ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **20**, Number 1, 2017, 42–56

Flat maximal parallelism in tissue P systems with promoters

Linqiang Pan[1], Yanfeng Wang[1], Suxia Jiang[1], Bosheng Song[2,*]

[1] School of Electric and Information Engineering,
Zhengzhou University of Light Industry,
Zhengzhou 450002, Henan, China

[2] Key Laboratory of Image Information Processing and Intelligent Control of Education Ministry of China
School of Automation, Huazhong University of Science and Technology,
Wuhan 430074, Hubei, China
Email: boshengsong@hust.edu.cn

Abstract. Tissue P systems with promoters are distributed parallel computing models inspired by the structure of tissues and the way of communicating substances regulated by promoters. In this work, we consider tissue P systems with promoters working in a flat maximally parallel way, 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 proved that tissue P systems with at most two promoters associated with any rule and using only symport rules of length 1 are Turing universal. Moreover, a uniform solution to the SAT problem is provided by tissue P systems with at most two promoters associated with any rule and using cell division rules and symport rules of length 1.

Key-words: Tissue P system, Promoter, Flat maximal parallelism, SAT problem, Universality

1. Introduction

Membrane computing, initiated by Gh. Păun, is a branch of natural computing, which is focused on abstracting computing concepts from the structure and the functioning of living cells, and the ways in which cells are organized in tissues and higher order biological structures [19]. All computing devices considered in membrane computing are usually called *P systems*. Since the seminal definition of P systems, membrane computing has developed rapidly on both theoretical results [12,30,34,35] and application of solving real problems [23,24,36,38]. According to the membrane structure of P systems, there exist two main families: *cell-like P systems*, which have a hierarchical arrangement of membranes [19]; and *tissue-like P systems* [13] or *neural-like P systems* [10], which have a net of membranes. For general information about membrane computing, one may consult [20,22].

Tissue P systems are inspired by the structure of tissues and the way of communicating substances between two cells or between a cell and the environment. In a tissue P system, cells are placed in the nodes of a graph and the environment is considered as a distinguished node, an arc between two nodes corresponds to a communication channel between two regions (two cells or a cell and the environment). If a communication channel between two regions exists, then they can communicate by means of communication (symport/antiport) rules [17]. Symport rules move objects across a membrane together in one direction, whereas antiport rules move objects across a membrane in opposite directions.

The computational power of tissue P systems has been investigated widely. Many variants of tissue P systems are proved to be computationally complete [1,2,7,8,11, 18]. Tissue P systems are also proved to be computationally efficient. In particular, by introducing cell division rules or cell separation rules into tissue P systems, an exponential workspace can be generated in polynomial time, which is successfully used for designing solutions to **NP**-complete problems [4–6,16,21,32,37].

An interesting variant of tissue P systems, called *tissue P systems with promoters*, was proposed in [3,26,29]. In [3], a set of techniques for the problem of counting cells inspired from the treatment of digital images via tissue P systems with promoters was presented; in [26], tissue P systems with promoters were used to implement a cell complex based algorithm for thinning images. In [29], the computational power of tissue P systems with promoters has been investigated, where rules of such systems are used 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).

Flat maximal parallelism was investigated for cell-like P systems, 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 [15]. In this work, the flat maximal parallelism of using rules is investigated for tissue P systems with promoters. Specifically, it is proved that tissue P systems with at most two promoters associated with any rule and using only symport rules of length 1 are Turing universal. Moreover, a uniform solution to the `SAT` problem is provided by tissue P systems with at most two promoters associated with any rule and using cell division rules and symport rules of length 1.

It is known that tissue P systems with promoters using only symport rules of length 1, working in the maximally parallel way, are able to compute only finite sets of non-negative integers [29]. The comparison with the result about universality obtained in this work shows that the flat maximally parallel way of applying rules can increase the computational power of tissue P systems with promoters.

2. Prerequisites and tissue P systems

In this section, we first recall some basic notions from formal language theory [27], then the notions of (recognizer) tissue P systems with promoters and cell division introduced in [29].

An *alphabet* Γ is a finite non-empty set of *symbols*. Any sequence of symbols from an alphabet Γ is called *string* over Γ. The *length* of the string u, denoted by $|u|$ is the total number of occurrences of symbols in u. The empty string (with length 0) is denoted by λ. The set of all strings over an alphabet Γ is denoted by Γ^*, and by $\Gamma^+ = \Gamma^* \backslash \{\lambda\}$ we denote the set of all non-empty strings.

A *multiset* m over an alphabet Γ is a pair (Γ, f), where f is a mapping from Γ onto the set of natural numbers $\mathbb{N}$. If alphabet $\Gamma = \{a_1, \ldots, a_k\}$, then we denote $m = \{a_1^{f(a_1)}, \ldots, a_k^{f(a_k)}\}$. We denote by $\emptyset$ the empty multiset and by $M_f(\Sigma)$ the set of all finite multisets over Σ. We denote by NRE the family of sets of numbers which are Turing computable.

Definition 1 *A tissue P system with promoters and cell division, of degree $q \geq 1$, is a tuple*

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

where

- Γ *is a finite non-empty set of objects;*
- $\mathcal{E} \subseteq \Gamma$ *is a set of objects initially located in the environment;*
- w_i, $1 \leq i \leq q$, *are finite multisets over* Γ*;*
- $\mathcal{R}$ *is a finite set of rules of the following forms:*
  - *Communication rules:*
    - *Symport rules:* $(pro \mid i, u/\lambda, j)$ *or* $(pro \mid i, \lambda/u, j)$, *where* $0 \leq i \neq j \leq q$, $pro, u \in M_f(\Gamma)$, $|u| > 0$*;*
    - *Antiport rules:* $(pro \mid i, u/v, j)$, *where* $0 \leq i \neq j \leq q$, $pro, u, v \in M_f(\Gamma)$, $|u| > 0$, $|v| > 0$*;*
  - *Division rules:*
    - $[\ a\ ]_i \rightarrow [\ b\ ]_i [\ c\ ]_i$, *where* $i \in \{1, \ldots, q\}$, $i \neq i_{out}$, $a, b, c \in \Gamma$*;*
- $i_{out} \in \{0, 1, \ldots, q\}$.

A tissue P system with promoters and cell division of degree $q \geq 1$ can be viewed as a set of q cells labelled by $1, \ldots, q$, arranged in the nodes of a directed graph, such that: (a) $w_1, \ldots, w_q$ represent the finite multisets of objects initially placed in the q cells of the system; (b) $\mathcal{E}$ is the set of objects initially located in the environment of the system, all of them are available in an arbitrary number of copies; (c) $\mathcal{R}$ is a finite set of rules; (d) i_{out} is a distinguished region which will encode the output of the system. We use the term region i $(0 \leq i \leq q)$ to refer to cell i in case $1 \leq i \leq q$ and to refer to the environment in case $i = 0$. The length of a symport rule $(pro \mid i, u/\lambda, j)$ or $(pro \mid i, \lambda/u, j)$ (resp., an antiport rule $(pro \mid i, u/v, j)$) is defined as $|u|$ (resp., $|u| + |v|$) (the number of occurrences of symbols it contains).

For an alphabet Γ, a *multiset* over Γ is a pair (Γ, f) where $f : \Gamma \to \mathbb{N}$ is a mapping; $\mathbb{N}$ is the set of natural numbers.

A *configuration* of a tissue P system with promoters and cell division at any instant is described by all multisets of objects over Γ associated with all cells in the system, and the multiset of objects over $\Gamma \setminus \mathcal{E}$ associated with the environment at that moment. Note that the objects from $\mathcal{E}$ have an arbitrary large number of copies, hence they are not properly changed along the computation. The *initial configuration* is $(w_1, \ldots, w_q; \emptyset)$.

A symport rule $(pro \mid i, u/\lambda, j) \in \mathcal{R}$ (resp., $(pro \mid i, \lambda/u, j) \in \mathcal{R}$) is applicable to a configuration if region i (resp., j) contains multiset u, and the objects of the promoter *pro* are present in region i (resp., j). When such a rule is applied, multiset u is sent from region i to region j (resp., from region j to region i).

An antiport rule $(pro \mid i, u/v, j) \in \mathcal{R}$ is applicable to a configuration if the following conditions hold: (1) region i contains multiset u; (2) region j contains multiset v; (3) the objects of the promoter *pro* are present in cell i. When such a rule is applied, the objects of the multiset u are sent from region i to region j, and simultaneously, the objects of multiset v are sent from region j to region i.

A division rule $[\ a\ ]_i \to [\ b\ ]_i [\ c\ ]_i$ is applicable to a configuration if cell i contains object a, and cell i is not the output cell. When such a rule is applied, cell i is divided into two cells with the same label: in one cell, object a is replaced by object b, in the other cell, object a is replaced by object c, and all the objects in the original cell, different from the object triggering the rule, are replicated in the two new cells.

We remark that in a tissue P system with promoters and cell division, the presence of the promoter objects makes it possible to use the associated rule as many times as possible, without any restriction, that is, a promoter is valid for any number of rules (if these rules are associated with this promoter) in one step. Moreover, the promoter objects do not directly participate in the rules, hence the promoters only guide the computation process. We also note that the promoters are not modified by the application of the rule. If the promoters are empty, then we simply write a rule $(i, u/\lambda, j)$ (resp., $(i, u/v, j)$) instead of $(\emptyset \mid i, u/\lambda, j)$ (resp., $(\emptyset \mid i, u/v, j)$).

The rules of a tissue P system with promoters and cell division considered in this work are applied in a flat maximally parallel way: in each step, in each membrane, a maximal set of concurrently applicable rules is chosen and each rule in the set is applied exactly once. This way of applying rules has only one restriction: when a cell is divided, the division rule is the only one which is applied for that cell at that step, the objects inside that cell do not evolve by means of communication rules. If the new cells resulting from division do not divide once again, then these cells could participate in the interaction with other cells or with the environment by means of communication rules at the next step.

Starting from the initial configuration and applying the rules as described above, one obtains a sequence of consecutive configurations. Each passage from a configuration to a next configuration is called a *transition*. A configuration is a *halting* one if no rule of the system is applicable to it. A sequence of transitions starting in the initial configuration is a *computation*. Only a computation reaching a halting configuration gives a result, encoded by the multiset of objects present in the output region i_{out}.

In order to investigate the computational efficiency, the notions of recognizer tissue P systems with promoters and cell division and a family of such P systems, which is used to solve a decision problem are adapted, for detailed information one can refer to [25, 29].

3. Computational power of tissue P systems with promoters working in the flat maximally parallel way

In this section, we investigate the computational power of tissue P systems with promoters working in the flat maximally parallel way. The notions of (recognizer) tissue P systems with promoters and cell division are given in [29]. Note that tissue P systems with promoters constructed in this work are working in the flat maximally parallel way, that is, in each step, in each cell, a maximal set of concurrently applicable rules is chosen and each rule in the set is applied exactly once. This way of applying rules has only one restriction: when a cell is divided, the division rule is the only one which is applied for that cell at that step, the objects inside that cell do not evolve by means of communication rules. If the new cells resulting from division do not divide at the next step, then these cells could participate in the interaction with other cells or with the environment by means of communication rules.

We remark that symport rules of the form $(pro \mid 0, a/\lambda, i)$ or $(pro \mid i, \lambda/a, 0)$ ($a \in \mathcal{E}$, i is a cell) are not allowed when a tissue P system with promoters working in the maximally parallel way, because its application (in the maximally parallel way) would lead to an infinite iteration of introducing object a into a cell.

The set of natural numbers computed by system Π is denoted by $N(\Pi)$. By $NOtP_m(pro_k, sym_{t_1}, anti_{t_2}, flat)$ we denote the family of all sets of numbers computed by systems with at most m cells, using symport/antiport rules (symport rules of length at most t_1, antiport rules of length at most t_2) with at most k promoters associated with each rule working in the flat maximally parallel way. If one of the parameters m, k, t_1, t_2 is not bounded, then it is replaced with $*$.

We denote by $\mathbf{PMC}^{f}_{\mathbf{TP_{k_1}DC}(k_2)}$ (resp., $\mathbf{PMC}^{f}_{\mathbf{TP_{k_1}DA}(k_2)}$ or $\mathbf{PMC}^{f}_{\mathbf{TP_{k_1}DS}(k_2)}$) the set of all decision problems which can be solved in a uniform way and polynomial time by means of recognizer tissue P systems with promoters and cell division, and using communication rules (resp., only antiport rules or only symport rules) of length at most k_2 with at most k_1 promoters associated with each rule working in the flat maximally parallel way.

Theorem 1 $NOtP_4(pro_2, sym_1, flat) = NRE$.

Proof. We only need to prove the inclusion $NOtP_4(pro_2, sym_1, flat) \supseteq NRE$, for the inclusion $NOtP_4(pro_2, sym_1, flat) \subseteq NRE$, we can invoke Turing-Church thesis.

Since we will use the characterization of NRE by register machines, we here introduce the notion of register machines. 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 (l_0 is the label of the initial instruction and l_h is the label of the halting instruction), and I is a set of instructions labeled in a one-to-one way with the elements of H, of the following forms: (a) $l_i : (\mathtt{ADD}(r), l_j, l_k)$ (add 1 to register r and then go to one of the instructions with labels l_j, l_k, non-deterministically chosen); (b) $l_i : (\mathtt{SUB}(r), l_j, l_k)$ (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); (c) $l_h : \mathtt{HALT}$ (the halt instruction). A configuration of a register machine is described by the contents (i.e., by the number stored in the register) of each register and by the current label, which indicates the next instruction to be executed. Computations start by executing the instruction l_0 of I, and terminate with reaching the HALT instruction l_h. It is known that register machines generate all sets of numbers which are Turing computable, hence they characterize NRE [14].

Let $M = (m, H, l_0, l_h, I)$ be a register machine. We construct the tissue P system Π, of order 4, to simulate register machine M.

$$\Pi = (\Gamma, \mathcal{E}, w_1, w_2, w_3, w_4, \mathcal{R}, 1),$$

where

- $\Gamma = \{a_i \mid 1 \leq i \leq m\} \cup \{l, l', l'', l''', l^{iv} \mid l \in H\} \cup \{b\}$,
- $\mathcal{E} = \{a_i \mid 1 \leq i \leq m\} \cup \{l, l', l'', l''', l^{iv} \mid l \in H\}$,
- $w_1 = \{b, l_0\}$, $w_2 = w_3 = w_4 = \emptyset$,

and the set $\mathcal{R}$ of rules is as follows:

- For each ADD instruction $l_i : (\mathtt{ADD}(r), l_j, l_k)$ of M, we introduce the following rules in $\mathcal{R}$:

$$\begin{aligned}
r_1 &\equiv (l_i \mid 1, \lambda/l'_i, 0),\\
r_2 &\equiv (1, l_i/\lambda, 0),\\
r_3 &\equiv (l'_i \mid 1, \lambda/a_r, 0),\\
r_4 &\equiv (1, l'_i/\lambda, 3),\\
r_5 &\equiv (1, l'_i/\lambda, 4),\\
r_6 &\equiv (l'_i \mid 3, \lambda/l_j, 0),\\
r_7 &\equiv (3, l'_i/\lambda, 0),\\
r_8 &\equiv (l'_i \mid 4, \lambda/l_k, 0),\\
r_9 &\equiv (4, l'_i/\lambda, 0),\\
r_{10} &\equiv (1, \lambda/l_j, 3),\\
r_{11} &\equiv (1, \lambda/l_k, 4).
\end{aligned}$$

An ADD instruction $l_i : (\texttt{ADD}(r), l_j, l_k)$ is simulated in four steps. At step 1, rules r_1, r_2 are enabled. By using rule r_1, one copy of object l_i' is sent into cell 1 (by the flat maximal parallelism, this rule is used only once). Note that we assume that $l_j \neq l_i$ and $l_k \neq l_i$. By applying rule r_2, object l_i is sent to the environment. At the next step, with the presence of object l_i' in cell 1, rule r_3 is used, one copy of object a_r is sent into cell 1 due to the flat maximally parallel use of rules (the number of copies of object a_r is increased by one, which simulates the fact that the number stored in register r is increased by one). Simultaneously, one of rules r_4 and r_5 is non-deterministically chosen and used. By using rule r_4 (resp., r_5), object l_i' is sent into cell 3 (resp., cell 4). At step 3, one of sets of rules $\{r_6, r_7\}$ and $\{r_8, r_9\}$ is non-deterministically chosen and used. By using rules r_6, r_7 (resp., r_8, r_9), one copy of object l_j is sent into cell 3 (resp., cell 4) due to the flat maximally parallel use of rules and object l_i' is sent to the environment from cell 3 (resp., cell 4). At step 4, one of rules r_{10} and r_{11} is non-deterministically chosen and used. By using rule r_{10} (resp., r_{11}), object l_j (resp., l_k) is sent into cell 1 from cell 3 (resp., cell 4). Hence, the system starts to simulate an instruction with label l_j or l_k. So the instruction l_i of M is correctly simulated by Π.

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

$$\begin{aligned}
r_{12} &\equiv (bl_i \mid 1, \lambda/l_i', 0),\\
r_{13} &\equiv (b \mid 1, l_i/\lambda, 0),\\
r_{14} &\equiv (l_i' \mid 1, a_r/\lambda, 2),\\
r_{15} &\equiv (l_i' \mid 1, \lambda/l_i'', 0),\\
r_{16} &\equiv (1, l_i'/\lambda, 0),\\
r_{17} &\equiv (l_i'' \mid 1, \lambda/l_k, 0),\\
r_{18} &\equiv (l_i'' \mid 1, b/\lambda, 0),\\
r_{19} &\equiv (1, l_i''/\lambda, 0),\\
r_{20} &\equiv (a_r \mid 2, \lambda/l_i''', 0),\\
r_{21} &\equiv (a_r \mid 2, \lambda/l_j, 0),\\
r_{22} &\equiv (2, a_r/\lambda, 0),\\
r_{23} &\equiv (l_i''' \mid 2, l_j/\lambda, 1),\\
r_{24} &\equiv (l_i''' \mid 2, \lambda/l_k, 1),\\
r_{25} &\equiv (1, \lambda/b, 0),\\
r_{26} &\equiv (l_i''' \mid 2, \lambda/l_i^{iv}, 0),\\
r_{27} &\equiv (2, l_i'''/\lambda, 0),\\
r_{28} &\equiv (l_i^{iv} \mid 2, l_k/\lambda, 0),\\
r_{29} &\equiv (2, l_i^{iv}/\lambda, 0).
\end{aligned}$$

A SUB instruction $l_i : (\mathtt{SUB}(r), l_j, l_k)$ is simulated in the following way. At step 1, by applying rule r_{12}, one copy of object l'_i is sent into cell 1 due to the flat maximal parallelism, simultaneously object l_i is sent to the environment by using rule r_{13}. In what follows, there are two cases.

- There is at least one copy of object a_r in cell 1 (corresponding to the fact that the number stored in register r is greater than 0). In this case, at step 2, rules r_{14}, r_{15} and r_{16} are enabled (we assume that $l_k \neq l_i$). By applying rule r_{14}, object a_r is sent into cell 2. By applying rule r_{15}, one copy of object l''_i is sent into cell 1. Object l'_i is sent to the environment by using rule r_{16}. At step 3, with the presence of object l''_i in cell 1, object b is sent to the environment and one copy of object l_k is sent into cell 1 by using rules r_{18} and r_{17}, respectively. By using rule r_{19}, object l''_i in cell 1 is sent to the environment. With the presence of object a_r in cell 2, one copy of object l'''_i and one copy of object l_j are sent into cell 2 by applying rules r_{20} and r_{21}, respectively. Simultaneously, object a_r in cell 2 is sent to the environment (the number of copies of object a_r is decreased by one, which simulates the fact that the number stored in register r is decreased by one). At step 4, by using rules r_{23}, r_{24}, object l_j in cell 2 is sent to cell 1 and object l_k in cell 1 is sent to cell 2, respectively. One copy of object l_i^{iv} from the environment is sent into cell 2 by using rule r_{26}. Simultaneously, object b is sent into cell 1 by using rule r_{25}. At step 5, with the presence of object l_i^{iv} in cell 2, rule r_{28} is enabled and used, object l_k in cell 2 is sent to the environment, and by applying rule r_{29}, object l_i^{iv} is also sent to the environment at this step. Note that at the last step of the simulation of a SUB instruction l_i, the next instruction l_j is simulated in cell 1, and there is no influence that the objects l_k, l_i^{iv} in cell 2 are sent to the environment. In this case, one copy of object a_r is consumed in cell 1, and the system starts to simulate the instruction l_j.

- There is no object a_r in cell 1 (corresponding to the fact that the number stored in register r is 0). In this case, at step 2, only rules r_{15}, r_{16} are enabled and applied, one copy of object l''_i is sent into cell 1 and object l'_i is sent to the environment (we assume that $l_k \neq l_i$). At step 3, with the appearance of object l''_i in cell 1, object b is sent to the environment and one copy of object l_k is sent into cell 1 by using rules r_{18} and r_{17}; simultaneously, object l''_i is sent to the environment by using rule r_{19}. At step 4, only rule r_{25} is used, object b in the environment is sent back to the cell 1. Hence, the system starts to simulate the instruction l_k.

When object l_h appears in cell 1, the computation will halt at that step or at the next step (in the case of the simulation of a SUB instruction that there is at least one copy of object a_r in cell 1). The number of copies of object a_1 in cell 1 corresponds to the result of the computation, hence $N(M) = N(\Pi)$.

Note that Turing universality is obtained by tissue P systems with at most two promoters associated with any rule and using only symport rules of length 1. So, the theorem holds. □

4. Solving SAT problem by tissue P systems with promoters and cell division

Let us start by recalling the well-known **NP**-complete `SAT` problem [9]: 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.

The following theorem shows that the `SAT` problem can be solved in linear time by a uniform family of tissue P systems with promoters and cell division, using non-cooperative symport rules and at most 2 promoters associated with each rule, working in the flat maximally parallel manner.

Theorem 2 $\texttt{SAT} \in \mathbf{PMC}^{f}_{\mathbf{TP_2DS}(1)}$.

Proof. Let us consider a propositional formula $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, with $C_i = y_{i,1} \vee \cdots \vee y_{i,p_i}$, for some $m \geq 1, p_i \geq 1$, and $y_{i,j} \in \{x_k, \neg x_k \mid 1 \leq k \leq n\}$, for each $1 \leq i \leq m, 1 \leq j \leq p_i$, where $\neg x_k$ is the negation of a propositional variable x_k, the two connections $\vee, \wedge$ are *or*, *and*, respectively.

For the given propositional formula φ, we construct the tissue P system with promoters and cell division

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

where:

- $\Gamma = \Sigma \cup \{a_i, t_i, f_i \mid 1 \leq i \leq n\} \cup \{b_j, c_j \mid 1 \leq j \leq m\} \cup \{z_i \mid 0 \leq i \leq 2n+m+3\} \cup \{a_{n+1}, b_{m+1}, p, \texttt{yes}, \texttt{no}\}$ is the alphabet,
- $\Sigma = \{x_{i,j}, \bar{x}_{i,j} \mid 1 \leq i \leq n, 1 \leq j \leq m\}$ is the input alphabet,
- $\mathcal{E} = \{a_i \mid 2 \leq i \leq n+1\} \cup \{b_j, c_j \mid 1 \leq j \leq m\} \cup \{z_i \mid 1 \leq i \leq 2n+m+1\} \cup \{b_{m+1}\}$,
- $w_1 = \{a_1\}$ is the initial multiset contained in cell 1,
- $w_2 = \{z_0, p, \texttt{yes}, \texttt{no}\}$ is the initial multiset contained in cell 2,
- $i_{in} = 1$ is the input cell,
- $i_{out} = 0$,
- the set $\mathcal{R}$ consists of the following rules:

  $r_{1,i} : [\ a_i\ ]_1 \rightarrow [\ t_i\ ]_1 [\ f_i\ ]_1$, $1 \leq i \leq n$,

  $r_{2,i,j} : (t_i x_{i,j} \mid 1, \lambda/c_j, 0)$, $1 \leq i \leq n, 1 \leq j \leq m$,

  $r_{3,i,j} : (t_i \mid 1, x_{i,j}/\lambda, 0)$, $1 \leq i \leq n, 1 \leq j \leq m$,

  $r_{4,i} : (t_i \mid 1, \lambda/a_{i+1}, 0)$, $1 \leq i \leq n$,

  $r_{5,i} : (1, t_i/\lambda, 0)$, $1 \leq i \leq n$,

$r_{6,i,j} : (f_i\bar{x}_{i,j} \mid 1, \lambda/c_j, 0)$, $1 \leq i \leq n, 1 \leq j \leq m$,

$r_{7,i,j} : (f_i \mid 1, \bar{x}_{i,j}/\lambda, 0)$, $1 \leq i \leq n, 1 \leq j \leq m$,

$r_{8,i} : (f_i \mid 1, \lambda/a_{i+1}, 0)$, $1 \leq i \leq n$,

$r_{9,i} : (1, f_i/\lambda, 0)$, $1 \leq i \leq n$,

$r_{10} : (a_{n+1} \mid 1, \lambda/b_1, 0)$,

$r_{11} : (1, a_{n+1}/\lambda, 0)$,

$r_{12,j} : (b_j c_j \mid 1, \lambda/b_{j+1}, 0)$, $1 \leq j \leq m$,

$r_{13,j} : (1, b_j/\lambda, 0)$, $1 \leq j \leq m$,

$r_{14} : (1, b_{m+1}/\lambda, 2)$,

$r_{15} : (b_{m+1} \mid 2, \mathtt{yes}/\lambda, 0)$,

$r_{16} : (b_{m+1} \mid 2, p/\lambda, 0)$,

$r_{17,i} : (z_i \mid 2, \lambda/z_{i+1}, 0)$, $0 \leq i \leq 2n + m + 2$,

$r_{18,i} : (2, z_i/\lambda, 0)$, $0 \leq i \leq 2n + m + 2$,

$r_{19} : (pz_{2n+m+3} \mid 2, \mathtt{no}/\lambda, 0)$.

An Overview of the Computation

We consider the polynomial encoding (cod, s) of instances from SAT in Π defined as follows: $s(\varphi) = \langle n, m\rangle$ and $cod(\varphi) = \{x_{i,j} \mid x_i \in C_j\} \cup \{\bar{x}_{i,j} \mid \neg x_i \in C_j\}$ for each instance. Hence, the Boolean formula φ will be processed by the system $\Pi(s(\varphi))$ with the input multiset $cod(\varphi)$.

In what follows, we give the overview of a computation to show how an instance of the SAT problem is solved by the system defined above. Generally, the computation process can be separated into three phases: generation phase, checking phase and output phase.

Generation phase. All truth assignments for the variables associated with the Boolean formula $\varphi(x_1, \ldots, x_n)$ will be generated by applying division rules in cells with label 1. Moreover, the system checks the clauses which are satisfiable by the corresponding truth assignment in a cell 1. In this way, after completing the phase, there exist 2^n cells with label 1 such that each of them contains some objects from the set $\{c_1, \ldots, c_m\}$, whose elements denote the clauses satisfied by the truth assignments of the variables.

In the initial configuration of the system, we have objects $a_1, cod(\varphi)$ in cell 1, objects $z_0, p, \mathtt{yes}, \mathtt{no}$ in cell 2.

The generation phase consists of a loop with n iterations, each iteration taking two steps. Thus this phase takes $2n$ steps. Specifically, this phase has two parallel processes. On the one hand, in cells with label 1, n loops of division and checking satisfied clauses are executed; on the other hand, in cell with label 2, the counter z evolves from z_0 to z_{2n} by using the rules $r_{17,i}, r_{18,i}$.

At the first step of the i-th loop ($1 \leq i \leq n$), the division rule $r_{1,i}$ is applied, cell 1 is divided into two copies of cell 1, which contain the objects t_i and f_i, respectively. In cell 2, the counter object z increases its subscript by one.

At the second step of the i-th loop ($1 \leq i \leq n$), with the appearance of object $t_i, x_{i,j}$ (resp., $f_i, \bar{x}_{i,j}$) in a cell 1, all the rules $r_{2,i,j}$ (resp., $r_{6,i,j}$) ($1 \leq j \leq m$) are enabled and applied, and the corresponding objects c_j are produced in that cell 1. Simultaneously, by using the rules $r_{3,i,j}, r_{4,i}$ (resp., $r_{7,i,j}, r_{8,i}$), objects $x_{i,j}$ ($1 \leq j \leq m$) are sent to the environment and one copy of object a_{i+1} is sent into cell 1 due to the flat maximal parallelism (resp., objects $\bar{x}_{i,j}$ ($1 \leq j \leq m$) are sent to the environment and one copy of object a_{i+1} is sent into cell 1). By using rule $r_{5,i}$ (resp., $r_{6,i}$), object t_i (resp., f_i) is sent to the environment. In addition, the subscript of the counter object z in cell 2 increases by one.

Checking phase. In this phase, the system checks whether or not the formula is satisfied by some truth assignment.

When the generation phase completes, each cell with label 1 contains some objects from the set $\{c_1, \ldots, c_m\}$. If there is at least one cell with label 1 that contains all the objects $c_1, \ldots, c_m$, this means that the corresponding truth assignment in that cell satisfies all clauses, hence formula φ is satisfiable; if there is no cell with label 1 that contains all the objects $c_1, \ldots, c_m$, the formula φ is not satisfiable. The checking phase begins at step $2n+1$, and it takes $m+1$ steps.

At step $2n+1$, with the appearance of object a_{n+1} in cell 1, by using rule r_{10}, one copy of object b_1 is sent into each cell 1 due to the flat maximal parallelism. Simultaneously, object a_{n+1} is sent to the environment by using rule r_{11}. In cell 2, the counter object z increases its subscript.

At step $2n+1+j$ ($1 \leq j \leq m$), the object c_j is checked in cell 1. If the objects b_j, c_j appear in a cell 1, rules $r_{12,j}$ and $r_{13,j}$ are applied at the same step, one copy of object b_{j+1} is sent into cell 1 and object b_j is sent to the environment, respectively. Note that the fact that the object b_j appears in a cell 1 means that the truth-assignment encoded in that cell 1 makes clauses $C_1, \ldots, C_{j-1}$ true. In cell 2, the counter object z increases its subscript.

Output phase. In this phase, the system sends the right answer into the environment according to the result of the previous phase.

If the input formula φ is satisfiable, then object b_{m+1} appears in cell 1 at step $2n+m+1$. With the appearance of object b_{m+1} in cell 1, rule r_{14} is enabled and applied, object b_{m+1} is sent to cell 2. At step $2n+m+3$, when object b_{m+1} appears in cell 2, objects `yes` and p are sent to the environment. Simultaneously, at step $2n+m+3$, object z_{2n+m+3} appears in cell 2 and the computation halts. Thus, the answer of the system is *affirmative*.

If the input formula φ is not satisfiable, then there is no object b_{m+1} in cell 1 at step $2n+m+1$. At the next two steps, only rules $r_{17,i}, r_{18,i}$ are enabled and used, and object z_{2n+m+3} will appear in cell 2. At step $2n+m+4$, with the appearance of objects p, z_{2n+m+3}, rule r_{19} is used, object `no` is sent to the environment, and the computation halts. Thus, the answer of the system is *negative*.

Formal Verification

According to the analysis above, we can check that the P system $\Pi(\langle m,n\rangle)$ with input multiset $cod(\varphi)$ always halts and sends to the environment one of the object `yes` or `no` in the last step, that is, at step $2n+m+3$, object `yes` is sent to the environment and the system halts, while object `no` is sent to the environment at step $2n+m+4$ and the system halts. Therefore, there exists a polynomial bound for the number of steps of the computation.

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. The necessary resources for defining each system are of polynomial order.

- Size of the alphabet: $2nm+5n+3m+9$.
- Initial number of cells: 2.
- Initial number of objects: 5.
- Number of rules: $4nm+9n+4m+12$.
- Maximal length of a rule (the promoters evolved in a rule are included): 3.

Therefore, the family $\mathbf{\Pi} = \{\Pi(\langle m,n\rangle) \mid m,n \in \mathbb{N}\}$ defined above is polynomially uniform by Turing machines, and this concludes the proof. $\square$

Since the complexity class $\mathbf{PMC}^{f}_{\mathbf{TP_2DS}(1)}$ is closed under polynomial time reductions, we have the following corollary.

Corrolary 1 $\mathbf{NP} \cup \mathbf{co-NP} \subseteq \mathbf{PMC}^{f}_{\mathbf{TP_2DS}(1)}$.

5. Conclusions and remarks

In this work, we investigated the computational efficiency and universality of tissue P systems with promoters working in the flat maximally parallel way. Specifically, we proved that tissue P systems with at most two promoters associated with any rule and using only symport rules of length 1 are Turing universal. Moreover, a uniform solution to the `SAT` problem has been provided by tissue P systems with at most two promoters associated with any rule and using cell division rules and symport rules of length 1.

The universality and computational efficiency given in section 2 and section 3 were obtained by tissue P systems with at most two promoters associated with any rule. It is of interest to investigate the computational power of tissue P systems with at most one promoter associated with any rule, and using antiport rules or symport rules of length at most 2. In this way, it is expected to obtain a characterization of the borderline of **P** and **NP** by the the number of promoters. Moreover, it is of interest to investigate the computational power of tissue P systems with inhibitors.

Inspired by the fact that the execution time of different biochemical reactions is hard to know precisely, in [28,31–33], time-free solutions to **NP**-complete problems by various P systems have been investigated. It remains open how we construct tissue P systems with promoters and cell division to solve **NP**-complete problems in the context of time-freeness.

Acknowledgements
The work was supported by National Natural Science Foundation of China (61320106005, 61033003, 61472372, and 61602192), Ph.D. Programs Foundation of Ministry of Education of China (20120142130008), China Postdoctoral Science Foundation (2016M600592), and the Innovation Scientists and Technicians Troop Construction Projects of Henan Province (154200510012).

References

[1] Alhazov, A., Freund, R., Leporati, A., Oswald, M., Zandron, C.: (Tissue) P Systems with Unit Rules and Energy Assigned to Membranes. Fund. Inform. 74(4), 391–408 (2006)

[2] Alhazov, A., Freund, R., Oswald, M.: Tissue P Systems with Antiport Rules and Small Numbers of Symbols and Cells. In: Felice, C.D., Restivo, A. (eds.) LNCS, vol. 3572, pp. 100–111. Springer, Heidelberg (2005)

[3] Ardelean, I., Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Peña-Cantillana, F., Reina-Molina, R., Sarchizian, I.: Counting Cells with Tissue-like P Systems. In: Proceedings of the Tenth Brainstorming Week on Membrane Computing, pp. 69–78. Fénix Editora, Sevilla (2012)

[4] Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Solving Subset Sum in Linear Time by Using Tissue P System with Cell Division. In: Mira, J., Álvarez, J.R. (eds.) LNCS, vol. 4527, pp. 170–179. Springer, Heidelberg (2007)

[5] Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A.: A Uniform Family of Tissue P System with Cell Division Solving 3-Col in a Linear Time. Theor. Comput. Sci. 404(1), 76–87 (2008)

[6] Díaz-Pernil, D., Pérez-Jiménez, M.J., Riscos-Núñez, A., Romero-Jiménez, Á.: Computational Efficiency of Cellular Division in Tissue-like Membrane Systems. Rom. J. Inf. Sci. Tech. 11(3), 229–241 (2008)

[7] Freund, R., Păun, Gh., Pérez-Jiménez, M.J.: Tissue P Systems with Channel States. Theor. Comput. Sci. 330(1), 101–116 (2005)

[8] Freund, R., Verlan, S.: (Tissue) P Systems Working in The k-Restricted Minimally or Maximally Parallel Transition Mode. Nat. Comput. 10, 821–833 (2011)

[9] Garey, M.R., Johnson, D.J.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, San Francisco (1979)

[10] Ionescu, M., Păun, Gh., Yokomori, T.: Spiking Neural P Systems. Fund. Inform. 71(2-3), 279–308 (2006)

[11] Krishna, S.N., Lakshmanan, K., Rama, R.: Tissue P Systems with Contextual and Rewriting Rules. In: Păun, Gh., Rozenberg, G., Salomaa, A., Zandron, C. (eds.) LNCS, vol. 2597, pp. 339–351. Springer, Heidelberg (2003)

[12] Leporati, A., Manzoni, L., Mauri, G., Porreca, A.E., Zandron, C.: Membrane Division, Oracles, and The Counting Hierarchy. Fund. Inform. 138(1-2), 97–111 (2015)

[13] Martín-Vide, C., Pazos, J., Păun, Gh., Rodriguez-Paton, A.: Tissue P Systems. Theor. Comput. Sci. 296(2), 295–326 (2003)

[14] Minsky, M.L.: Computation: Finite and Infinite Machines. Prentice-Hall, NJ (1967)

[15] Pan, L., Păun, Gh., Song, B.: Flat Maximal Parallelism in P Systems with Promoters. Theor. Comput. Sci. 623, 83–91 (2016)

[16] Pan, L., Pérez-Jiménez, M.J.: Computational Complexity of Tissue-like P Systems. J. Complexity 26(3), 296–315 (2010)

[17] Păun, A., Păun, Gh.: The power of Communication: P Systems with Symport/Antiport. New Generat. Comput. 20(3), 295–305 (2002)

[18] Păun, A., Păun, Gh., Rozenberg, G.: Computing by Communication in Networks of Membranes. Int. J. Found. Comput. S. 13, 779–798 (2002)

[19] Păun, Gh.: Computing with Membranes. J. Comput. Syst. Sci. 61(1), 108–143 (2000)

[20] Păun, Gh.: Membrane Computing: An Introduction. Springer-Verlag, Berlin (2002)

[21] Păun, Gh., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Tissue P Systems with Cell Division. Int. J. Comput. Commun. 3(3), 295–303 (2008)

[22] Păun, Gh., Rozenberg, G., Salomaa, A.(Eds.): The Oxford Handbook of Membrane Computing. Oxford University Press, New York (2010)

[23] Peng, H., Wang, J., Pérez-Jiménez, M.J., Riscos-Núñez, A.: An Unsupervised Learning Algorithm for Membrane Computing. Inf. Sci. 304, 80–91 (2015)

[24] Peng, H., Wang, J., Pérez-Jiménez, M.J., Wang, H., Shao, J., Wang, T.: Fuzzy Reasoning Spiking Neural P System for Fault Diagnosis. Inf. Sci. 235, 106–116 (2013)

[25] Pérez-Jiménez, M.J., An Approach to Computational Complexity in Membrane Computing. In: Mauri, G., Păun, Gh., Pérez-Jiménez, M.J., Rozenberg, G., Salomaa, A. (eds.) LNCS, vol. 3365, pp. 85–109. Springer, Heidelberg (2005)

[26] Reina-Molina, R., Díaz-Pernil, D., Gutiérrez-Naranjo, M.A.: Cell Complexes and Membrane Computing for Thinning 2D and 3D Images. In: Proceedings of the Tenth Brainstorming Week on Membrane Computing, pp. 167–186. Fénix Editora, Sevilla (2012)

[27] Rozenberg, G., Salomaa, A. (Eds.), Handbook of Formal Languages, 3 vols., Springer, Berlin, 1997.

[28] Song, B., Pan, L.: Computational Efficiency and Universality of Timed P Systems with Active Membranes. Theor. Comput. Sci. 567, 74–86 (2015)

[29] Song, B., Pan, L.: The Computational Power of Tissue-like P Systems with Promoters. Theor. Comput. Sci. 641, 43–52 (2016)

[30] Song, B., Pérez-Jiménez, M.J., Pan, L.: Efficient Solutions to Hard Computational Problems by P Systems with Symport/Antiport Rules and Membrane Division. BioSystems 130, 51–58 (2015)

[31] Song, B., Pérez-Jiménez, M.J., Pan, L.: Computational Efficiency and Universality of Timed P Systems with Membrane Creation. Soft Comput. 19(11), 3043–3053 (2015)

[32] Song, B., Pérez-Jiménez, M.J., Pan, L.: An Efficient Time-free Solution to SAT Problem by P Systems with Proteins on Membranes. J. Comput. Syst. Sci. 82, 1090–1099 (2016)

[33] Song, B., Song, T., Pan, L.: Time-free Solution to SAT Problem by P Systems with Active Membranes and Standard Cell Division Rules. Nat. Comput. 14(4), 673–681 (2015)

[34] Song, B., Zhang, C., Pan, L.: Tissue-like P Systems with Evolutional Symport/Antiport Rules. Inf. Sci. 378, 177–193 (2017)

[35] Sosík, P., Langer, M.: Small (Purely) Catalytic P Systems Simulating Register Machines. Theor. Comput. Sci. 623, 65–74 (2016)

[36] Zhang, G., Gheorghe, M., Pan, L., Pérez-Jiménez, M.J.: Evolutionary Membrane Computing: A Comprehensive Survey and New Results. Inf. Sci. 279, 528–551 (2014)

[37] Zhang, X., Wang, S., Niu, Y., Pan, L.: Tissue P Systems with Cell Separation: Attacking the Partition Problem. Sci. China Inf. Sci. 54(2), 293–304 (2011)

[38] Zhang, G., Rong, H., Neri, F., Pérez-Jiménez, M.J.: An Optimization Spiking Neural P System for Approximately Solving Combinatorial Optimization Problems. Int. J. Neural Syst. 24(5), 1–16 (2014)