Minimal Cooperation in P Systems with Symport/Antiport: A Complexity Approach

Luis Valencia-Cabrera[1], Bosheng Song[2], Luis F. Macías-Ramos[1], Linqiang Pan[2], Agustín Riscos-Núñez[1], and Mario J. Pérez-Jiménez[1]

[1] Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
Universidad de Sevilla
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
E-mail: { lvalencia, lfmaciasr, ariscosn, marper } @us.es

[2] Key Laboratory of Image Information Processing and Intelligent Control,
School of Automation, Huazhong University of Science and Technology,
Wuhan 430074, Hubei, China
E-mail: boshengsong@163.com, lqpan@mail.hust.edu.cn

Summary. Membrane systems with symport/antiport rules compute by just moving objects among membranes, and not by changing the objects themselves. In these systems the environment plays an active role because, not only it receives objects from the system, but it also sends objects into the system. Actually, in this framework it is commonly assumed that an arbitrarily large number of copies of some objects are initially available in the environment. This special feature has been widely exploited for the design of efficient solutions to computationally hard problems in the framework of tissue like P systems able to create an exponential workspace in polynomial time (e.g. via cell division or cell separation rules).

This paper deals with cell-like P systems which use symport/antiport rules as communication rules, and the role played by the *minimal cooperation* is studied from a computational complexity point of view. Specifically, the limitations on the efficiency of P systems with membrane separation whose symport/antiport rules involve at most two objects are established. In addition, a polynomial time solution to HAM-CYCLE problem, a well known **NP**-complete problem, by using a family of such kind of P systems with membrane division, is provided. Therefore, in the framework of cell-like P systems with minimal cooperation in communication rules, passing from membrane separation to membrane division amounts to passing from tractability to **NP**–hardness.

1 Introduction

The **P** versus **NP** problem is one of the most important open problems in theoretical computer science. Broadly speaking, we can say that this problem analyzes whether or not *finding solutions* is harder than *checking the correctness* of possible

solutions. It is widely believed that it is harder *to solve* a problem than *to check* that a solution is valid/good; that is, it is widely believed that $\mathbf{P} \neq \mathbf{NP}$. The classical approach to solve this problem consists on considering a single **NP**–complete problem and trying to prove whether that problem belongs to the class **P** or not. In the first case, the answer of the conjecture is negative. If the **NP**–complete problem considered does not belong to **P**, then the answer of the conjecture is positive.

In this paper we follow the lines of previous works [3, 4, 5, 6, 8, 10, 13, 14], and new tools to tackle the **P** versus **NP** problem are given in the framework of *Membrane Computing* paradigm. Specifically, we deal with cell-like P systems whose communication is implemented by means of symport/antiport rules abstracting trans-membrane transport of couples of chemical substances, in the same or in opposite directions. Besides, in order to achieve the efficiency of these models, membrane division rules abstracting cell division process and membrane separation rules inspired by membrane fission process, are also included. It is worth pointing out some relevant differences of cell-like approach with respect to tissue-like approach. First, communication rules are not given in a single set within the description of the model, but are organized into subsets, each one of them associated with a membrane label. Second, the structure of the system is a rooted tree given in an explicit way, instead of a directed graph given by means of the set of rules of the system. Third, communication is only produced between inner compartments if they have a parent-child relationship, and the communication with the environment is restricted to the skin membrane. Finally, only elementary membranes can be divided.

In the framework of cell-like P systems which use symport/antiport rules working with minimal cooperation (at most two objects are involved in these rules), we analyze the role played by membrane division and membrane separation as a tool to create an exponential workspace in linear time. On the one hand, we study the limitations on the efficiency of this kind of P systems with membrane separation; that is, we prove that the corresponding polynomial complexity class, denoted by $\mathbf{PMC}_{\mathbf{CSC}(\mathbf{2})}$, is equal to class **P**. On the other hand, we analyze the efficiency of the systems that use membrane division instead of membrane separation, by giving a polynomial time solution to `HAM-CYCLE` problem (that is, showing that $\texttt{HAM-CYCLE} \in \mathbf{PMC}_{\mathbf{CDC}(\mathbf{2})}$). Therefore, in the framework of cell-like P systems with minimal cooperation in communication rules, passing from membrane separation to membrane division amounts to passing from tractability to **NP**–hardness.

The paper is structured as follows. We first recall some preliminaries concerning definitions, concepts and results needed in order to make the paper self-contained. The specific models of cell-like P systems with symport/antiport rules that we use in this work and the corresponding complexity classes are introduced in Section 3.1. Next section is devoted to analyze the limitations about the computational efficiency of P systems with minimal cooperation which use membrane separation rules. Section 5 presents a polynomial time solution of `HAM-CYCLE` problem by means of a family of P systems with membrane division using symport/antiport

rules with length at most 2. Conclusions and some open problems are formulated at the last section.

2 Preliminaries

2.1 Languages and Multisets

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

A *multiset* over an alphabet Γ is an ordered pair (Γ, f), where f is a mapping from Γ onto the set of natural numbers $\mathbb{N}$. For each $x \in \Gamma$ we say that $f(x)$ is the *multiplicity* of x in that multiset. The *support* of a multiset $m = (\Gamma, f)$ is defined as $supp(m) = \{x \in \Gamma \mid f(x) > 0\}$. A multiset is finite if its support is a finite set. The *size* of a finite multiset m, denoted by $|m|$, is the sum of the multiplicities of each element of Γ (obviously that sum is a natural number). We denote by $\emptyset$ the empty multiset. Let us note that a set is a particular case of a multiset where each symbol of the support has multiplicity 1.

Let $m_1 = (\Gamma, f_1)$, $m_2 = (\Gamma, f_2)$ be multisets over Γ. Then, the union of m_1 and m_2, denoted by $m_1 + m_2$, is the multiset (Γ, g), where $g(x) = f_1(x) + f_2(x)$ for each $x \in \Gamma$. We say that m_1 is contained in m_2, and we denote it by $m_1 \subseteq m_2$, if $f_1(x) \leq f_2(x)$ for each $x \in \Gamma$. The relative complement of m_2 in m_1, denoted by $m_1 \setminus m_2$, is the multiset (Γ, g), where $g(x) = f_1(x) - f_2(x)$ if $f_1(x) \geq f_2(x)$, and $g(x) = 0$ otherwise.

2.2 Graphs and Hamiltonian cycles

Let us recall that a *free tree* (*tree*, for short) is a connected, acyclic, undirected graph. A *rooted tree* is a tree in which one of the vertices (called *the root of the tree*) is distinguished from the others. In a rooted tree, the concepts of ascendants and descendants are defined in a usual way. Given a node x (different from the root), if the last edge on the (unique) path from the root of the tree to the node x is $\{x, y\}$ (in this case, $x \neq y$), then y is **the** *parent* of node x and x is **a** *child* of node y. The root is the only node in the tree with no parent. A node with no children is called a *leaf* (see [1] for details).

Let $G = (V, E)$ be a directed graph, where $V = \{1, \ldots, n\}$ and the set of arcs is $E = \{(u_1, v_1), \ldots, (u_m, v_m)\} \subset V \times V$. We say that a finite sequence $\gamma = (u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}, u_{\alpha_{r+1}})$ of nodes of G is a *simple path of G of length* $r \geq 1$ if the following holds:

- $\forall i \ (1 \leq i \leq r \rightarrow (u_{\alpha_i}, u_{\alpha_{i+1}}) \in E)$.
- $|\{u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}\}| = r$.

If $u_{\alpha_{r+1}} \notin \{u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}\}$, then we say that γ is a simple path of length r from u_{α_1} to $u_{\alpha_{r+1}}$. If $u_{\alpha_{r+1}} = u_{\alpha_1}$ and $r \geq 2$, then we say that γ is a *simple cycle* of length r.

A *Hamiltonian path* of G from $a \in V$ to $b \in V$ $(a \neq b)$ is a simple path $\gamma = (u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}, u_{\alpha_{r+1}})$ from a to b such that $a = u_{\alpha_1}$, $b = u_{\alpha_{r+1}}$, and $V = \{u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}, u_{\alpha_{r+1}}\}$. A *Hamiltonian cycle* of G is a simple cycle $\gamma = (u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}, u_{\alpha_{r+1}})$ of G such that $V = \{u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}\}$.

If $\gamma = (u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_r}, u_{\alpha_{r+1}})$ is a simple path of G then we also denote it by the set $\{(u_{\alpha_1}, u_{\alpha_2})_1, (u_{\alpha_2}, u_{\alpha_3})_2, \ldots, (u_{\alpha_r}, u_{\alpha_{r+1}})_r\}$. That is, $(u_{\alpha_k}, u_{\alpha_{k+1}})_k$ can be interpreted as the k-th arc of the path γ, for each k $(1 \leq k \leq r)$.

Let $G = (V, E)$ be a directed graph with $V = \{1, \ldots, n\}$. Throughout this paper, $A_G = \{(i,j)_k \mid i, j, k \in \{1, \ldots, n\} \wedge (i,j) \in E\}$, $A'_G = \{(i,j)'_k \mid (i,j)_k \in A_G\}$ and $A''_G = \{(i,j)''_k \mid (i,j)_k \in A_G\}$.

Proposition 2.1 *Let $G = (V, E)$ be a directed graph such that $V = \{1, \ldots, n\}$. If $B \subseteq A_G$ then the following assertions are equivalent:*

1. *B is a Hamiltonian cycle.*
2. *$|B| = n$ and the following holds: for each $i, i', j, j', k, k' \in \{1, \ldots, n\}$,*
   (a) $[(i,j)_k \in B \wedge (i',j')_{k'} \in B \wedge (i,j)_k \neq (i',j')_{k'} \rightarrow k \neq k']$
   (b) $[(i,j)_k \in B \wedge (i',j')_{k'} \in B \wedge (i,j)_k \neq (i',j')_{k'} \rightarrow i \neq i']$
   (c) $[(i,j)_k \in B \wedge (i',j')_{k'} \in B \wedge (i,j)_k \neq (i',j')_{k'} \rightarrow j \neq j']$
   (d) $[(i,j)_k \in B \wedge (i',j')_{k+1} \in B \rightarrow j = i']$

Proof: Let $B = \{(u_{\alpha_1}, u_{\alpha_2})_1, (u_{\alpha_2}, u_{\alpha_3})_2 \ldots, (u_{\alpha_m}, u_{\alpha_{r+1}})_n\}$ be a Hamiltonian cycle of G. Then, $|B| = n$ and conditions (a), (b), (c) and (d) from (2) hold.

Let $B \subseteq A_G$ such that $|B| = n$ and conditions (a), (b), (c) and (d) from (2) hold. Then, from (a) the set B must to be of the form

$$B = \{(u_{\alpha_1}, v_{\alpha_1})_1, (u_{\alpha_2}, v_{\alpha_2})_2 \ldots, (u_{\alpha_n}, v_{\alpha_n})_n\}$$

where:

- From (d) we deduce that $\forall s$ $(1 \leq s \leq n-1 \rightarrow v_{\alpha_s} = u_{\alpha_{s+1}})$.
- From (b) we have $V = \{u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_n}\}$.

Finally, on the one hand we have $v_{\alpha_n} \in \{u_{\alpha_1}, u_{\alpha_2} \ldots, u_{\alpha_n}\}$. On the other hand, by condition (c) we deduce that $v_{\alpha_n} \notin \{v_{\alpha_1}, \ldots, v_{\alpha_{n-1}}\} = \{u_{\alpha_2}, \ldots, u_{\alpha_n}\}$. Thus, $v_{\alpha_n} = u_{\alpha_1}$.

□

Remark 1: Let $B \subseteq A_G$ be a Hamiltonian cycle of G. For each $i, i', j, j', k, k' \in \{1, \ldots, n\}$ the following holds:

1. If $(i,j)_k \in B$ and $j \neq j'$ then $(i,j')_{k'} \notin B$.
2. If $(i,j)_k \in B$ and $i \neq i'$ then $(i',j)_{k'} \notin B$.
3. If $(i,j)_k \in B$ and $(i,j) \neq (i',j')$ then $(i',j')_k \notin B$.
4. If $(i,j)_k \in B$ and $(i',j')_{k+1} \in B$ then $j = i'$.

Remark 2: Let us notice that if $(u_{\alpha_1}, u_{\alpha_2}, \ldots, u_{\alpha_n}, u_{\alpha_1})$ is a Hamiltonian cycle of G of length n, then we can describe it by the following subset of A_G:

$$B_1 = \{(u_{\alpha_1}, u_{\alpha_2})_1, (u_{\alpha_2}, u_{\alpha_3})_2, \ldots, (u_{\alpha_n}, u_{\alpha_1})_n\}$$

But $(u_{\alpha_2}, u_{\alpha_3}, \ldots, u_{\alpha_m}, u_{\alpha_1}, u_{\alpha_2})$ also represents the same Hamiltonian cycle. It can be described as follows: $B_2 = \{(u_{\alpha_2}, u_{\alpha_3})_1, (u_{\alpha_3}, u_{\alpha_4})_2, \ldots, (u_{\alpha_1}, u_{\alpha_2})_n\}$. Thus, given a Hamiltonian cycle γ of G, there are exactly n different subsets of A_G codifying that cycle.
Remark 3: Let us suppose that the total number of Hamiltonian cycles of G is q. Then, the number of different subsets B of A_G verifying conditions (a), (b), (c), and (d) from Proposition 2.1 is exactly $n \cdot q$.

2.3 Encoding ordered pairs of natural numbers

The *pair function* $\langle n, m\rangle = ((n + m)(n + m + 1)/2) + n$ is a polynomial–time computable function from $\mathbb{N} \times \mathbb{N}$ onto $\mathbb{N}$ which is also a primitive recursive and bijective function.

3 P systems with symport/antiport rules

In this section we introduce a kind of cell-like P systems that use communication rules capturing the biological phenomenon of trans-membrane transport of several chemical substances. Specifically, two processes have been considered. The first one allows a multiset of chemical substances to pass through a membrane in the same direction. In the second one, two multisets of chemical substances, located in different biological membranes, only pass with the help of each other (yielding an *exchange* of objects between both membranes).

Next, we introduce an abstraction of these operations in the framework of P systems with symport/antiport rules following [9]. In these models, the membranes are not polarized.

Definition 1. *A P system with symport/antiport rules* (*SA P system, for short*) *of degree* $q \geq 1$ *is a tuple* $\Pi = (\Gamma, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$, *where:*

1. Γ *is a finite alphabet;*
2. $\mathcal{E} \subseteq \Gamma$*;*
3. Σ *is an (input) alphabet strictly contained in* Γ *such that* $\mathcal{E} \subseteq \Gamma \setminus \Sigma$*;*
4. μ *is a rooted tree whose nodes are injectively labelled by* $1, \ldots, q$ (*the root of the tree is labelled by* 1)*;*
5. $\mathcal{M}_1, \ldots, \mathcal{M}_q$ *are finite multisets over* $\Gamma \setminus \Sigma$*;*
6. $\mathcal{R}_i$, $1 \leq i \leq q$, *are finite sets of communication rules over* Γ *of the form:*
   (*a*) Symport rules*:* (u, out) *or* (u, in), *where* u *is a finite multiset over* Γ *such that* $|u| > 0$*;*

(*b*) Antiport rules*:* $(u, out; v, in)$*, where* u, v *are finite multisets over* Γ *such that* $|u| > 0$ *and* $|v| > 0$*;*

7. $i_{in} \in \{1, \ldots, q\}$ *and* $i_{out} \in \{0, 1, \ldots, q\}$.

A SA P system of degree q $\Pi = (\Gamma, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{out})$ can be viewed as a set of q membranes, labelled by $1, \ldots, q$, arranged in a hierarchical structure μ (given by a rooted tree whose root is called the *skin membrane*), such that: (a) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ represent the finite multisets of objects initially placed into the q membranes of the system; (b) $\mathcal{E}$ is the set of objects initially located in the environment of the system (labelled by 0), all of them available in an arbitrary number of copies; (c) $\mathcal{R}_1, \ldots, \mathcal{R}_q$ are finite sets of communication rules over Γ ($\mathcal{R}_i$ is associated with the membrane i of μ); and (d) i_{out} represents a distinguished *region* which will encode the output of the system. We use the term *region* i ($0 \leq i \leq q$) to refer to membrane i in the case $1 \leq i \leq q$ and to refer to the environment in the case $i = 0$. The length of rule (u, out) or (u, in) (resp. $(u, out; v, in)$) is defined as $|u|$ (resp. $|u| + |v|$).

For each membrane $i \in \{2, \ldots, q\}$ (different from the skin membrane) we denote by $p(i)$ the parent of membrane i in the rooted tree μ. We define $p(1) = 0$, that is, by convention the "parent" of the skin membrane is the environment.

An *instantaneous description* or a *configuration* at an instant t of a SA P system is described by the membrane structure at instant t, all multisets of objects over Γ associated with all the membranes present in the system, and the multiset of objects over $\Gamma \setminus \mathcal{E}$ associated with the environment at that moment. Recall that we assume that there are infinitely many copies of objects from $\mathcal{E}$ in the environment, and hence it does not make sense to keep record of their multiplicity along the computation. The *initial configuration* of the system is $(\mu, \mathcal{M}_1, \ldots, \mathcal{M}_q; \emptyset)$.

A symport rule $(u, out) \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}_t$ at an instant t if membrane i is in $\mathcal{C}_t$ and multiset u is contained in that membrane. When applying a rule $(u, out) \in \mathcal{R}_i$, the objects specified by u are sent out of membrane i into the region immediately outside (the parent $p(i)$ of i), which can be the environment in the case of the skin membrane. A symport rule $(u, in) \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}_t$ at an instant t if membrane i is in $\mathcal{C}_t$ and multiset u is contained in the parent of i. When applying a rule $(u, in) \in \mathcal{R}_i$, the multiset of objects u is taken from the parent membrane of i and enters into the region defined by membrane i.

An antiport rule $(u, out; v, in) \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}_t$ at an instant t if membrane i is in $\mathcal{C}_t$ and multiset u is contained in that membrane, and multiset v is contained in the parent of i. When applying a rule $(u, out; v, in) \in \mathcal{R}_i$, the objects specified by u are sent out of membrane i into the parent of i and, at the same time, the objects specified by v are brought into membrane i.

With respect to the semantics of SA P systems, the rules of such P systems are applied in a non-deterministic maximally parallel manner.

Let Π be a P system with symport/antiport rules. We say that configuration $\mathcal{C}_t$ yields configuration $\mathcal{C}_{t+1}$ in one *transition step*, denoted by $\mathcal{C}_t \Rightarrow_\Pi \mathcal{C}_{t+1}$, if we can pass from $\mathcal{C}_t$ to $\mathcal{C}_{t+1}$ by applying the rules from the system following the

semantics described above. A *computation* of Π is a (finite or infinite) sequence of configurations such that: (a) the first term is the initial configuration of the system; (b) for each $n \geq 2$, the n-th configuration of the sequence is obtained from the previous configuration in one transition step; and (c) if the sequence is finite (called *halting computation*) then the last term is a *halting configuration* (a configuration where no rule of the system is applicable to it). All the computations start from an initial configuration and proceed as stated above; only a halting computation gives a result, which is encoded by the objects present in the output region i_{out} associated with the halting configuration. If $\mathcal{C} = \{\mathcal{C}_t\}_{t<r+1}$ of Π is a halting computation, then the *length* of $\mathcal{C}$, denoted by $|\mathcal{C}|$, is r. For each i $(1 \leq i \leq q)$, we denote by $\mathcal{C}_t(i)$ the finite multiset of objects over Γ contained in all membranes labelled by i (by applying division rules different membranes with the same label can be created) at configuration $\mathcal{C}_t$.

Definition 2. *A P system with symport/antiport rules and membrane division (SAD P system, for short) of degree $q \geq 1$ is a tuple*

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

where:

1. $\Pi = (\Gamma, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ *is a P system with symport/antiport rules of degree q;*
2. $\mathcal{R}_i$, $1 \leq i \leq q$, *are finite sets of rules over Γ of the following types:*
   (*a*) Symport/antiport rules.
   (*b*) Division rules*:* $[\,a\,]_i \rightarrow [\,b\,]_i\,[\,c\,]_i$, *where $a, b, c \in \Gamma$, $i \in \{2, \ldots, q\}$, $i \neq i_{out}$, and i is the label of a leaf of the tree μ;*
3. $i_{in} \in \{1, \ldots, q\}$ *and* $i_{out} \in \{0, 1, \ldots, q\}$.

A SAD P system of degree q is a P system with symport/antiport rules of degree q where membrane division rules (for only elementary membranes) are allowed.

A division rule $[a]_i \rightarrow [b]_i[c]_i \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}_t$ at an instant t if the following holds: (a) membrane i is in $\mathcal{C}_t$; (b) object a is contained in that membrane; and (c) membrane i is elementary, and it is neither the skin membrane nor the output membrane (if $i_{out} \in \{1, \ldots, q\}$). When applying a division rule $[a]_i \rightarrow [b]_i[c]_i$, under the influence of object a, the membrane with label i is divided into two membranes with the same label; in the first copy, object a is replaced by object b, and in the second one, object a is replaced by object c; all the other objects residing in the membrane are replicated, and a copy of each one of them is placed in each of the two new membranes.

With respect to the semantics of SAD P systems, the rules of such P systems are applied in a non-deterministic maximally parallel manner with the following important remark: when a membrane i is divided by a division rule at a computation step, this is the only one from $\mathcal{R}_i$ which can be applied to that membrane at that step. The new membranes resulting from division could participate in the interaction with other membranes or the environment by means of communication rules at the next step – providing that they are not divided once again.

Definition 3. *A P system with symport/antiport rules and membrane separation (SAS P system, for short) of degree $q \geq 1$ is a tuple*

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

where

1. $\Pi = (\Gamma, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ *is a P system with symport/antiport rules of degree q;*
2. $\{\Gamma_0, \Gamma_1\}$ *is a partition of* Γ*, that is,* $\Gamma = \Gamma_0 \cup \Gamma_1$*,* $\Gamma_0, \Gamma_1 \neq \emptyset$*,* $\Gamma_0 \cap \Gamma_1 = \emptyset$*;*
3. $\mathcal{R}_i$*,* $1 \leq i \leq q$*, are finite sets of rules over* Γ *of the following types:*
   (*a*) Symport/antiport rules.
   (*b*) Separation rules: $[\,a\,]_i \rightarrow [\,\Gamma_0\,]_i\,[\,\Gamma_1\,]_i$*, where* $a \in \Gamma$*,* $i \in \{2, \ldots, q\}$*,* $i \neq i_{out}$*, and i is the label of a leaf of the tree;*
4. $i_{in} \in \{1, \ldots, q\}$ *and* $i_{out} \in \{0, 1, \ldots, q\}$.

A SAS P system of degree q is a P system with symport/antiport rules of degree q where membrane separation rules (for only elementary membranes) are allowed.

A separation rule $[a]_i \rightarrow [\Gamma_0]_i[\Gamma_1]_i \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}_t$ at an instant t, if there exists an elementary membrane labelled by i in $\mathcal{C}_t$, different from the skin membrane and from the output membrane, such that it contains an object a. When applying a separation rule $[a]_i \rightarrow [\Gamma_0]_i\,[\Gamma_1]_i \in \mathcal{R}_i$ to a membrane labelled by i in a configuration $\mathcal{C}_t$, that membrane is separated into two membranes with the same label; at the same time, object a is consumed, and the rest of the contents of the membrane are distributed as follows: the objects from Γ_0 are placed in the first membrane, while those from Γ_1 are placed in the second membrane. In this way, several membranes with the same label $i \neq 1$ can be present in the new membrane structure μ' of the system: a new node i and a new arc $(p(i), i)$ are added to μ' each time a membrane separation rule $[a]_i \rightarrow [\Gamma_0]_i[\Gamma_1]_i$ is applied.

With respect to the semantics of these variants, the rules of such P systems are applied in a non-deterministic maximally parallel manner with the following important remark: when a membrane i is separated, the membrane separation rule is the only one from $\mathcal{R}_i$ which is applied for that membrane at that step. The new membranes resulting from separation could participate in the interaction with other membranes or the environment by means of communication rules at the next step – providing that they are not separated once again.

3.1 Recognizer P systems with symport/antiport rules

Recognizer P systems were introduced in [12], and they provide a natural framework to solve decision problems by means of computational devices in membrane computing (i.e., P systems).

Definition 4. *A recognizer P system with symport/antiport rules (and membrane division or membrane separation) of degree $q \geq 1$ is a P system with symport/antiport rules (and membrane division or membrane separation) such that:*

1. *Alphabet Γ has two distinguished symbols* `yes` *and* `no`*;*
2. *initial multisets are finite multisets over $\Gamma \setminus \Sigma$ such that at least one copy of* `yes` *or* `no` *is present in some of them;*
3. *the output region is the environment ($i_{out} = 0$);*
4. *all computations halt;*
5. *if $\mathcal{C}$ is a computation of the system, then either symbol* `yes` *or symbol* `no` *(but not both) must have been released to the environment, and only at the last step of the computation.*

Let us notice that, if a recognizer P system has a symport rule of the type $(u, in) \in \mathcal{R}_1$, then the multiset u must contain some object from $\Gamma \setminus \mathcal{E}$; otherwise there might exist non-halting computations of Π.

We say that a computation $\mathcal{C}$ of a recognizer P system is an *accepting computation* (respectively, *rejecting computation*) if object `yes` (respectively, object `no`) appears in the environment associated with the corresponding halting configuration of $\mathcal{C}$, and neither object `yes` nor `no` appears in the environment associated with any non–halting configuration of $\mathcal{C}$.

We denote by $\mathbf{CDC}(k)$ (respectively, $\mathbf{CSC}(k)$) the class of all recognizer P systems with symport/antiport rules and membrane division (respectively, membrane separation) for elementary membranes such that the length of the communication rules of the system is at most k.

3.2 Polynomial complexity classes of recognizer P systems with symport/antiport rules

Next, according to [11], we define what solving a decision problem by a family of recognizer P systems with symport/antiport rules and membrane division or membrane separation means.

Definition 5. *A decision problem $X = (I_X, \theta_X)$ is solvable in polynomial time by a family $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ of recognizer P systems with symport/antiport rules and membrane division or membrane separation, if the following holds:*

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

– *the family* $\mathbf{\Pi}$ *is complete with regard to* (X, cod, s)*; that is, for each* $u \in I_X$*, if* $\theta_X(u) = 1$*, then every computation of* $\Pi(s(u)) + cod(u)$ *is an accepting one.*

According to Definition 5, we say that the family $\mathbf{\Pi}$ provides a *uniform solution* to the decision problem X. We also say that ordered pair (cod, s) is a polynomial encoding from X in $\mathbf{\Pi}$ and s is the size mapping associated with that solution. It is worth pointing out that, for each instance $u \in I_X$, the P system $\Pi(s(u)) + cod(u)$ is *confluent*, in the sense that all possible computations of the system must give the same answer.

If $\mathbf{R}$ is a class of recognizer P systems, then we denote by $\mathbf{PMC_R}$ the set of all decision problems which can be solved in polynomial time (and in a uniform way) by means of recognizer P systems from $\mathbf{R}$. The class $\mathbf{PMC_R}$ is closed under complement and polynomial–time reductions (see [11] for details). Besides, we have $\mathbf{P} \subseteq \mathbf{PMC_R}$. Indeed, if $X \in \mathbf{P}$, then we consider the family $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ where $\Pi(n) = \Pi(0)$, for each $n \in \mathbb{N}$, and $\Pi(0)$ is a P system from $\mathbf{R}$ of degree 1 containing only two rules $(\texttt{yes}, out)$ and $(\texttt{no}, out)$. Let us consider the polynomial encoding from X in $\mathbf{\Pi}$ defined as follows: (a) $s(u) = 0$, for each $u \in I_X$; and (b) $cod(u) = \texttt{yes}$ if $\theta_X(u) = 1$ and $cod(u) = \texttt{no}$ if $\theta_X(u) = 0$. Then, the family $\mathbf{\Pi}$ solves X according to Definition 5.

4 Computational efficiency of systems in CSC(2)

In this section, we study the limitations on the computational efficiency (ability to solve hard problems in polynomial time) of systems from $\mathbf{CSC}(2)$. Specifically, we show that only problems in class $\mathbf{P}$ can be efficiently solved in polynomial time by means of families of recognizer P systems with membrane separation that use symport/antiport rules involving at most two objects (i.e., with *minimal cooperation*). Hence, we prove that $\mathbf{P} = \mathbf{PMC_{CSC(2)}}$.

Let us first introduce a new representation for the membrane structure of recognizer P systems with membrane separation. Let $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ be a recognizer P system of degree $q \geq 1$ from $\mathbf{CSC}(2)$. In order to identify the membranes created by the application of a separation rule, we modify the labels of the new membranes in the following recursive manner:

- The label of a membrane will be a pair (i, σ), where $1 \leq i \leq q$ and σ is a string over $\{0, 1\}$. At the initial configuration, the labels of the membranes are $(1, \lambda), \ldots, (q, \lambda)$.
- If a separation rule from $\mathcal{R}_i$ is applied to a membrane labelled by (i, σ), then the new created membranes will be labelled by $(i, \sigma 0)$ and $(i, \sigma 1)$, respectively. Membrane $(i, \sigma 0)$ will only contain the objects of membrane (i, σ) which belong to Γ_0, and membrane $(i, \sigma 1)$ will only contain the objects of membrane (i, σ) which belong to Γ_1. The skin membrane cannot be separated, so the label of

the skin membrane, $(1, \lambda)$, is not changed along any computation. Note that we can consider a lexicographical order over the set of labels of cells in the system along any computation.

If a membrane labelled by (i, σ) is engaged by a communication rule, then, after the application of the rule, the membrane keeps its label.

A configuration at an instant t of a P system from $\mathbf{CSC}(2)$ is described by the current membrane structure, the multisets of objects over Γ contained in each membrane, and the multiset of objects over $\Gamma \setminus \mathcal{E}$ currently in the environment. Hence, a configuration of Π can be described by a multiset of labelled objects

$$\{(a, i, \sigma) \mid a \in \Gamma \cup \{\lambda\}, 1 \leq i \leq q, \sigma \in \{0,1\}^*\} \cup \{(a, 0) \mid a \in \Gamma \setminus \mathcal{E}\}.$$

Let us notice that the number of labels we need to identify all membranes appearing along any computation of a P system from $\mathbf{CSC}(2)$ is quadratic in the size of the initial configuration of the system and the length of the computation.

Let $r = (ab, out) \in \mathcal{R}_i$, $2 \leq i \leq q$, be a symport rule of Π and $n \in \mathbb{N}$. We denote by $n \cdot LHS(r, (i, \sigma), (p(i), \tau))$ the multiset of objects $(a, i, \sigma)^n (b, i, \sigma)^n$, and we denote by $n \cdot RHS(r, (i, \sigma), (p(i), \tau))$ the multiset $(a, p(i), \tau)^n (b, p(i), \tau)^n$. In a similar way, $n \cdot LHS(r, (i, \sigma), (p(i), \tau))$ and $n \cdot RHS(r, (i, \sigma), (p(i), \tau))$ are defined when r is of the form $(a, out) \in \mathcal{R}_i$. Note that, at a given instant of the computation, for each membrane (i, σ) there is a unique parent membrane $(p(i), \tau)$, according to the current membrane structure.

Let $r = (ab, out) \in \mathcal{R}_1$ be a symport rule of Π and $n \in \mathbb{N}$. We denote by $n \cdot LHS(r, (1, \lambda), 0)$ the multiset of objects $(a, 1, \lambda)^n (b, 1, \lambda)^n$. We denote by $n \cdot RHS(r, (1, \lambda), 0)$ the following multiset of objects:

$$\begin{cases} (a,0)^n (b,0)^n, & \text{if } a, b \in \Gamma \setminus \mathcal{E}; \\ (a,0)^n, & \text{if } a \in \Gamma \setminus \mathcal{E} \text{ and } b \in \mathcal{E}; \\ (b,0)^n, & \text{if } b \in \Gamma \setminus \mathcal{E} \text{ and } a \in \mathcal{E}; \\ \emptyset, & \text{if } a, b \in \mathcal{E}. \end{cases}$$

In a similar way, $n \cdot LHS(r, (1, \lambda), 0)$ and $n \cdot RHS(r, (1, \lambda), 0)$ are defined when r is of the form $(a, out) \in \mathcal{R}_1$.

Let $r = (ab, in) \in \mathcal{R}_i$, $2 \leq i \leq q$, be a symport rule of Π and $n \in \mathbb{N}$. We denote by $n \cdot LHS(r, (i, \sigma), (p(i), \tau))$ the multiset of objects $(a, p(i), \tau)^n (b, p(i), \tau)^n$. We denote by $n \cdot RHS(r, (i, \sigma), (p(i), \tau))$ the multiset of objects $(a, i, \sigma)^n (b, i, \sigma)^n$. In a similar way, $n \cdot LHS(r, (i, \sigma), (p(i), \tau))$ and $n \cdot RHS(r, (i, \sigma), (p(i), \tau))$ are defined when r is of the form $(a, in) \in \mathcal{R}_i$.

Let $r = (ab, in) \in \mathcal{R}_1$ be a symport rule of Π and $n \in \mathbb{N}$. We denote by $n \cdot LHS(r, (1, \lambda), 0)$ the following multiset of objects:

$$\begin{cases} (a,0)^n (b,0)^n, & \text{if } a, b \in \Gamma \setminus \mathcal{E}; \\ (a,0)^n, & \text{if } a \in \Gamma \setminus \mathcal{E} \text{ and } b \in \mathcal{E}; \\ (b,0)^n, & \text{if } b \in \Gamma \setminus \mathcal{E} \text{ and } a \in \mathcal{E}; \\ \emptyset, & \text{if } a, b \in \mathcal{E}. \end{cases}$$

We denote by $n \cdot RHS(r,(1,\lambda),0)$ the multiset of objects $(a,1,\lambda)^n(b,1,\lambda)^n$. In a similar way, $n \cdot LHS(r,(1,\lambda),0)$ and $n \cdot RHS(r,(1,\lambda),0)$ are defined when r is of the form $(a,in) \in \mathcal{R}_1$.

Let $r = (a,out;b,in) \in \mathcal{R}_i$, $2 \leq i \leq q$, be an antiport rule of Π and $n \in \mathbb{N}$. We denote by $n \cdot LHS(r,(i,\sigma),(p(i),\tau))$ the multiset of objects $(a,i,\sigma)^n(b,p(i),\tau)^n$. Similarly, we denote by $n \cdot RHS(r,(i,\sigma),(p(i),\tau))$ the multiset of objects $(a,p(i),\tau)^n(b,i,\sigma)^n$.

Let $r = (a,out;b,in) \in \mathcal{R}_1$ be an antiport rule of Π. We denote by $n \cdot LHS(r,(1,\lambda),0)$ the following multiset of objects:

$$\begin{cases} (a,1,\lambda)^n(b,0)^n, & \text{if } b \in \Gamma \setminus \mathcal{E}; \\ (a,1,\lambda)^n, & \text{if } b \in \mathcal{E}. \end{cases}$$

Similarly, we denote by $n \cdot RHS(r,(1,\lambda),0)$ the following multiset of objects:

$$\begin{cases} (a,0)^n(b,1,\lambda)^n, & \text{if } a \in \Gamma \setminus \mathcal{E}; \\ (b,1,\lambda)^n, & \text{if } a \in \mathcal{E}. \end{cases}$$

If $\mathcal{C}_t$ is a configuration of Π, then we denote by $\mathcal{C}_t + \{(x,i,\sigma)/\sigma'\}$ the multiset obtained by replacing in $\mathcal{C}_t$ every occurrence of (x,i,σ) by (x,i,σ'). Besides, $\mathcal{C}_t + m$ (resp., $\mathcal{C}_t \setminus m$) is used to denote that a multiset m of labelled objects is added (resp., removed) to the configuration.

4.1 Characterizing class P by means of systems from CSC(2)

In order to show that only tractable problems can be solved efficiently by using families of P systems from **CSC**(2), we first state a technical result concerning recognizer P systems from **CSC**(2) (see [7] for more details).

Lemma 4.1 *Let $\Pi = (\Gamma, \Gamma_0, \Gamma_1, \mathcal{E}, \Sigma, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \ldots, \mathcal{R}_q, i_{in}, i_{out})$ be a recognizer P system of degree $q \geq 1$ from* **CSC**(2). *Let $M = |\mathcal{M}_1 + \ldots + \mathcal{M}_q|$ and let $\mathcal{C} = \{\mathcal{C}_0, \ldots, \mathcal{C}_r\}$ be a computation of Π. Then, we have*

(1) $|\mathcal{C}_0^*| = M$, and for each t, $0 \leq t < r$, $\mathcal{C}_{t+1}^* \cap (\Gamma \setminus \mathcal{E}) \subseteq \mathcal{C}_t^* \cap (\Gamma \setminus \mathcal{E})$;
(2) for each t, $0 \leq t \leq r$, $\mathcal{C}_t^* \cap (\Gamma \setminus \mathcal{E}) \subseteq (\mathcal{M}_1 + \ldots + \mathcal{M}_q) \cap (\Gamma \setminus \mathcal{E})$, and $|\mathcal{C}_t^* \cap (\Gamma \setminus \mathcal{E})| \leq M$;
(3) for each t, $0 \leq t < r$, $|\mathcal{C}_{t+1}^*| \leq |\mathcal{C}_t^*| + M$;
(4) for each t, $0 \leq t \leq r$, $|\mathcal{C}_t^*| \leq M \cdot (1+t)$;
(5) the number of membranes created along computation $\mathcal{C}$ by the application of separation rules is bounded by $2M \cdot (1+r)$.

Next, we present a deterministic algorithm $\mathcal{A}$ working in polynomial time that receives as an input a P system Π from **CSC**(2) and an input multiset m of Π, in such manner that algorithm $\mathcal{A}$ reproduces the behaviour of a computation of $\Pi + m$. In particular, if Π is confluent, then algorithm $\mathcal{A}$ will provide the same answer of the system Π.

The pseudocode of the algorithm $\mathcal{A}$ is described as follows:

Input: A P system Π from **CSC**(2) and an input multiset m
  Initialization phase: $\mathcal{C}_0$ is the initial configuration of $\Pi + m$
  $t \leftarrow 0$
  while $\mathcal{C}_t$ is a non halting configuration **do**
    Selection phase: Input $\mathcal{C}_t$, Output $(\mathcal{C}'_t, A)$
    Execution phase: Input $(\mathcal{C}'_t, A)$, Output $\mathcal{C}_{t+1}$
    $t \leftarrow t + 1$
  end while
Output: *Yes* if object yes appears in the environment associated with the halting configuration $\mathcal{C}_t$, *No* otherwise

The algorithm $\mathcal{A}$ receives a recognizer P system

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

from **CSC**(2) and an input multiset m. Let $M = |\mathcal{M}_1 + \ldots + \mathcal{M}_q|$, $p \in \mathbb{N}$ be a natural number such that any computation of $\Pi + m$ performs, at most, p transition steps. Hence, from Lemma 4.1, we know that the number of membranes in the system along any computation is bounded by $2M(1 + p) + q$.

A transition step of a recognizer P system $\Pi + m$ is performed by the selection and the execution phases. Specifically, the selection phase receives as an input a configuration $\mathcal{C}_t$ of $\Pi + m$ at an instant t. The output of this phase is a pair $(\mathcal{C}'_t, A)$, where A encodes a multiset of rules selected to be applied to $\mathcal{C}_t$, and $\mathcal{C}'_t$ is the configuration obtained from $\mathcal{C}_t$ once the labelled objects corresponding to the left-hand side of the rules from A have been consumed. The execution phase receives as an input the pair $(\mathcal{C}'_t, A)$, and the output of this phase is the next configuration $\mathcal{C}_{t+1}$ of $\mathcal{C}_t$. More precisely, configuration $\mathcal{C}_{t+1}$ is obtained from $\mathcal{C}'_t$ by adding the labelled objects produced by the application of rules from A; that is, the labelled objects corresponding to the right-hand side of the rules from A.

Selection phase.

Input: A configuration $\mathcal{C}_t$ of $\Pi + m$ at instant t
  $\mathcal{C}'_t \leftarrow \mathcal{C}_t$; $A \leftarrow \emptyset$; $B \leftarrow \emptyset$
  for $r = (u, out; v, in) \in \mathcal{R}_i, 2 \leq i \leq q$ according to the order chosen **do**
    for each membrane (i, σ) of $\mathcal{C}'_t$ according to the lexicographical order **do**
      $n_r \leftarrow$ maximum number of times that r is applicable to (i, σ)
      if $n_r > 0$ **then**
        $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus n_r \cdot LHS(r, (i, \sigma), (p(i), \tau))$
        $A \leftarrow A \cup \{(r, n_r, (i, \sigma), (p(i), \tau))\}$
        $B \leftarrow B \cup \{(i, \sigma), (p(i), \tau)\}$
      end if
    end for
  end for

for $r = (u, out; v, in) \in \mathcal{R}_1$ `according to the order chosen` **do**
  $n_r \leftarrow$ `maximum number of times that` r `is applicable to` $(1, \lambda)$
  if $n_r > 0$ **then**
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus n_r \cdot LHS(r, (1, \lambda), 0)$
    $A \leftarrow A \cup \{(r, n_r, (1, \lambda), 0)\}$
  end if
end for
for $r = [\, a \,]_i \rightarrow [\, \Gamma_0 \,]_i [\, \Gamma_1 \,]_i \in \mathcal{R}_i$ $(i \neq 1)$ `according to the order chosen` **do**
  for each $(a, i, \sigma) \in \mathcal{C}'_t$ `according to the lexicographical order, and such that` $(i, \sigma) \notin B$ **do**
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t \setminus \{(a, i, \sigma)\}$
    $A \leftarrow A \cup \{(r, 1, (i, \sigma))\}$
    $B \leftarrow B \cup \{(i, \sigma)\}$
  end for
end for

This algorithm is deterministic and works in polynomial time. Indeed, the running time of the previous algorithm is polynomial in the size of Π because: the number of cycles of the first main loop **for** is of order $O(|\mathcal{R}| \cdot M \cdot p \cdot q)$; the number of cycles of the second main loop **for** is of order $O(|\mathcal{R}|)$; and the number of cycles of the third main loop **for** is of order $O(|\mathcal{R}| \cdot M \cdot p \cdot q \cdot |\Gamma|)$.

Execution phase.

Input: `The output` $(\mathcal{C}'_t, A)$ `of the selection phase`
  for each $(r, n_r, (i, \sigma), (p(i), \tau)) \in A$ **do**
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + n_r \cdot RHS(r, (i, \sigma), (p(i), \tau))$
  end for
  for each $(r, n_r, (1, \lambda), 0) \in A$ **do**
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + n_r \cdot RHS(r, (1, \lambda), 0)$
  end for
  for each $(r, 1, (i, \sigma)) \in A$ **do**
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(\lambda, i, \sigma)/\sigma 0\}$
    $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(\lambda, i, \sigma 1)\}$
    for each $(x, i, \sigma) \in C'_t$ `according to the lexicographical order` **do**
      if $x \in \Gamma_0$ **then**
        $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x, i, \sigma)/\sigma 0\}$
      else
        $\mathcal{C}'_t \leftarrow \mathcal{C}'_t + \{(x, i, \sigma)/\sigma 1\}$
      end if
    end for

end for
$\mathcal{C}_{t+1} \leftarrow \mathcal{C}'_t$

This algorithm is deterministic and works in polynomial time. Indeed, the running time of the previous algorithm is polynomial in the size of Π because: the number of cycles of the first main loop **for** is of order $O(|\mathcal{R}| \cdot M \cdot p \cdot q)$; the number of cycles of the second main loop **for** is of order $O(|\mathcal{R}|)$; and the number of cycles of the third main loop **for** is of order $O(|\mathcal{R}| \cdot M \cdot p \cdot q \cdot |\Gamma|)$.

Theorem 4.2 $\mathbf{P} = \mathbf{PMC}_{\mathbf{CSC}(2)}$.

Proof. It suffices to show that $\mathbf{PMC}_{\mathbf{CSC}(2)} \subseteq \mathbf{P}$. Let $X \in \mathbf{PMC}_{\mathbf{CSC}(2)}$ and let $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ be a family of recognizer P systems from $\mathbf{CSC}(2)$ solving X, according to Definition 5. Let (cod, s) be a polynomial encoding associated with that solution. If $u \in I_X$ is an instance of the problem X, then u will be processed by the system $\Pi(s(u)) + cod(u)$.
Let us consider the following deterministic algorithm $\mathcal{A}'$:

Input: `an instance` u `of the problem` X
`Construct the system` $\Pi(s(u)) + cod(u)$.
`Run algorithm` $\mathcal{A}$ `with input` $\Pi(s(u)) + cod(u)$.
Output: *Yes* `if algorithm` $\mathcal{A}$ `returns` *Yes*,
No `otherwise.`

The algorithm $\mathcal{A}'$ receives as an input an instance u of the decision problem $X = (I_X, \theta_X)$ and works in polynomial time with respect to the size of the input. The following assertions are equivalent:

- $\theta_X(u) = 1$; that is, the answer of problem X to instance u is affirmative.
- Every computation of $\Pi(s(u)) + cod(u)$ is an accepting computation.
- The output of algorithm $\mathcal{A}'$ with input u is *Yes*.

Hence, $X \in \mathbf{P}$.

□

5 Computational efficiency of systems in CDC(2)

In this section we study the ability to solve **NP**–complete problems of families of recognizer P systems with membrane division whose communication rules (of type symport/antiport) use a minimal cooperation (i.e., communication rules involving at most two objects). Specifically, we give a polynomial time solution to `HAM-CYCLE` problem, a well known **NP**-complete problem [2], by means of a family of such kind of recognizer P systems, according to Definition 5 (see [15] for more details).

Let us recall that `HAM-CYCLE` problem is the following: *Given a directed graph, determine whether or not there exists a Hamiltonian cycle in the graph.*

5.1 A polynomial time solution of HAM-CYCLE problem in CDC(2)

For each $n, m \in \mathbb{N}$, we consider the recognizer P system with symport/antiport rules and membrane division of degree $11 + 2n + n^3$

$$\begin{aligned}\Pi(\langle n,m\rangle) = (\Gamma, \mathcal{E}, \Sigma, \mu, \mathcal{M}_r\,(1 \le r \le 11)\,, \mathcal{M}_{a_{1,j}}\,(1 \le j \le n),\ \mathcal{M}_{a_{2,j}}\,(1 \le j \le n)\,,\\ \mathcal{M}_{e_{i,j,k}}(1 \le i,j,k \le n)\,, \mathcal{R}_r\,(1 \le r \le 11)\,, \mathcal{R}_{a_{1,j}}\,(1 \le j \le n)\,,\\ \mathcal{R}_{a_{2,j}}\,(1 \le j \le n)\ \mathcal{R}_{e_{i,j,k}}(1 \le i,j,k \le n))\end{aligned}$$

defined as follows:

(1) Working alphabet:
$\Gamma = \Sigma \cup \mathcal{E} \cup \{\beta_r \mid 0 \le r \le n^3+7\} \cup \{b'_r, b''_r, b'''_r, c'_r, c''_r, c'''_r, c''''_r \mid 1 \le r \le n^3\} \cup$
$\{(i,j)'_k\,,\ (i,j)''_k \mid 1 \le i,j,k \le n\} \cup \{(i,j)''_{k,r} \mid 1 \le i,j,k \le n \wedge 1 \le r \le n^3\} \cup$
$\{\alpha_0, a, a', a'', b, b', b'', b''', c, c', c'', c''', c'''', \mathtt{yes}, \mathtt{no}\}$,
where the input alphabet is $\Sigma = \{(i,j)_k \mid 1 \le i,j,k \le n\}$, and the alphabet of the environment is $\mathcal{E} = \{\alpha_r \mid 1 \le r \le n^3+6\}$

(2) Membrane structure μ: the root is labelled by 1, and the remaining nodes are children of the root, being labelled by

$$2, 3, \ldots, 11, a_{1,j}\,(1 \le j \le n), a_{2,j}\,(1 \le j \le n), e_{i,j,k}\,(1 \le i,j,k \le n),$$

respectively.

(3) Initial multisets:
$\mathcal{M}_1 = \{\alpha_0\} \cup \{\beta_r \mid 1 \le r \le n^3+7\} \cup \{b'_r, b''_r, b'''_r, c'_r, c''_r, c'''_r, c''''_r \mid 1 \le r \le n^3-1\}$;
$\mathcal{M}_2 = \{a^n, b, c\}$;
$\mathcal{M}_3 = \{b'_{n^3}\}$; $\mathcal{M}_4 = \{b''_{n^3}\}$; $\mathcal{M}_5 = \{b'''_{n^3}\}$;
$\mathcal{M}_6 = \{c'_{n^3}\}$; $\mathcal{M}_7 = \{c''_{n^3}\}$; $\mathcal{M}_8 = \{c'''_{n^3}\}$; $\mathcal{M}_9 = \{c''''_{n^3}\}$;
$\mathcal{M}_{10} = \{\mathtt{yes}\}$; $\mathcal{M}_{11} = \{\mathtt{no}, \beta_0\}$;
$\mathcal{M}_{a_{1,j}} = \{a'_{n^3}\}$, $\mathcal{M}_{a_{2,j}} = \{a''_{n^3}\}, 1 \le j \le n$;
$\mathcal{M}_{e_{i,j,k}} = \{(i,j)''_{k,n^3}\}, 1 \le i,j,k \le n$.

(4) Rules of the system:

• Rules in $\mathcal{R}_1$:

1.1 Rules to control the output of the computations by counters of type α_r.

$$(\alpha_r\,,\, out\,;\, \alpha_{r+1}\,,\, in)\ ,\ 0 \le r \le n^3+5.$$

Rules 1.2 and 1.3 produce the output of the computations:

1.2 $(\mathtt{yes}\,,\, out)$

1.3 $(\mathtt{no}\,\alpha_{n^3+6}\,,\, out)$

• Rules in $\mathcal{R}_2$:

2.1 Rules to produce all possible subsets of A'_G in membranes labelled by 2 at configuration $\mathcal{C}_{n^3+1}$:

$$[\,(i,j)_k\,]_2 \to [\,(i,j)'_k\,]_2\ [\,\#\,]_2,\ 1 \le i,j,k \le n.$$

Rules 2.2, 2.3, 2.4 and 2.5 allow to introduce objects a', a'', b', b'', c''', c', c'', c''' and c'''' in membranes labelled by 2 at configurations $\mathcal{C}_{n^3+2}$, $\mathcal{C}_{n^3+3}$, $\mathcal{C}_{n^3+4}$ and $\mathcal{C}_{n^3+5}$, respectively:

2.2 $(a\,,\,out\,;\,a'\,,\,in)$; $(a'\,,\,out\,;\,a''\,,\,in)$;
2.3 $(b\,,\,out\,;\,b'\,,\,in)$; $(b'\,,\,out\,;\,b''\,,\,in)$; $(b''\,,\,out\,;\,b'''\,,\,in)$;
2.4 $(c\,,\,out\,;\,c'\,,\,in)$; $(c'\,,\,out\,;\,c''\,,\,in)$; $(c''\,,\,out\,;\,c'''\,,\,in)$; $(c'''\,,\,out\,;\,c''''\,,\,in)$;
2.5 $(a''\,b'''\,,\,out)$; $(b'''\,c''''\,,\,out)$.
2.6 Rules to produce in each membrane labelled by 2 at configuration $\mathcal{C}_{n^3+2}$ a subset of A''_G from a subset of A'_G at configuration $\mathcal{C}_{n^3+1}$:

$$((i,j)'_k\,,\,out\,;\,(i,j)''_k\,,\,in)\ ,\ 1 \leq i,j,k \leq n.$$

2.7 Rules to generate in each membrane labelled by 2 at configuration $\mathcal{C}_{n^3+1}$ a subset of A''_G encoding a possible Hamiltonian cycle.
$((i,j)''_k\ (i,j')''_{k'}\ ,\,out),\ 1 \leq i,i',j,j',k,k' \leq n;$
$((i,j)''_k\ (i',j)''_{k'}\ ,\,out),\ 1 \leq i,i',j,j',k,k' \leq n;$
$((i,j)''_k\ (i',j')''_{k+1}\ ,\,out),\ 1 \leq i,i',j,j',k,k' \leq n,\ j \neq i';$
$((i,j)''_k\ (i',j')''_k\ ,\,out),\ 1 \leq i,i',j,j',k,k' \leq n.$
2.8 Rules to check if the subset represented by each membrane with label 2 at configuration $\mathcal{C}_{n^3+3}$ encodes a Hamiltonian cycle of the input graph:

$$(a''\,(i,j)''_k\,,\,out),\ 1 \leq i,j,k \leq n.$$

• Rules in $\mathcal{R}_3$:
Rules to produce $2^{n \cdot p}$ copies of objects b' in the skin membrane of configuration $\mathcal{C}_{n^3+1}$:
3.1 $(b'_r\,,\,out\,;\,b'_{r-1}\,,\,in),\ n \cdot m + 1 \leq r \leq n^3;$
3.2 $[\,b'_r\,]_3 \rightarrow [\,b'_{r-1}\,]_3\ [\,b'_{r-1}\,]_3,\ 2 \leq r \leq n \cdot m;$
3.3 $[\,b'_1\,]_3 \rightarrow [\,b'\,]_3\ [\,b'\,]_3;$
3.4 $(b'\ ,\,out)$.
• Rules in $\mathcal{R}_4$:
Rules to produce $2^{n \cdot p}$ copies of objects b'' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:
4.1 $(b''_r\,,\,out\,;\,b''_{r-1}\,,\,in),\ n \cdot m + 1 \leq r \leq n^3;$
4.2 $[\,b''_r\,]_4 \rightarrow [\,b''_{r-1}\,]_4\ \ b''_{r-1}\,]_4,\ 2 \leq r \leq n \cdot m;$
4.3 $[\,b''_1\,]_4 \rightarrow [\,b''\,]_4\ [\,b''\,]_4;$
4.4 $(b''\ ,\,out)$.
• Rules in $\mathcal{R}_5$:
Rules to produce $2^{n \cdot p}$ copies of objects b''' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:
5.1 $(b'''_r\,,\,out\,;\,b'''_{r-1}\,,\,in),\ n \cdot m + 1 \leq r \leq n^3;$
5.2 $[\,b'''_r\,]_5 \rightarrow [\,b'''_{r-1}\,]_5\ [\,b'''_{r-1}\,]_5,\ 2 \leq r \leq n \cdot m;$
5.3 $[\,b'''_1\,]_5 \rightarrow [\,b'''\,]_5\ [\,b'''\,]_5;$
5.4 $(b'''\ ,\,out)$.
• Rules in $\mathcal{R}_6$:
Rules to produce $2^{n \cdot p}$ copies of objects c' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:

6.1 $(c'_r\,,\,out\,;\,c'_{r-1}\,,\,in),\ n\cdot m+1\leq r\leq n^3$;
6.2 $[\,c'_r\,]_6\rightarrow[\,c'_{r-1}\,]_6\,[\,c'_{r-1}\,]_6,\ 2\leq r\leq n\cdot m$;
6.3 $[\,c'_1\,]_6\rightarrow[\,c'\,]_6\,[\,c'\,]_6$;
6.4 $(c'\,,\,out)$.

• Rules in $\mathcal{R}_7$:
Rules to produce $2^{n\cdot p}$ copies of objects c'' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:
7.1 $(c''_r\,,\,out\,;\,c''_{r-1}\,,\,in),\ n\cdot m+1\leq r\leq n^3$;
7.2 $[\,c''_r\,]_7\rightarrow[\,c''_{r-1}\,]_7\,[\,c''_{r-1}\,]_7,\ 2\leq r\leq n\cdot m$;
7.3 $[\,c''_1\,]_7\rightarrow[\,c''\,]_7\,[\,c''\,]_7$;
7.4 $(c''\,,\,out)$.

• Rules in $\mathcal{R}_8$:
Rules to produce $2^{n\cdot p}$ copies of objects c''' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:
8.1 $(c'''_r\,,\,out\,;\,c'''_{r-1}\,,\,in),\ n\cdot m+1\leq r\leq n^3$;
8.2 $[\,c'''_r\,]_8\rightarrow[\,c'''_{r-1}\,]_8\,[\,c'''_{r-1}\,]_8,\ 2\leq r\leq n\cdot m$;
8.3 $[\,c'''_1\,]_8\rightarrow[\,c'''\,]_8\,[\,c'''\,]_8$;
8.4 $(c'''\,,\,out)$.

• Rules in $\mathcal{R}_9$:
Rules to produce $2^{n\cdot p}$ copies of objects c'''' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:
9.1 $(c''''_r\,,\,out\,;\,c''''_{r-1}\,,\,in),\ n\cdot m+1\leq r\leq n^3$;
9.2 $[\,c''''_r\,]_9\rightarrow[\,c''''_{r-1}\,]_9\,[\,c''''_{r-1}\,]_9,\ 2\leq r\leq n\cdot m$;
9.3 $[\,c''''_1\,]_9\rightarrow[\,c''''\,]_9\,[\,c''''\,]_9$;
9.4 $(c''''\,,\,out)$.

• Rules in $\mathcal{R}_{10}$:
Rules to produce an affirmative answer:
10.1 $(\alpha_{n^3+6}\,c''''\,,\,in)\ ;\ (c''''\,\texttt{yes}\,,\,out)$

• Rules in $\mathcal{R}_{11}$:
Rules to control the negative answer of the computations by counters β_r:
11.1 $(\beta_r\,out\,;\,\beta_{r+1}\,,\,in),\ 0\leq r\leq n^3+6$;
11.2 $(\beta_{n^3+7}\,\texttt{no}\,,\,out)$.

• Rules in $\mathcal{R}_{a_{1,j}},\ 1\leq j\leq n$:
Rules to produce 2^{n^3} copies of objects a' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:
$\mathbf{a_{1,j}.1}$ $[\,a'_r\,]_{a_{1,j}}\rightarrow[\,a'_{r-1}\,]_{a_{1,j}}\,[\,a'_{r-1}\,]_{a_{1,j}},\ 2\leq r\leq n^3$;
$\mathbf{a_{1,j}.2}$ $[\,a'_1\,]_{a_{1,j}}\rightarrow[\,a'\,]_{a_{1,j}}\,[\,a'\,]_{a_{1,j}}$;
$\mathbf{a_{1,j}.3}$ $(a'\,,\,out)$.

• Rules in $\mathcal{R}_{a_{2,j}},\ 1\leq j\leq n$:
Rules to produce 2^{n^3} copies of objects a'' in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:
$\mathbf{a_{2,j}.1}$ $[\,a''_r\,]_{a_{2,j}}\rightarrow[\,a''_{r-1}\,]_{a_{2,j}}\,[\,a''_{r-1}\,]_{a_{2,j}},\ 2\leq r\leq n^3$;

$\mathbf{a_{2,j}.2}$ $[\, a_1'' \,]_{a_{2,j}} \rightarrow [\, a'' \,]_{a_{2,j}}\ [\, a'' \,]_{a_{2,j}}$;
$\mathbf{a_{2,j}.3}$ (a'', out).

• Rules in $\mathcal{R}_{e_{i,j,k}}$, $1 \leq i, j, k \leq n$:

Rules to produce 2^{n^3} copies of objects $(i,j)_k''$ in the skin membrane at configuration $\mathcal{C}_{n^3+1}$:

$\mathbf{e_{i,j,k}.1}$ $[\, (i,j)_{k,r}'' \,]_{e_{i,j,k}} \rightarrow [\, (i,j)_{k,r-1}'' \,]_{e_{i,j,k}}\ [\, (i,j)_{k,r-1}'' \,]_{e_{i,j,k}}$, $2 \leq r \leq n^3$;
$\mathbf{e_{i,j,k}.2}$ $[\, (i,j)_{k,1}'' \,]_{e_{i,j,k}} \rightarrow [\, (i,j)_k'' \,]_{e_{i,j,k}}\ [\, (i,j)_k'' \,]_{e_{i,j,k}}$;
$\mathbf{e_{i,j,k}.3}$ $((i,j)_k'', out)$.

(5) The input membrane is the membrane labelled by 2 and the output region is the environment of the system (labelled by 0).

5.2 An overview of the computations

Now we briefly show how each system $\Pi(\langle n, m \rangle)$ works in order to process any directed graph with n nodes and m arcs.

We consider the ensuing polynomial encoding (cod, s) from HAM-CYCLE in $\mathbf{\Pi}$: for each instance $G = (V, E)$ of HAM-CYCLE problem, with $V = \{1, \ldots, n\}$ and $E = \{(i_1, j_1), \ldots, (i_m, j_m)\}$, we define $s(G) = \langle n, m \rangle$ and $cod(G) = \{(i,j)_k \mid (i,j) \in E, 1 \leq k \leq n\}$. The expression $(i,j)_k$ in $cod(G)$ can be interpreted as follows: arc (i,j) is "placed" in "position k" in a potential path. According to this polynomial encoding, graph G will be processed by system $\Pi(s(G))$ with input multiset $cod(G)$. In what follows, we informally describe how system $\Pi(s(G)) + cod(G)$ works. The solution is structured in the following stages:

- *Generation Stage*: All possible combinations of arcs from the input graph, including a code of their position in potential paths, are generated by using cell division in an adequate way.
- *Checking Stage*: It is checked whether or not the different combinations of arcs generated in the previous stage encode Hamiltonian cycles of the input graph.
- *Output Stage*: The system sends the right answer to the environment according to the results obtained in the previous stage.

Generation stage

At this stage, the system generates all the possible subsets of arcs of the graph (in fact, subsets of A'_G) which contain their potential positions in a path according to the notations introduced in Subsection 2.2. In this way, by applying rules of type 2.1 at configuration $\mathcal{C}_{2^{n \cdot m}}$, there will be $2^{n \cdot m}$ membranes labelled by 2 such that each of them encodes a different combination of arcs from the input graph. Simultaneously, by applying rules of types 1, 2 and 3 from $\mathcal{R}_3$, $\mathcal{R}_4$, $\mathcal{R}_5$, $\mathcal{R}_6$, $\mathcal{R}_7$, $\mathcal{R}_8$ and $\mathcal{R}_9$, $2^{n \cdot m}$ copies of objects $b', b'', b''', c', c'', c'''$ and c'''' are produced in membranes labelled by $3, 4, 5, 6, 7, 8, 9$, respectively, and 2^{n^3} copies of objects a', a'' and $(i,j)_k''$ are produced in membranes labelled by $a_{1,j}, a_{2,j}$, and $e_{i,j,k}$, respectively. The generation stage takes n^3 steps.

Checking stage

At this stage, the system checks whether or not there exists a membrane labelled by 2 at configuration $\mathcal{C}_{n^3+5}$ containing a subset of A''_G that encodes a Hamiltonian cycle of G. This is done in 4 steps.

At step n^3+1, the contents of membranes labelled by $3, 4, 5, 6, 7, 8, 9$, $a_{1,j}$ $(1 \leq j \leq n), a_{2,j}$ $(1 \leq j \leq n)$ and $e_{i,j,k}$ $(1 \leq i, j, k \leq n)$ are sent to the skin membrane by applying rules 3.4, 4.4, 5.4, 6.4, 7.4, 8.4, 9.4, $a_{1,j}.2, a_{2,j}.2, e_{i,j,k}.3$. From this moment on, none of these membranes will participate in the evolution of the configurations.

At step n^3+2, objects a, b, c in membrane labelled by 2 at configuration $\mathcal{C}_{n^3+1}$ are replaced by objects a', b', c' from the skin membrane by applying rules 2.2, 2.3, and 2.4. Simultaneously, by applying rules 2.6, each subset of A'_G contained in a membrane labelled by 2 at configuration $\mathcal{C}_{n^3+1}$ produces the "corresponding" subset of A''_G. Besides, $\mathcal{C}_{n^3+2}(10) = \{\texttt{yes}\}$ and $\mathcal{C}_{n^3+2}(11) = \{\beta_{n^3+2}\,, \texttt{no}\}$.

At step n^3+3, by applying rules 2.3 and 2.4, objects a', b', c' in membranes labelled by 2 at configuration $\mathcal{C}_{n^3+2}$ are replaced by objects a'', b'', c'' from the skin membrane. Simultaneously, by applying rules of type 2.7, each subset contained in a membrane labelled by 2 at configuration $\mathcal{C}_{n^3+2}$ is transformed into a subset encoding each possible path in the input graph. This way, according to Proposition 2.1, we have that the input graph (with n nodes and m arcs) has a Hamiltonian cycle if and only if at configuration $\mathcal{C}_{n^3+3}$ there exists some membrane labelled by 2 at configuration $\mathcal{C}_{n^3+3}$ such that the subset of A''_G contained in it has size equal to n. Besides, $\mathcal{C}_{n^3+3}(10) = \{\texttt{yes}\}$ and $\mathcal{C}_{n^3+3}(11) = \{\beta_{n^3+3}\,, \texttt{no}\}$.

At step n^3+4, by applying rules 2.3 and 2.4, objects b'', c'' in membranes labelled by 2 are substituted by objects b''', c''' from the skin membrane. Simultaneously, by applying rules 2.8, each object contained in the subset associated with each membrane labelled by 2 at configuration $\mathcal{C}_{n^3+3}$ is sent to the skin membrane cooperating with an object a''. Therefore, the number of copies of object a'' appearing in a membrane labelled by 2 at configuration $\mathcal{C}_{n^3+4}$ is equal to $n - \gamma$, where γ is the size of the path in the input graph encoded by that membrane. Then, the input graph (with n nodes and m arcs) has a Hamiltonian cycle if and only if there exists a membrane labelled by 2 at configuration $\mathcal{C}_{n^3+4}$ such that it does not contain any object a''.

At step n^3+5, by applying rules of type 2.5, objects a'' and b''' in membrane labelled by 2 at configuration $\mathcal{C}_{n^3+5}$ are sent to the skin membrane. Simultaneously, rule $(c''', out\,; c'''', in)$ produces an object c'''' in each membrane labelled by 2 at configuration $\mathcal{C}_{n^3+5}$.

Output stage

Finally, the output stage takes 4 steps. Only membranes labelled by 2 at configuration $\mathcal{C}_{n^3+5}$ containing some object b''' (i.e., membrane encoding a Hamiltonian cycle) can evolve, and only rule $(c''', out\,; c'''', in) \in \mathcal{R}_2$ is applicable to that membrane. In this case, an object c'''' will appear in each membrane labelled by 2

at that configuration. Besides, if a membrane with label 2 at the mentioned configuration does not encode a Hamiltonian cycle of the input graph, then it contains objects b'', so rule $(a''\, b''' , out) \in \mathcal{R}_2$ will be applied. That is, the input graph has a Hamiltonian cycle if and only if some object c'''' appears in the skin membrane at configuration $\mathcal{C}_{n^3+6}$. Besides, $\mathcal{C}_{n^3+6}(10) = \{\texttt{yes}\}$ and $\mathcal{C}_{n^3+6}(11) = \{\beta_{n^3+6} , \texttt{no}\}$.

If the input graph has a Hamiltonian cycle, then only rules $(\alpha_{n^3+6}\, c'''' , in) \in \mathcal{R}_{10}$ and $(\beta_{n^3+6} , out ; \beta_{n^3+7} , in) \in \mathcal{R}_{11}$ are applicable to configuration $\mathcal{C}_{n^3+6}$. Otherwise, only rule $(\beta_{n^3+6}\, out ; \beta_{n^3+7} , in)$ is applicable to that configuration. Therefore, the answer of the problem is affirmative if and only if $\mathcal{C}_{n^3+7}(10) = \{\alpha_{n^3+6}\, c'''' , \texttt{yes}\}$. Besides, in any case, $\mathcal{C}_{n^3+7}(11) = \{\beta_{n^3+7} , \texttt{no}\}$. Then, if there exists a Hamiltonian path, then rules $(c''''\, \texttt{yes} , out) \in \mathcal{R}_{10}$ and $(\beta_{n^3+7}\, \texttt{no} , out) \in \mathcal{R}_{11}$ are applicable to configuration $\mathcal{C}_{n^3+7}$. Otherwise, only rule $(\beta_{n^3+7}\, \texttt{no} , out) \in \mathcal{R}_{11}$ is applicable to that configuration. Hence, the answer of the problem is affirmative if and only if the skin membrane at configuration $\mathcal{C}_{n^3+8}$ contains object $\texttt{yes}$ (together with objects $c'''', \beta_{n^3+7}, \texttt{no}$), but no object α_{n^3+6}. Otherwise, the skin membrane at configuration $\mathcal{C}_{n^3+8}$ contains objects $\beta_{n^3+7}, \texttt{no}, \alpha_{n^3+6}$, but no object $\texttt{yes}$.

At the last step, in cases when an affirmative answer results, rule $(\texttt{yes} , out)$ is applied to configuration $\mathcal{C}_{n^3+8}$, producing an object $\texttt{yes}$ in the environment, and the computation halts. Otherwise, rule $(\texttt{no}\, \alpha_{n^3+6} , out)$ is applied to that configuration, thus producing a negative answer.

5.3 Main result

Theorem 5.1 $\texttt{HAM-CYCLE} \in \mathbf{PMC}_{\mathbf{CDC(2)}}$.

Proof. The family of P systems with symport/antiport rules and membrane division constructed in Section 3.2 verifies the following:

(a) Every system of the family $\mathbf{\Pi}$ is a recognizer P system with membrane division and symport/antiport rules of length at most 2.

(b) The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines because, for each $n, m \in \mathbb{N}$, the rules of $\mathit{\Pi}(\langle n, m\rangle)$ of the family are recursively defined from $n, m \in \mathbb{N}$, and the amount of resources needed to build an element of the family is of a polynomial order in n, as shown below:
- Size of the alphabet: $n^6 + 12n^3 + 29 \in \Theta(n^6)$;
- Initial number of membranes: $n^3 + 2n + 11 \in \Theta(n^3)$;
- Initial number of objects: $9n^3 + 3n + 13 \in \Theta(n^3)$;
- Number of rules: $n^6 + 4n^5 + n^4 + 13n^3 + 2n + 30 \in \Theta(n^6)$;
- Maximal length of a rule: $2 \in \Theta(1)$.

(c) The pair (cod, s) of polynomial–time computable functions defined in Subsection 5.2 is a polynomial encoding from $\texttt{HAM} - \texttt{CYCLE}$ to $\mathbf{\Pi}$.

(d) The family $\mathbf{\Pi}$ is polynomially bounded, sound and complete with regard to $(\texttt{HAM-CYCLE}, cod, s)$ (see Subsection 5.2).

Therefore, according to Definition 5, the family $\mathbf{\Pi}$ from $\mathbf{CDC(2)}$ solves HAM-CYCLE problem in polynomial time with respect to the number of nodes.

□

Corollary 5.2 $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC_{CDC(2)}}$.

Proof. It suffices to notice that HAM-CYCLE problem is an **NP**-complete problem, HAM-CYCLE$\in \mathbf{PMC_{CDC(2)}}$, and the complexity class $\mathbf{PMC_{CDC(2)}}$ is closed under polynomial-time reduction and under complement.

□

6 Conclusions and open problems

The ability of cell-like P systems with symport/antiport rules involving at most two objects to efficiently solve computationally hard problems, has been studied. Specifically, if further membrane separation rules are allowed (while keeping the minimal cooperation restriction), then only problems in **P** can be solved in polynomial time. Nevertheless, if membrane division rules are allowed (instead of membrane separation rules), then **NP**–complete problems can be solved in polynomial time. In summary, we have two important results concerning the polynomial complexity classes associated with these kind of systems: (a) $\mathbf{P} = \mathbf{PMC_{CSC(2)}}$; and (b) $\mathbf{NP} \cup \mathbf{co\text{-}NP} \subseteq \mathbf{PMC_{CDC(2)}}$.

Therefore, assuming that **P** is different from **NP**, a new frontier of the efficiency has been obtained in Membrane Computing in terms of the kind of rules (separation versus division) able to produce an exponential workspace in linear time. That is, passing from allowing membrane separation rules to allowing membrane division rules in the framework of P systems with symport/antiport rules which use minimal cooperation, amounts to passing from non–efficiency to efficiency.

Acknowledgements

The work of L. Valencia-Cabrera, L.F. Macías-Ramos, A. Riscos-Núñez , and M.J. Pérez-Jiménez was supported by Project TIN2012-37434 of the Ministerio de Economía y Competitividad of Spain, cofinanced by FEDER funds. The work of B. Song and L. Pan was supported by National Natural Science Foundation of China (61033003, 91130034 and 61320106005).

References

1. T.H. Cormen, C.E. Leiserson, R.L. Rivest. *An Introduction to Algorithms*. The MIT Press, Cambridge, Massachussets, 1994.

2. M.R. Garey, D.S. Johnson. *Computers and Intractability A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, (1979).
3. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez, F.J. Romero-Campero. On the efficiency of cell-like and tissue-like recognizing membrane systems. *International Journal of Intelligent Systems*, **24**, 7 (2009), 747-765.
4. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez, F.J. Romero-Campero. On the power of dissolution in P systems with active membranes. In R. Freund, Gh. Paun, Gr. Rozenberg, A. Salomaa (eds.) Membrane Computing, 6th International Workshop, WMC 2005, Vienna, Austria, July 18-21, 2005, Revised Selected and Invited Papers. *Lecture Notes in Computer Science*, **3850** (2006), 224-240.
5. L.F. Macías-Ramos, M.J. Pérez-Jiménez, A. Riscos-Núñez, M. Rius-Font. The efficiency of tissue P systems with cell separation relies on the environment. In E. Csuhaj-Varjú, M. Gheorghe, G. Rozenberg, A. Salomaa, G. Vaszil (eds.) Membrane Computing- 13th International Conference CMC 2012 Budapest, Hungary, August 28-31, 2012, Revised Selected Papers. *Lecture Notes in Computer Science*, **7762** (2013), 243-256.
6. L.F. Macías-Ramos, M.A. Martínez-del-Amor, M.J. Pérez-Jiménez, A. Riscos-Núñez, L. Valencia-Cabrera. The Role of the Direction in Tissue P Systems with Cell Separation. *Journal of Automata, Languages and Combinatorics*, **19**, 1-4 (2014), 185-199.
7. L.F. Macías-Ramos, L. Valencia-Cabrera, B. Song, T. Song, L. Pan, M.J. Pérez-Jiménez. Membrane Fission: A Computational Complexity Perspective. *Complexity*, 2015, in press.
8. L. Pan, M.J. Pérez-Jiménez. Computational complexity of tissue–like P systems. *Journal of Complexity*, **26**, 3 (2010), 296–315.
9. A. Păun, Gh. Păun. The power of communication: P systems with symport/antiport, *New Generation Computing*, **20**, 3 (2002), 295–305.
10. I. Pérez-Hurtado, M.J. Pérez-Jiménez, A. Riscos-Núñez, M.A. Gutiérrez, M. Rius-Font. On a partial affirmative answer for a Paun's conjecture. *International Journal of Foundations of Computer Science*, **22**, 1 (2011), 55-64.
11. M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini, F. Complexity classes in models of cellular computing with membranes. *Natural Computing*, **2**, 3 (2003), 265–285.
12. M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini, A polynomial complexity class in P systems using membrane division, *Journal of Automata, Languages and Combinatorics*, **11**, 4 (2006) 423–434.
13. M.J. Pérez-Jiménez, A. Riscos-Núñez, M. Rius-Font, F.J. Romero- Campero. A polynomial alternative to unbounded environment for tissue P systems with cell division. *International Journal of Computer Mathematics*, **90**, 4 (2013), 760-775.
14. M.J. Pérez-Jiménez, A. Riscos-Núñez, M. Rius-Font, L. Valencia-Cabrera. The relevance of the environment on the efficiency of tissue P systems. In A. Alhazov, S. Cojocaru, M. Gheorghe, Y. Rogozhin G. Rozenberg, A. Salomaa (eds.) Membrane Computing- 14th International Conference CMC 2013 Chisinau, Republic of Moldova, August 20-23, 2013, Revised Selected Papers. *Lecture Notes in Computer Science*, **8340** (2014), 308-321.
15. L. Valencia-Cabrera, B. Song, T. Song, L.F. Macías-Ramos, L. Pan, M.J. Pérez-Jiménez. The Role of Cooperation in the Efficiency of Bioinspired Computing Devices, submitted 2015.