# Solving Problems Through a Single Membrane System

David Orellana-Martín, Luis Valencia-Cabrera,
Agustín Riscos-Núñez, Mario J. Pérez-Jiménez

Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
Universidad de Sevilla
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
{dorellana,lvalencia,ariscosn,marper}@us.es

**Summary.** The tape of a deterministic Turing machine contains an unbounded number of cells. Thanks to that, a single machine can solve decision problems with an infinite number of instances. Nevertheless, in the framework of membrane computing, traditionally a "solution" to an abstract decision problem consists of a *family* of membrane systems (where each system of the family is associated with a finite set of instances of the problem to be solved). An interesting question is to analyze the possibility to find a single membrane system able to deal with the infinitely many instances of a decision problem.

In this context, it is fundamental to define precisely how the instances of the problem are introduced into the system. In this paper, two different methods are considered.

The first one relies on a pre-computing process, where a polynomial-time computable function will be in charge of producing a multiset of objects associated with the instance to be solved. On the other hand, the second one assumes that the input alphabet of the system is equal to the alphabet of instances, and therefore instances are directly introduced in the initial configuration of the system. Polynomial complexity classes associated with these two approaches are introduced and some complexity aspects are studied.

## 1 Introduction

In the 17th Brainstorming Week on Membrane Computing, an apparently innocent problem was presented by the authors: the `ONLY-ONE-OBJECT` problem. The goal is to build a system able to distinguish whether in a given region, at a given moment, there is only one copy of an object, or if the multiplicity of the object is strictly greater than one. Besides, the notion of efficient solvability by means of a single recognizer polarizationless P system with active membranes, without dissolution rules and using division for elementary and non-elementary membranes, was proposed. Following a reasoning based on the dependency graph technique, a

negative answer to the previous question was concluded (i.e. the problem is not solvable in the proposed framework).

In some sense, the previous question links up with others that were proposed by P. Sosík [17], which raise the possibility of being able to solve **P**-complete problems or **NP**-complete problems by means of a single membrane system. Specifically, two "open problems" were "formulated" in [17], expressed in an informal way as follows:

- **Open Problem 1.** *Is there any known standard model of P system capable of solving a P-complete problem in polynomial time without the use of families, i.e., all instances are solved by the same P systems?*
- **Open Problem 2.** *How to design a natural (not much "extraordinary") model of P system capable of solving an NP-complete problem in polynomial time without the use of families?*

Of course, these questions should be expressed in a formal way and their answers will depend on the definitions given about what *solving a decision problem through a single membrane system* means.

For instance, two possible definitions could be considered according to the way of entering the input inside the membrane system: (a) by using *precomputed* resources (that is, waiting for a polynomial time *prior* to the initial step of the computation, to calculate which is the input multiset that has to be provided to the system); or (b) by directly introducing the input multiset without any preprocessing, that is, *free* of external resources.

For a comprehensive introduction to membrane systems, we refer the reader to [12, 15].

## 2 The Complexity Class $\mathbf{PMC}_{\mathcal{R}}^{\mathbf{1}p}$

First, let us define a solution to a decision problem through a single membrane system allowing the possibility to use (external) *precomputed* resources for providing the input multiset to the system. In other words, we assume that there is an available device able to execute the function that computes the input multiset, and this process should be performed before the computation of the membrane system starts.

**Definition 1.** *Let $\mathcal{R}$ be a class of recognizer membrane system. Let $X = (I_X, \theta_X)$ be a decision problem. We say that problem $X$ is solvable in polynomial time by a single membrane system $\Pi$ from $\mathcal{R}$ with precomputed resources, denoted by $X \in \mathbf{PMC}_{\mathcal{R}}^{1p}$, if the following hold:*

- *There exists a* polynomial encoding *cod from $X$ to $\Pi$ providing a "reasonable encoding scheme" which maps problem instances into the multisets describing them [3]; that is, there exists a polynomial time computable function, cod, whose domain is $I_X$ such that for every instance $u \in I_X$, $cod(u)$ is a multiset over the input alphabet of $\Pi$.*

- *The system $\Pi$ is* polynomially bounded *with regard to $(X, cod)$; that is, there exists a polynomial $p(r)$ such that for each instance $u \in I_X$, every computation of the system $\Pi$ with input multiset $cod(u)$ performs at most $p(|u|)$ steps.*
- *The system $\Pi$ is* sound *with regard to $(X, cod)$; that is, for each instance $u \in I_X$, if there exists an accepting computation of the system $\Pi$ with input multiset $cod(u)$ then $\theta_X(u) = 1$.*
- *The system $\Pi$ is* complete *with regard to $(X, cod)$; that is, for each instance $u \in I_X$ such that $\theta_X(u) = 1$, every computation of the system $\Pi$ with input multiset $cod(u)$ is an accepting computation.*

In this definition, the input multiset that is allocated into the initial configuration of the system is precomputed by means of a polynomial-time computable function.

**Proposition 1.** *If $\mathcal{R}$ is a class of recognizer membrane systems, then*

$$\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{R}}^{1p} \subseteq \mathbf{PMC}_{\mathcal{R}}$$

*Proof.* In order to show that $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{R}}^{1p}$, let $X = (I_X, \theta_X)$ be a decision problem in class $\mathbf{P}$. Let us consider the deterministic recognizer (cell-like) membrane system $\Pi = \{\Gamma, \Sigma, \mu, \mathcal{M}_1, \mathcal{R}, i_{in}\}$ of degree 1 defined as follows:

- $\Gamma = \Sigma = \{\mathtt{yes}, \mathtt{no}\}$.
- $\mu = [\ ]_1$.
- $\mathcal{M}_1 = \emptyset$
- $\mathcal{R} = \{[\mathtt{yes}]_1 \to \mathtt{yes}\ [\ ]_1;\ [\mathtt{no}]_1 \to \mathtt{no}\ [\ ]_1\}$
- $i_{in} = 1$.

Let us consider *cod* as the map whose domain is $I_X$ defined as follows: for every $u \in I_X$, $cod(u) = \{\mathtt{yes}\}$ if $\theta_X(u) = 1$, and $cod(u) = \{\mathtt{no}\}$, otherwise. Since $X \in \mathbf{P}$, *cod* is a polynomial-time function. Then, we have:

- The system $\Pi$ is *polynomially bounded* with regard to $(X, cod)$: for every instance $u \in I_X$, the computation of $\Pi$ with input multiset $cod(u)$ performs 1 transition step.
- For every instance $u \in I_X$, the computation of the system $\Pi$ with input multiset $cod(u)$ is an accepting computation if and only if $\theta_X(u) = 1$.

This definition can be easily adjusted for any class of recognizer membrane systems $\mathcal{R}$, in such a way that we have $X \in \mathbf{PMC}_{\mathcal{R}}^{1p}$. Then, we conclude that $\mathbf{P} \subseteq \mathbf{PMC}_{\mathcal{R}}^{1p}$.

In order to show that $\mathbf{PMC}_{\mathcal{R}}^{1p} \subseteq \mathbf{PMC}_{\mathcal{R}}$, let $X = (I_X, \theta_X)$ be a decision problem such that $X \in \mathbf{PMC}^{1p}$. Let $\Pi'$ a membrane system from $\mathcal{R}$ solving $X$ according to Definition 1, being $cod'$ a *polynomial encoding* from $X$ to $\Pi$ associated with that solution. Let us consider the family $\mathbf{\Pi} = \{\Pi(t) \mid t \in \mathbb{N}\}$ defined as follows $\Pi(t) = \Pi'$, for each $t \in \mathbb{N}$. Let us consider the polynomial encoding $(cod, s)$ from the problem $X$ to the family $\mathbf{\Pi}$ defined as follows: $cod = cod'$ and $s(u) = 0$, for each $u \in I_X$. Then it is easy to check that the family $\mathbf{\Pi}$ is polynomially uniform

by Turing machines, polynomially bounded with regard to $(X, cod, s)$, and sound and complete with regard to $(X, cod, s)$. Thus, $X \in \mathbf{PMC}_{\mathcal{R}}$.

□

## 3 The Complexity Class $\mathbf{PMC}_{\mathcal{R}}^{\mathbf{1}f}$

The second definition refers to the case in which the input multiset is directly introduced inside the system as it is ("*free*" of external dependencies or resources), and thus the input alphabet should be chosen so that the system is able to "read" the instances of the problem to be solved.

**Definition 2.** *Let $\mathcal{R}$ be a class of recognizer membrane systems. Let $X = (I_X, \theta_X)$ be a decision problem such that $I_X$ is a language over a finite alphabet $\Sigma_X$. We say that problem $X$ is solvable in polynomial time by a single membrane system $\Pi$ from $\mathcal{R}$ free of external resources, denoted by $X \in \mathbf{PMC}_{\mathcal{R}}^{1f}$, if the following hold:*

- *The input alphabet of $\Pi$ is $\Sigma_X$.*
- *The system $\Pi$ is* polynomially bounded *with regard to $X$; that is, there exists a polynomial $p(r)$ such that for each instance $u \in I_X$, every computation of the system $\Pi$ with input multiset $u$ performs at most $p(|u|)$ steps.*
- *The system $\Pi$ is* sound *with regard to $X$; that is, for each instance $u \in I_X$, if there exists an accepting computation of the system $\Pi$ with input multiset $u$ then $\theta_X(u) = 1$.*
- *The system $\Pi$ is* complete *with regard to $X$; that is, for each instance $u \in I_X$ such that $\theta_X(u) = 1$, every computation of the system $\Pi$ with input multiset $u$ is an accepting computation.*

**Proposition 2.** *Let $\mathcal{R}$ be a class of recognizer membrane systems. Then we have $\mathbf{PMC}_{\mathcal{R}}^{1f} \subseteq \mathbf{PMC}_{\mathcal{R}}^{1p}$.*

*Proof.* Let us assume that $X \in \mathbf{PMC}_{\mathcal{R}}^{1f}$. Let $\Pi'$ a membrane system from $\mathcal{R}$ whose input alphabet is $\Sigma_X$ (the working alphabet of the problem $X$) such that it is polynomially bounded, sound and complete with regard to $X$. Let us consider the polynomial encoding $cod$ from $X$ to $\Pi'$ defined as follows: $cod(u) = u$, for every instance $u \in I_X$. Then, $\Pi'$ is polynomially bounded, sound and complete with regard to $(X, cod)$. Thus, $X \in \mathbf{PMC}_{\mathcal{R}}^{1p}$. □

## 4 Decision Problems with a Finite Number of Instances

In this section, we work with decision problems whose set of instances is a finite set.

**Proposition 3.** *Let $\mathcal{T}(so)$ the class of all recognizer transition P systems which make use of send-out communication rules only. Then, if $X = (I_X, \theta_X)$ is a decision problem whose set of instances is a finite set, then $X \in \mathbf{PMC}_{\mathcal{T}(so)}^{1f}$.*

*Proof.* Let $X = (I_X, \theta_X)$ be a decision problem whose set of instances $I_X$ is a finite language over the alphabet $\Sigma_X$. Let us consider the recognizer transition P system $\Pi = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \mathcal{R}_1, i_{in})$, defined as follows:

- The working alphabet is $\Gamma = \Sigma_X \cup \{\texttt{yes}, \texttt{no}\}$ and the input alphabet $\Sigma$ is $\Sigma_X$.
- The membrane structure is $\mu = [\ ]_1$ and the initial multiset is $\mathcal{M}_1 = \emptyset$.
- The set $\mathcal{R}_1$ of rules is

$$\{[\,u\,]_1 \to \texttt{yes}\ [\ ]_1 \mid \theta_X(u) = 1\} \cup \{[\,u\,]_1 \to \texttt{no}\ [\ ]_1 \mid \theta_X(u) = 0\}$$

- The input membrane is labelled by 1.

Obviously, membrane system $\Pi$ belongs to the class $\mathcal{T}(so)$ and it solves problem $X$, according to Definition 2.

### 4.1 The logic gate problems

**Definition 3.** *A Boolean function of arity $n \geq 1$ is a total function $f$ from $\{0,1\}^n$ to $\{0,1\}$.*

Usually, in this context, Boolean values 0, 1 can be associated with `false` and `true`. Specifically, value 0 is associated with the logical value `false` (denoted by $0^*$) and value 1 is associated with the logical value `true` (denoted by $1^*$). The logical connective $\neg$ can be considered as a unary Boolean function and the logical connectives $\wedge, \vee$ can be considered as binary Boolean functions.

Any Boolean expression $\varphi$ whose set of variables is $Var(\varphi) = \{x_1, \ldots, x_n\}$, can be viewed as the $n$-ary Boolean function $f$ verifying the following: for any tuple $(t_1, \ldots, t_n) \in \{0,1\}^n$ we have $f(t_1, \ldots, t_n) = 1$ if and only if $\sigma(\varphi) = \texttt{true}$, where $\sigma$ is the truth assignment $(t_1^*, \ldots, t_n^*)$.

Any Boolean function of arity $n \geq 1$ has associated a decision problem $X_f = (I_{X_f}, \theta_{X_f})$, in a natural way, as follows: $I_{X_f} = \{0,1\}^n$ and $\theta_{X_f}(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$, for each $(x_1, \ldots, x_n) \in \{0,1\}^n$.

As an interesting case of Boolean functions, we consider the following decision problems associated with logic gates.

- `NOT-GATE`= $(\{0,1\}, \theta)$, where $\theta(0) = 1$ and $\theta(1) = 0$.
- `OR-GATE`= $(\{0,1\} \times \{0,1\}, \theta)$, where $\theta(u) = 0$, if $u = (0,0)$, and $\theta(u) = 1$, otherwise.
- `AND-GATE`= $(\{0,1\} \times \{0,1\}, \theta)$, where $\theta(u) = 1$, if $u = (1,1)$, and $\theta(u) = 0$, otherwise.

**Proposition 4.** *Let $\mathcal{T}(nc, so)$ the class of all recognizer transition P systems which makes only use of non-cooperative send-out communication rules. Then, problems* `NOT-GATE`, `OR-GATE`, `AND-GATE` *belong to* $\mathbf{PMC}^{1f}_{\mathcal{T}(nc,so)}$.

*Proof.* Let us consider the P system $\Pi = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \mathcal{R}_1, i_{in})$ defined as follows:

- The working alphabet is $\Gamma = \Sigma \cup \{\texttt{yes}, \texttt{no}\}$ and the input alphabet $\Sigma = \{0, 1\}$ .
- The membrane structure is $\mu = [\ ]_1$ and the initial multiset is $\mathcal{M}_1 = \emptyset$.
- The set $\mathcal{R}_1$ of rules is $\{[\,0\,]_1 \to \texttt{yes}\ ;\ [\,1\,]_1 \to \texttt{no}\ [\ ]_1\}$.
- The input membrane is labelled by 1.

Obviously, system $\Pi$ from class $\mathcal{T}(nc, so)$ solves the NOT-GATE problem, according to Definition 2.

With respect to the OR-GATE problem, let us consider the P system $\Pi = (\Gamma, \Sigma, \mu, \mathcal{M}_1, \mathcal{R}_1, i_{in})$ defined as follows:

- The working alphabet $\Gamma = \Sigma \cup \{\texttt{yes}, \texttt{no}\}$ and the input alphabet $\Sigma = \{0, 1\} \times \{0, 1\}$.
- The membrane structure is $\mu = [\ ]_1$ and the initial multiset is $\mathcal{M}_1 = \emptyset$.
- The set $\mathcal{R}_1$ of rules is $\{[\,(u, v)\,]_1 \to \texttt{yes}\ [\ ]_1 \mid (u, v) \in \{0, 1\} \times \{0, 1\}, u + v \geq 1\}$ $\cup\ \{[\,(0, 0)\,]_1 \to \texttt{no}\ [\ ]_1\}$.
- The input membrane is labelled by 1.

Obviously, system $\Pi$ from class $\mathcal{T}(nc, so)$ solves the OR-GATE problem, according Definition 2. In a similar way, it can be shown that the AND-GATE problem belongs to $\mathbf{PMC}^{1f}_{\mathcal{T}(nc,so)}$.

It is worth pointing out that in this kind of solution by means of a single membrane system using non-cooperative rules, the representation of the instances is specially relevant. For instance, in the case of the OR-GATE problem and the AND-GATE problem, the cooperation in the rules of the system can be avoided because the set of instances is described by symbols of the language $\{0, 1\} \times \{0, 1\}$.

## 5 The NONE-OBJECT Problem

In this section, we consider the NONE-OBJECT problem which informally corresponds to the task of determining whether there is any input object or not in the system. Formally, let $X = (I_X, \theta_X)$ be the decision problem defined as follows:

$$I_X = \{\emptyset\} \cup \{a^n \mid n \in \mathbb{N}, n \geq 1\}\ ,\ \theta_X(\emptyset) = 1, \text{and } \theta_X(a^n) = 0 \text{ for each } n \geq 1$$

That is, the problem $X$ distinguishes two types of situations: absence of objects on one hand, and at least one copy of object $a$, on the other hand.

**Theorem 1.** *Let $\mathcal{T}(nc, ev, so, dis, pr)$ the class of all non-cooperative recognizer P systems which makes use of minimal production in object evolution rules (that is, only one object in the right-hand side of the rule), send-out communication rules, dissolution rules and priorities. Then,* NONE-OBJECT$\in \mathbf{PMC}^{1f}_{\mathcal{T}(nc,ev,so,dis,pr)}$.

*Proof.* Let us consider the system $\Pi$ from $\mathcal{T}(nc, ev, so, dis, pr)$ defined as follows:

- The working alphabet is $\Gamma = \{a, b, c\}$ and the input alphabet is $\Sigma = \{a\}$.

- The membrane structure $\mu$ is $\mu = [\,[\,\,]_2\,]_1$ and the initial multisets are $\mathcal{M}_1 = \emptyset$ and $\mathcal{M}_2 = \{c\}$.
- The set $\mathcal{R}$ of rules of $\Pi$ is the following:

$$\{[\,a \to b]_2;\ [\,b\,]_2 \to \texttt{no};\ [\,c\,]_2 \to \texttt{yes};\ [\,\texttt{yes}]_1 \to \texttt{yes}\,[\,\,]_1;\ [\,\texttt{no}]_1 \to \texttt{no}\,[\,\,]_1\}$$

- The set of priorities $\mathcal{P}$ among rules of $\Pi$ is the following:

$$\big\{([\,a \to b]_2, [\,c\,]_2 \to \texttt{yes});\ ([\,b\,]_2 \to \texttt{no}, [\,c\,]_2 \to \texttt{yes})\big\}$$

- The input membrane is labelled by 2.

Then, the following hold:

- For each natural number $n \geq 1$, the system $\Pi$ with input multiset $\{a^n\}$ is deterministic, the computation of $\Pi + \{a^n\}$ performs three transition steps and it is a rejecting computation.
- The system $\Pi$ with input multiset $\emptyset$ is deterministic, the computation of $\Pi + \emptyset$ performs two transition steps and it is an accepting computation.

Thus, NONE-OBJECT$\in \mathbf{PMC}^{1f}_{\mathcal{T}(nc,ev,so,dis,pr)}$. □

## 6 The ONLY-ONE-OBJECT Problem

In this section, the problem of telling apart "one" from "more-than-one" object is considered. Formally, let $X = (I_X, \theta_X)$ be the decision problem defined as follows:

$$I_X = \{a^n \mid n \in \mathbb{N}, n \geq 1\} \text{ and } \theta_X(a^n) = 1 \text{ if and only if } n = 1$$

That is, the problem $X$ distinguishes the case when there is only one copy of object $a$ from the rest of possible cases with several copies of that object. We denote that problem as the ONLY-ONE-OBJECT problem. Obviously, the ONLY-ONE-OBJECT problem belongs to class $\mathbf{P}$ since it is easy to design a deterministic Turing machine solving that problem which takes two computation steps. Thus, ONLY-ONE-OBJECT$\in \mathbf{P}$. Bearing in mind that for every class $\mathcal{R}$ of recognizer membrane systems, we have we $\mathbf{P} \subseteq \mathbf{PMC}^{1p}_{\mathcal{R}}$, we deduce that ONLY-ONE-OBJECT$\in \mathbf{PMC}^{1p}_{\mathcal{R}}$.

It is easy to prove that ONLY-ONE-OBJECT$\in \mathbf{PMC}^{1f}_{\mathcal{T}(nc,ev,so,dis,pr)}$, but the following result shows that this problem cannot be solved by a membrane system from $\mathcal{AM}^0(-d, +ne)$ without using precomputed resources, being $\mathcal{AM}^0(-d, +ne)$ the class of polarizationless P systems without dissolution rules and with division rules for elementary and non-elementary membranes.

**Theorem 2.** *There does not exist a recognizer membrane system $\Pi' \in \mathcal{AM}^0(-d, +ne)$ solving the* ONLY-ONE-OBJECT *problem in a polynomial time by a single membrane system and free of resources. That is,* ONLY-ONE-OBJECT$\notin \mathbf{PMC}^{1f}_{\mathcal{AM}^0(-d,+ne)}$.

*Proof.* (Reasoning by *reductio ad absurdum*) Let us assume that there exists a recognizer membrane system $\Pi'$ from $\mathcal{AM}^0(-d,+ne)$ verifying the following:

(a) The input alphabet of $\Pi'$ is the singleton $\{a\}$.
(b) Every computation of $\Pi'$ with input multiset $\{a\}$ is an accepting computation.
(c) Every computation of $\Pi'$ with input multiset $\{a^n\}$, for each $n > 1$, is a rejecting computation.

Let us denote by $G_{\Pi'+\{a\}}$ (respectively, $G_{\Pi'+\{a^n\}}$, for each $n > 1$) the *dependency graph*[1] associated with the system $\Pi' + \{a\}$ (resp. $\Pi' + \{a^n\}$). Then, we have:

- For each $n > 1$, $G_{\Pi'+\{a\}} = G_{\Pi'+\{a^n\}}$. Indeed, in both graphs there is only one edge starting from $s$, specifically, the edge $\{s, (a, i_{in})\}$, and the rest of edges are given by the rules of $\Pi'$, due to $\Pi' \in \mathcal{AM}^0(-d,+ne)$.
- A computation of $\Pi' + \{a\}$ is an accepting computation if and only if there exists a path in $G_{\Pi'+\{a\}}$ from $s$ to $(\texttt{yes}, env)$.
- For each $n > 1$, a computation of $\Pi' + \{a^n\}$ is an accepting computation if and only if there exists a path in $G_{\Pi'+\{a^n\}}$ from $s$ to $(\texttt{yes}, env)$.

Thus, bearing in mind that $G_{\Pi'+\{a\}} = G_{\Pi'+\{a^n\}}$ we deduce that every computation of $\Pi' + \{a\}$ is an accepting computation if and only if every computation of $\Pi' + \{a^n\}$, for each $n > 1$, is an accepting computation. Hence, conditions (b) and (c) are contradictory.

□

**Corollary 1.** $\mathbf{PMC}^{1f}_{\mathcal{AM}^0(-d,+ne)} \subsetneq \mathbf{P} \subseteq \mathbf{PMC}^{1p}_{\mathcal{AM}^0(-d,+ne)}$.

## 7 A Version of the PARITY Problem

In this section, a version of the PARITY problem is considered. Specifically, let $\texttt{PARITY} = (I_{\texttt{PARITY}}, \theta_{\texttt{PARITY}})$ be the decision problem defined as follows:

$$I_{\texttt{PARITY}} = \{a^n \mid n \in \mathbb{N}, n \geq 1\} \text{ and } \theta_{\texttt{PARITY}}(a^n) = 1 \text{ if and only if } n \text{ is even}$$

That is, the PARITY problem distinguishes an even number of copies of object $a$ from an odd number of copies of that object. Obviously, this version of the PARITY problem belongs to class **P** since it is easy to design a deterministic Turing machine solving that problem.

**Theorem 3.** *Let $\mathcal{T}(mcmp, so, dis, pr)$ the class of all recognizer P systems which make use of minimal cooperation and minimal production in object evolution rules, send-out communication rules, dissolution rules and priorities. Then,* $\texttt{PARITY} \in \mathbf{PMC}^{1f}_{\mathcal{T}(mcmp,so,dis,pr)}$.

[1] We will not recall the formal definition here (see [2, 18] for details). The dependency graph can be intuitively seen as a map of "reactants-product" relationship between objects: the nodes are pairs $(object, region)$ and for every rule of the system there will be an arc connecting each object on the left-hand-side to each object on the right-hand side.

*Proof.* Let us consider the system $\Pi$ from $\mathcal{T}(mcmp, so, dis, pr)$ defined as follows:

(a) The working alphabet is $\Gamma = \{a, b\}$ and the input alphabet is $\Sigma = \{a\}$.
(b) The membrane structure is $\mu = [\ [\ ]_2\ ]_1$, and the initial multisets are $\mathcal{M}_1 = \emptyset$ and $\mathcal{M}_2 = \emptyset$.
(d) The set $\mathcal{R}$ of rules of $\Pi$ is the following:

$$\{[\ a^2 \to b]_2;\ [\ b^2 \to b]_2;\ [\ a]_2 \to \texttt{no};\ [\ b]_2 \to \texttt{yes}\} \cup$$
$$\{[\ \texttt{no}]_1 \to \texttt{no}\,[\ ]_1;\ [\ \texttt{yes}]_1 \to \texttt{yes}\,[\ ]_1\}$$

(e) The set of priorities $\mathcal{P}$ among rules of $\Pi$ is the following:

$$\big\{([\ a^2 \to b]_2, [\ a]_2 \to \texttt{no});\ ([\ b^2 \to b]_2, [\ a]_2 \to \texttt{no});\ ([\ a^2 \to b]_2, [\ b]_2 \to \texttt{yes});$$
$$([\ b^2 \to b]_2, [\ b]_2 \to \texttt{yes});\ ([\ a]_2 \to \texttt{no}, [\ b]_2 \to \texttt{yes})\big\}$$

(f) The input membrane is labelled by 2.

Then, for each natural number $n \geq 1$, the following hold:

- The system $\Pi$ with input multiset $\{a^n\}$ is deterministic.
- The computation of $\Pi + \{a^n\}$ performs $2 + \lfloor \log_2(n) \rfloor$ transition steps.
- The natural number $n$ is odd if and only if the configuration $\mathcal{C}_{\lfloor \log_2(n) \rfloor}$ contains a copy of object $a$.
- The natural number $n$ is even if and only if the computation of $\Pi + \{a^n\}$ is an accepting computation.

Thus, $\texttt{PARITY} \in \mathbf{PMC}^{1f}_{\mathcal{T}(mcmp,so,dis,pr)}$. □

## 8 Conclusions

In this work, the ability of solving problems by single "stand-alone" membrane systems instead of families of membrane systems is studied. While using precomputed resources, it is easy to see that problems from **P** can be solved by a single membrane system using only send-out rules. A question arises from here: What if we cannot access to a precomputed encoding and we have the raw instance as input? In this paper, the power of single membrane systems free of precomputed resources is also studied, giving, on the one hand, solutions to decision problems by means of a single membrane system solving them, and on the other hand demonstrating the inability of systems from $\mathcal{AM}^0(-d, +ne)$ to solve the ONLY-ONE-OBJECT problem by using the dependency graph technique in a novel way.

While talking about recognizer membrane systems, we suppose that they can, at least, send an object to the environment to return the answer. Even with this minimal definition, the lower bound for $\mathbf{PMC}^{1p}_{\mathcal{R}}$ has been demonstrated to be

**P**. On the other hand, logic gates have been solved by a system using only non-cooperative send-out rules. This result gives a tool to tackle problems below **P** in the framework of Membrane Computing.

An interesting question is to obtain a lower bound of these systems using only unary alphabets; that is, not allowing cooperation implicit in the instance of the problem. It could be also worth investigating other "weaker" variants, for example obtained removing priorities.

Some open problems remain for future work, e.g. looking for upper bounds of the complexity classes $\mathbf{PMC}_{\mathcal{R}}^{1p}$ and $\mathbf{PMC}_{\mathcal{R}}^{1f}$.

## Acknowledgements

This work was supported in part by the research project TIN2017-89842-P, co-financed by Ministerio de Economía, Industria y Competitividad (MINECO) of Spain, through the Agencia Estatal de Investigación (AEI), and by Fondo Europeo de Desarrollo Regional (FEDER) of the European Union.

## References

1. A. Alhazov, T.-O. Ishdorj: Membrane operations in P systems with active membranes. In Proceedings of the Second Brainstorming Week on Membrane Computing, Sevilla, 2-7 February 2004, 37-44.
2. A. Cordón-Franco, M. A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez. Weak metrics on configurations of a P system. In Gh. Paun, A. Riscos, Á. Romero, F. Sancho (eds.) *Proceedings of the Second Brainstorming Week on Membrane Computing*, Report RGNC 01/2004, 2004, pp. 139-151.
3. M.R. Garey, D.S. Johnson D.S. *Computers and intractability*, W.H. Freeman and Company, New York, 1979.
4. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, F.J. Romero-Campero: A linear solution of Subset Sum problem by using membrane creation. *Lecture Notes in Computer Science*, **3561** (2005), 258-267.
5. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, A. Riscos-Núñez, F.J. Romero-Campero, A. Romero-Jiménez: Characterizing tractability by cell–like membrane systems. In K.G. Subramanian, K. Rangarajan, M. Mukund (eds.) *Formal models, languages and applications*, World Scientific, Singapore, 2006, pp. 137–154.
6. M.A. Gutiérrez-Naranjo, M.J. Pérez-Jiménez, F.J. Romero-Campero: A linear time solution for QSAT with membrane creation. *Lecture Notes in Computer Science*, **3850** (2006), 241-252.
7. P.L. Luisi. The Chemical Implementation of Autopoiesis, *Self-Production of Supramolecular Structures* (G.R. Fleishaker et al., eds.), Kluwer, Dordrecht, 1994.
8. C. Martín-Vide, Gh. Păun, A. Rodríguez-Patón. On P Systems with Membrane Creation. *Computer Science Journal of Moldova*, **9**, 2(26) 2001, 134-145.
9. M. Mutyam, K. Krithivasan: P systems with membrane creation: Universality and efficiency. *Lecture Notes in Computer Science*, **2055** (2001), 276–287.

10. L. Pan, T.-O. Ishdorj: P systems with active membranes and separation rules. *Journal of Universal Computer Science*, **10**, 5 (2004), 630–649.
11. Gh. Păun: Computing with membranes. *Journal of Computer and System Sciences*, **61**, 1 (2000), 108–143, and *Turku Center for CS-TUCS Report* No. 208, 1998
12. Gh. Păun. Membrane Computing: An Introduction. *Springer Natural Computing Series*, 2002.
13. Gh. Păun. P systems with active membranes: attacking **NP**–complete problems, *Journal of Automata, Languages and Combinatorics*, **6**, 1 (2001), 75-90.
14. Gh. Păun, M.J. Pérez-Jiménez, Gr. Rozenberg. Spike trains in spiking neural P systems. *International Journal of Foundations of Computer Science*, **17**, 4 (2006), 975-1002.
15. Gh. Păun, G. Rozenberg, A. Salomaa (eds). The Oxford Handbook of Membrane Computing. *Oxford University Press*, Oxford, U.K., 2009.
16. M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini: Complexity classes in cellular computing with membranes. *Natural Computing*, **2**, 3 (2003), 265–285.
17. P. Sosík. Active Membranes, Proteins on Membranes, Tissue P Systems: Complexity-Related Issues and Challenges. *Lecture Notes in Computer Science*, **8340** (2014), 40–55.
18. L. Valencia-Cabrera, D. Orellana-Martín, I. Pérez-Hurtado, M.J. Pérez-Jiménez. Dependency graph technique revisited. *in this volume*
19. G. Zhang, M.J. Pérez-Jiménez, M. Gheorghe. *Real-Life Modelling with Membrane Computing*. Series: Emergence, Complexity and Computation, Volume 25. Springer International Publishing, 2017, X + 367 pages.