Monodirectional Tissue *P* Systems With Promoters

Bosheng Song, Xiangxiang Zeng, *Senior Member, IEEE*, Min Jiang, *Senior Member, IEEE*, and Mario J. Pérez-Jiménez

***Abstract*—Tissue *P* systems with promoters provide nondeterministic parallel bioinspired devices that evolve by the interchange of objects between regions, determined by the existence of some special objects called *promoters*. However, in cellular biology, the movement of molecules across a membrane is transported from high to low concentration. Inspired by this biological fact, in this article, an interesting type of tissue *P* systems, called monodirectional *tissue P systems with promoters*, where communication happens between two regions only in one direction, is considered. Results show that finite sets of numbers are produced by such *P* systems with one cell, using any length of symport rules or with any number of cells, using a maximal length 1 of symport rules, and working in the maximally parallel mode. Monodirectional tissue *P* systems are Turing universal with two cells, a maximal length 2, and at most one promoter for each symport rule, and working in the maximally parallel mode or with three cells, a maximal length 1, and at most one promoter for each symport rule, and working in the flat maximally parallel mode. We also prove that monodirectional tissue *P* systems with two cells, a maximal length 1, and at most one promoter for each symport rule (under certain restrictive conditions) working in the flat maximally parallel mode characterizes regular sets of natural numbers. Besides, the computational efficiency of monodirectional tissue *P* systems with promoters is analyzed when cell division rules are incorporated. Different uniform solutions to the Boolean satisfiability problem (SAT problem) are provided. These results show that with the restrictive condition of "monodirectionality," monodirectional tissue *P* systems with promoters are still computationally powerful. With the powerful computational power, developing membrane algorithms for monodirectional tissue *P* systems with promoters is potentially exploitable.**

***Index Terms*—Bioinspired computing, membrane computing, monodirectional tissue *P* system, NP-complete problem tissue-like network, universality.**

Manuscript received November 21, 2019; revised March 2, 2020 and April 20, 2020; accepted June 14, 2020. Date of publication July 10, 2020; date of current version December 22, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61972138 and Grant 61602192, in part by the Fundamental Research Funds for the Central Universities under Grant 531118010355, in part by the Natural Science Foundation of Hunan Province of China under Grant 2020JJ4215, in part by the Research Project under Grant TIN2017-89842-P, cofinanced by Ministerio de Economía, Industria y Competitividad (MINECO) of Spain, through the Agencia Estatal de Investigación, and in part by the Fondo Europeo de Desarrollo Regional (FEDER) of the European Union. This article was recommended by Associate Editor L. Cheng. *(Corresponding author: Xiangxiang Zeng.)*

Bosheng Song and Xiangxiang Zeng are with the College of Information Science and Engineering, Hunan University, Changsha 410082, China (e-mail: boshengsong@hnu.edu.cn; xzeng@foxmail.com).

Min Jiang is with the School of Information Science and Engineering, Xiamen University, Xiamen 361005, China (e-mail: minjiang@xmu.edu.cn).

Mario J. Pérez-Jiménez is with the Department of Computer Science and Artificial Intelligence, Universidad de Sevilla, 41012 Sevilla, Spain (e-mail: marper@us.es).

Digital Object Identifier 10.1109/TCYB.2020.3003060

I. Introduction

BIOINSPIRED computing focuses on the designs and developments of computer algorithms and models based on biological mechanisms and living phenomena, which includes quantum computing [27]; DNA computing [33], [63], [67]; membrane computing [42]; etc. In this article, we focus on the research area of membrane computing, which is an active bioinspired field initiated by Păun [42], who discussed computing models motivated by the behavior and structure of cells, understanding the processes that take place in the compartments as computations. All the computing devices considered in this paradigm are called *P systems* [42]. Since the initial model proposes in this area, various *P* system models have been presented and investigated in the aspects of computer science [1], [22], [52]; mathematics [7], [28]; and biochemistry [15], [23]. A brief summary of membrane computing can be found in monographs [43], [45], while a review of applications of this field can be found in [14] and [21].

An essential and important component of *P* systems is membrane structures that can be classified into two categories: 1) hierarchical arrangements of membranes corresponding to trees (*cell-like P systems* [42]) and 2) nets of membranes (neurons) corresponding to arbitrary graphs (*neural-like* [26], [46], [56], [58], [61], [62], [64] or *tissue-like P* systems [31]). The basic models studied in this work are tissue-like *P* systems, which are motivated by the structure of tissue and the way of communicating substances moving from one region to another region. In a tissue *P* system, cells can communicate with other cells directly through channels and with the environment via symport/antiport rules [41]. A rule is called *symport* if objects go the same direction; and a rule is called *antiport* if some objects go the opposite directions at the same time between two regions.

Inspired by various biological phenomena of cells, various tissue *P* systems were constructed and investigated, most of them are turned out to be Turing complete [3], [4], [20], [40]. Besides, from mathematical, biological, and computational point of view, it is natural to incorporate mitosis (corresponding to computational rules "cell division") [44] or membrane fission (corresponding to computational rules "cell separation") [36] into tissue *P* systems, giving them the capacity to generate an exponential space of computational units in linear time. Thanks to this mechanism, different *NP*-complete problems, for instance, 3-coloring [17]; vertex cover [18]; subset sum [16], [47], [55]; and satisfiability problem (SAT problem) [36], [44], [57], can be efficiently solved by means of tissue *P* systems with cell separation or cell division. We

TABLE I
RESULTS BETWEEN TISSUE *P* SYSTEMS WITH PROMOTERS AND MONODIRECTIONAL TISSUE *P* SYSTEMS WITH PROMOTERS, WHERE pro_k REPRESENTS AT MOST k PROMOTERS ASSOCIATED WITH EACH RULE, sym_{t_1} REPRESENTS SYMPORT RULES OF LENGTH AT MOST t_1, anti_{t_2} REPRESENTS ANTIPORT RULES OF LENGTH AT MOST t_2, $*$ REPRESENTS UNBOUNDED ON THE PARAMETER, AND ? REPRESENTS UNKNOWN RESULT

Strategies of using rules		Tissue P systems with promoters	This paper
Maximal parallelism	Computational power	$NOtP_*(pro_*, sym_1) \subseteq NFIN.$ [53] $NOtP_1(pro_1, sym_2) \subseteq NRE.$ [53]	$NOtP_1^{mon}(pro_*, sym_*) \subseteq NFIN.$ $NOtP_*^{mon}(pro_*, sym_1) \subseteq NFIN.$ $NOtP_2^{mon}(pro_1, sym_2) = NRE.$
	Computational efficiency	$\texttt{SAT} \in \mathbf{PMC}_{\mathbf{TPDC}(pro_1, anti_2)}$. [53]	?
Flat maximal parallelism	Computational power	$NOtP_4(pro_2, sym_1) = NRE.$ [37]	$NOtP_3^{mon}(pro_1, sym_1) = NRE.$
	Computational efficiency	$\texttt{SAT} \in \mathbf{PMC}_{\mathbf{TPDC}(pro_2, sym_1)}$. [37]	$\texttt{SAT} \in \mathbf{PMC}_{\mathbf{MTPDC}(pro_2, sym_1)}$. $\texttt{SAT} \in \mathbf{PMC}_{\mathbf{MTPDC}(pro_1, sym_2)}$.

remark that under the hypothesis $P \neq NP$, *NP*-complete problems cannot be solved by *P* systems without cell division in polynomial time [48].

Tissue *P* systems with promoters, motivated by the fact that biological reactions may occur in the presence of certain chemicals, were raised in [8] and [50]. In [8] and [50], tissue *P* systems with promoters were investigated in the way of application; that is, such models were used in image processing. Moreover, tissue *P* systems with promoters as generating devices of numbers were studied in [53], which was shown that such *P* systems are Turing complete when different lengths of communication rules are combined (the length of a rule is defined by all numbers of objects in such a rule).

Inspired by the biological fact that the movement of molecules across a membrane is transported from high to low concentration, the notion of "monodirectionality" was first proposed in cell-like *P* systems (more precisely, *P* systems with active membranes [29], readers can refer [6], [19], [22], and [39] for more details about *P* systems with active membranes), where for two given regions, communication happens only in one direction and never in the opposite direction, that is, for two given regions, either object *send-in* rules or object *send-out* rules can be used.

The motivation of this work aims at building a bridge between tissue *P* systems with promoters and a variety of applications that involve information representation and information processing, thereby extending tissue *P* systems with promoters to serve as a class of suitable and attractive models for these applications. Motivated by the monodirectional nature in cellular biology, a novel type of tissue *P* systems with promoters, called monodirectional *tissue P systems with promoters*, is introduced, where communication happens between two regions only in one direction, and never in the opposite direction (hence, antiport rules are forbidden to be used systems).

Many applications require a monodirectional mechanism, including information acquisition device for power systems [30], [59]; some controllers for mobile robots [9], [60]; etc. In this way, the computational models are usually required to have the monodirectional nature in the sense that the transmission of information in devices can only be one way.

The computational power of monodirectional tissue *P* systems with promoters is examined as number generators. As a result, finite sets of numbers are produced by such *P* systems with one cell, using any length of symport rules or with any number of cells, using a maximal length 1 of symport rules, and working in the maximally parallel mode. Monodirectional tissue *P* systems are Turing universal with two cells, a maximal length 2, and at most one promoter for each symport rule, and working in the maximally parallel mode or with three cells, a maximal length 1, and at most one promoter for each symport rule, and working in the flat maximally parallel mode (see Table I). We also prove that monodirectional tissue *P* systems with two cells, a maximal length 1, and at most one promoter for each symport rule (under certain restrictive conditions) working in the flat maximally parallel mode characterizes regular sets of natural numbers.

The computational efficiency of monodirectional tissue *P* systems with promoters is also investigated by introducing cell division rules. It is proved that the SAT problem is solved by such systems with a maximal length 1 and at most two promoters for each symport rule or with a maximal length 2 and at most one promoter for each symport rule, working in the flat maximally parallel mode (see Table I).

The main contributions of this article are summarized as follows.

1) A novel type of tissue *P* systems with promoters, called monodirectional *tissue P systems with promoters*, is developed by introducing the notion of monodirectionality into tissue *P* systems with promoters. More precisely, such *P* systems have a network architecture with the capability of complex topology representation. Moreover, the achieved monodirectional tissue *P* systems with promoters are proved to be Turing universal. These results manifest that a Turing universal monodirectional paradigm of tissue *P* systems with promoters is theoretically possible and potentially exploitable.
2) A monodirectionality control strategy is introduced into tissue *P* systems with promoters to control the application of communication rules, thus making monodirectional tissue *P* systems with promoters be more suitable for some applications which require a monodirectional mechanism.

3) By employing network architecture as a model structure and monodirectionality as information processing, monodirectional tissue P systems with promoters are attractive to some real-world problems, which involve monodirectional nature and require networking model precisely.
4) Furthermore, by incorporating cell division into monodirectional tissue P systems with promoters, the information nature in cells can be replicated, thus making monodirectional tissue P systems with promoters be a more powerful modeling tool to develop various membrane algorithms, which makes training such systems presumable and enhances its potential for practical applications.

The remainder of this article is organized as follows. Section II presents some fundamental conceptions of language and automata theory and the notion of monodirectional tissue P systems with promoters and cell division. The computational power of the proposed P systems is investigated in Section IV. In Section V, computational efficiency of the proposed P systems is presented. Finally, conclusions and some future works are given in Section VI.

II. Preliminaries and Model Description

In this section, several fundamental conceptions from language and automata theory are recalled [51]. Also, the notion of (recognizer) monodirectional tissue P systems with promoters and cell division is introduced.

We define an alphabet (denoted by Γ) to be any nonempty finite set of abstract symbols. The set of strings that is concatenated by any number of symbols is denoted by Γ^*. $\Gamma^+ = \Gamma^* \backslash \{\lambda\}$ is the set that excludes the empty string (if a string does not have any symbols at all, it is called an *empty* string, denoted by λ). The number of symbols in a string u is the *length* of u, which is denoted by $|u|$.

Let Γ be an alphabet, and a *multiset* $\mathcal{M}$ is two tuples (Γ, f) such that f is a function from Γ to $\mathbb{N}$ (the set of natural numbers). $\mathcal{M}(\Gamma)$ [respectively, $\mathcal{M}^+(\Gamma)$] represents the set of all multisets (respectively, nonempty multisets). If $\Gamma = \{a_1, \ldots, a_k\}$, then the multiset $\mathcal{M} = (\Gamma, f)$ can be denoted by $\{a_1^{f(a_1)}, \ldots, a_k^{f(a_k)}\}$. If we have two multisets $\mathcal{M}_1 = (\Gamma, f_1)$ and $\mathcal{M}_2 = (\Gamma, f_2)$, then $\mathcal{M}_1 + \mathcal{M}_2$ is the union of $\mathcal{M}_1$ and $\mathcal{M}_2$, which is defined by $(\Gamma, f_1(x) + f_2(x))$ such that each $x \in \Gamma$.

The family of finite sets of natural numbers is denoted by *NFIN*, the family of regular sets of natural numbers is denoted by *NREG*, and we denote by NRE the family of recursively enumerable sets of natural numbers recognized by Turing machines.

In order to characterize NRE, the notion of *program machines* is used. Readers can refer [45] for more details about program machines. It is known that program machines and Turing machines are equivalent; that is, both of them characterize NRE [32].

Next, we give the notion of monodirectional tissue P systems with promoters and cell division (the readers are suggested to refer to [53] for more information about tissue P systems with promoters).

Definition 1: A monodirectional tissue P system with promoters and cell division of degree $q \geq 1$ has the following construction:

$$\Pi = \left(\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{\text{out}}\right)$$

where

1) Γ is a finite set of alphabet of objects;
2) $\mathcal{E}$ is a set of alphabet of objects initially placed in the environment, such that $\mathcal{E} \subseteq \Gamma$;
3) $\mathcal{M}_i$, $1 \leq i \leq q$, are multisets of objects initially placed in q cells;
4) $\mathcal{R}$ is a set of division rules and symport rules with the following restriction.
   a) *Symport Rules:* Either rules of form $(\text{pro}|i, u/\lambda, j)$ or rules of form $(\text{pro}|i, \lambda/u, j)$ for two given regions exist in the system such that $0 \leq i \neq j \leq q$, $\text{pro} \in \mathcal{M}(\Gamma)$, and $u \in \mathcal{M}^+(\Gamma)$.
   b) *Division Rules:* $[\ a\ ]_i \rightarrow [\ b\ ]_i[\ c\ ]_i$ such that $i \in \{1, \ldots, q\}$, $i \neq i_{\text{out}}$, $a, b, c \in \Gamma$.
5) $i_{\text{out}} \in \{0, 1, \ldots, q\}$ is an output region.

Note that if a system does not contain cell division rules, then it is simply called a monodirectional *tissue P system with promoters*.

Rules in monodirectional tissue P systems with promoters (and cell division) are applied in the *maximally parallel mode* (a maximum degree of each rule is used in parallel) [42] or in the *flat maximally parallel mode* (for each computation step, a maximal applicable set of rules is selected and applied exactly once for each rule in this set) [5], [35], [54].

A *configuration* of monodirectional tissue P systems with promoters at an instant t is defined by a tuple $(N_1, \ldots, N_q, N_e)$, where N_i $(1 \leq i \leq q)$ are multisets of objects over Γ and N_e is a multiset of objects over $\Gamma \backslash \mathcal{E}$.

The notions of computation and halting computation are described in [42]; in particular, a configuration is a *halting configuration* if no rule of the system is applicable to it. Only a computation reaching a halting configuration gives a result, which is encoded by the multiset of specified objects present in the output region i_{out} (i.e., some objects present in the output region i_{out} may not be counted).

A division rule $[\ a\ ]_i \rightarrow [\ b\ ]_i[\ c\ ]_i$ is applicable if and only if object a occurs in cell i, and cell i is not the output cell. When applying such a rule, cell i is divided into two cells with the same label: object a specified in the cell i is replaced by object b and object c in the newly generated cells, respectively; and all objects in the original cell, different from the object triggering the rule, are duplicated in the two new cells.

For the application of symport rules, one is referred to [53]. Note that in monodirectional tissue P systems 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 promoters do not directly participate in the rules. If a symport rule does not involve promoters, then such rule is simply written by $(i, u/\lambda, j)$.

A P system Π that computes a set of natural numbers is denoted by $N(\Pi)$, and we denote by

$NOtP_m^{\mathrm{mon}}(\mathrm{pro}_k, \mathrm{sym}_t, \max)$ [respectively, $NOtP_m^{\mathrm{mon}}(\mathrm{pro}_k, \mathrm{sym}_t, f\max)$] the family of all sets $N(\Pi)$ of natural numbers computed by systems Π with at most m cells, using a maximal length t and at most k promoters for each symport rule, and working in the maximally parallel mode (respectively, in the flat maximally parallel mode). If no bound on any of parameters m, k, and t is forced, then we use symbol $*$ to replace it.

The definition of recognizer tissue *P* systems with promoters and cell division was proposed in [53], such a model is considered to solve decision problems in a uniform way. The readers are referred to [53] for details.

We denote by $\mathbf{PMC}_{\mathbf{MTPDS}(\mathrm{pro}_k, \mathrm{sym}_t, f\max)}$ the set of all decision problems that are solved by a family of recognizer monodirectional tissue *P* systems with cell division, using a maximal length t and at most k promoters for each symport rule in a uniform and polynomial time, and working in the flat maximally parallel mode.

III. Two Examples

In order to illustrate the difference between monodirectional tissue *P* systems and tissue *P* systems using only symport rules (here, we consider that both of these two kinds of *P* systems are worked in the maximally parallel mode), the following two examples are presented.

Example 1: Let $\Pi_1 = (\Gamma, \mathcal{E}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, 1)$ be a monodirectional tissue *P* system (cell 1 is the output cell), where $\Gamma = \{a\}$, $\mathcal{M}_1 = \mathcal{M}_2 = \{a\}$, and $\mathcal{R} = \{r_1 : (1, a/\lambda, 2)\}$. Note that rules of form $(1, \lambda/a, 2)$ are not allowed. At step 1, object a is sent to cell 2 from cell 1, then no rule can be used in the system, and the computation halts. So the result of computation is empty set.

Example 2: Let $\Pi_2 = (\Gamma, \mathcal{E}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, 1)$ be a tissue *P* system using only symport rules (cell 1 is the output cell), where $\Gamma = \{a\}$, $\mathcal{M}_1 = \mathcal{M}_2 = \{a\}$, and $\mathcal{R} = \{r_1 : (1, a/\lambda, 2); r_2 : (1, \lambda/a, 2)$. Now, we analyze how such *P* system works: in such *P* system, both rules r_1 and r_2 can be chosen and used. Obviously, both rules r_1 and r_2 are used in every step, and the computation never halts. So no result is obtained.

IV. Computational Power of Monodirectional Tissue *P* Systems With Promoters

In this section, monodirectional tissue *P* systems with promoters as generating devices for numbers are investigated.

A. Computational Power of Monodirectional Tissue P Systems With Promoters Working the Maximally Parallel Mode

In this section, monodirectional tissue *P* systems with promoters as generating devices for natural numbers working in the maximally parallel mode are investigated.

Theorem 1: $NOtP_1^{\mathrm{mon}}(\mathrm{pro}_*, \mathrm{sym}_*, \max) \subseteq NFIN$.

Proof: Keep in mind that the system contains only one cell, objects move between the environment and cell by using symport rules in one direction. Hence in such a system, communication occurs in the following two cases: 1) objects in the environment are moved to cell and 2) objects in the cell are moved to the environment. For case 1), symport rules are not allowed because for objects in set $\mathcal{E}$ (the environment), each of them has any number of copies (if symport rules are allowed in this direction, the computation will never stop); and for case 2), since there is a finite number of objects in the cell, the numbers of computation steps and reachable configurations are finite. Therefore, finite sets of numbers are generated by the system and the theorem holds. ■

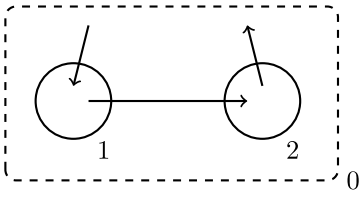

Fig. 1. Membrane structure for the constructed monodirectional tissue *P* system with promoters, where circles represent cells, numbers on the side of the circles represent labels of cells, all cells are placed in the environment with label 0, and arrows indicate directions that objects are moved between two regions.

Theorem 2: $NOtP_*^{\mathrm{mon}}(\mathrm{pro}_*, \mathrm{sym}_1, \max) \subseteq NFIN$.

Proof: It is clear that symport rules of form $(\mathrm{pro}_*|i, \lambda/a, 0)$ $(i \neq 0, a \in \mathcal{E})$ are not allowed. So objects in set $\mathcal{E}$ cannot be sent into any cells. Besides, there is a finite number of objects in the system, so the numbers of computation steps and reachable configurations are finite. Hence, finite sets of numbers are generated by the system and the theorem holds. ■

Theorem 3: $NOtP_2^{\mathrm{mon}}(\mathrm{pro}_1, \mathrm{sym}_2, \max) = \mathrm{NRE}$.

Proof: Let $M = (m, H, l_0, l_h, I)$ be a program machine. The following monodirectional tissue *P* system with promoters Π (see Fig. 1) is constructed to simulate M:

$$\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, 1)$$

where

1) $\Gamma = \{l, l', l'', l''', l^{iv}, l^{v}, l^{vi}, l^{vii} | l \in H\} \cup \{a_r | 1 \leq r \leq m\}$;
2) $\mathcal{E} = \{l, l^{v}, l^{vi}, l^{vii} | l \in H\} \cup \{a_r | 1 \leq r \leq m\}$;
3) $\mathcal{M}_1 = \{l', l'', l''', l^{iv} | l \in H\} \cup \{l_0\}$, $\mathcal{M}_2 = \emptyset$.

We design the following finite set $\mathcal{R}$ of symport rules.

The number of object a_r in cell 1 corresponds to the value of register r in M. Each ADD instruction (respectively, SUB instruction) in M can be simulated by six steps (respectively, eight steps) in Π. Initially, cell 1 contains multisets of objects $\{l', l'', l''', l^{iv} | l \in H\} \cup \{l_0\}$, cell 2 is empty, and the environment contains multisets of objects $\{l, l^{v}, l^{vi}, l^{vii} | l \in H\} \cup \{a_r | 1 \leq r \leq m\}$ (each of these objects has an arbitrary number of copies). The program machine M halts corresponding to cell 1 has object l_h and no rule can be applied in system Π. When the designed *P* system Π halts, the computation result for M is the number of object a_1 deposited in cell 1 at this moment.

1) The following rules in $\mathcal{R}$ are constructed to simulate each ADD instruction l_i of M:

$$r_{1,i} : (1, l_i l_i'/\lambda, 2), \qquad r_{2,i} : (2, l_i l_i'/\lambda, 0)$$
$$r_{3,i} : (1, \lambda/l_i' l_i^{v}, 0), \qquad r_{4,i} : (l_i^{v} | 1, l_i'' l_i'''/\lambda, 2)$$
$$r_{5,i} : (1, l_i^{v}/\lambda, 2), \qquad r_{6,i} : (2, l_i'' l_i'''/\lambda, 0)$$
$$r_{7,i} : (2, l_i^{v}/\lambda, 0), \qquad r_{8,i} : (1, \lambda/l_i''' a_r, 0)$$

$$r_{9,i} : (1, \lambda/l_i'' l_j, 0), \qquad r_{10,i} : (1, \lambda/l_i'' l_k, 0).$$

By applying rules $r_{1,i}$, $r_{2,i}$, and $r_{3,i}$ one by one, cell 1 will appear object l_i^v. At step 4, rules $r_{4,i}$ and $r_{5,i}$ are applied in parallel, cell 2 receives objects l_i'', l_i''', and l_i^v, and at the next step, the environment will receive all these objects. At step 6, cell 1 receives objects l_i''' and a_r by means of applying rule $r_{8,i}$; meanwhile, the system nondeterministically chooses $r_{9,i}$ and $r_{10,i}$, and using one of these rules, the label object l_j or l_k is present in cell 1. Hence, during the process of simulation, the output cell increases one copy of object a_r, which will simulate the next instruction l_j or l_k.

1) The following rules in $\mathcal{R}$ are constructed to simulate each SUB instruction l_i of M:

$$\begin{aligned}
&r_{11,i} : (1, l_i l_i'/\lambda, 2), && r_{12,i} : (2, l_i l_i'/\lambda, 0)\\
&r_{13,i} : (1, \lambda/l_i' l_i^v, 0), && r_{14,i} : (l_i^v|1, l_i'' a_r/\lambda, 2)\\
&r_{15,i} : (l_i^v|1, l_i''' l_i^{iv}/\lambda, 2), && r_{16,i} : (1, l_i^v/\lambda, 2)\\
&r_{17,i} : (2, l_i'' a_r/\lambda, 0), && r_{18,i} : (2, l_i''' l_i^{iv}/\lambda, 0)\\
&r_{19,i} : (2, l_i^v/\lambda, 0), && r_{20,i} : (l_i''|0, l_i''' l_i^{vi}/\lambda, 1)\\
&r_{21,i} : (l_i''|1, \lambda/l_i''' l_i^{vii}, 0), && r_{22,i} : (l_i^{vi}|1, \lambda/l_i'' l_j, 0)\\
&r_{23,i} : (l_i^{vi}|1, \lambda/l_i^{iv}, 0), && r_{24,i} : (1, l_i^{vi}/\lambda, 2)\\
&r_{25,i} : (l_i^{vii}|1, \lambda/l_i^{iv} l_k, 0), && r_{26,i} : (1, l_i^{vii}/\lambda, 2)\\
&r_{27,i} : (2, l_i^{vi}/\lambda, 0), && r_{28,i} : (2, l_i^{vii}/\lambda, 0).
\end{aligned}$$

The system simulates an SUB instruction l_i as follows. By applying rules $r_{11,i}$, $r_{12,i}$, and $r_{13,i}$ one by one, object l_i^v will appear in cell 1 after three steps. At step 4, we have two cases to check whether cell 1 contains object a_r or not.

1) Cell 1 contains object a_r. Rules $r_{14,i}$, $r_{15,i}$, and $r_{16,i}$ are applied in parallel, cell 2 receives objects l_i'', l_i''', l_i^{iv}, l_i^v, and a_r, and these objects will be sent to the environment by applying rules $r_{17,i}$, $r_{18,i}$, and $r_{19,i}$. When the environment contains object l_i'', objects l_i''' and l_i^{vi} are sent into cell 1. If cell 1 contains object l_i^{vi}, such cell will receive objects l_i'', l_i^{iv}, and l_j; meanwhile, cell 2 receives object l_i^{vi}, and the environment will receive object l_i^{vi} at the next step. Hence, during this process, the output cell decreases one copy of object a_r (corresponding to subtract one from register r), which will simulate the next instruction l_j.
2) Cell 1 does not contain object a_r. In this case, by applying rules $r_{15,i}$ and $r_{16,i}$ in parallel, cell 2 receives objects l_i''', l_i^{iv}, and l_i^v, and the environment will receive these objects at the next step by applying rules $r_{18,i}$ and $r_{19,i}$. At this moment, if cell 1 contains object l_i'', such cell will receive objects l_i''' and l_i^{vii}. When cell 1 contains object l_i^{vii}, objects l_i^{iv} and l_k are sent into cell 1; meanwhile, cell 2 receives object l_i^{vii}, and the environment will receive object l_i^{vii} at the next step. So when the simulation of instruction l_i is finished, the system will start to simulate instruction l_k.

So an SUB instruction can be effectively simulated for both cases.

The computation halts only when cell 1 contains object l_h. The number of object a_1 stored in cell 1 at this moment represents the computation result of M. Hence, $N(M) = N(\Pi)$, and this concludes the proof. ■

B. Computational Power of Monodirectional Tissue P Systems With Promoters Working in the Flat Maximally Parallel Mode

In this section, the computational power of monodirectional tissue P systems with promoters working in the flat maximally parallel mode is presented.

Theorem 4: $NOtP_2^{\text{mon}}(\text{pro}_1, \text{sym}_1, f\max) = NREG$, where two cells are labeled by 1 and 2, respectively; one cell (we assume cell 1, and cell 1 is the output cell) can receive objects from the environment and cell 2 has no communication with the environment; and objects are moved from cell 1 to cell 2.

Proof: We first prove that $NOtP_2^{\text{mon}}(\text{pro}_1, \text{sym}_1, f\max) \subseteq NREG$.

Let Π' be an arbitrary monodirectional tissue P system with a maximal length 1 and at most 1 promoter for each symport rule (under the above restrictive conditions). The maximal number of objects increased in the system Π' is described as follows: each object initially placed in cell 1 and each object initially placed in the environment (at some step, these objects are sent into cell 1) can be viewed as promoters, with the influence of the promoter, some copies of objects are introduced into the system; simultaneously, the promoter must be sent to cell 2, otherwise, the system never halts. Since the number of different objects initially placed in cell 1 and the number of different objects initially placed in the environment are finite, the numbers of computation steps and reachable configurations are finite. We denote these configurations by C_i, $0 \le i \le L$, and C_0 is the initial configuration.

We construct a regular grammar $G = (N, T, C_0, P)$, where $N = \{C_i | 0 \le i \le L\}$, T is a set of terminal objects, and P contains the following productions.

1) $C_i \to aC_j$, for $C_i, C_j \in N, a \in T$ such that there is a transition $C_i \Rightarrow C_j$ between two configurations of Π' during which the rules introducing terminal object a into the output cell are used.
2) $C_i \to C_j$, for $C_i, C_j \in N$ such that there is a transition $C_i \Rightarrow C_j$ between two configurations of Π' during which the output cell does not receive terminal object.
3) $C_i \to \lambda$, for $C_i \in N$ such that C_i is a halting configuration of Π'.

Note that if there are several copies of terminal objects introduced into the system (or output cell) at one step, then we can increase the number of configurations such that each subconfiguration introduces one copy of the terminal object.

We can check that each set of natural numbers generated by system Π' can be generated by the regular grammar G. Hence, $NOtP_2^{\text{mon}}(\text{pro}_1, \text{sym}_1, f\max) \subseteq NREG$.

In what follows, we prove that the converse inclusion also holds. Let $G' = (N', T', S', P')$ be an arbitrary regular grammar with the productions of the form $A \to aB$ and $A \to a$, where $A, B \in N'$, $a \in T'$, $S' = A$, and P' is the set of productions. We construct a monodirectional tissue P system Π''

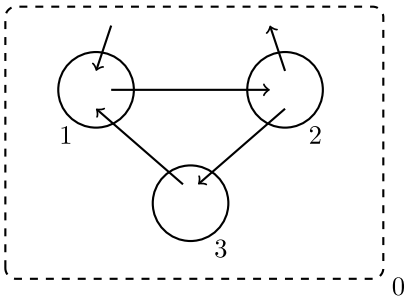

Fig. 2. Membrane structure of the constructed monodirectional tissue P system with promoters, where circles represent cells, numbers on the side of the circles represent labels of cells, all cells are placed in the environment with label 0, and arrows indicate directions that objects are moved between two regions.

under the above restrictive conditions. Initially, N' is the alphabet of system Π'', and object $S' \in N'$. The set of rules in Π'' has the forms $(p|1, \lambda/q, 0)$ and $(1, p/\lambda, 2)$. A production of the form $A \to aB$ can be simulated by using rules $(A|1, \lambda/a, 0)$, $(A|1, \lambda/B, 0)$, and $(1, A/\lambda, 2)$ in one step in the mode of flat maximally parallelism. A production of the form $A \to a$ can be simulated by using rules $(A|1, \lambda/a, 0)$ and $(1, A/\lambda, 2)$ in one step in mode of flat maximally parallelism. We can check that system Π'' can generate the sets of natural numbers generated by grammar G'. Hence, $NOtP_2^{\text{mon}}(\text{pro}_1, \text{sym}_1, f\max) \supseteq NREG$. ■

The following corollary is easy to obtain according to Theorem 4, here we omit the proof process.

Corollary 1: $NOtP_2^{\text{mon}}(\text{pro}_1, \text{sym}_1, f\max) = NREG$, where two cells are labeled by 1 and 2, respectively; both cells (we assume cell 1 is the output cell) can receive objects from the environment, and objects are moved from cell 1 to cell 2.

Theorem 5: $NOtP_3^{\text{mon}}(\text{pro}_1, \text{sym}_1, f\max) = \text{NRE}$.

Proof: Let $M = (m, H, l_0, l_h, I)$ be a program machine. The following monodirectional tissue P system with promoters Π (see Fig. 2) is constructed to simulate M:

$$\Pi = (\Gamma, \mathcal{E}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{R}, 1)$$

where

1) $\Gamma = \{l, l', l'', l''', l^{iv}, l^v | l \in H\} \cup \{a_r | 1 \le r \le m\} \cup \{d\}$;
2) $\mathcal{E} = \{l, l^v | l \in H\} \cup \{a_r | 1 \le r \le m\}$;
3) $\mathcal{M}_1 = \{l', l'', l''' | l \in H\} \cup \{l_0\}$, $\mathcal{M}_2 = \{l_i^{iv} | l \in H\} \cup \{d\}$, $\mathcal{M}_3 = \emptyset$.

We design the following finite set $\mathcal{R}$ of symport rules.

The number of object a_r in cell 1 corresponds to the value of register r in M. Each ADD instruction (respectively, SUB instruction) in M can be simulated by four steps (respectively, eight steps) in Π. Initially, cell 1 contains multisets of objects $\{l', l'', l''' | l \in H\} \cup \{l_0\}$, cell 2 contains multisets of objects $\{l_i^{iv} | l \in H\} \cup \{d\}$, cell 3 is empty, and the environment contains multisets of objects $\{l, l^v | l \in H\} \cup \{a_r | 1 \le r \le m\}$ (each of these objects has an arbitrary number of copies). The program machine M halts corresponding to cell 1 has object l_h and no rule can be applied in system Π. When the designed P system Π halts, the computation result for M is the number of object a_1 deposited in cell 1 at this moment.

1) The following rules in $\mathcal{R}$ are constructed to simulate each ADD instruction l_i of M:

$$\begin{aligned}
&r_{1,i} : (1, l_i/\lambda, 2), && r_{2,i} : \left(l_i|2, l_i^{iv}/\lambda, 0\right)\\
&r_{3,i} : (2, l_i/\lambda, 0), && r_{4,i} : \left(l_i^{iv}|0, a_r/\lambda, 1\right)\\
&r_{5,i} : \left(0, l_i^{iv}/\lambda, 1\right), && r_{6,i} : \left(1, l_i^{iv}/\lambda, 2\right)\\
&r_{7,i} : \left(l_i^{iv}|1, \lambda/l_j, 0\right), && r_{8,i} : \left(l_i^{iv}|1, \lambda/l_k, 0\right).
\end{aligned}$$

At step 1, cell 2 receives object l_i by applying rule $r_{1,i}$. When cell 2 contains object l_i, the environment will obtain object l_i^{iv}; meanwhile, the environment receives object l_i. When the environment contains object l_i^{iv}, cell 1 will receive object l_i^{iv} and one copy of object a_r (as flat maximal parallelism). Next, object l_i^{iv} is sent back to cell 2; meanwhile, the system nondeterministically chooses $r_{7,i}$ and $r_{8,i}$, and using one of these rules, and cell 1 will receive only one copy of label object l_j or l_k as using rules in the flat maximally parallel mode. Hence, during the process, the output cell increases one copy of object a_r, which will simulate the next instruction l_j or l_k.

1) The following rules in $\mathcal{R}$ are constructed to simulate each SUB instruction l_i of M:

$$\begin{aligned}
&r_{9,i} : (1, l_i/\lambda, 2), && r_{10,i} : \left(l_i|2, l_i^{iv}/\lambda, 0\right)\\
&r_{11,i} : (2, l_i/\lambda, 0), && r_{12,i} : \left(1, \lambda/l_i^{iv}, 0\right)\\
&r_{13,i} : \left(l_i^{iv}|1, l_i'/\lambda, 2\right), && r_{14,i} : \left(l_i^{iv}|1, \lambda/l_i^v, 0\right)\\
&r_{15,i} : \left(1, l_i^{iv}/\lambda, 2\right), && r_{16,i} : \left(l_i'|2, \lambda/a_r, 1\right)\\
&r_{17,i} : \left(2, l_i'/\lambda, 0\right), && r_{18,i} : \left(1, l_i^v/\lambda, 2\right)\\
&r_{19} : (a_r|2, d/\lambda, 0), && r_{20,i} : \left(l_i^v|2, \lambda/l_i'', 1\right)\\
&r_{21,i} : \left(l_i^v|2, \lambda/l_i''', 1\right), && r_{22,i} : \left(1, \lambda/l_i', 0\right)\\
&r_{23,i} : \left(2, l_i^v/\lambda, 0\right), && r_{24,i} : \left(l_i''|2, a_r/\lambda, 0\right)\\
&r_{25,i} : \left(a_r|2, l_i''/\lambda, 3\right), && r_{26,i} : \left(a_r|2, l_i'''/\lambda, 0\right)\\
&r_{27} : (1, \lambda/d, 0), && r_{28,i} : \left(d|2, l_i''/\lambda, 0\right)\\
&r_{29,i} : \left(d|2, l_i'''/\lambda, 3\right), && r_{30,i} : \left(1, \lambda/l_i''', 0\right)\\
&r_{31,i} : \left(1, \lambda/l_i'', 3\right), && r_{32} : (1, d/\lambda, 2)\\
&r_{33,i} : \left(l_i'''|0, l_j/\lambda, 1\right), && r_{34,i} : \left(1, \lambda/l_i'', 0\right)\\
&r_{35,i} : \left(1, \lambda/l_i''', 3\right), && r_{36,i} : \left(l_i''|0, l_k/\lambda, 1\right).
\end{aligned}$$

By the application of rules $r_{9,i}$, $r_{10,i}$, $r_{11,i}$, and $r_{12,i}$, the environment will receive object l_i^{iv} from cell 2, this object will then be sent to cell 1. When cell 1 contains object l_i^{iv}, cell 2 will receive object l_i' from cell 1, and cell 1 receives only one copy of object l_i^v; meanwhile, cell 2 receives object l_i^{iv}. Next, we have two cases to check whether cell 1 contains object a_r or not.

1) Cell 1 contains object a_r. Rules $r_{16,i}$, $r_{17,i}$, and $r_{18,i}$ are applied in parallel, object l_i^v is sent to cell 2, and the environment will receive this object later; besides, the environment and cell 1 will receive object l_i' in turn; and cell 2 receives one copy of object a_r. When cell 2 contains object a_r, the environment, cell 1, and cell 2 will receive object d in turn. If cell 2 contains object l_i^v, cell 2 obtains objects l_i'' and l_i'''. Next, the environment will receive object a_r; if cell 2 contains object a_r, object l_i'' is sent to cell 3, and the environment receives

object l_i'''. At step 8, objects l_i'' and l_i''' are sent to cell 1 and the environment, respectively; meanwhile, when the environment contains object l_i''', cell 1 will receive one copy of object l_j. Hence during the process, the output cell decreases one copy of object a_r (corresponding to subtract one from register r), which will simulate the next instruction l_j.

2) Cell 1 does not contain object a_r. Only rules $r_{17,i}$ and $r_{18,i}$ are applied in parallel at step 5, cell 2 receives object l_i^v, and the environment will receive object l_i^v; in parallel, the environment receives object l_i', and then cell 1 will obtain this object. If cell 2 contains object l_i^v, this cell will obtain objects l_i'' and l_i''', and the environment and cell 3 will receive this object, respectively, (when object d still presents in cell 2). Next, cell 1 will obtain objects l_i'' and l_i'''; meanwhile, when the environment contains object l_i'', cell 1 will receive object l_k. So when the simulation is finished, the system is passed to simulate the next instruction l_k.

So an SUB instruction can be effectively simulated for both cases.

The computation halts only when cell 1 contains object l_h. The number of object a_1 stored in cell 1 at this moment represents the computational result of M. Hence, $N(M) = N(\Pi)$, and this concludes the proof. ■

V. Solving SAT Problem by Monodirectional Tissue P Systems With Promoters and Cell Division

In this section, the SAT problem is efficiently solved by monodirectional tissue P systems with cell division by using a maximal length 1 and at most two promoters for each symport rule or using a maximal length 2 and at most one promoter for each symport rule, working in the flat maximally parallel mode.

The *propositional* SAT problem is defined as follows: determining whether there exists an assignment to variables that satisfies a given propositional formula or not. The SAT problem was proved to be *NP*-complete problem [24].

Consider a propositional formula $\varphi = C_1 \wedge \cdots \wedge C_m$ such that $C_i = y_{i,1} \vee \cdots \vee y_{i,p_i}$, $p_i \geq 1$, $y_{i,j} \in \{x_k, \neg x_k | 1 \leq k \leq n\}$, $1 \leq i \leq m$, $1 \leq j \leq p_i$, and $\vee$ and $\wedge$ represent *or* and *and*, respectively.

Theorem 6: $\mathtt{SAT} \in \mathbf{PMC}_{\mathbf{MTPDS}(\mathrm{pro}_2, \mathrm{sym}_1, f\max)}$.

Proof: A uniform solution to the propositional SAT problem is given by a family of recognizer monodirectional tissue P systems with promoters and cell division $\Pi = \{\Pi(t) | t \in \mathbb{N}\}$, where each system $\Pi(t)$ $(t = \langle n, m\rangle = ((n+m)(n+m+1)/2)+n)$ with the input multiset $cod(\varphi)$ will process all Boolean formulas φ, which have m clauses and n variables.

The recognizer monodirectional tissue P system with promoters and cell division is constructed as follows:

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

where

1) $\Gamma = \Sigma \cup \mathcal{E} \cup \{a_1, p, \mathtt{yes}, \mathtt{no}\}$;
2) $\Sigma = \{x_{i,j}, \bar{x}_{i,j} | 1 \leq i \leq n, 1 \leq j \leq m\}$;
3) $\mathcal{E} = \{a_i | 2 \leq i \leq n\} \cup \{b_j, c_j | 1 \leq j \leq m\} \cup \{\beta_i | 0 \leq i \leq 2n+m+3\} \cup \{b_{m+1}\}$;
4) $\mathcal{M}_1 = \{a_1\}$, $\mathcal{M}_2 = \{p, \beta_0, \mathtt{yes}, \mathtt{no}\}$, $\mathcal{M}_3 = \emptyset$;
5) $i_{in} = 1$ and $i_{\text{out}} = 0$ are the input cell and the output region, respectively;
6) The finite set of symport rules and division rules in $\mathcal{R}$ is constructed as follows:

$$\begin{aligned}
&r_{1,i} : [\ a_i\ ]_1 \to [\ t_i\ ]_1 [\ f_i\ ]_1, && 1 \leq i \leq n\\
&r_{2,i,j} : (t_i x_{i,j} | 1, \lambda/c_j, 0), && 1 \leq i \leq n, 1 \leq j \leq m\\
&r_{3,i,j} : (f_i \bar{x}_{i,j} | 1, \lambda/c_j, 0), && 1 \leq i \leq n, 1 \leq j \leq m\\
&r_{4,i} : (t_i | 1, \lambda/a_{i+1}, 0), && 1 \leq i \leq n\\
&r_{5,i} : (f_i | 1, \lambda/a_{i+1}, 0), && 1 \leq i \leq n\\
&r_{6,i} : (1, t_i/\lambda, 2), && 1 \leq i \leq n\\
&r_{7,i} : (1, f_i/\lambda, 2), && 1 \leq i \leq n\\
&r_8 : (a_{n+1} | 1, \lambda/b_1, 0)\\
&r_9 : (1, a_{n+1}/\lambda, 2)\\
&r_{10,j} : (b_j c_j | 1, \lambda/b_{j+1}, 0), && 1 \leq j \leq m\\
&r_{11,j} : (1, b_j/\lambda, 2), && 1 \leq j \leq m\\
&r_{12} : (1, b_{m+1}/\lambda, 2)\\
&r_{13} : (b_{m+1} | 2, \mathtt{yes}/\lambda, 3)\\
&r_{14} : (b_{m+1} | 2, p/\lambda, 3)\\
&r_{15} : (3, \mathtt{yes}/\lambda, 0)\\
&r_{16,i} : (\beta_i | 2, \lambda/\beta_{i+1}, 0), && 0 \leq i \leq 2n+m+2\\
&r_{17,i} : (2, \beta_i/\lambda, 3), && 0 \leq i \leq 2n+m+2\\
&r_{18} : (p\beta_{2n+m+3} | 2, \mathtt{no}/\lambda, 3)\\
&r_{19} : (3, \ \mathtt{no}/\lambda, 0).
\end{aligned}$$

The P system $\Pi(\langle n, m\rangle)$ that solved the SAT problem consists of three stages: 1) the generation stage; 2) the checking stage; and 3) the output stage. We remark that during the computational process, a counter object β is used for counting the computation steps (by using rule $r_{16,i}$); that is, for each computation step, the subscript of β is increased by 1.

Generation Stage: In this stage, by applying division rules, all truth assignments for the formula $\varphi(x_1, \ldots, x_n)$ will be produced (see Fig. 3). Meanwhile, the system checks the value of all clauses by the corresponding truth assignment. This stage consists of n iterations, and two steps are consumed for each iteration. So this stage takes $2n$ steps.

At step $i = 1$ for $1 \leq i \leq n$ iterations, by using a rule of $r_{1,i}$, two cells with the same label are obtained from dividing a cell with label 1, where objects t_i and f_i are distributed in each of these two cells, respectively.

At step $i = 2$ for $1 \leq i \leq n$ iterations, rules $r_{2,i,j}$, $r_{3,i,j}$, $r_{4,i}$, $r_{5,i}$, $r_{6,i}$, and $r_{7,i}$ are used in parallel. When objects t_i and $x_{i,j}$ (respectively, f_i and $\bar{x}_{i,j}$) appear in cell 1, objects a_{i+1} and c_j are introduced into that cell 1; meanwhile, by using rules $r_{6,i}$ and $r_{7,i}$, objects t_i and f_i in cells 1 are sent to cell 2. The effect of rules $r_{6,i}$ and $r_{7,i}$ is to ensure that objects t_i and f_i appear in cell 1 only in one step, so only one copy of object a_{i+1} and one copy of object c_j are introduced into a cell due to the flat maximal parallelism.

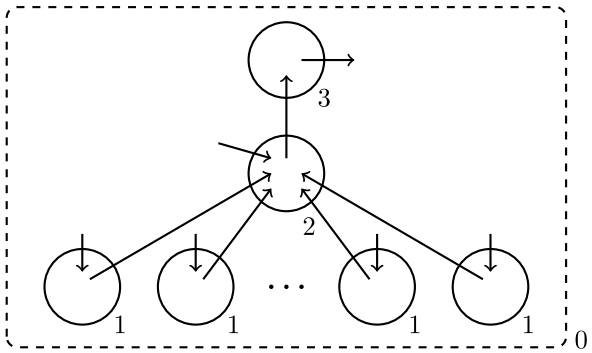

Fig. 3. Membrane structure of the constructed monodirectional tissue *P* system with promoters and cell division at the moment when the generation stage completes, where circles represent cells, numbers on the side of the circles represent labels of cells, all cells are placed in the environment with label 0, and arrows indicate directions that objects are moved between two regions.

Checking Stage: The system starts to check whether or not φ assesses true by some truth assignments. Specifically, if all objects $c_1, \ldots, c_m$ exist in cell 1, then it means in that cell the truth assignment satisfies all clauses, so φ assesses to TRUE; if every cell 1 does not contain all the objects $c_1, \ldots, c_m$, then it means in each cell with label 1, the truth assignment does not satisfy at least one clause, so φ assesses to FALSE.

This stage begins at step $2n+1$, which costs $m+1$ steps.

At step $2n+1$, when cell 1 contains object a_{n+1}, cell 1 will obtain object b_1; meanwhile, cell 2 receives object a_{n+1} from each cell 1 by using rule r_9.

At step $2n+1+j$ ($1 \le j \le m$), the system checks whether object c_j exists in each cell 1. With the appearance of objects b_j and c_j in cell 1, this cell will receive object b_{j+1}; meanwhile, cell 2 receives object b_j. We remark that if cell 1 presents object b_{j+1}, it means clauses $C_1, \ldots, C_j$ are true according to the truth assignment assigned in that cell 1.

Output Stage: In the output stage, the computation result is sent to the output region (i.e., the environment).

If the object b_{m+1} appears in cell 1 at step $2n+m+2$, then it means the Boolean formula evaluates to TRUE. Specifically, at step $2n+m+2$, by applying rule r_{12}, cell 2 receives object b_{m+1}. When cell 2 contains object b_{m+1}, cell 3 receives objects p, yes, and the environment will obtain object yes at step $2n+m+4$. The system halts and the computation result is *affirmative*.

If every cell 1 does not contain object b_{m+1}, then it means the Boolean formula evaluates to FALSE. Specifically, at step $2n+m+2$, rules r_{16} and r_{17} are used in parallel, and object β_{2n+m+2} is sent into cell 2, which will be evolved to β_{2n+m+3} at the next step. When cell 2 contains objects p, β_{2n+m+3}, cell 3 and the environment will obtain object no in turn. The system halts and the computation result is *negative*.

In what follows, we give the resources for constructing the system: 1) the size of the alphabet: $2nm+3n+3m+8 \in O(nm)$; 2) the number of cells initially needed in the system: $3 \in O(1)$; 3) the number of objects initially needed in the system: $5 \in O(1)$; 4) the total number of rules in the system: $2nm+9n+4m+14 \in O(nm)$; and 5) the maximum length of a rule in the system (promoters are not included): $1 \in O(1)$. Therefore, a Turing machine exists that can construct the system $\Pi(\langle n, m\rangle)$ in polynomial time.

The designed *P* system $\Pi(\langle n, m\rangle)$ halts at step $2n+m+4$ (the last step) for the output yes or at step $2n+m+5$ (the last step) for the output no. Thus, the system $\Pi(\langle n, m\rangle)$ is polynomially bounded concerning the numbers of clauses and variables for the formula φ.

Therefore, the family of *P* systems Π offers an efficient solution to the SAT problem. ■

Theorem 7: SAT $\in \mathbf{PMC}_{\mathbf{MTPDS}(\mathrm{pro}_1, \mathrm{sym}_2, f\max)}$.

Proof: We design a family of recognizer monodirectional tissue *P* systems with promoters and cell division $\Pi = \{\Pi(t) | t \in \mathbb{N}\}$ solving the SAT problem. Let $\Pi(t)$ ($t = \langle n, m\rangle = ((n+m)(n+m+1)/2)+n$) be such a tissue *P* system [associated with input $cod(\varphi)$], which will deal with all Boolean formulas φ with m clauses and n variables.

We design the recognizer monodirectional tissue *P* system with promoters and cell division as follows:

$$\Pi(\langle n, m\rangle) = (\Gamma, \mathcal{E}, \mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_5, \mathcal{R}, i_{in}, i_{\text{out}})$$

where

1) $\Gamma = \Sigma \cup \mathcal{E} \cup \{b_i, d_i | 1 \le i \le n\} \cup \{b'_i, h_i, h'_i | 1 \le i \le m\} \cup \{g_i | 2 \le i \le n+1\} \cup \{a_1, a'_{n+1}, b_0, d_0, q, z_1, \mathtt{yes}, \mathtt{no}\}$;
2) $\Sigma = \{x_{i,j}, \bar{x}_{i,j} | 1 \le i \le n, 1 \le j \le m\}$;
3) $\mathcal{E} = \{a_i | 2 \le i \le n+1\} \cup \{c_j, \bar{c}_j, p_j, p'_j | 1 \le j \le m\} \cup \{\beta_i | 0 \le i \le 7n+4m+7\}$;
4) $\mathcal{M}_1 = \{b_0, b_1, \ldots, b_n, b'_1, \ldots, b'_m\}$, $\mathcal{M}_2 = \{\beta_0, a_1, g_2, \ldots, g_{n+1}\}$, $\mathcal{M}_3 = \{d_0, d_1, \ldots, d_n, q, \mathtt{yes}, \mathtt{no}\}$, $\mathcal{M}_4 = \{z_1, a'_{n+1}, h_1, \ldots, h_m, h'_1, \ldots, h'_m\}$, and $\mathcal{M}_5 = \emptyset$;
5) $i_{in} = 1$ and $i_{\text{out}} = 0$ are the input cell and the output region, respectively;
6) the finite set of symport rules and division rules in $\mathcal{R}$ is constructed as follows, the computation process consists of the generation stage, the checking stage, and the output stage, and an overview of the computation is presented.

Generation Stage:

$$\begin{aligned}
&r_1 : [\ z_1\ ]_4 \to [\ z'_2\ ]_4[\ z''_2\ ]_4 \\
&r_{2,i} : [\ z'_i\ ]_4 \to [\ z'_{i+1}\ ]_4[\ z''_{i+1}\ ]_4, 2 \le i \le n \\
&r_{3,i} : [\ z''_i\ ]_4 \to [\ z'_{i+1}\ ]_4[\ z''_{i+1}\ ]_4, 2 \le i \le n \\
&r_4 : \left(1, \lambda / z'_{n+1}, 4\right) \\
&r_5 : \left(z'_{n+1} | 1, \lambda / a_1, 2\right) \\
&r_{6,i} : [\ a_i\ ]_1 \to [\ t_i\ ]_1[\ f_i\ ]_1, \qquad 1 \le i \le n \\
&r_{7,i,j} : \left(t_i | 1, x_{i,j} / \lambda, 0\right), \qquad 1 \le i \le n, 1 \le j \le m \\
&r_{8,i,j} : \left(f_i | 1, \bar{x}_{i,j} / \lambda, 0\right), \qquad 1 \le i \le n, 1 \le j \le m \\
&r_{9,i} : (t_i | 1, b_i / \lambda, 0), \qquad 1 \le i \le n \\
&r_{10,i} : (f_i | 1, b_i / \lambda, 0), \qquad 1 \le i \le n \\
&r_{11,i,j} : \left(0, x_{i,j} c_j / \lambda, 2\right), \qquad 1 \le i \le n, 1 \le j \le m \\
&r_{12,i,j} : \left(0, \bar{x}_{i,j} \bar{c}_j / \lambda, 2\right), \qquad 1 \le i \le n, 1 \le j \le m \\
&r_{13,i} : (0, b_i a_{i+1} / \lambda, 4), \qquad 1 \le i \le n \\
&r_{14,i,j} : \left(t_i | 1, \lambda / c_j, 2\right), \qquad 1 \le i \le n, 1 \le j \le m \\
&r_{15,i,j} : \left(f_i | 1, \lambda / \bar{c}_j, 2\right), \qquad 1 \le i \le n, 1 \le j \le m \\
&r_{16,i} : (4, a_i / \lambda, 3), \qquad 2 \le i \le n+1
\end{aligned}$$

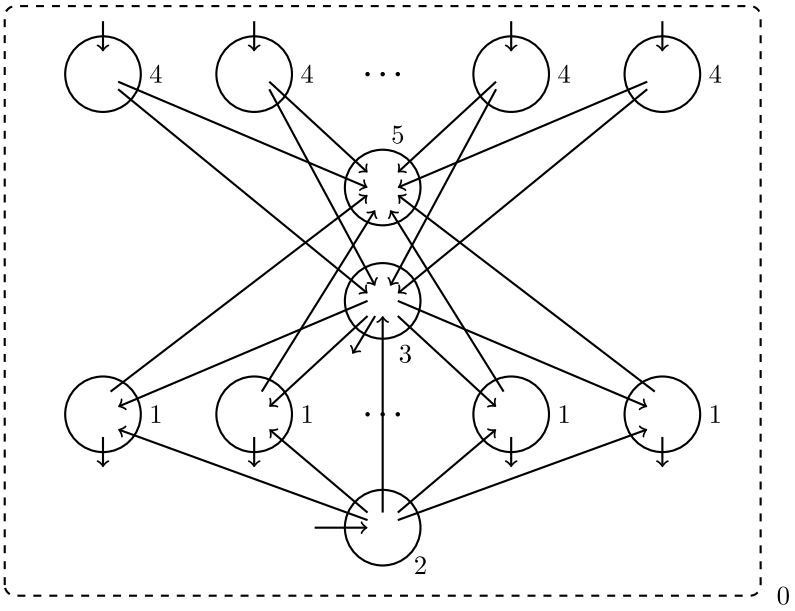

Fig. 4. Membrane structure of the constructed monodirectional tissue P system with promoters and cell division at the moment when the generation phase completes, where circles represent cells, numbers on the side of the circles represent labels of cells, all cells are placed in the environment with label 0, and arrows indicate directions that objects are moved between two regions.

$$
\begin{aligned}
&r_{17,i} : (a_i|3, \lambda/g_i, 2), && 2 \le i \le n+1\\
&r_{18,i} : (a_i|3, d_{i-1}/\lambda, 0), && 2 \le i \le n+1\\
&r_{19,i} : (d_i|0, \lambda/t_i, 1), && 1 \le i \le n\\
&r_{20,i} : (d_i|0, \lambda/f_i, 1), && 1 \le i \le n\\
&r_{21,i} : (g_i|3, a_i/\lambda, 1), && 2 \le i \le n+1.
\end{aligned}
$$

In the generation stage, the system works as follows. At the first n steps, cells with label 4 are divided by using rules r_1, $r_{2,i}$, and $r_{3,i}$, and 2^n cells with label 4 are produced. Cells with label 4 play an auxiliary role, which is used for checking the clauses that are satisfied. At the next two steps, by the sequential application of rules r_4 and r_5, object a_1 will be sent into cell 1 (the auxiliary object z'_{n+1} is sent to cell 1, which is used as a promoter).

In the following steps, the system generates all truth assignments for the variables $x_1, \ldots, x_n$ and checks the clauses that are satisfiable. This process consists n iterations, and six steps are costed for each iteration, so it takes $6n$ steps for this stage.

At step $i = 1$ for $1 \le i \le n$ iterations, by using rule $r_{6,i}$, two copies of cell 1 are produced from dividing cell 1, and each of the generated cell obtains objects t_i and f_i, respectively.

At step $i = 2$ for $1 \le i \le n$ iterations, when cell 1 contains object t_i (respectively, f_i), the environment receives objects $x_{i,j}$ and b_i (respectively, $\bar{x}_{i,j}$ and b_i) by applying rules $r_{7,i,j}$ and $r_{9,i}$ (respectively, $r_{8,i,j}$ and $r_{10,i}$) in parallel.

At step $i = 3$ for $1 \le i \le n$ iterations, cell 2 receives $x_{i,j}, c_j$ (respectively, $\bar{x}_{i,j}, \bar{c}_j$) by applying rule $r_{11,i,j}$ (respectively, $r_{12,i,j}$); meanwhile, cell 4 receives objects b_i and a_{i+1} by using rule $r_{13,i}$. Note that the number of cell 4 is more than cell 1 at this moment, one copy of object b_i and one copy of object a_{i+1} enter one cell with label 4 because of the flat maximal parallelism.

At step $i = 4$ for $1 \le i \le n$ iterations, as using rules in mode of flat maximally parallelism, rule $r_{14,i,j}$ (respectively, $r_{15,i,j}$) is applied, cell 1 that it contains object t_i (respectively, f_i) receives an object c_j (respectively, $\bar{c}_j$). Meanwhile, cell 3 receives object a_i by using rule $r_{16,i}$.

At step $i = 5$ for $1 \le i \le n$ iterations, if object a_i appears in cell 3, such cell will receive object g_i from cell 2, the environment receives object d_{i-1} from cell 3.

At step $i = 6$ for $1 \le i \le n$ iterations, the environment receives objects t_i and f_i; meanwhile, cell 1 receives object a_i.

In general, the generation stage costs $7n + 2$ steps (see Fig. 4).

Checking Stage:

$$
\begin{aligned}
&r_{22} : (1, a_{n+1}c_1/\lambda, 0)\\
&r_{23} : (1, a_{n+1}\bar{c}_1/\lambda, 0)\\
&r_{24,j} : \left(1, p_j c_{j+1}/\lambda, 0\right), && 1 \le j \le m-1\\
&r_{25,j} : \left(1, p_j \bar{c}_{j+1}/\lambda, 0\right), && 1 \le j \le m-1\\
&r_{26} : \left(a_{n+1}|1, b'_1/\lambda, 0\right)\\
&r_{27} : \left(a_{n+1}|1, \lambda/a'_{n+1}, 4\right)\\
&r_{28,j} : \left(p_j|1, b'_{j+1}/\lambda, 0\right), && 1 \le j \le m-1\\
&r_{29,j} : \left(h'_j|1, \lambda/p'_j, 4\right), && 1 \le j \le m-1\\
&r_{30,j} : \left(1, h'_j/\lambda, 0\right), && 1 \le j \le m-1\\
&r_{31} : \left(a_{n+1}|1, a'_{n+1}/\lambda, 0\right)\\
&r_{32,j} : \left(0, b'_j/\lambda, 4\right), && 1 \le j \le m\\
&r_{33,j} : \left(p_j|1, p'_j/\lambda, 0\right), && 1 \le j \le m-1\\
&r_{34,j} : \left(b'_j|4, \lambda/p_j p'_j, 0\right), && 1 \le j \le m\\
&r_{35,j} : \left(4, b'_j h_j/\lambda, 5\right), && 1 \le j \le m\\
&r_{36} : \left(a'_{n+1}|1, \lambda/p_1 h'_1, 4\right)
\end{aligned}
$$

$$r_{37} : \left(h_1|5, \lambda/a'_{n+1}, 1\right)$$
$$r_{38,j} : \left(p'_j|1, \lambda/p_{j+1}h'_{j+1}, 4\right), \qquad 1 \le j \le m-1$$
$$r_{39,j} : \left(h_{j+1}|5, \lambda/p'_j, 1\right), \qquad 1 \le j \le m-1.$$

The system starts to check whether or not φ assesses true by some truth assignments. Specifically, if the set of objects $\{c_1, \bar{c}_1, \ldots, c_m, \bar{c}_m\}$ with the subscript from 1 to m exists in a cell 1, then it means all clauses of the formula are satisfied in that cell, so the formula evaluates to TRUE; if every cell 1 does not contain the set of objects $\{c_1, \bar{c}_1, \ldots, c_m, \bar{c}_m\}$ with the subscript from 1 to m, then it means in each cell with label 1, at least one clause of the formula is not satisfied in that cell, so the formula evaluates to FALSE.

The checking stage consists of m iterations, and four steps are needed for each iteration, thus this stage costs 4 m steps.

At step 1 of the first iteration, rules r_{26}, r_{27}, and r_{22} (or r_{23}) are used, if cell 1 contains an object c_1 or $\bar{c}_1$, then the environment receives objects a_{n+1}, b'_1, and c_1 (or $\bar{c}_1$) from cell 1; meanwhile, cell 1 receives object a'_{n+1} from cell 4. Note that the system has the same number of cell 4 and cell 1, in cell 4 each copy of object a'_{i+1} enters one cell 1 because of the flat maximal parallelism.

At step 2 of the first iteration, if object a_{n+1} still places in a cell 1 (it means object c_1 or $\bar{c}_1$ does not appear in that cell 1), the environment receives object a'_{n+1} by applying rule r_{31}. By using rule $r_{32,1}$, each cell 4 will obtain an object b'_1 because of the flat maximal parallelism.

At step 3 of the first iteration, when cell 4 contains object b'_1, such cell will receive one copy of object p_1 and one copy of object p'_1 from the environment; by using rule $r_{35,j}$, cell 5 receives objects b'_1 and h_1 from cell 4.

At step 4 of the first iteration, if cell 1 still contains object a'_{n+1} (it means that cell 1 has object c_1 or $\bar{c}_1$), rule r_{36} is used, and cell 1 receives objects p_1 and h'_1 from cell 4. Meanwhile, cell 5 receives object a'_{n+1} from cell 1 by using rule r_{37}.

At step $i = 1$ for $2 \le i \le n$ iterations, if cell 1 contains an object c_j or $\bar{c}_j$, then rule $r_{24,j}$ or $r_{25,j}$ is applied, the environment receives objects p_{j-1}, h'_{j-1}, b'_j, and c_j (or $\bar{c}_j$) from cell 1; meanwhile, when cell 1 contains object h'_{j-1}, rule $r_{29,j-1}$ is used, cell 1 receives object p'_{j-1} from cell 4.

At step $i = 2$ for $2 \le i \le n$ iterations, if object p_{j-1} still places in a cell 1 (it means object c_j or $\bar{c}_j$ does not appear in that cell with label 1), rule $r_{33,j-1}$ is used, the environment receives object p'_{j-1}. Meanwhile, a cell with label 4 will obtain an object b'_j due to the flat maximal parallelism.

At step $i = 3$ for $2 \le i \le n$ iterations, rules $r_{34,j}$ and $r_{35,j}$ are used, objects p_j and p'_j are sent to the cell 4 that contains object b'_j, and objects b'_j and h_j in cell 4 are sent to cell 5.

At step $i = 4$ for $2 \le i \le n$ iterations, if cell 1 still contains object p'_{j-1} (it means that cell 1 has object c_j or $\bar{c}_j$), rule $r_{38,j-1}$ is used, cell 1 receives objects p_j and h'_j from cell 4, which contains object p'_{j-1}. Meanwhile, cell 5 receives object p'_{j-1} from cell 1 by using rule $r_{39,j-1}$.

Output Stage:

$$r_{40,i} : (\beta_i|2, \lambda/\beta_{i+1}, 0), \qquad 0 \le i \le 7n + 4m + 6$$
$$r_{41,i} : (2, \beta_i/\lambda, 3), \qquad 0 \le i \le 7n + 4m + 6$$
$$r_{42} : (1, b_0 p_m/\lambda, 0)$$
$$r_{43} : (b_0|0, \lambda/d_0, 3)$$
$$r_{44} : (0, d_0 p_m/\lambda, 2)$$
$$r_{45} : (2, p_m/\lambda, 3)$$
$$r_{46} : (p_m|3, \texttt{yes}q/\lambda, 0)$$
$$r_{47} : (q|3, \beta_{7n+4m+6}\texttt{no}/\lambda, 0).$$

From step 1 to step $7n+4m+7$, a counter object β is used for counting the computation steps (by using rules $r_{40,i}$ and $r_{41,i}$); that is, for each computation step, the subscript of β is increased by one.

If the object p_m appears in cell 1 at step $7n+4m+3$, then it means the Boolean formula evaluates to TRUE. Specifically, by the sequential application of rules r_{42}, r_{43}, r_{44}, and r_{45}, object p_m will appear in cell 3. When cell 3 contains object p_m, rule r_{46} is used, the environment receives objects q, yes. The P system halts and the computation result is *affirmative*.

If every cell 1 does not have object p_m, then it means the Boolean formula evaluates to FALSE. Specifically, from step $7n+4m+3$ to step $7n+4m+7$, only rules $r_{40,i}$ and $r_{41,i}$ are used in parallel, object $\beta_{7n+4m+6}$ will be present in cell 3. At step $7n+4m+8$, if object q still presents in cell 3, rule r_{47} is used, the environment receives object no. The P system halts and the computation result is *negative*.

In what follows, we give the necessary resources for constructing the system: 1) the size of alphabet: $2nm + 11n + 11m + 16 \in O(nm)$; 2) the number of cells initially needed in the system: $5 \in O(1)$; 3) the number of objects initially needed in the system: $3n + 3m + 9 \in O(n + m)$; 4) the total number of rules in the system: $6nm + 26n + 19m + 20 \in O(nm)$; and 5) the maximum length of a rule in the system (promoters is not included): $2 \in O(1)$. Therefore, a Turing machine exists that can construct the system $\Pi(\langle n, m\rangle)$ in polynomial time.

The designed P system $\Pi(\langle n, m\rangle)$ halts and the output is yes at step $7n+4m+7$ (the last step) or no at step $7n+4m+8$ (the last step). Thus, the system $\Pi(\langle n, m\rangle)$ is polynomially bounded concerning the numbers of clauses and variables for the formula φ.

Therefore, the family of P systems Π offers an efficient solution to the SAT problem. ■

VI. Conclusion

On the one hand, the hierarchical architecture of cell-like P systems with symport/antiport rules lacks the capability of complex topology representation; on the other hand, these kinds of cell-like P systems lack the capability of processing monodirectional information. As a result, cell-like P systems with symport/antiport rules are not suitable for a mass of applications which involve monodirectional information representation and complex topology representation, and how to extend cell-like P systems to make them suitable and attractive to some of these applications becomes a crucial issue. In this work, motivated by the monodirectional nature in cellular biology, a novel type of tissue P systems with promoters is proposed, called monodirectional tissue P systems with promoters. Specifically, monodirectional tissue P systems are Turing universal with two cells, a maximal length 2, and at

most one promoter for each symport rule, and working in the maximally parallel mode or with three cells, a maximal length 1 and at most one promoter for each symport rule, and working in the flat maximally parallel mode. We have also proved that monodirectional tissue P systems with two cells, a maximal length 1, and at most one promoter for each symport rule (under certain restrictive conditions) working in the flat maximally parallel mode characterize regular sets of natural numbers. Besides, two efficient solutions to the SAT problem have been provided by such systems working in the flat maximally parallel mode (an obvious open problem is to investigate the computational efficiency of monodirectional tissue P systems with promoters working in the maximally parallel mode).

The advantages of proposing monodirectional tissue P systems with promoters are summarized as follows.

1) Monodirectional tissue P systems with promoters are developed by combining the monodirectional features and network architecture of tissue P systems, thereby being more suitable and attractive to various applications which refer to complex topology representation.
2) A monodirectionality control strategy is introduced into tissue P systems with promoters to control the application of communication rules, thus making monodirectional tissue P systems with promoters be more suitable for some applications which require a monodirectional mechanism.
3) By employing cell division into monodirectional tissue P systems with promoters, the information nature in cells can be replicated, thus making monodirectional tissue P systems with promoters be a more powerful modeling tool to develop various membrane algorithms, which makes training such systems presumable and enhances its potential for practical applications.

Communication for two given regions considered in this work is monodirectional; however, for a single cell, communication may be bidirectional; that is, objects can both enter and exit a cell, for instance, cell 1 or cell 2 in Theorem 3. It is interesting to study the computational power of tissue P systems with promoters in the sense that communication is monodirectional for each region in a computation. (For each region, objects either enter this region or exit this region during a computation.)

Membrane fission is a process by which a biological cell is split into two new ones in the manner that the content of the initial cell is separated and distributed to the new cells [34], [49]. Consequently, the computational efficiency of monodirectional tissue P systems with promoters and cell separation deserves to be investigated.

Several features for rule application inspired by biological or mathematical facts are introduced in membrane computing, such as asynchronism: any number of applicable rules in a system may be used in each computational step [22]; minimal parallelism: at least one rule must be used in a membrane if there exists at least one applicable rule in such membrane [13]; time freeness: the same result is generated by a system which is not related to the times that rules are costed [2], [11], [55]; and sequentiality: exactly one rule is applied in one derivation step [25]. It is interesting to consider the computational power of monodirectional tissue P systems with promoters by using rules in the asynchronous mode, or in the minimal parallel mode, or in the time-free mode, or in the sequential mode.

In [10], an interesting variant of spiking neural P systems, called spiking neural P systems with scheduled synapses, was proposed, where a duration or schedule was added to each synapse, and each synapse was available only in the duration of its schedule. Consequently, it deserves to investigate monodirectional tissue P systems with promoters and with scheduled rules, where duration was added to each rule, and each rule was available only in the duration of its schedule.

Monodirectional tissue P systems with promoters have a network architecture with a stronger capability of complex topology representation than the hierarchical architecture of cell-like P systems. Consequently, the prospect of monodirectional tissue P systems with promoters for applications is optimistic and promising, and the practical applications of monodirectional tissue P systems with promoters are well worth investigating, for example, information acquisition devices for power systems or other real-world problems (e.g., membrane-inspired evolutionary algorithms for multiobjective optimization [12], [65], [66]) that need a networking model and symbolic computing techniques.

REFERENCES

[1] A. Alhazov and R. Freund, "Variants of small universal P systems with catalysts," *Fundamenta Informaticae*, vol. 138, nos. 1–2, pp. 227–250, 2015.

[2] A. Alhazov, R. Freund, S. Ivanov, L. Pan, and B. Song, "Time-freeness and clock-freeness and related concepts in P systems," *Theor. Comput. Sci.*, vol. 805, pp. 127–143, Mar. 2020.

[3] A. Alhazov, R. Freund, A. Leporati, M. Oswald, and C. Zandron, "(Tissue) P systems with unit rules and energy assigned to membranes," *Fundamenta Informaticae*, vol. 74, no. 4, pp. 391–408, 2006.

[4] A. Alhazov, R. Freund, and M. Oswald, *Tissue P Systems With Antiport Rules and Small Numbers of Symbols and Cells* (LNCS 3572). Heidelberg, Germany: Springer, 2005, pp. 100–111.

[5] A. Alhazov, R. Freund, and S. Verlan, *P Systems Working in Maximal Variants of the Set Derivation Mode* (LNCS 10105). Cham, Switzerland: Springer, 2017, pp. 83–102.

[6] A. Alhazov and L. Pan, "Polarizationless P systems with active membranes," in *Grammars*, vol. 7. Heidelberg, Germany: Springer, 2004, pp. 141–159.

[7] A. Alhazov and M. J. Pérez-Jiménez, *Uniform Solution of QSAT Using Polarizationless Active Membranes* (LNCS 4664). Heidelberg, Germany: Springer, 2007, pp. 122–133.

[8] I. Ardelean, D. Díaz-Pernil, M. A. Gutiérrez-Naranjo, F. Peña-Cantillana, R. Reina-Molina, and I. Sarchizian, "Counting cells with tissue-like P systems," in *Proc. 10th Brainstorming Week Membrane Comput.*, Sevilla, Spain, 2012, pp. 69–78.

[9] C. Buiu, C. Vasile, and O. Arsene, "Development of membrane controllers for mobile robots," *Inf. Sci.*, vol. 187, pp. 33–51, Mar. 2012.

[10] F. G. C. Cabarle, H. N. Adorna, M. Jiang, and X. Zeng, "Spiking neural P systems with scheduled synapses," *IEEE Trans. Nanobiosci.*, vol. 16, no. 8, pp. 792–801, Dec. 2017.

[11] M. Cavaliere and D. Sburlan, *Time–Independent P Systems* (LNCS 3365). Heidelberg, Germany: Springer, 2005, pp. 239–258.

[12] J. Cheng, G. Zhang, and T. Wang, "A membrane-inspired evolutionary algorithm based on population P systems and differential evolution for multi-objective optimization," *J. Comput. Theor. Nanosci.*, vol. 12, no. 7, pp. 1150–1160, 2015.

[13] G. Ciobanu, L. Pan, G. Păun, and M. J. Pérez-Jiménez, "P systems with minimal parallelism," *Theor. Comput. Sci.*, vol. 378, pp. 117–130, Jul. 2007.

[14] G. Ciobanu, Gabriel, M. J. Pérez-Jiménez, and G. Păun, *Applications of Membrane Computing*. Berlin, Germany: Springer-Verlag, 2006.

[15] M. À. Colomer, A. Margalida, D. Sanuy, and M. J. Pérez-Jiménez, "A bio-inspired computing model as a new tool for modeling ecosystems: The avian scavengers as a case study," *Ecol. Model.*, vol. 222, no. 1, pp. 33–47, 2011.

[16] D. Díaz-Pernil, M. A. Gutiérrez-Naranjo, M. J. Pérez-Jiménez, and A. Riscos-Núñez, *Solving Subset Sum in Linear Time by Using Tissue P System With Cell Division* (LNCS 4527). Heidelberg, Germany: Springer, 2007, pp. 170–179.

[17] D. Díaz-Pernil, M. A. Gutiérrez-Naranjo, M. J. Pérez-Jiménez, and A. Riscos-Núñez, "A uniform family of tissue *P* system with cell division solving 3-COL in a linear time," *Theor. Comput. Sci.*, vol. 404, nos. 1–2, pp. 76–87, 2008.

[18] D. Díaz-Pernil, M. J. Pérez-Jiménez, A. Riscos-Núñez, and Á. Romero-Jiménez, "Computational efficiency of cellular division in tissue-like membrane systems," *Romanian J. Inf. Sci. Technol.*, vol. 11, no. 3, pp. 229–241, 2008.

[19] R. Freund and A. Păun, "P systems with active membranes and without polarizations," *Soft Comput.*, vol. 9, no. 9, pp. 657–663, 2005.

[20] R. Freund, G. Păun, and M. J. Pérez-Jiménez, "Tissue *P* systems with channel states," *Theor. Comput. Sci.*, vol. 330, no. 1, pp. 101–116, 2005.

[21] P. Frisco, M. Gheorghe, and M. J. Pérez-Jiménez, *Applications of Membrane Computing in Systems and Synthetic Biology*. Cham, Switzerland: Springer, 2013.

[22] P. Frisco, G. Govan, and A. Leporati, "Asynchronous P systems with active membranes," *Theor. Comput. Sci.*, vol. 429, pp. 74–86, Apr. 2012.

[23] M. García-Quismondo, M. Levin, and D. Lobo, "Modeling regenerative processes with membrane computing," *Inf. Sci.*, vol. 381, pp. 229–249, May 2017.

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

[25] O. H. Ibarra, A. Păun, and A. Rodríguez-Patón, "Sequential SNP systems based on min/max spike number," *Theor. Comput. Sci.*, vol. 410, nos. 30–32, pp. 2982–2991, 2009.

[26] M. Ionescu, G. Păun, and T. Yokomori, "Spiking neural P systems," *Fundamenta Informaticae*, vol. 71, nos. 2–3, pp. 279–308, 2006.

[27] N. Klco and M. J. Savage, "Digitization of scalar fields for quantum computing," *Phys. Rev. A*, vol. 99, no. 5, 2019, Art. no. 052335.

[28] S. N. Krishna and R. Rama, "A variant of P systems with active membranes: Solving NP–complete problems," *Romanian J. Inf. Sci. Technol.*, vol. 2, no. 4, pp. 357–367, 1999.

[29] A. Leporati, L. Manzoni, G. Mauri, A. E. Porreca, and C. Zandron, "Monodirectional P systems," *Nat. Comput.*, vol. 15, no. 4, pp. 551–564, 2016.

[30] W. Liu *et al.*, "A fault diagnosis method for power transmission networks based on spiking neural P systems with self-updating rules considering biological apoptosis mechanism," *Complexity*, vol. 2020, Jan. 2020, Art. no. 2462647.

[31] C. Martín-Vide, J. Pazos, G. Păun, and A. Rodriguez-Paton, "Tissue *P* systems," *Theor. Comput. Sci.*, vol. 296, no. 2, pp. 295–326, 2003.

[32] M. L. Minsky, *Computation: Finite and Infinite Machines*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1967.

[33] L. Pan *et al.*, "Aptamer-based regulation of transcription circuits," *Chem. Commun.*, vol. 55, pp. 7378–7381, Jun. 2019.

[34] L. Pan and T.-O. Ishdorj, "P systems with active membranes and separation rules," *J. Univ. Comput. Sci.*, vol. 10, no. 5, pp. 630–649, 2004.

[35] L. Pan, G. Păun, and B. Song, "Flat maximal parallelism in P systems with promoters," *Theor. Comput. Sci.*, vol. 623, pp. 83–91, Apr. 2016.

[36] L. Pan and M. J. Pérez-Jiménez, "Computational complexity of tissue-like P systems," *J. Complexity*, vol. 26, no. 3, pp. 296–315, 2010.

[37] L. Pan, Y. Wang, S. Jiang, and B. Song, "Flat maximal parallelism in tissue *P* systems with promoters," *Romanian J. Inf. Sci. Technol.*, vol. 20, no. 1, pp. 42–56, 2017.

[38] C. H. Papadimitriou, *Computational Complexity*. Reading, MA, USA: Addison–Wesley, 1994.

[39] A. Păun, "On P systems with active membranes," in *Proc. 2nd Int. Conf. Unconventional Models Comput.*, 2000, pp. 187–201.

[40] A. Păun, G. Păun, and G. Rozenberg, "Computing by communication in networks of membranes," *Int. J. Found. Comput. Sci.*, vol. 13, no. 6, pp. 779–798, 2002.

[41] A. Păun and G. Păun, "The power of communication: P systems with symport/antiport," *New Gen. Comput.*, vol. 20, no. 3, pp. 295–305, 2002.

[42] G. Păun, "Computing with membranes," *J. Comput. Syst. Sci.*, vol. 61, no. 1, pp. 108–143, 2000.

[43] G. Păun, *Computing With Membranes: An Introduction*. Berlin, Germany: Springer-Verlag, 2002.

[44] G. Păun, M. J. Pérez-Jiménez, and A. Riscos-Núñez, "Tissue *P* systems with cell division," *Int. J. Comput. Commun. Control*, vol. 3, no. 3, pp. 295–303, 2008.

[45] G. Păun, G. Rozenberg, and A. Salomaa, Eds., *The Oxford Handbook of Membrane Computing*. New York, NY, USA: Oxford Univ. Press, 2010.

[46] H. Peng and J. Wang, "Coupled neural P systems," *IEEE Trans. Neural Netw. Learn.*, vol. 30, no. 6, pp. 1672–1682, Oct. 2019.

[47] M. J. Pérez-Jiménez and A. Riscos-Núñez, "Solving the subset-sum problem by active membranes," *New Gen. Comput.*, vol. 23, no. 4, pp. 339–356, 2005.

[48] M. J. Pérez-Jiménez, A. Romero-Jiménez, and F. Sancho-Caparrini, *The P Versus NP Problem Through Cellular Computing With Membranes* (LNCS 2950). Heidelberg, Germany: Springer, 2004, pp. 338–352.

[49] M. J. Pérez-Jiménez and P. Sosík, "An optimal frontier of the efficiency of tissue *P* systems with cell separation," *Fundamenta Informaticae*, vol. 138, nos. 1–2, pp. 45–60, 2015.

[50] R. Reina-Molina, D. Díaz-Pernil, and M. A. Gutiérrez-Naranjo, "Cell complexes and membrane computing for thinning 2D and 3D images," in *Proc. 10th Brainstorming Week Membrane Comput.*, Sevilla, Spain, 2012, pp. 167–186.

[51] G. Rozenberg and A. Salomaa, Eds., *Handbook of Formal Languages*, vol. 3. Berlin, Germany: Springer, 1997.

[52] B. Song, K. Li, D. Orellana-Martín, L. Valencia-Cabrera, and M. J. Pérez-Jiménez, "Cell-like P systems with evolutional symport/antiport rules and membrane creation," *Inf. Comput.*, to be published. [Online]. Available: https://doi.org/10.1016/j.ic.2020.104542

[53] B. Song and L. Pan, "The computational power of tissue-like P systems with promoters," *Theor. Comput. Sci.*, vol. 641, pp. 43–52, May 2016.

[54] B. Song, M. J. Pérez-Jiménez, G. Păun, and L. Pan, "Tissue *P* systems with channel states working in the flat maximally parallel way," *IEEE Trans. Nanobiosci.*, vol. 15, no. 7, pp. 645–656, Oct. 2016.

[55] B. Song and Y. Kong, "Solution to PSPACE-complete problem using P systems with active membranes with time-freeness," *Math. Probl. Eng.*, vol. 2019, Jun. 2019, Art. no. 5793234.

[56] T. Song, A. Rodríguez-Patón, P. Zheng, and X. Zeng, "Spiking neural P systems with colored spikes," *IEEE Trans. Cogn. Develop. Syst.*, vol. 10, no. 4, pp. 1106–1115, Dec. 2018.

[57] B. Song, C. Zhang, and L. Pan, "Tissue-like P systems with evolutional symport/antiport rules," *Inf. Sci.*, vol. 378, pp. 177–193, Feb. 2017.

[58] T. Song, P. Zheng, D. M. Wong, and X. Wang, "Design of logic gates using spiking neural P systems with homogeneous neurons and astrocytes-like control," *Inf. Sci.*, vol. 372, pp. 380–391, Dec. 2016.

[59] T. Wang *et al.*, "Modeling fault propagation paths in power systems: A new framework based on event SNP systems with neurotransmitter concentration," *IEEE Access*, vol. 7, pp. 12798–12808, 2019.

[60] X. Wang *et al.*, "Design and implementation of membrane controllers for trajectory tracking of nonholonomic wheeled mobile robots," *Integr. Comput-Aided Eng.*, vol. 23, no. 1, pp. 15–30, 2016.

[61] T. Wu, F. D. Bîlbîe, A. Păun, L. Pan, and F. Neri, "Simplified and yet turing universal spiking neural P systems with communication on request," *Int. J. Neural Syst.*, vol. 28, no. 8, 2018, Art. no. 1850013.

[62] T. Wu, A. Păun, Z. Zhang, and L. Pan, "Spiking neural P systems with polarizations," *IEEE Trans. Neural Netw. Learn.*, vol. 29, no. 8, pp. 3349–3360, Aug. 2018.

[63] J. Yang *et al.*, "Entropy-driven DNA logic circuits regulated by DNAzyme," *Nucl. Acids Res.*, vol. 46, no. 16, pp. 8532–8541, 2018.

[64] X. Zhang, L. Pan, and A. Păun, "On the universality of axon P systems," *IEEE Trans. Neural Netw. Learn.*, vol. 26, no. 11, pp. 2816–2829, Nov. 2015.

[65] G. Zhang, M. Gheorghe, L. Pan, and M. J. Pérez-Jiménez, "Evolutionary membrane computing: A comprehensive survey and new results," *Inf. Sci.*, vol. 279, pp. 528–551, Sep. 2014.

[66] G. Zhang, H. Rong, J. Cheng, and Y. Qin, "A population-membrane-system-inspired evolutionary algorithm for distribution network reconfiguration," *Chin. J. Electron.*, vol. 23, no. 3, pp. 437–441, 2014.

[67] X. Zheng, J. Yang, C. Zhou, C. Zhang, Q. Zhang, and X. Wei, "Allosteric DNAzyme-based DNA logic circuit: Operations and dynamic analysis," *Nucl. Acids Res.*, vol. 47, no. 3, pp. 1097–1109, 2019.

Bosheng Song received the Ph.D. degree in control science and engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2015.

He spent 18 months working with the Research Group on Natural Computing, University of Seville, Seville, Spain, from 2013 to 2015. He was a Postdoctoral Researcher with the School of Automation, Huazhong University of Science and Technology, from 2016 to 2019. He is currently an Associate Professor with the College of Information Science and Engineering, Hunan University, Changsha, China. His current research interests include membrane computing and bioinformatics.

Xiangxiang Zeng (Senior Member, IEEE) received the B.S. degree in automation from Hunan University, Changsha, China, in 2005, and the Ph.D. degree in system engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2011.

In 2019, he was with the Department of Computer Science, Xiamen University, Xiamen, China. He is a Yuelu Distinguished Professor with the College of Information Science and Engineering, Hunan University. His main research interests include membrane computing, neural computing, and bioinformatics.

Min Jiang (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees in computer science from Wuhan University, Wuhan, China, in 2001 and 2007, respectively.

From 2005 to 2007, he was a visiting student with the Department of Computer Science, City University of Hong Kong, Hong Kong. He is currently a Professor with the School of Information Science and Engineering, Xiamen University, Xiamen, China. His current research interests mainly include machine learning and computational intelligence.

Mario J. Pérez-Jiménez received the B.Sc. degree in mathematics from Barcelona University, Barcelona, Spain, in 1971, and the Ph.D. degree in mathematics from the University of Seville, Seville, Spain, in 1992.

He is currently a Full Professor with the Department of Computer Science and Artificial Intelligence, University of Seville, where he is the Head of the Research Group on Natural Computing. He has published 17 books in computer science and mathematics, and over 300 scientific papers in international journals (collaborating with researchers worldwide). His main research interests include theory of computation, computational complexity theory, natural computing (DNA computing and membrane computing), bioinformatics and computational modeling for systems biology, and population dynamics.

Dr. Pérez-Jiménez is currently a Numerary Member of the Academia Europaea (Academy of Europe).