On Efficiency of P Systems with Symport/Antiport and Membrane Division

Luis F. Macías-Ramos[1], Bosheng Song[2],
Tao Song[2], Linqiang Pan[2], 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: lfmaciasr,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, songtao0608@hotmail.com,
lqpan@mail.hust.edu.cn

Summary. Classical membrane systems with symport/antiport rules observe the *conservation law*, in the sense that they compute by changing the places of objects with respect to the membranes, and not by changing the objects themselves. In these systems the environment plays an active role because the systems not only send objects to the environment, but also bring objects from the environment. In the initial configuration of a system, there is a special alphabet whose elements appear in an arbitrary large number of copies. The ability of these computing devices to have infinite copies of some objects has been widely exploited in the design of efficient solutions to computationally hard problems.

This paper deals with computational aspects of P systems with symport/antiport and membrane division rules where there is not an environment having the property mentioned above. Specifically, we establish the relationships between the polynomial complexity class associated with P systems with symport/antiport, membrane division rules, and with or without environment. As a consequence, we prove that the role of the environment is irrelevant in order to solve **NP**–complete problems in an efficient way.

Key words: Membrane Computing, P System with Symport/Antiport, Membrane Division, Computational Complexity.

1 Preliminaries

An *alphabet* Γ is a non–empty set whose elements are called *symbols*. An ordered finite sequence of symbols is a *string* or *word*. If u and v are strings over Γ, then so

is their *concatenation* uv, obtained by juxtaposition, that is, writing u and v one after the other. The number of symbols in a string u is the *length* of the string and it is denoted by $|u|$. As usual, the empty string (with length 0) will be denoted by λ. The set of all strings over an alphabet Γ is denoted by Γ^*. In algebraic terms, Γ^* is the free monoid generated by Γ under the operation of concatenation. Subsets of Γ^* are referred to as *languages* over Γ. The set of symbols occurring in a string $u \in \Gamma^*$ is denoted by $alph(u)$.

The *Parikh vector* associated with a string $u \in \Gamma^*$ with respect to the alphabet $\Sigma = \{a_1, \ldots, a_r\} \subseteq \Gamma$ is $\Psi_\Sigma(u) = (|u|_{a_1}, \ldots, |u|_{a_r})$, where $|u|_{a_i}$ denotes the number of occurrences of symbol a_i in string u. The application Ψ_Σ is called the *Parikh mapping* associated with Σ. Notice that, in this definition, the ordering of the symbols from Σ is relevant. If $\Sigma_1 = \{a_{i_1}, \ldots, a_{i_r}\} \subseteq \Gamma$, then we define $\Psi_{\Sigma_1}(u) = (|u|_{a_{i_1}}, \ldots, |u|_{a_{i_r}})$, for each $u \in \Gamma^*$.

A *multiset* m over a set A is a pair (A, f) where $f : A \to \mathbb{N}$ is a mapping. If $m = (A, f)$ is a multiset then its *support* is defined as $supp(m) = \{x \in A \,|\, f(x) > 0\}$. A multiset is empty (resp. finite) if its support is the empty set (resp. a finite set). If $m = (A, f)$ is a finite multiset over A and $supp(m) = \{a_1, \ldots, a_k\}$, then it will be denoted as $m = \{a_1^{f(a_1)}, \ldots, a_k^{f(a_k)}\}$. That is, superscripts indicate the multiplicity of each element, and if $f(x) = 0$ for $x \in A$, then element x is omitted. A finite multiset $m = \{a_1^{f(a_1)}, \ldots, a_k^{f(a_k)}\}$ can also be represented by the string $a_1^{f(a_1)} \ldots a_k^{f(a_k)}$ over the alphabet $\{a_1, \ldots, a_k\}$. Nevertheless, all permutations of this string identify the same multiset m precisely. We denote by $\emptyset$ the empty multiset and we denote by $\mathcal{M}_f(\Gamma)$ the set of all finite multisets over Γ. Throughout this paper, we speak about "the finite multiset m" where m is a string, meaning "the finite multiset represented by the string m". If $m_1 = (A, f_1)$, $m_2 = (A, f_2)$ are multisets over A, then we define the union of m_1 and m_2 as $m_1 + m_2 = (A, g)$, where $g = f_1 + f_2$, that is, $g(a) = f_1(a) + f_2(a)$, for each $a \in A$.

For any sets A and B the *relative complement* $A \setminus B$ of B in A is defined as follows: $A \setminus B = \{x \in A \,|\, x \notin B\}$. For any set A we denote $|A|$ the cardinal (number of elements) of A, as usual.

In what follows, we assume the reader is already familiar with the basic notions and terminology of P systems. For details, see [7].

2 P Systems with Symport/Antiport Rules and Membrane Division

Cell division is an elegant process that enables organisms to grow and reproduce. Mitosis is a process of cell division which results in the production of two daughter cells from a single parent cell. Daughter cells are identical to one another and to the original parent cell. Through a sequence of steps, the replicated genetic material in a parent cell is equally distributed to two daughter cells. While there are some subtle differences, mitosis is remarkably similar across organisms.

Before a dividing cell enters mitosis, it undergoes a period of growth where the cell replicates its genetic material and organelles. Replication is one of the most important functions of a cell. DNA replication is a simple and precise process that creates two complete strands of DNA (one for each daughter cell) where only one existed before (from the parent cell).

Next, we introduce an abstraction of these operation in the framework of P systems with symport/antiport rules. In these models, the membranes are not polarized; the membranes obtained by division have the same labels as the original membrane, and if a membrane is divided, its interaction with other membranes or with the environment is locked during the division process. In some sense, this means that while a membrane is dividing it closes its communication channels.

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

1. Γ *is a finite* alphabet.
2. $\mathcal{E} \subseteq \Gamma$.
3. μ *is a membrane structure (a rooted tree) whose nodes are injectively labelled with* $1, 2 \ldots, q$.
4. $\mathcal{M}_1, \ldots, \mathcal{M}_q$ *are multises over* Γ.
5. $\mathcal{R}_1, \cdots, \mathcal{R}_q$ *are finite set of rules of the following forms:*
   (a) Communication rules*:* $(u, out), (u, in), (u, out; v, in)$*, for* u, v *multisets over* Γ *and* $|u| + |v| > 0$*;*
   (b) Division rules*:* $[a]_i \rightarrow [b]_i[c]_i$*, where* $i \neq i_{out}$ *and* $a, b, c \in \Gamma$*;*
6. $i_{out} \in \{0, 1, \ldots, q\}$.

A P system with symport/antiport rules and membrane division

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

of degree q can be viewed as a set of q membranes, labelled by $1, \ldots, q$, arranged in a hierarchical structure, such that: (a) $\mathcal{M}_1, \ldots, \mathcal{M}_q$ represent the finite multisets of objects initially placed in the q membranes of the system; (b) $\mathcal{E}$ is the set of objects initially located in the environment of the system, all of them available in an arbitrary number of copies; and (c) 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 $i = 0$.

A rule of the type (u, out) or (u, in) is called a *symport* rule. A rule of the type $(u, out; v, in)$, where $|u| + |v| > 0$, is called an *antiport* rule. A P system with symport rules (resp. with antiport rules) is a P system with only symport rules (resp. only antiport rules) as communication rules. The length of rule (u, out) or (u, in) (resp. $(u, out; v, in)$) is defined as $|u|$ (resp. $|u| + |v|$).

An *instantaneous description* or a *configuration* at an instant t of a P system with symport/antiport and membrane division is described by all multisets of objects over Γ associated with all the membranes present in the system, and the

multiset of objects over $\Gamma - \mathcal{E}$ associated with the environment at that moment. Recall that there are infinitely many copies of objects from $\mathcal{E}$ in the environment, and hence this set is not properly changed along the computation. The *initial configuration* is $(\mathcal{M}_1, \cdots, \mathcal{M}_q; \emptyset)$.

A rule $(u, out) \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}$ at an instant t if membrane i is in $\mathcal{C}$ and multiset u is contained in such 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 (its father), this can be the environment in the case of the skin membrane.

A rule $(u, in) \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}$ at an instant t if membrane i is in $\mathcal{C}$ and multiset u is contained in the immediately upper region (its father), this is the environment in the case when the rule is associated with the skin membrane (the root of the tree μ). When applying a rule $(u, in) \in \mathcal{R}_i$, the multiset of objects u enters the region defined by the membrane i from the immediately upper region (its father), this is the environment in the case when the rule is associated with the skin membrane (the root of the tree μ).

A rule $(u, out; v, in) \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}$ at an instant t if membrane i is in $\mathcal{C}$ and multiset u is contained in such membrane, and multiset v is contained in the immediately upper region (its father). 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 region immediately outside (its father), at the same time bringing the objects specified by v into membrane i.

A rule $[a]_i \to [b]_i[c]_i \in \mathcal{R}_i$ is *applicable* to a configuration $\mathcal{C}$ at an instant t if the following holds: (a) membrane i is in $\mathcal{C}$; (b) object a is contained in such membrane; and (c) membrane i is neither the skin membrane nor the output membrane (if $i_{out} \in \{1, \ldots, q\}$). When applying a division rule $[a]_i \to [b]_i[c]_i$, 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, in the second one, object a is replaced by object c; all the other objects residing in membrane i are replicated and copies of them are placed in the two new membranes. The output membrane i_{out} cannot be divided.

The rules of a P system with symport/antiport rules and membrane division are applied in a non-deterministic maximally parallel manner (at each step we apply a multiset of rules which is maximal, no further applicable rule can be added), with the following important remark: if a membrane divides, then the division rule is the only one which is applied for that membrane at that step; the objects inside that membrane do not evolve by means of communication rules. In other words, before division a membrane interrupts all its communication channels with the other membranes and with the environment. The new membranes resulting from division will interact with other membranes or with the environment only at the next step – providing that they do not divide once again. The label of a membrane precisely identifies the rules which can be applied to it.

Let us fix a P system with symport/antiport rules and membrane division Π. We say that configuration $\mathcal{C}_1$ yields configuration $\mathcal{C}_2$ in one *transition step*, denoted

by $\mathcal{C}_1 \Rightarrow_{\Pi} \mathcal{C}_2$, if we can pass from $\mathcal{C}_1$ to $\mathcal{C}_2$ by applying the rules from $\mathcal{R}_1 \cup \cdots \cup \mathcal{R}_q$ following the previous remarks. A *computation* of Π is a (finite or infinite) sequence of configurations such that:

1. the first term of the sequence is the initial configuration of the system;
2. each non-initial configuration of the sequence is obtained from the previous configuration by applying rules of the system in a maximally parallel manner with the restrictions previously mentioned; and
3. if the sequence is finite (called *halting computation*) then the last term of the sequence is a *halting configuration* (a configuration where no rule of the system is applicable to it).

All computations start from an initial configuration and proceed as stated above; only halting computations give a result, which is encoded by the objects present in the output region i_{out} in the halting configuration.

If $\mathcal{C} = \{\mathcal{C}_t\}_{t<r+1}$ of Π ($r \in \mathbb{N}$) is a halting computation, then the *length of* $\mathcal{C}$, denoted by $|\mathcal{C}|$, is r, that is, $|\mathcal{C}|$ is the number of non-initial configurations which appear in the finite sequence $\mathcal{C}$. We denote by $\mathcal{C}_t(i), 1 \leq i \leq q$, the 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 $\mathcal{C}_t$. We denote by $\mathcal{C}_t(0)$ the multiset of objects over $\Gamma \setminus \mathcal{E}$ contained in the environment at configuration $\mathcal{C}_t$. Finally, we denote by $\mathcal{C}_t^*$ the multiset $\mathcal{C}_t(0)+\mathcal{C}_t(1)+\cdots+\mathcal{C}_t(q)$.

Definition 2. *A P system with symport/antiport rules and membrane division* $\Pi = (\Gamma, \mathcal{E}, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \cdots, \mathcal{R}_q, i_{out})$*, where* $\mathcal{E} = \emptyset$*, is called a P system with symport/antiport rules, membrane division and without environment.*

Usually, we omit the alphabet of the environment in the tuple describing such P system.

3 Recognizer P systems with symport/antiport rules

Let us recall that a *decision problem* is a pair (I_X, θ_X) where I_X is a language over a finite alphabet (whose elements are called *instances*) and θ_X is a total boolean function over I_X. Many abstract problems are not decision problems. For example, in *combinatorial optimization problems* some value must be optimized (minimized or maximized). In order to deal with such problems, they can be transformed into roughly equivalent decision problems by supplying a target/threshold value for the quantity to be optimized, and then asking whether this value can be attained.

There exists a correspondence between decision problems and formal languages. So that, the solvability of decision problems is defined through the recognition of the languages associated with them.

In order to study the computing efficiency of membrane systems, the notions from classical *computational complexity theory* are adapted for membrane computing, and a special class of cell-like P systems is introduced in [10]: *recognizer P systems* (called *accepting P systems* in a previous paper [9]).

Definition 3. *A* recognizer P system *with symport/antiport rules and membrane division of degree $q \geq 1$ is a tuple*

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

where:

- $(\Gamma, \mathcal{E}, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \cdots, \mathcal{R}_q, i_{out})$ *is a P system with symport/antiport rules and membrane division of degree $q \geq 1$, as defined in the previous section;*
- *The working alphabet Γ has two distinguished objects* `yes` *and* `no`*, at least one copy of them present in some initial multisets $\mathcal{M}_1$, ..., $\mathcal{M}_q$, but none of them is present in $\mathcal{E}$;*
- *Σ is an (input) alphabet strictly contained in Γ such that $\mathcal{E} \cap \Sigma = \emptyset$;*
- *$\mathcal{M}_1, \ldots, \mathcal{M}_q$ are multisets over $\Gamma \setminus \Sigma$;*
- *$i_{in} \in \{1, \ldots, q\}$ is the input membrane;*
- *The output region i_{out} is the environment;*
- *All computations halt;*
- *If $\mathcal{C}$ is a computation of Π, then either object* `yes` *or object* `no` *(but not both) must have been released into the output region (the environment), and only at the last step of the computation.*

Definition 4. *A* recognizer P system *with symport/antiport rules, membrane division and without environment of degree $q \geq 1$ is a tuple*

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

where:

- $(\Gamma, \mathcal{E}, \Sigma, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \cdots, \mathcal{R}_q, i_{out})$ *is a P system with symport/antiport rules and membrane division.*
- *The working alphabet Γ has two distinguished objects* `yes` *and* `no`*, at least one copy of them present in some initial multisets $\mathcal{M}_1$, ..., $\mathcal{M}_q$, but none of them is present in $\mathcal{E}$.*
- *$\mathcal{E} = \emptyset$.*
- *Σ is an (input) alphabet strictly contained in Γ such that $\mathcal{E} \cap \Sigma = \emptyset$.*
- *$\mathcal{M}_1, \ldots, \mathcal{M}_q$ are multisets over $\Gamma \setminus \Sigma$.*
- *$i_{in} \in \{1, \ldots, q\}$ is the input membrane.*
- *$i_{out} \in \{1, \ldots, q\}$ is the output membrane.*
- *All computations halt.*
- *If $\mathcal{C}$ is a computation of Π, then either object* `yes` *or object* `no` *(but not both) must have been released into the output region, and only at the last step of the computation.*

For each multiset $m \in \mathcal{M}_f(\Sigma)$, the *computation of the system Π with input $m \in \mathcal{M}_f(\Sigma)$* starts from the configuration of the form $(\mathcal{M}_1, \ldots, \mathcal{M}_{i_{in}} + m, \ldots, \mathcal{M}_q; \emptyset)$, that is, the input multiset m has been added to the contents of the input membrane i_{in}, and we denote it by $\Pi + m$. Therefore, we have an initial configuration

associated with each input multiset m (over the input alphabet Σ) in this kind of systems.

Given a recognizer P system with symport/antiport rules (with or without environment) and a halting computation $\mathcal{C} = \{\mathcal{C}_t\}_{t<r+1}$ of Π ($r \in \mathbb{N}$), we define the result of $\mathcal{C}$ as follows:

$$Output(\mathcal{C}) = \begin{cases} \texttt{yes}, & \text{if } \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{r,i_{out}}) = (1,0) \wedge \\ & \quad \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{t,i_{out}}) = (0,0) \text{ for } t = 0,\ldots,r-1 \\ \texttt{no}, & \text{if } \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{r,i_{out}}) = (0,1) \wedge \\ & \quad \Psi_{\{\texttt{yes},\texttt{no}\}}(M_{t,i_{out}}) = (0,0) \text{ for } t = 0,\ldots,r-1 \end{cases}$$

where Ψ is the Parikh mapping, and $M_{t,i_{out}}$ is the multiset over $\Gamma \setminus \mathcal{E}$ associated with the output region at the configuration $\mathcal{C}_t$, in particular, $M_{r,i_{out}}$ is the multiset over $\Gamma \setminus \mathcal{E}$ associated with the output region at the halting configuration $\mathcal{C}_r$.

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

Let us notice that if a recognizer P system

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

has a symport rule of the type $(i, \lambda/u, 0)$ then $alph(u) \cap (\Gamma \setminus \mathcal{E}) \neq \emptyset$, that is, the multiset u must contains some object from $\Gamma \setminus \mathcal{E}$ because on the contrary, all computations of Π would be not halting.

For each natural number $k \geq 1$, we denote by $\mathbf{CDC}(k)$ (respectively, $\mathbf{CDS}(k)$ or $\mathbf{CDA}(k)$) the class of recognizer P systems with membrane division and with symport/antiport rules (resp., allowing only symport or antiport rules) of length at most k. In the case of P systems without environment, we denote by $\widehat{\mathbf{CDC}}(k)$ ($\widehat{\mathbf{CDS}}(k)$ or $\widehat{\mathbf{CDA}}(k)$ respectively) the class of recognizer P systems with membrane division without environment and with symport/antiport rules (allowing only symport or only antiport rules respectively) of length at most k.

4 Polynomial Complexity Classes of P Systems with Symport/Antiport

In this section, we define what solving a decision problem in the framework of P systems with symport/antiport rules in a uniform and efficient way, means. Bearing in mind that they provide devices with a finite description, a numerable family of membrane systems will be necessary in order to solve a decision problem.

Definition 5. *We say that a decision problem $X = (I_X, \theta_X)$ is solvable in a uniform way and 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 (with or without environment) 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* $\Pi(s(u))$ *with input* $cod(u)$ *is halting and it performs at most* $p(|u|)$ *steps;*
  - *the family* $\mathbf{\Pi}$ *is* sound *with regard to* (X, cod, s), *that is, for each* $u \in I_X$, *if there exists an accepting computation of* $\Pi(s(u))$ *with input* $cod(u)$, *then* $\theta_X(u) = 1$;
  - *the family* $\mathbf{\Pi}$ *is* complete *with regard to* (X, cod, s), *that is, for each* $u \in I_X$, *if* $\theta_X(u) = 1$, *then every computation of* $\Pi(s(u))$ *with input* $cod(u)$ *is an accepting one.*

From the soundness and completeness conditions above we deduce that every P system $\Pi(n)$ is *confluent*, in the following sense: every computation of a system with the *same* input multiset must always give the *same* answer.

Let $\mathbf{R}$ be a class of recognizer P systems with symport/antiport rules. We denote by $\mathbf{PMC_R}$ the set of all decision problems which can be solved in a uniform way and polynomial time by means of families of systems from $\mathbf{R}$. The class $\mathbf{PMC_R}$ is closed under complement and polynomial–time reductions [9].

In what follows, we prove two technical results concerning recognizer P systems.

Proposition 1. *Let* $\Pi = (\Gamma, \mathcal{E}, \Sigma, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}_1, \cdots, \mathcal{R}_q, i_{in}, i_{out})$ *be a recognizer P systems with symport/antiport rules with length at most* k, $k \geq 2$, *and without membrane division. Let* $M = |\mathcal{M}_1 + \cdots + \mathcal{M}_q|$ *and let* $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_m)$ *be a computation of* Π *Then,* $|\mathcal{C}_0^*| = M$, *and for each* t, $0 \leq t < m$, *we have*

$$|\mathcal{C}_{t+1}^*| \leq |\mathcal{C}_t^*| \cdot k, \text{ and } |\mathcal{C}_{t+1}^*| \leq M \cdot k^t$$

Proof: Obviously, $|\mathcal{C}_0^*| = |\mathcal{C}_0(0) + \mathcal{C}_0(1) + \cdots + \mathcal{C}_0(q)| = |\mathcal{M}_1 + \cdots + \mathcal{M}_q| = M$. Suppose $0 \leq t < m$, and let us compute $\mathcal{C}_{t+1}^* = \mathcal{C}_{t+1}(0) + \mathcal{C}_{t+1}(1) + \cdots + \mathcal{C}_{t+1}(q)$. Bearing in mind that only the skin membrane can send and receive objects from the environment, we have

$$\mathcal{C}_{t+1}(0) + \mathcal{C}_{t+1}(2) + \mathcal{C}_{t+1}(3) + \cdots + \mathcal{C}_{t+1}(q) \subseteq \mathcal{C}_t(0) + \mathcal{C}_t(1) + \cdots + \mathcal{C}_t(q)$$

Next, let us see what objects membrane 1 can receive at step $t + 1$.

- On the one hand, membrane 1 can receive objects from $\mathcal{C}_t(0)$.
- On the other hand, membrane 1 can receive objects from $\mathcal{E}$ by means of rules in the skin membrane of the types:

- $(a\ e_{i_1} \dots\ e_{i_r}, in)$ with $a \in \mathcal{C}_t(0)$ and $e_{i_1}, \dots, e_{i_r} \in \mathcal{E}$, $r \leq k-1$.
- $(a, out; e_{i_1} \dots\ e_{i_r}, in)$ with $a \in \mathcal{C}_t(1)$ and $e_{i_1}, \dots, e_{i_r} \in \mathcal{E}$, $r \leq k-1$.

Then, $|\mathcal{C}_{t+1}(1)| \leq |\mathcal{C}_t(0) + \mathcal{C}_t(1)| \cdot (k-1)$. So, we have

$$\begin{aligned} |\mathcal{C}^*_{t+1}| &= |\mathcal{C}_{t+1}(0) + \mathcal{C}_{t+1}(2) + \mathcal{C}_{t+1}(3) + \cdots + \mathcal{C}_{t+1}(q)| + |\mathcal{C}_{t+1}(1)| \\ &\leq |\mathcal{C}_t(0) + \mathcal{C}_t(1) + \cdots + \mathcal{C}_t(q)| + |\mathcal{C}_t(0) + \mathcal{C}_t(1)| \cdot (k-1) \\ &\leq |\mathcal{C}^*_t| + |\mathcal{C}^*_t| \cdot (k-1) \leq |\mathcal{C}^*_t| \cdot k \end{aligned}$$

Finally, let us see that $|\mathcal{C}^*_{t+1}| \leq M \cdot k^t$ by induction on t. For $t = 1$ the result is trivial because of $|\mathcal{C}^*_1| \leq (|\mathcal{C}^*_0| + M) \cdot (k-1) = 2M \cdot (k-1)$.

Let t be such that $1 < t < m$ and the result holds for t. Then,

$$|\mathcal{C}^*_{t+1}| \leq |\mathcal{C}^*_t| \cdot k \overset{h.i}{\leq} M \cdot k^{t-1} \cdot k = M \cdot k^t$$

□

Proposition 2. *Let* $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ *a family of recognizer P systems from* $\mathbf{CDC}(k)$*, where* $k \geq 2$*, solving a decision problem* $X = (I_X, \theta_X)$ *in polynomial time according to Definition 5. Let* (cod, s) *be a polynomial encoding associated with that solution. There exists a polynomial function* $r(n)$ *such that for each instance* $u \in I_X$*,* $2^{r(|u|)}$ *is an upper bound of the number of objects in all membranes of the system* $\Pi(s(u)) + cod(u)$ *along any computation.*

Proof: Let $p(n)$ be a polynomial function 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.

Let $u \in I_X$ be an instance of X and

$$\Pi(s(u)) + cod(u) = (\Gamma, \mathcal{E}, \Sigma, \mathcal{M}_1, \dots, \mathcal{M}_q, \mathcal{R}_1, \dots, \mathcal{R}_q, i_{in}, i_{out})$$

Let $M = |\mathcal{M}_1 + \cdots + \mathcal{M}_q|$. Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots, \mathcal{C}_m)$, $0 \leq m \leq p(|u|)$, be a computation of Π.

First, let us suppose that we apply only communication rules at m consecutive transition steps. From Proposition 1 we deduce that $|\mathcal{C}^*_0| = M$ and $|\mathcal{C}^*_{t+1}| \leq M \cdot k^t$, for each t, $0 \leq t < m$. Thus, if we apply in a consecutive way the maximum possible number of communication rules (without applying any division rules) to the system $\Pi(s(u)) + cod(u)$, in any instant of any computation of the system, $M \cdot k^{p(|u|)}$ is an upper bound of the number of objects in the whole system.

Now, let us consider the effect of applying in a consecutive way the maximum possible number of division rules (without applying any communication rules) to the system $\Pi(s(u)) + cod(u)$ when the initial configuration has $M \cdot k^{p(|u|)}$ objects. After that, an upper bound of the number of objects in the whole system by any computation is $M \cdot k^{p(|u|)} \cdot 2^{p(|u|)} \cdot p(|u|)$. Then, we consider a polynomial function $r(n)$ such that $r(|u|) \geq \log(M) + p(|u|) \cdot (1 + \log k) + \log(p(|u|))$, for each instance $u \in I_X$. The polynomial function $r(n)$ fulfills the property required.

□

Corollary 1. *Let* $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ *a family of recognizer P systems with symport/antiport rules and membrane division, solving a decision problem* $X = (I_X, \theta_X)$ *in polynomial time according to Definition 5. Let* (cod, s) *a polynomial encoding associated with that solution. Then, there exists a polynomial function* $r(n)$ *such that for each instance* $u \in I_X$, $2^{r(|u|)}$ *is an upper bound of the number of objects from* $\mathcal{E}$ *which are moved from the environment to all membranes of the system* $\Pi(s(u)) + cod(u)$ *along any computation.*

Proof: It suffices to note that from Proposition 2 there exists a polynomial function $r(n)$ such that for each instance $u \in I_X$, $2^{r(|u|)}$ is an upper bound of the number of objects in all membranes of the system $\Pi(s(u)) + cod(u)$. □

5 Simulating Systems from CDC(k) by Means of Systems from $\widehat{\text{CDC}}(k)$

The goal of this section is to show that any P system with symport/antiport rules and membrane division can be simulated by a P system symport/antiport rules, membrane division and without environment, in an efficient way.

First of all, we define the meaning of efficient simulations in the framework of recognizer P systems with symport/antiport rules.

Definition 6. *Let* Π *and* Π' *be recognizer P systems with symport/antiport rules. We say that* Π' simulates Π in an efficient way *if the following holds:*

1. Π' *can be constructed from* Π *by a deterministic Turing machine working in polynomial time.*
2. *There exists an injective function,* f*, from the set* $\mathbf{Comp}(\Pi)$ *of computations of* Π *onto the set* $\mathbf{Comp}(\Pi')$ *of computations of* Π' *such that:*
   - ⋆ *There exists a deterministic Turing machine that constructs computation* $f(\mathcal{C})$ *from computation* $\mathcal{C}$ *in polynomial time.*
   - ⋆ *A computation* $\mathcal{C} \in \mathbf{Comp}(\Pi)$ *is an accepting computation if and only if* $f(\mathcal{C}) \in \mathbf{Comp}(\Pi')$ *is an accepting one.*
   - ⋆ *There exists a polynomial function* $p(n)$ *such that for each* $\mathcal{C} \in \mathbf{Comp}(\Pi)$ *we have* $|f(\mathcal{C})| \leq p(|\mathcal{C}|)$.

Now, for every family of recognizer P system with symport/antiport rules and membrane division solving a decision problem, we design a family of recognizer P systems with symport/antiport rules, membrane division and *without environment* efficiently simulating it, according to Definition 6.

In what follows throughout this Section, let $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ a family of recognizer P systems with symport/antiport rules and membrane division solving a decision problem $X = (I_X, \theta_X)$ in polynomial time according to Definition 5, and let $r(n)$ be a polynomial function such that for each instance $u \in I_X$, $2^{r(|u|)}$ is an upper bound of the number of objects from $\mathcal{E}$ which are moved from the environment to all membranes of the system by any computation of $\Pi(s(u)) + cod(u)$.

Definition 7. *For each $n \in \mathbb{N}$, let*

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

an element of the previous family $\mathbf{\Pi}$*, and for the sake of simplicity we denote r instead of $r(n)$ and 1 is the label of the skin membrane. Let us consider the recognizer P system with symport/antiport rules of degree $q_1 = 1 + q \cdot (r+2) + |\mathcal{E}|$, with membrane division and without environment*

$$\mathbf{S}(\Pi(n)) = (\Gamma', \Sigma', \mu', \mathcal{M}'_0, \mathcal{M}'_1, \ldots, \mathcal{M}'_{q_1}, \mathcal{R}'_0, \mathcal{R}'_1, \ldots, \mathcal{R}'_{q_1}, i'_{in}, i'_{out})$$

defined as follows:

- $\Gamma' = \Gamma \ \cup \ \{\alpha_i : \ 0 \leq i \leq r-1\}$.
- $\Sigma' = \Sigma$.
- *Each membrane $i \in \{1, \ldots, q\}$ of Π provides a membrane of $\mathbf{S}(\Pi(n))$ with the same label. In addition, $\mathbf{S}(\Pi(n))$ has:*
  - ⋆ *$r+1$ new membranes, labelled by $(i,0), (i,1), \ldots, (i,r)$, respectively, for each $i \in \{1, \ldots, q\}$.*
  - ⋆ *A distinguished membrane labelled by 0.*
  - ⋆ *A new membrane, labelled by l_b, for each $b \in \mathcal{E}$.*
- *μ' is the rooted tree obtained from μ as follows:*
  - ⋆ *Membrane 0 is the root of μ' and it is the father of the root of μ.*
  - ⋆ *For each $b \in \mathcal{E}$, membrane 0 is the father of membrane l_b.*
  - ⋆ *We consider a linear structure whose nodes are $(i,0), (i,1), \ldots, (i,r)$ and such that (i,j) is the father of $(i,j-1)$, for each $1 \leq i \leq q$ and $1 \leq j \leq r$.*
  - ⋆ *For each membrane i of μ we add the previous linear structure being membrane i the father of membrane (i,r).*
- *Initial multisets: $\mathcal{M}'_0 = \emptyset$, $\mathcal{M}'_{l_b} = \{\alpha_0\}$, for each $b \in \mathcal{E}$, and*

$$(1 \leq i \leq q) \begin{cases} \mathcal{M}'_{(i,0)} = \mathcal{M}_i \\ \mathcal{M}'_{(i,1)} = \ \emptyset \\ \ldots \qquad \ldots \\ \mathcal{M}'_{(i,r)} = \ \emptyset \\ \mathcal{M}'_i \quad = \ \emptyset \end{cases}$$

- *Set of rules:*

$$\mathcal{R}'_0 \cup \mathcal{R}'_1 \cup \cdots \cup \mathcal{R}'_q \cup \{\mathcal{R}'_{(i,j)} : 1 \leq i \leq q, 0 \leq j \leq r\} \cup \{\mathcal{R}'_{l_b} : b \in \mathcal{E}\}$$

where $\mathcal{R}'_0 = \emptyset$, $\mathcal{R}'_i = \mathcal{R}_i$ for $1 \leq i \leq q$, and

$$\begin{array}{rl} \mathcal{R}'_{(i,j)} & = \{(a, out; \lambda, in) : \ a \in \Gamma\}, \ \text{for } 1 \leq i \leq q \ \wedge \ 0 \leq j \leq r\} \\ \mathcal{R}'_{l_b} & = \{[\alpha_j]_{l_b} \to [\alpha_{j+1}]_{l_b} \ [\alpha_{j+1}]_{l_b} : \ 0 \leq j \leq r-2\} \cup \\ & \quad \{[\alpha_{r-1}]_{l_b} \to [b]_{l_b} \ [b]_{l_b}, (l_b, out; \lambda, in)\}, \ \text{for } b \in \mathcal{E} \end{array}$$

- $i'_{in} = (i_{in}, 0)$, *and* $i'_{out} = 0$.

Let us notice that $\mathbf{S}(\Pi(n))$ can be considered as an extension of $\Pi(n)$ without environment, in the following sense:

- $\star$ $\Gamma \subseteq \Gamma', \Sigma \subseteq \Sigma'$ and $\mathcal{E} = \emptyset$.
- $\star$ Each membrane in Π is also a membrane in $\mathbf{S}(\Pi(n))$.
- $\star$ There is a distinguished membrane in $\mathbf{S}(\Pi(n))$ labelled by 0 which plays the role of environment of $\Pi(n)$.
- $\star$ μ is a subtree of μ'.
- $\star$ $\mathcal{R} \subseteq \mathcal{R}'$, and now 0 is the label of a "ordinary membrane" in $\mathbf{S}(\Pi(n))$.

Next, we analyze the structure of the computations of system $\mathbf{S}(\Pi(n))$ and we compare them with the computations of $\Pi(n)$.

Lemma 1. *Let $\mathcal{C}' = (\mathcal{C}'_0, \mathcal{C}'_1, \dots)$ be a computation of $\mathbf{S}(\Pi(n))$. For each t $(1 \leq t \leq r)$ the following holds:*

- *$\mathcal{C}'_t(i) = \emptyset$, for $0 \leq i \leq q$.*
- *For each $1 \leq i \leq q$, and $0 \leq j \leq r$ we have:*

$$\mathcal{C}'_t(i,j) = \begin{cases} \mathcal{M}_i, \text{ if } & j = t \\ \emptyset, \text{ if } & j \neq t \end{cases}$$

- *For each $b \in \mathcal{E}$, there exist 2^t membranes labelled by l_b whose father is membrane 0 and their content is:*

$$\mathcal{C}'_t(l_b) = \begin{cases} \{\alpha_t\}, \text{ if } & 1 \leq t \leq r-1 \\ \{b\}, \text{ if } & t = r \end{cases}$$

Proof: By induction on t.

Let us start with the basic case $t = 1$. The initial configuration of system $\mathbf{S}(\Pi(n))$ is the following:

- $\mathcal{C}'_0(i) = \emptyset$, for $0 \leq i \leq q$.
- For each $1 \leq i \leq q$ we have $\mathcal{C}'_0(i,0) = \mathcal{M}_i$, and $\mathcal{C}'_0(i,j) = \emptyset$, for $1 \leq j \leq r$.
- For each $b \in \mathcal{E}$, there exists only one membrane labelled by l_b whose contents is $\{\alpha_0\}$.

At configuration $\mathcal{C}'_0$, only the following rules are applicable:

- $[\alpha_0]_{l_b} \to [\alpha_1]_{l_b}\ [\alpha_1]_{l_b}$, for each $b \in \mathcal{E}$.
- $\big(a, out; \lambda, in\big) \in \mathcal{R}_{(i,0)}$, for each $a \in supp(\mathcal{M}_i)$.

Thus,

(a) For each i $(1 \leq i \leq q)$ we have:

$$\begin{cases} \mathcal{C}'_1(i) & = \emptyset \\ \mathcal{C}'_1(0) & = \emptyset \\ \mathcal{C}'_1(i,0) & = \emptyset \\ \mathcal{C}'_1(i,1) & = \mathcal{M}_i \\ \mathcal{C}'_1(i,j) & = \emptyset, \text{ for } 2 \leq j \leq r \end{cases}$$

(b) For each $b \in \mathcal{E}$, there are 2 membranes labelled by l_b whose father is membrane 0 and their content is $\{\alpha_1\}$.

Hence, the result holds for $t = 1$.

By induction hypothesis, let t be such that $1 \leq t < r$, and let us suppose the result holds for t, that is,

- $\mathcal{C}'_t(i) = \emptyset$, for $0 \leq i \leq q$.
- For each $1 \leq i \leq q$, and $0 \leq j \leq r$ we have:

$$\mathcal{C}'_t(i,j) = \begin{cases} \mathcal{M}_i, \text{ if } & j = t \\ \emptyset, \text{ if } & j \neq t \end{cases}$$

- For each $b \in \mathcal{E}$, there exist 2^t membranes labelled by l_b whose father is membrane 0 and their content is $\mathcal{C}'_t(l_b) = \{\alpha_t\}$ (because $t \leq r-1$).

Then, at configuration $\mathcal{C}'_t$ only the following rules are applicable:

(1) If $t \leq r-2$, the rules $[\alpha_t]_{l_b} \rightarrow [\alpha_{t+1}]_{l_b}\ [\alpha_{t+1}]_{l_b}$, for each $b \in \mathcal{E}$.
(2) If $t = r-1$, the rules $[\alpha_t]_{l_b} \rightarrow [b]_{l_b}\ [b]_{l_b}$, for each $b \in \mathcal{E}$.
(3) $\big(a, out; \lambda, in\big) \in \mathcal{R}_{(i,t)}$, for each $a \in supp(\mathcal{M}_i)$.

From the application of rules of types (1) or (2) at configuration $\mathcal{C}'_t$, we deduce that there are 2^{t+1} membranes labelled by l_b in $\mathcal{C}'_{t+1}$, for each $b \in \mathcal{E}$, whose father is membrane 0 and their content is $\{\alpha_{t+1}\}$, if $t \leq r-2$, or $\{b\}$, if $t = r-1$.

From the application of rules of type (3) at configuration $\mathcal{C}'_t$, we deduce that

$$\mathcal{C}'_{t+1}(i,j) = \begin{cases} \mathcal{M}_i, \text{ if } & j = t+1 \\ \emptyset, \text{ if } & 0 \leq j \leq r \ \wedge\ j \neq t+1 \end{cases}$$

Bearing in mind that no other rule of system $\mathbf{S}(\Pi(n))$ is applicable, we deduce that $\mathcal{C}'_{t+1}(i) = \emptyset$, for $0 \leq i \leq q$.

This completes the proof of this Lemma.

□

Lemma 2. *Let $\mathcal{C}' = (\mathcal{C}'_0, \mathcal{C}'_1, \dots)$ be a computation of the P system $\mathbf{S}(\Pi(n))$. Configuration $\mathcal{C}'_{r+1}$ is the following:*

(1) $\mathcal{C}'_{r+1}(0) = b_1^{2^r} \dots b_\alpha^{2^r}$, where $\mathcal{E} = \{b_1, \dots, b_\alpha\}$.
(2) $\mathcal{C}'_{r+1}(i) = \mathcal{M}_i = \mathcal{C}_0(i)$, for $1 \leq i \leq q$.
(3) $\mathcal{C}'_{r+1}(i,j) = \emptyset$, for $1 \leq i \leq q$, $0 \leq j \leq r$.
(4) For each $b \in \mathcal{E}$, there exist 2^r membranes labelled by l_b whose father is membrane 0 and their content is empty.

Proof: From Lemma 1, the configuration $\mathcal{C}'_r$ is the following:

- $\mathcal{C}'_r(i) = \emptyset$, for $0 \leq i \leq q$.
- For each i $(1 \leq i \leq q)$ we have

$$\mathcal{C}'_r(i,j) = \begin{cases} \mathcal{M}_i, \text{ if } & j = r \\ \emptyset, \text{ if } & j \neq r \end{cases}$$

- For each $b \in \mathcal{E}$, there exist 2^r membranes labelled by l_b whose father is membrane 0 and their content is $\{b\}$.

At configuration $\mathcal{C}'_r$ only the following rules are applicables:

- $\big(a, out; \lambda, in\big) \in \mathcal{R}_{(i,r)}$, for each $a \in \Gamma \cap supp(\mathcal{M}_i)$.
- $\big(b, out; \lambda, in\big) \in \mathcal{R}_{l_b}$, for each $b \in \mathcal{E}$.

Thus,

- $\mathcal{C}'_{r+1}(0) = b_1^{2^r} \dots b_\alpha^{2^r}$, where $\mathcal{E} = \{b_1, \dots, b_\alpha\}$.
- $\mathcal{C}'_{r+1}(i) = \mathcal{M}_i = \mathcal{C}_0(i)$, for $1 \leq i \leq q$.
- $\mathcal{C}'_{r+1}(i,j) = \emptyset$, for $1 \leq i \leq q$ and $0 \leq j \leq r$.
- For each $b \in \mathcal{E}$, there exist 2^r membranes labelled by l_b whose father is membrane 0 and their content is empty.

□

Definition 8. *Let $\mathcal{C} = (\mathcal{C}_0, \mathcal{C}_1, \dots, \mathcal{C}_m)$ be a halting computation of $\Pi(n)$. Then we define the computation $\mathbf{S}(\mathcal{C}) = (\mathcal{C}'_0, \mathcal{C}'_1, \dots, \mathcal{C}'_r, \mathcal{C}'_{r+1}, \dots, \mathcal{C}'_{r+1+m})$ of $\mathbf{S}(\Pi(n))$ as follows:*

(1) *The initial configuration is:*
$$\begin{cases} \mathcal{C}'_0(i) & = \emptyset, \text{ for } 0 \leq i \leq q \\ \mathcal{C}'_0(i,0) & = \mathcal{C}_0(i), \text{ for } 1 \leq i \leq q \\ \mathcal{C}'_0(i,j) & = \emptyset, \text{ for } 1 \leq i \leq q \text{ and } 1 \leq j \leq r \\ \mathcal{C}'_0(l_b) & = \alpha_0, \text{ for each } b \in \mathcal{E} \end{cases}$$
(2) *The configuration $\mathcal{C}'_t$, for $1 \leq t \leq r$, is described by Lemma 1.*
(3) *The configuration $\mathcal{C}'_{r+1}$ is described by Lemma 2.*
(4) *The configuration $\mathcal{C}'_{r+1+s}$, for $0 \leq s \leq m$, coincides with the configuration $\mathcal{C}_s$ of Π, that is, $\mathcal{C}_s(i) = \mathcal{C}'_{r+1+s}(i)$, for $1 \leq i \leq q$. The content of the remaining membranes (excluding membrane 0) at configuration $\mathcal{C}'_{r+1+s}$ is equal to the content of that membrane at configuration $\mathcal{C}'_{r+1}$, that is, these membranes do not evolve after step $r+1$.*

That is, every computation $\mathcal{C}$ of $\Pi(n)$ can be "reproduced" by a computation $\mathbf{S}(\mathcal{C})$ of $\mathbf{S}(\Pi(n))$ with a delay: from step $r+1$ to step $r+1+m$, the computation $\mathbf{S}(\mathcal{C})$ restricted to membranes $1, \dots, q$ provides the computation $\mathcal{C}$ of $\Pi(n)$.

From Lemma 1 and Lemma 2 we deduce the following:

(a) $\mathbf{S}(\mathcal{C})$ is a computation of $\mathbf{S}(\Pi(n))$.
(b) $\mathbf{S}$ is an injective function from $\mathbf{Comp}(\Pi(n))$ onto $\mathbf{Comp}(S(\Pi(n)))$.

Proposition 3. *The P system $\mathbf{S}(\Pi(n))$ defined in Definition 7 simulates $\Pi(n)$ in an efficient way.*

Proof. In order to show that $\mathbf{S}(\Pi(n))$ can be constructed from $\Pi(n)$ by a deterministic Turing machine working in polynomial time, it is enough to note that the amount of resources needed to construct $\mathbf{S}(\Pi(n))$ from $\Pi(n)$ is polynomial in the size of the initial resources of $\Pi(n)$. Indeed,

1. The size of the alphabet of $\mathbf{S}(\Pi(n))$ is $|\Gamma'| = |\Gamma| + r$.
2. The initial number of membranes of $\mathbf{S}(\Pi(n))$ is $1 + q \cdot (r + 2) + |\mathcal{E}|$.
3. The initial number of objects of $\mathbf{S}(\Pi(n))$ is the initial number of objects of $\Pi(n)$ plus $|\mathcal{E}|$.
4. The number of rules of $\mathbf{S}(\Pi(n))$ is $|\mathcal{R}'| = |\mathcal{R}| + (r + 1) \cdot |\mathcal{E}| + |\Gamma| \cdot q \cdot (r + 1)$.
5. The maximal length of a communication rule of $\mathbf{S}(\Pi(n))$ is equal to the maximal length of a communication rule of $\Pi(n)$.

From Lemma 1 and Lemma 2 we deduce that:

(a) Every computation $\mathcal{C}'$ of $\mathbf{S}(\Pi(n))$ has associated a computation $\mathcal{C}$ of $\Pi(n)$ such that $\mathbf{S}(\mathcal{C}) = \mathcal{C}'$ in a natural way.
(b) The function $\mathbf{S}$ is injective.
(c) A computation $\mathcal{C}$ of $\Pi(n)$ is an accepting computation if and only if $\mathbf{S}(\mathcal{C})$ is an accepting computation of $\mathbf{S}(\Pi(n))$.

Finally, let us notice that if $\mathcal{C}$ is a computation of $\Pi(n)$ with length m, then $\mathbf{S}(\mathcal{C})$ is a computation of $\mathbf{S}(\Pi(n))$ with length $r + 1 + m$. □

6 Computational Complexity Classes of P Systems with Membrane Division and Without Environment

In this Section, we analyze the role of the environment in the efficiency of P systems with membrane division. That is, we study the ability of these P systems with respect to the computational efficiency when the alphabet of the environment is an empty set.

Theorem 1. *For each $k \in \mathbb{N}$ we have* $\mathbf{PMC}_{\mathbf{CDC}(k+1)} = \mathbf{PMC}_{\widehat{\mathbf{CDC}}(k+1)}$.

Proof: Let us recall that $\mathbf{PMC}_{\mathbf{CDC}(1)} = \mathbf{P}$ (see [4] for details). Then,

$$\mathbf{P} \subseteq \mathbf{PMC}_{\widehat{\mathbf{CDC}}(1)} \subseteq \mathbf{PMC}_{\mathbf{CDC}(1)} = \mathbf{P}$$

Thus, the result holds for $k = 0$. Let us show the result holds for $k \geq 1$. Since $\widehat{\mathbf{CDC}}(k+1) \subseteq \mathbf{CDC}(k+1)$ it suffices to prove that $\mathbf{PMC}_{\mathbf{CDC}(k+1)} \subseteq \mathbf{PMC}_{\widehat{\mathbf{CDC}}(k+1)}$. For that, let $X \in \mathbf{PMC}_{\mathbf{CDC}(\mathbf{k+1})}$.

Let $\{\Pi(n) \mid n \in N\}$ be a family of P systems from $\mathbf{CDC}(k+1)$ solving X according to Definition 5. Let (cod, s) be a polinomial encoding associated with that solution. Let $u \in I_X$ be an instance of the problem X that will be processed by the system $\Pi(s(u)) + cod(u)$. According to Proposition 2, let $r(n)$ be a polynomial function that $2^{r(|u|)}$ is an upper bound of the number of objects from $\mathcal{E}$ which are moved from the environment to all membranes of the system by any computation of

$$\Pi(s(u)) + cod(u) = (\Gamma, \mathcal{E}, \Sigma, \mathcal{M}_1, \ldots, \mathcal{M}_{i_{in}} + cod(u), \ldots, \mathcal{M}_q, \mathcal{R}, i_{in}, i_{out})$$

Then, we consider the P system without environment

$$\mathbf{S}(\Pi(s(u))) + cod(u) = (\Gamma', \Sigma', \mathcal{M}'_0, \mathcal{M}'_1, \ldots, \mathcal{M}'_{i_{in}} + cod(u), \ldots, \mathcal{M}'_{q_1}, \mathcal{R}', i'_{in}, i'_{out})$$

according to Definition 7, where $q_1 = 1 + q \cdot (r(|u|) + 2) + |\mathcal{E}|$.

Therefore, $\mathbf{S}(\Pi(s(u))) + cod(u)$ is a P system from $\widehat{\mathbf{CDC}}(k+1)$ such that verifies the following:

- A distinguished membrane labelled by 0 has been considered, which will play the role of the environment at the system $\Pi(s(u)) + cod(u)$.
- At the initial configuration, it has enough objects in membrane 0 in order to simulate the behaviour of the environment of the system $\Pi(s(u))) + cod(u)$.
- After $r(n) + 1$ step, computations of $\Pi(s(u)) + cod(u)$ are reproduced by the computations of $\mathbf{S}(\Pi(s(u))) + cod(u)$ exactly.

Let us suppose that $\mathcal{E} = \{b_1, \ldots, b_\alpha\}$. In order to simulate $\Pi(s(u)) + cod(u)$ by a P system without environment in an efficient way, we need to have enough objects in the membrane of $\mathbf{S}(\Pi(s(u))) + cod(u)$ labelled by 0 available. Specifically, $2^{r(n)}$ objects in that membrane are enough.

In order to start the simulation of any computation $\mathcal{C}$ of $\Pi(s(u)) + cod(u)$, it would be enough to have $2^{r(n)}$ copies of each object $b_j \in \mathcal{E}$ in the membrane of $\mathbf{S}(\Pi(s(u))) + cod(u)$ labelled by 0. For this purpose,

- For each $b \in \mathcal{E}$ we consider a membrane in $\mathbf{S}(\Pi(s(u))) + cod(u)$ labelled by l_b which only contains object α_0 initially. We also consider the following rules:
  - $[\alpha_j]_{l_b} \to [\alpha_{j+1}]_{l_b}\ [\alpha_{j+1}]_{l_b}$, for $0 \leq j \leq r(|u|) - 2$,
  - $[\alpha_{p(n)-1}]_{l_b} \to [b]_{l_b}\ [b]_{l_b}$,
  - $(l_b, b/\lambda, 0)$.
- By applying the previous rules, after $r(|u|)$ transition steps we get $2^{r(|u|)}$ membranes labelled by l_b, for each $b \in \mathcal{E}$ in such a way that each of them contains only object b. Finally, by applying the third rule we get $2^{r(|u|)}$ copies of objects b in membrane 0, for each $b \in \mathcal{E}$.

Therefore, after the execution of $r(|u|) + 1$ transition steps in each computation of $\mathbf{S}(\Pi(s(u))) + cod(u)$ in membrane 0 of the corresponding configuration, we have $2^{r(|u|)}$ copies of each object $b \in \mathcal{E}$. This number of copies is enough to simulate any computation $\mathcal{C}$ of $\Pi(s(u)) + cod(u)$ through the system $\mathbf{S}(\Pi(s(u)) + cod(u))$.

From Proposition 3 we deduce that the family $\{\mathbf{S}(\Pi(n))|\ n \in N\}$ solves X in polynomial time according to Definition 5. Hence, $X \in \mathbf{PMC}_{\widehat{\mathbf{CDC}}(k+1)}$. □

7 Conclusions and Further Works

Initial configurations of ordinary P systems with symport/antiport rules have an arbitrarily large amount of copies of some kind of objects belonging to a distinguished alphabet which specifies the *environment* of the system.

The previous condition is no too nice from the computational complexity point of view. In this paper, we show that in P systems with with symport/antiport rules and membrane division the environment can be "removed" without a loss of efficiency.

Acknowledgements

The work of L. Pan was supported by National Natural Science Foundation of China (61033003, 91130034 and 61320106005). The work of M.J. Pérez and L.F. Macías was supported by Project TIN2012-37434 of the Ministerio de Ciencia e Innovación of Spain.

References

1. Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A. Romero-Jiménez. . Computational efficiency of cellular division in tissue-like P systems. *Romanian Journal of Information Science and Technology*, **11**, 3 (2008), 229-241.
2. Gutiérrez-Escudero, R., Pérez-Jiménez, M.J. and Rius-Font, M. Characterizing tractability by tissue-like P systems. *Lecture Notes in Computer Science* **5957**, 5957 (2010), 289-300.
3. Macías-Ramos, L.F., Pérez-Jiménez, M.J., Riscos-Núñez, A., Rius-Font, M. and Valencia-Cabrera, L. The efficiency of tissue P systems with cell separation rely on the environment. Submitted, 2012
4. Macías-Ramos, L.F., Song B., Song T., Pan L., Pérez-Jiménez, M.J. Limits on efficient computation in P systems with symport/antiport rules. Proceedings of the Fifteenth Brainstorming Week on Membrane Computing. Submitted 2017.
5. Pan, L. and Ishdorj, T.-O. P systems with active membranes and separation rules. *Journal of Universal Computer Science*, **10**, 5, (2004), 630–649.
6. Păun, Gh. Attacking **NP**-complete problems. In *Unconventional Models of Computation, UMC'2K* (I. Antoniou, C. Calude, M. J. Dinneen, eds.), Springer-Verlag, 2000, 94-115.
7. Păun, Gh. *Membrane Computing. An Introduction*. Springer–Verlag, Berlin, (2002).
8. Păun, Gh., Pérez-Jiménez, M.J. and Riscos-Núñez, A. Tissue P System with cell division. *In. J. of Computers, communications & control*, **3**, 3, (2008), 295–303.
9. Pérez-Jiménez, M.J., Romero-Jiménez, A. and Sancho-Caparrini, F. Complexity classes in models of cellular computing with membranes. *Natural Computing*, **2**, 3 (2003), 265–285.
10. Pérez-Jiménez, M.J., Romero-Jiménez, A. and Sancho-Caparrini, F. A polynomial complexity class in P systems using membrane division. *Journal of Automata, Languages and Combinatorics*, **11**, 4, (2006), 423-434.
11. Porreca, A.E., Murphy, N. Pérez-Jiménez, M.J. An efficient solution of Ham Cycle problem in tissue P systems with cell division and communication rules with length at most 2. In M. Garca-Quismondo, L.F. Macas-Ramos, Gh. Paun, I. Prez Hurtado, L. Valencia-Cabrera (eds.) *Proceedings of the Tenth Brainstorming Week on Membrane*

Computing, Volume II, Seville, Spain, January 30- February 3, 2012, Report RGNC 01/2012, Fnix Editora, 2012, pp. 141-166.