mathematical Structures in Computer Science; doi:10.1016/j.tcs.2015.10.027.

[33] Ciobanu G, Pan L, Păun G, Pérez-Jiménez MJ. P systems with minimal parallelism. Theoretical Computer Science. 2007;378(1):117–130. doi:10.1016/j.tcs.2007.03.044.

[34] Pan L, Păun G, Song B. Flat maximal parallelism in P systems with promoters. Theoretical Computer Science. 2015; doi:10.1016/j.tcs.2015.10.027.

[35] Pan L, Pérez-Jiménez MJ. Computational complexity of tissue-like P systems. Journal of Complexity. 2010;26(3):296–315. doi:10.1016/j.jco.2010.03.001.

[36] Gutiérrez-Naranjo MA, Pérez-Jiménez MJ, Riscos-Núñez A, Romero-Campero FJ. Computational efficiency of dissolution rules in membrane systems. International Journal of Computer Mathematics. 2006;83(7):593–611. doi:10.1080/00207160601065413.