Research Article

Solution to PSPACE-Complete Problem Using P Systems with Active Membranes with Time-Freeness

Bosheng Song¹ and Yuan Kong²

¹College of Information Science and Engineering, Hunan University, Changsha 410082, Hunan, China
²College of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao, 266590, China

Correspondence should be addressed to Yuan Kong; kongyuan@sdust.edu.cn

Received 12 May 2019; Revised 13 June 2019; Accepted 18 June 2019; Published 27 June 2019

Academic Editor: Francesca Vipiana

Copyright © 2019 Bosheng Song and Yuan Kong. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

P systems with active membranes are powerful parallel natural computing models, which were inspired by cell structure and behavior. Inspired by the parallel processing of biological information and with the idealistic assumption that each rule is completed in exactly one time unit, P systems with active membranes are able to solve computational hard problems in a feasible time. However, an important biological fact in living cells is that the execution time of a biochemical reaction cannot be accurately divided equally and completed in one time unit. In this work, we consider time as an important factor for the computation in P systems with active membranes and investigate the computational efficiency of such P systems. Specifically, we present a time-free semiuniform solution to the quantified Boolean satisfiability problem (QSAT problem, for short) in the framework of P systems with active membranes, where the solution to such problem is correct, which does not depend on the execution time for the used rules.

1. Introduction

Membrane computing is a member of natural computing that seeks to discover new computational models from the study of biological cells [1, 2]. Since the first computational model was proposed in 1998, many variants of models were constructed [3–7], and all computational models considered in such framework are called P systems. Inspired by the biological structure of cells, there are two kinds of P systems: neural-like P systems [8, 9] or tissue-like P systems [10, 11] (arbitrary graph structure) and cell-like P systems [2, 12] (tree structure). For more information about the area of membrane computing, we refer the reader to [13–18].

A P system with active membranes is one of basic cell-like P systems; all membranes are embedded in a skin membrane [19, 20]. In such P systems, each membrane delimits a region (also called compartment), which contains multisets of objects and evolution rules. If a membrane does not contain any other membrane, then it is called elementary; if a membrane contains at least an inner membrane, then it is called nonelementary. An interesting feature of P systems with active membranes is that there exists one of electrical charge (positive (+), negative (−), or neutral (0)) on each membrane, and such P systems are evolved according to the following types of rules: (a) send-in communication rules, (b) send-out communication rules, (c) evolution rules, (d) division rules, and (e) dissolving rules. When one of such rules is applied, the charge on the involved membrane may be changed.

There are many interesting results obtained by P systems with active membranes and their variants [19, 21, 22]. Because P systems with active membranes contain membrane division rules, which can generate arbitrary numbers of membranes, that is, an exponential workspace in polynomial time has been generated, they provide a way to theoretically solve NP-complete problems or PSPACE-complete problem in feasible time (in polynomial time or in linear time) by a time-space trade-off [23–27]. In [28], the QSAT problem is solved in linear time by P systems with restricted active membranes; however, in [29], the QSAT problem is solved in polynomial time by P systems with active membranes and without polarizations.

Inspired by biological fact that the execution time of a biochemical reaction cannot be accurately divided equally and completed in one time unit, in [30], timed P systems were proposed such that a natural number representing the execution time is associated with each rule. The notion of time-free is also proposed in [30], which is considered to
solve hard computational problems [31–37]. In all these time-free solutions to hard computational problems, the correct computation results are obtained when any execution time is given for each rule. Note that, in [32, 36], the QSAT problem was solved by P systems in a time-free uniform way.

In this work, the notion of time-freeness is incorporated into P systems with active membranes, and a time-free uniform solution to the PSACE-complete problem, QSAT problem, is presented, where the solution to such problem is correct, which does not depend on the execution time for the used rules.

2. Preliminary and Model Description

In this section, we first give the basic conception of formal language theory and the notion of P systems with active membranes [20, 38].

For an alphabet $\Gamma$ (a finite nonempty set of symbols), we denote by $\Gamma^*$ the set of all strings over $\Gamma$, and by $\Gamma^+ = \Gamma^* \setminus \{\lambda\}$ we denote the set of nonempty strings. The length of a string $u$, denoted by $|u|$, is the total number of symbols in the string.

We denote by $(\Gamma, f)$ a multiset $m$ over an alphabet $\Gamma$, where $f$ is a mapping from $\Gamma$ onto $\mathbb{N}$ (the set of natural numbers). If $\Gamma = \{a_1, \ldots, a_k\}$, then the multiset of $m$ is denoted by $m = \{a_1^{f(a_1)}, \ldots, a_k^{f(a_k)}\}$, which can also be represented by $m = a_1^{f(a_1)} \ldots a_k^{f(a_k)}$ or by any permutation of this string.

Definition 1. A P system with active membranes is a tuple
$$\Pi = (O, H, \mu, u_1, \ldots, u_m, R),$$
where

(i) $O$ is an alphabet which includes all objects used in the system;

(ii) $H$ is a label set which marks for each membrane;

(iii) $\mu$ is an initial membrane structure;

(iv) $u_i, 1 \leq i \leq m$, are multisets of objects that are placed in membranes at the initial configuration;

(v) $R$ is a finite set of rules with the following forms:

(a) Object evolution rules: $[a \rightarrow v]_h$, for $h \in H, e \in C, a \in O, v \in O^*$.

(b) Send-in communication rules: $[a]_h \rightarrow [b]_h$, for $h \in H, e_1, e_2 \in C, a, b \in O$.

(c) Send-out communication rules: $[a]_h \rightarrow [b]_h$, for $h \in H, e_1, e_2 \in C, a, b \in O$.

(d) Dissolving rules: $[a]_h \rightarrow b$, for $h \in H, e \in C, a, b \in O$.

(e) Division rules for elementary membranes or nonelementary membranes: $[a]_h \rightarrow [b]_h[e]_h$, for $h \in H, e_1, e_2, e_3 \in C, a, b, c \in O$.

The current membrane structure (including polarization associates with each membrane) and the multisets of objects in each membrane at a moment are considered a configuration of the P system. A P system starts from the initial configuration; using rules in a maximally parallel manner, a sequence of consecutive configurations is archived. If there is no rule used in the system, then halting configuration is reached. The result of a system is encoded by the objects present in the output membrane or emitted from the skin membrane when the system reaches halting configuration.

Next we give the definitions of timed P systems with active membranes and the corresponding recognizer version of such model; time-free solutions to decision problems by such P systems are also introduced [33, 39].

Definition 2. A timed P system with active membranes is a pair $(\Pi, e)$, where $\Pi$ is a P system with active membranes and $e$ is a mapping from the finite set of rules in the system into the set of natural numbers $\mathbb{N}$ (representing the execution times for rules in such system).

A timed P system with active membranes $\Pi(e)$ works in the following way: an external clock is assumed, which marks time-units of equal length; the step $t$ of computation is defined by the period of time that goes from instant $t - 1$ to instant $t$. If a membrane contains a rule $r$ from types (a) – (e) selected to be executed, then execution of such rule takes $e(r)$ time units to complete. Therefore, if the execution is started at instant $j$, the rule is completed at instant $j + e(r)$ and the resulting objects and membranes become available only at the beginning of step $j + e(r) + 1$. When a rule $r$ from types (b) – (e) is started, then the occurrences of symbol-objects and the membrane subject to this rule cannot be subject to other rules from types (b) – (e) until the implementation of the rule completes. At one step, a membrane can be subject to several rules of type (a).

In timed P systems with active membranes, the evolution rules and division rules also follow the “bottom-up” manner (one can refer to [33, 39] for more details).

Definition 3. A recognizer timed P system with active membranes of degree $m \geq 1$ is a tuple $\Pi = (O, H, \mu, u_1, \ldots, u_m, R, e, i_{out})$, such that

(i) the tuple $(O, H, \mu, u_1, \ldots, u_m, R)$ is a P system with active membranes, where alphabet $O$ includes two elements, yes and no;

(ii) $i_{out} = 0$: the output zone is the environment;

(iii) $e$ is a time-mapping of $\Pi$;

(iv) all the computations halt;

(v) either object yes or object no (but not both) must appear in the output region (environment) when the system reaches the halting configuration.

The present work also uses rule starting steps (RS-steps, for short) as the computation steps; a computation step is called a RS-step if at this step at least one rule starts its execution [33, 39].

A decision problem $X$ is a pair $(X, \Theta_X)$ such that $X$ is a language over a finite alphabet (whose elements are called instances) and $\Theta_X$ is a total Boolean function (i.e., predicate) over $I_X$. 

Definition 4. Let \( X = (I_X, \Theta_X) \) be a decision problem. We say that \( X \) is solvable in time-free polynomial RS-steps by a family of recognizer P systems with active membranes \( \Pi = \{ \Pi_u : u \in I_X \} \) if the following items are true:

(i) the family \( \Pi \) is polynomially uniform by a Turing machine; that is, there exists a deterministic Turing machine working in polynomial time which constructs the system \( \Pi_u \) from the instance \( u \in I_X \).

(ii) the family \( \Pi \) is time-free sound (with respect to \( X \)); that is, for any time-mapping \( e \), the following property holds: if for each instance of the problem \( u \in I_X \) such that there exists an accepting computation of \( \Pi_u(e) \), we have \( \Theta_X(u) = 1 \).

(iii) the family \( \Pi \) is time-free complete (with respect to \( X \)); that is, for any time-mapping \( e \), the following property holds: if for each instance of the problem \( u \in I_X \) such that \( \Theta_X(u) = 1 \), every computation of \( \Pi_u(e) \) is an accepting computation.

(iv) the family \( \Pi \) is time-free polynomially bounded; that is, there exists a polynomial function \( p(n) \) such that, for every time-mapping \( e \), and for each \( u \in I_X \), all the computations in \( \Pi_u(e) \) halt in, at most, \( p(|u|) \) RS-steps.

We also say that the family \( \Pi \) provides an efficient time-free semiuniform solution to the decision problem \( X \).

3. A Time-Free Solution to QSAT Problem by P Systems with Active Membranes

Quantified Boolean formula problem (QBF, in short) is a well-known PSPACE-complete problem [40]; it asks whether a quantified sentential form over a set of Boolean variables is true or false. For more details about QBF, one can refer to [28].

Theorem 5. A family of P systems with active membranes using rules of types (a), (b), (c), and (e') can solve the QSAT problem in polynomial RS-steps in a time-free manner, where the solution to such problem is correct, which does not depend on the execution time for the used rules.

Proof. Let us consider a quantified Boolean formula:

\[
\varphi = \exists x_1 \forall x_2 \ldots \exists x_{2n-1} \forall x_{2n} (C_1 \land C_2 \land \ldots \land C_m),
\]

\[
C_i = y_{i,1} \lor \cdots \lor y_{i,m}, \quad 1 \leq i \leq m,
\]

where

\[
y_{i,k} \in \{x_j, \neg x_j \mid 1 \leq j \leq 2n\}, \quad 1 \leq i \leq m, \quad 1 \leq k \leq l_i.
\]

We construct the following P system with active membranes:

\[
\Pi = (O, H, \mu, w_1, w_2, \ldots, w_{2n+1}, R, i_{out}),
\]

with the following components:

\[
O = \{ q_i^{(1)} 1 \leq i \leq a_1 2, m q_i^{(2)}, t_i, f_i \}
\]

\[
\cup \{ q_i 1 \leq q_i 2, r m, r_i \}
\]

\[
\cup \{ yes, no, a_{2m+1}, c_{m+1}, s, t \}
\]

\[
H = \{ 1, 2, \ldots, 2n + m + 1 \},
\]

\[
\mu = \left[ \left[ \left[ \cdots \left[ \left[ \left[ 0, 1 \ldots 0 \right] \right] \right] \right] \right] \right]_{2n+1},
\]

\[
w_i = a_1,
\]

\[
w_i = \lambda, \quad 2 \leq i \leq 2n + m,
\]

\[
w_{2n+1} = no,
\]

\[
i_{out} = 0.
\]

The rules designed for solving the QSAT problem are divided into four parts: generation phase, checking phase, quantifier phase, and output phase. In what follows, we give the set \( R \) and its explanation. Let \( e \) be any time-mapping from \( R \) to \( \mathbb{N} \) (set of natural numbers representing the execution times for the rules).

Generation Phase

\[
G_{1,1} : [a_i^0] \rightarrow [t_i^0], 1 \leq i \leq 2n.
\]

\[
G_{2,1} : [t_i^0] \rightarrow [t_i^0], 1 \leq i \leq 2n-1, i + 1 \leq j \leq 2n.
\]

\[
G_{3,1} : [f_i^0] \rightarrow [f_i^0], 1 \leq i \leq 2n-1, i + 1 \leq j \leq 2n.
\]

\[
G_{4,1} : [s_i \rightarrow r_{a_{1,1}} \ldots r_{a_{1,m}} a_{1}^{(1)}], 1 \leq i \leq 2n, \text{ and the clauses } C_{h_{a_{1,1}}} \ldots C_{h_{a_{1,m}}} \text{ contain the literal } x_{i}.
\]

\[
G_{5,1} : [f_i \rightarrow r_{a_{1,1}} \ldots r_{a_{1,m}} a_{1}^{(2)}], 1 \leq i \leq 2n, \text{ and the clauses } C_{h_{a_{1,1}}} \ldots C_{h_{a_{1,m}}} \text{ contain the literal } \neg x_{i}.
\]

\[
G_{6,1} : [a_{1}^{(1)}] \rightarrow [a_{1}^{(1)}], 1 \leq i \leq 2n+1, i + 1 \leq j \leq 2n.
\]

\[
G_{7,1} : [a_{2}^{(2)}] \rightarrow [a_{2}^{(2)}], 1 \leq i \leq 2n+1, i + 1 \leq j \leq 2n.
\]

\[
G_{8,1} : [a_{1}^{(2)}] \rightarrow [a_{2}^{(1)}], 1 \leq i \leq 2n+1, i + 1 \leq j \leq 2n.
\]

\[
G_{9,1} : [a_{1}^{(1)}] \rightarrow [a_{2}^{(2)}], 1 \leq i \leq 2n+1, 1 \leq i \leq 2n.
\]

\[
G_{10,1} : [a_{1}^{(1)}] \rightarrow [a_{2}^{(2)}], 1 \leq i \leq 2n+1, 1 \leq i \leq 2n.
\]

\[
G_{11,1} : [a_{1}^{(2)}] \rightarrow [a_{1}^{(3)}], 1 \leq i \leq 2n+1.
\]

\[
G_{12,1} : [a_{1}^{(3)}] \rightarrow a_{1,1} a_{1,2} a_{1,3}, 1 \leq i \leq 2n.
\]

\[
G_{13,1} : [a_{1}^{(3)}] \rightarrow a_{1,1} a_{1,2} a_{1,3}, 2 \leq i \leq 2n.
\]
Initially, we have object $a_i$ in membrane 1, which corresponds to variable $x_j$, and object no in membrane $2n + m + 1$. At step $i$, rules $G_{1,i}$ and $O_i$ start to be used at the same step, but they may complete at different steps. By using rule $G_{1,i}$, membrane 1 is divided, objects $t_j$ (representing the true value true) and $f_j$ (representing the true value false) are generated, and each of the new produced membranes will obtain one of these two objects. Besides, object no exits the membrane $2n + m + 1$ by using rule $O_i$; polarization of membrane $2n + m + 1$ is changed to positive. During the execution of rule $G_{1,i}$, the system takes one RS-step.

Rules $G_{2,i}$ and $G_{12,i}$ start to be used at the same step only when rule $G_{1,i}$ completes; however, due to the fact that the execution for rules $G_{2,i}$ and $G_{12,i}$ may take different times, these two rules may complete at different steps. By using rule $G_{2,i}$ (resp., $G_{12,i}$), object $t_j$ (resp., $f_j$) enters membrane 2. After the execution of rule $G_{2,i}$ (resp., $G_{3,12,i}$), system starts to use rule $G_{2,13}$ (resp., $G_{3,13,i}$); object $t_j$ (resp., $f_j$) enters membrane 3. In this way, rules $G_{2,1,j}$ (resp., $G_{3,1,j}$) $(2 \leq j \leq 2n)$ will be applied one by one, and object $t_j$ (resp., $f_j$) will be transferred to membrane $2n$.

If membrane $2n$ has object $t_j$ (resp., $f_j$), rule $G_{4,i}$ (resp., $G_{5,1}$) starts to apply; these rules are used to look for the clauses satisfied by the truth-assignment true (resp., false) of variable $x_j$. Note that rules $G_{4,i}$ and $G_{5,1}$ may start to be used at different steps.

If membrane $2n$ contains object $a_i^{(1)}$ (resp., $a_i^{(2)}$), then rule $G_{6,1,2a,n}$ (resp., $G_{7,1,2a,n}$) is used; object $a_i^{(1)}$ (resp., $a_i^{(2)}$) exits membrane $2n$. When membrane $2n - 1$ contains object $a_i^{(1)}$ (resp., $a_i^{(2)}$), rule $G_{6,1,2n-1}$ (resp., $G_{7,1,2n-1}$) starts to apply; object $a_i^{(1)}$ (resp., $a_i^{(2)}$) exits membrane $2n - 1$. In this way, rules $G_{6,1,j}$ (resp., $G_{7,1,j}$) $(2 \leq j \leq 2n)$ will be applied one by one, and object $a_i^{(1)}$ (resp., $a_i^{(2)}$) will be transferred to membrane 1. Note that rules $G_{6,1,2a}$ and $G_{7,1,2a}$ may start at different steps; rules $G_{6,1,j}$ and $G_{7,1,j}$ $(2 \leq j \leq 2n - 1)$ may start at different steps.

When membrane 1 contains object $a_i^{(1)}$ (resp., $a_i^{(2)}$), system starts to use rule $G_{8,i}$ (resp., $G_{9,i}$); object $a_i^{(1)}$ (resp., $a_i^{(2)}$) exits membrane 1, and the polarization of membrane $i$ is changed from neutral to positive (resp., negative). Note that rules $G_{8,1}$ and $G_{9,1}$ may start to be used at different steps. Rule $G_{10,i}$ can be applied only when membrane $2n + m + 1$ has object $a_i^{(1)}$, and there is a membrane 1 having polarization negative; rule $G_{10,1}$ can be applied only when both rules $G_{8,1}$ and $G_{9,1}$ are completed. By applying rule $G_{10,1}$, object $a_i^{(1)}$ evolves to $a_i^{(3)}$, and object $a_i^{(5)}$ enters membrane 1; polarization of membrane 1 is changed to positive. Hence, rule $G_{10,1}$ has a synchronization function because $\Sigma_{2 \leq j \leq 2n} e(G_{2,1,j}) + e(G_{4,1}) + \Sigma_{2 \leq j \leq 2n} e(G_{6,1,j}) + e(G_{8,1})$ may not be equal to $\Sigma_{2 \leq j \leq 2n} e(G_{3,1,j}) + e(G_{5,1}) + \Sigma_{2 \leq j \leq 2n} e(G_{7,1,j}) + e(G_{9,1})$. Hence, after the execution of rule $G_{10,1}$, the system takes at most $8n + 1$ RS-steps.

After the execution of rule $G_{10,1}$, the system starts to use rule $G_{11,1}$; object $a_i^{(3)}$ exits membrane 1. Now we have the following two cases:

(i) The execution of rule $O_i$ has completed; in this case, rule $G_{12}$ starts to be used, the polarization of membrane $2n + m + 1$ changes to positive, and two copies of object $a_i$ are produced. Each membrane 1 will obtain one copy of object $a_i$, and polarization of the corresponding membrane is changed to neutral.

(ii) The system is still at the execution of rule $O_i$; rules $G_{12}$ and $G_{14,2}$ will be applied after the execution of rule $O_i$. After the execution of rule $O_i$, the system starts to use rules $G_{12}$ and $G_{14,2}$.

After the execution of rule $G_{14,2}$, rule $G_{15,2}$ starts to apply; object $a_i$ enters membrane 2. In all membranes 1, the system starts to use rule $G_{15,2}$ at the same step, so rule $G_{15,2}$ in all membranes 1 completes at the same step; that is, in each membrane 2, object $a_i$ is generated at the same step. In general, when membrane 2 has object $a_i$, the system takes at most $8n + 5$ RS-steps.

When membrane 2 has object $a_i$, the system starts to assign truth-assignment of variable $x_j$ and looks for clauses satisfied by such variable.

If membrane 2 has object $a_j$, rule $G_{3,2}$ starts to be used; the system starts to assign truth values true and false to variable $x_j$. In all membranes 1, the system starts and completes rule $G_{13,2}$ at the same step. The system starts to use rules $G_{2,2,3}$ and $G_{13,2}$ at the same step, but these rules may complete at different steps. Besides, the system starts and completes rules $G_{2,2,j}$ $(4 \leq j \leq 2n)$, $G_{3,2,j}$ $(4 \leq j \leq 2n)$, $G_{4,2}, G_{5,2}$, $G_{6,2,j}$ $(3 \leq j \leq 2n)$, $G_{7,2,j}$ $(3 \leq j \leq 2n)$, $G_{8,2}$, and $G_{9,2}$ at different steps. Note that rule $G_{10,2}$ can be used only when all the above rules have completed their executions. Thus, this process takes at most $8n + 1$ RS-steps. So we can deduce that if we consider this process in the worse case, then the system assigns from $x_1$ to $x_{2n-1}$ $(1 \leq i \leq 2n - 2)$; the number of RS-steps decreases by 4. Hence, the system assigns truth-assignment of variable $x_{2n-1}$ and looks for the clauses satisfied by this variable; this process takes at most 13 RS-steps. Therefore, the system takes at most $8n^2 + 14n - 9$ RS-steps for this process.

When membrane 2n contains object $a_{2n}$, the system starts to execute rule $G_{12,2}$. By using rule $G_{12,2}$, objects $t_j$ and $f_j$ are generated, which will be placed in two separate copies of membrane $2n$. Note that rule $G_{12,2}$ in all membranes $2n - 1$ starts and completes at the same step. Rules $G_{2,2}$ and $G_{5,2}$ start to be used at the same step, but they may complete at different steps. Rules $G_{6,2}$ and $G_{9,2}$ may start and complete at different steps. When both rules $G_{8,2}$ and $G_{9,2}$ have been completed, rule $G_{10,2}$ starts to apply: So, rule $G_{10,2}$ has a synchronization function as $e(G_{4,2}) + e(G_{8,2})$ may not be equal to $e(G_{5,2}) + e(G_{9,2})$. If rule $G_{10,2}$ completes, rules $G_{1,12,2}, G_{13,2}$, and $G_{14,2}$ starts to apply one by one. Hence, the system assigns truth-assignment of variable $x_{2n}$ and looks for the clauses satisfied for such variable, this process takes at most 8 RS-steps.

So this phase takes at most $8n^2 + 14n - 9$ RS-steps.
The membrane structure is described in Figure 1 when generation phase finishes.

Checking Phase

\[ C_1 : [a_{2n+1}] \rightarrow [c_1]^{m}_{2n} \]

\[ C_{2,j} : [c_j]^{f_{2n+j}}_{2n+j} \rightarrow [c_j]^{f_{2n+j}}_{2n+j}, 1 \leq j \leq m. \]

\[ C_{3,j} : [r_j]^{f_{2n+j}}_{2n+j} \rightarrow [r_j]^{f_{2n+j}}_{2n+j}, 1 \leq j \leq m. \]

\[ C_{4,j} : [c_j]^{f_{2n+j}}_{2n+j} \rightarrow [c_j]^{f_{2n+j}}_{2n+j}, 1 \leq j \leq m. \]

\[ C_5 : [c_m+1] \rightarrow t \]

After the execution of rule \( C_{2,1} \), object \( c_1 \) evolves to \( c'_1 \) and object \( c'_1 \) enters membranes \( 2n + 1 \); polarization of such membrane is changed from neutral to positive. After the execution of rule \( C_{3,1} \), object \( r_1 \) evolves to \( r'_1 \), and object \( r'_1 \) enters membrane \( 2n + 1 \); polarization of such membrane is changed from positive to neutral. When the charge of membrane \( 2n + 1 \) changes to neutral, rule \( C_{4,1} \) is used, object \( c'_1 \) evolves to \( c_2 \), and object \( c_2 \) exits membrane \( 2n + 1 \). When membrane \( 2n + 1 \) has object \( c_2 \), it means clause \( C_1 \) is satisfied, and the system starts to check whether membrane \( 2n \) has object \( r_2 \) or not.

If membrane \( 2n \) has object \( c_2 \), rule \( C_{2,2} \) starts to be used. When membrane \( 2n \) has object \( r_2 \), object \( c_3 \) will be obtained. If a membrane \( 2n \) does not contain an object \( r_j \), then this membrane will stop evolving when \( C_{3,j} \) is supposed to be used. In general, the system takes at most \( 3m + 1 \) RS-steps.

If there exists a membrane \( 2n \) which contains all objects \( r_1, r_2, \ldots, r_m \), then rule \( C_5 \) starts to apply; object \( c_{m+1} \) evolves to \( t \) in membrane \( 2n \).

This phase takes at most \( 3m + 2 \) RS-steps.

Quantifier Phase

\[ Q_1 : [t]^{0}_{2n} \rightarrow [t]^{0}_{2n} \]

\[ Q_2 : [t]^{0}_{2k} \rightarrow [t]^{0}_{2k}, 1 \leq k \leq n - 1. \]

\[ Q_3 : [t]^{0}_{2k-1} \rightarrow [t]^{0}_{2k-1}, 1 \leq k \leq n. \]

\[ Q_4 : [t]^{+}_{2k-1} \rightarrow [t]^{+}_{2k-1}, 1 \leq k \leq n. \]

After the execution of rule \( C_5 \) (if membranes \( 2n \) have object \( c_{m+1} \)), object \( t \) presents in membranes \( 2n \) at the same step. At that moment, rule \( Q_1 \) starts to be used; object \( t \) (if it exists) exits membranes \( 2n \). Note that in membranes \( 2n \) rule \( Q_1 \) starts and completes at the same step.

The quantifier \( \exists \) is simulated by transferring only one copy of object \( t \) to an upper level membrane; that is, in membrane \( 2k \) \((1 \leq k \leq n - 1)\), there exists at least one copy of object \( t \). Specifically, if membranes \( 2k \) \((1 \leq k \leq n - 1)\) have object \( t \), rule \( Q_2 \) starts to be used; object \( t \) exits membranes \( 2k \), and the polarization of such membrane is changed from neutral to positive. Note that, in each of the immediately lower level membranes, if there exist two copies of object \( t \) in a membrane \( 2k \) \((1 \leq k \leq n - 1)\), then in membranes \( 2k \) two copies of object \( t \) appear at the same step. Obviously, there are \( n - 1 \) levels of membranes \( 2k \) \((1 \leq k \leq n - 1)\), and rule \( Q_3 \) starts to be used at different steps for each level of membranes \( 2k \).

Hence, the simulation of all quantifiers \( \exists \) takes \( n - 1 \) RS-steps.

The quantifier \( \forall \) is simulated by transferring only one copy of \( t \) to an upper level membrane; that is, there are two copies of object \( t \) in membrane \( 2k - 1 \) \((1 \leq k \leq n)\). Specifically, if membranes \( 2k - 1 \) \((1 \leq k \leq n)\) contain two copies of \( t \), rule \( Q_4 \) starts to apply; object \( t \) evolves to \( s \), and object \( s \) exits membrane; the charge of such membrane is changed from neutral to positive. After the execution of rule \( Q_4 \), rule \( Q_5 \) starts to be used, object \( t \) exits this membrane, and the charge is changed from positive to neutral. Note that if each of the immediately lower level membranes of a membrane \( 2k - 1 \) \((1 \leq k \leq n)\) has one copy of \( t \), then two copies of \( t \) appear in membranes \( 2k - 1 \) at the same step. There are \( n \) levels.

Figure 1: The membrane structure of the system \( \Pi \) at the moment when the generation phase completes.
of membranes $2k - 1$ ($1 \leq k \leq n$), and rules $Q_3, Q_4$ start to be used at different steps for each level of the membranes $2k - 1$; thus, the simulation of all quantifiers $V$ takes $2n$ RS-steps.

In general, this phase takes $3n$ RS-steps.

**Output Phase**

\[
\begin{align*}
O_1 &: [\text{no}]^{2m+1} \rightarrow [1]^{2m+1}, \text{no}. \\
O_2 &: [\text{no}]^{2m+1} \rightarrow [\text{no}]^{2m+1}. \\
O_3 &: [t]^{2m+1} \rightarrow [1]^{2m+1}, \text{yes}.
\end{align*}
\]

At step 1, object no is sent out of the system by using rule $O_1$. Note that this operation takes no RS-step. After the quantifier phase, we have the following two cases.

If positive membrane $2n + m + 1$ does not contain object $t$, in this case, rules $O_2, O_3$ cannot be applied. Hence when computation halts, the environment has object no, which means the formula is not satisfiable.

If positive membrane $2n + m + 1$ contains object $t$, in this case, rule $O_3$ will be applied, object $t$ evolves to yes, which will be sent to the environment, and the polarization of membrane $2n + m + 1$ is changed from positive to negative. After the execution of rule $O_3$, rule $O_2$ starts to be used; object no enters membrane $2n + m + 1$. Hence when computation halts, the environment has one copy of yes, which means the formula is satisfiable. The output phase takes two RS-steps.

According to the constructed P systems, for any time-mapping $e: R \rightarrow N$, if the computation halts, object yes (resp., no) appears in the environment if and only if the formula $\varphi$ is satisfiable (resp., not satisfiable). Thus, the system $\Pi$ is time-free sound and time-free complete.

For any time-mapping $e: R \rightarrow N$, the computation takes at most $8r^2 + 17n + 3m + 3$ RS-steps when formula $\varphi$ is satisfiable, and the computation takes at most $8r^2 + 17n + 3m + 1$ RS-steps when formula $\varphi$ is not satisfiable. Therefore, the family $\Pi$ of P systems with active membrane is time-free polynomially bounded.

The family $\Pi = \{\Pi_\varphi \mid \varphi$ is an instance of the QSAT problem} is polynomially uniform:

(i) total number of objects: $12n + 4m + 6$;
(ii) number of initial membranes: $2n + m + 1$;
(iii) cardinality of the initial multiset: $2n + m + 1$;
(iv) total number of rules: $8r^2 + 19n + 3m + 4$;
(v) maximal length of a rule: $m + 4$.

Therefore, P systems with active membranes can solve the QSAT problem in polynomial RS-steps in a time-free manner; this concludes the proof.

\[\square\]

4. Conclusions and Future Work

In this work, a time-free way of using rules is considered into P systems with active membranes, and a time-free solution to the QSAT problem by using P systems with active membranes has been given, where the solution to such problem is correct, which does not depend on the execution time for the used rules.

P systems with active membranes presented in this work are semiuniform; that is, the P systems are designed from the instances of the problem. It remains open how we can construct a family of P systems to solve the QSAT problem in a time-free manner in the sense that P systems are designed from the size of instances.

The P system constructed in Section 3 has the polarization on membranes. It is of interest to investigate whether P systems with active membranes without polarization on membranes can still solve the QSAT problem in a time-free context.

The QSAT problem is one of the most important issues in many application areas, such as artificial intelligence aspect (e.g., planning, nonmonotonic reasoning, scheduling, model checking, and verification formal verification can be reduced to QSAT [41–45]) and fault localization in digital circuits aspect [46–51]. It is interesting to design new algorithms based on QSAT which can be used in the above-mentioned areas and other areas.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61602192 and 61602188), the Fundamental Research Funds for the Central Universities (531118010355), and Scientific Research Foundation of Shandong University of Science and Technology for Recruited Talents (2017RCJJ068 and 2017RCJJ069).

References


