Evolving by Maximizing the Number of Rules: Complexity Study

Oana Agrigoroaiei, Gabriel Ciobanu, and Andreas Resios

[1] Romanian Academy, Institute of Computer Science
Blvd. Carol I no.8, 700505 Iaşi, Romania
[2] "A.I.Cuza" University, Blvd. Carol I no.11, 700506 Iaşi, Romania
oanaag@iit.tuiasi.ro, gabriel@info.uaic.ro, andreas.resios@iit.tuiasi.ro

Summary. This paper presents the complexity of finding the multiset of rules in a P system in such a way to have a maximal number of rules applied. It is proved that the decision version of this problem is **NP**-complete. We study a number of subproblems obtained by considering that a rule can be applied at most once, and by considering the number of objects in the alphabet of the membrane as being fixed. When considering P systems with simple rules, the corresponding decision problem is in **P**. When considering P systems having only two types of objects, and P systems in which a rule is applied at most once, their corresponding decision problems are **NP**-complete. We compare these results with those obtained for *maxO* evolution.

1 Introduction

The reader is assumed to have basic knowledge of membrane computing; a good reference is [6]. Here we just mention the main biological inspiration of P systems, and some terminology concerning the variants of maximal parallelism we consider in this paper.

P systems are inspired by the structure and the functioning of the living cells. Inside the cell, several membranes define compartments where specific biochemical processes take place. Each compartment contains substances (ions, small molecules, macromolecules) and specific reactions. The substances are represented by multisets of objects, and the reactions by rules of form $u \to v$, where u and v are multisets of objects. The multisets are represented by strings, with the understanding that all permutations of a string represent the same multiset. We denote by O the alphabet of objects, and by R_i the set of rules associated with a compartment i. When such a system is evolving, the objects and the rules are chosen in a nondeterministic manner, and the rules are applied in parallel.

The most investigated way of using the rules in a P system is the maximal parallelism: in each membrane a multiset of rules is chosen which can be applied to the objects from that membrane and is maximal in the sense of inclusion, i.e.,

no further rule can be added such that the enlarged multiset is still applicable. We use "$maxP$" to refer to this evolution strategy.

Another natural idea is to apply the rules in such a way to have a maximal number of objects consumed in each membrane. This manner of evolution is denoted by "$maxO$". This strategy was explicitly considered in [1, 2], where it is proved that the problem of finding a multiset of rules consuming a maximal number of objects is **NP**-complete.

Yet a third idea is to apply the rules in such a way to have a maximal number of rules applied. We call this type of evolution "$maxR$". Note that any evolution of type $maxR$ or $maxO$ is also of type $maxP$.

The computing power of these strategies of applying a multiset of rules in membranes is studied in [3]. Specifically, P systems having multiset rewriting rules (with cooperative rules), symport/antiport rules, and active membranes are considered. The universality of the system is proved for any combination of type of system and type of evolution.

In previous papers [1, 2], two variants of membrane systems called simple P systems and maximum cooperative P systems are considered. They evolve at each step by consuming the maximum number of objects. The problem of distributing objects to rules in order to achieve a maximum consuming and non-deterministic evolution of simple P systems is studied in [1]; using the knapsack problem, the decision version of the resource mapping problem for simple P systems is proved to be **NP**-complete. In [2] the integer linear programming problem is used to prove that the resource mapping problem for maximum cooperative P systems is also **NP**-complete.

In this paper we study the complexity of finding a multiset of rules which evolves the membrane in the sense of $maxR$. We study a number of subproblems obtained by considering the number of objects in the alphabet of the membrane as being fixed or by considering that a rule can be applied at most once. We compare the results with those obtained for $maxO$ evolution.

2 *maxR* Complexity

We recall a number of notations for multisets and P systems. We represent multisets as strings of elements over their support alphabet together with their multiplicities (for example $w = a^2b^5c$ is a multiset over $\{a, b, c, d\}$). The union $v + w$ of two multisets over a set O is given by the sum of multiplicities for each element of O. We define $w(a) \in \mathbb{N}$ to be the multiplicity of a in w. We say that $w \leq w'$ if $w(a) \leq w'(a)$ for each element a of the multiset w. In this case we define $w' - w$ to be the multiset obtained by subtracting the multiplicity in w of an element from its multiplicity in w'.

We use the notation $i = \overline{1, n}$ to denote $i \in \{1, \ldots, n\}$.

Definition 1. *A transition P system of degree $n, n \geq 1$ is a construct*

$$\Pi = (O, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n)$$

where

- O *is an alphabet of objects;*
- μ *is a membrane structure, with the membranes labelled by natural numbers* $1, \ldots, m$*, in a one-to-one manner;*
- w_i *are multisets over* O *associated with the regions* $1, \ldots, m$ *defined by* μ*;*
- $R_1, \ldots, R_m$ *are finite sets of rules associated with the membranes with labels* $1, \ldots, m$*; the rules have the form* $u \to v$*, where* u *is a non-empty multiset of objects and* v *a multiset over messages of the form* $(a, here), (a, out), (a, in_j)$*.*

A configuration of the system is given by the membrane structure and the multisets contained in each membrane. For a rule $r = u \to v$ we use the notations $lhs(r) = u$ and $rhs(r) = v$. These notations are extended naturally to multisets of rules: given a multiset of rules $\mathcal{R}$, the left hand side of the multiset $lhs(\mathcal{R})$ is obtained by adding the left hand sides of the rules in the multiset, considered with their multiplicities.

We define the three evolution strategies as follows:

Definition 2. *Let* $i = \overline{1, n}$*. A multiset* $\mathcal{R}$ *of rules over* R_i *is applicable (in membrane* i*) with respect to the multiset* w_i *if* $lhs(\mathcal{R}) \leq w_i$ *and for each message* (a, in_j) *present in* $rhs(\mathcal{R})$ *we have that* j *is one of the children of membrane* i*.*

A multiset $\mathcal{R}$ *of rules over* R_i *which is applicable with respect to the multiset* w_i *is called:*

- $maxP$*-applicable with respect to* w_i *if there is no rule* r *in* R_i *such that* $\mathcal{R} + r$ *is applicable with respect to* w_i*;*
- $maxO$*-applicable with respect to* w_i *if for any other multiset* $\mathcal{R}'$ *of rules which is applicable with respect to* w_i *we have that*

$$\sum_{a \in O} lhs(\mathcal{R})(a) \geq \sum_{a \in O} lhs(\mathcal{R}')(a);$$

- $maxR$*-applicable with respect to* w_i *if for any other multiset* $\mathcal{R}'$ *of rules which is applicable with respect to* w_i *we have that*

$$\sum_{r \in R_i} \mathcal{R}(r) \geq \sum_{r \in R_i} \mathcal{R}'(r).$$

In other words, when choosing the $maxP$ evolution strategy we only apply multisets of rules which are maximal with respect to inclusion; when choosing $maxO$ we only apply multisets of rules which are maximal with respect to the number of objects (considered with their multiplicities) in the left hand side of the multiset; when choosing $maxR$ we only apply multisets of rules which are maximal with respect to the number of rules in the multiset (considered with their multiplicities). Note that any multiset of rules which is either $maxR$ or

$maxO$-applicable is also $maxP$-applicable. P systems generally employ the $maxP$ evolution strategy; however, a convincing case can be made for $maxO$ and $maxR$.

As it is mentioned in [3], maximizing the number of objects or the number of rules can be related to the idea of energy for controlling the evolutions of P systems. In the same paper, the complexity of finding the multiset of rules in a P system in the case of $maxR$ was presented as an open problem.

We denote by P_O and P_R the problems of finding a $maxO$ or $maxR$-applicable multiset of rules, with respect to a given multiset of objects w. We could consider similar problems for the entire system, but they are solved by splitting the problems into smaller ones, one for each membrane. So for our purposes we can just as well consider the system contains only one membrane, i.e. the degree of the P system is $n = 1$. In other words, all multisets of rules we consider from now on are over a set of rules R. We use the following notations:

- m is the cardinal of the alphabet O and we consider the objects to be denoted by $o_1, \ldots, o_m$;
- d is the number of rules associated to the membrane, and the rules are denoted by $r_1, \ldots, r_d$;
- C_a is the multiplicity of o_a in the multiset w which is in the membrane;
- $k_{i,a}$ is the multiplicity of o_a in the left hand side of the rule r_i.

The problem P_O can be described in the form of an integer linear programming problem as follows. Given the positive integers $m, d, k_{i,a}, C_a$ for $i = \overline{1,d}$ and $a = \overline{1,m}$, find positive integers x_i such that

- $\sum_{i=\overline{1,d}}(\sum_{a=\overline{1,m}} k_{i,a})x_i$ is maximal;
- $\sum_{i=\overline{1,d}} x_i \cdot k_{i,a} \leq C_a$, for all $a = \overline{1,m}$.

The decision version of this problem was shown to be **NP**-complete in [1, 2]. The proofs are based on the knapsack problem and integer linear programming [4, 5].

The problem P_R can be described as follows. Given the positive integers $m, d, k_{i,a}, C_a$ for $i = \overline{1,d}$ and $a = \overline{1,m}$, find positive integers x_i such that

- $\sum_{i=\overline{1,d}} x_i$ is maximal;
- $\sum_{i=\overline{1,d}} x_i \cdot k_{i,a} \leq C_a$, for all $a = \overline{1,m}$.

The decision version of P_R is denoted by DP_R: being given positive integers $m, d, t, k_{i,a}$ and C_a, find whether there exist positive integers x_i such that

- $\sum_{i=\overline{1,d}} x_i \geq t$;
- $\sum_{i=\overline{1,d}} x_i \cdot k_{i,a} \leq C_a$, for all $a = \overline{1,m}$.

The length of this instance of the problem can be considered to be $m + d + \max_{a,i}\{\log C_a, \log k_{i,a}\}$.

Proposition 1. *DP_R is **NP**-complete.*

Proof. First, we prove that DP_R is in **NP**. To show this we construct a Turing machine that computes the result in nondeterministic polynomial time by either accepting (output YES) or rejecting (output NO) the input string. The machine operates as follows:

1. nondeterministically assign values for x_i, $i = \overline{1,d}$;
2. if the assigned values verify the constraints
3. and $\sum_{i=\overline{1,d}} x_i \geq t$, then output YES;
4. in any other case, output NO.

It can be easily seen that the number of steps performed by the machine is polynomial with respect to the input size. Thus DP_R is in **NP**.

Secondly, we construct a polynomial-time reduction from $3CNFSAT$ to DP_R. The $3CNFSAT$ [4] problem asks whether a formula ϕ given in conjunctive normal form with 3 variables per clause is satisfiable, i.e. if there exists a variable assignment which makes the formula true.

Consider a formula ϕ with variables $x_1, \ldots, x_r$ and clauses $c_1, \ldots, c_s$. We describe a corresponding instance of DP_R:

- $d = 2r$, $m = r + s$, $t = r$;
- for each variable x_i of ϕ we consider two variables y_i and z_i and an inequality $y_i + z_i \leq 1$ in the instance of DP_R;
- for each clause c_a we consider the inequality

$$\sum_{i=\overline{1,r}} q_{i,a} y_i + \sum_{i=\overline{1,r}} l_{i,a} z_i \leq 2$$

  such that:
  - $q_{i,a} = 0$, $l_{i,a} = 1$ if the literal x_i appears in c_a;
  - $q_{i,a} = 1$, $l_{i,a} = 0$ if the literal $\neg x_i$ appears in c_a;
  - $q_{i,a} = l_{i,a} = 0$ if neither x_i nor $\neg x_i$ appear in c_a.

Since we have $t{=}r$, the first inequality in this instance of DP_R is $\sum_{i=\overline{1,r}} y_i + z_i \geq r$. This can be computed in polynomial time with respect to the size of the input. The idea behind the reduction is to set $x_i = 1$ if and only if $y_i = 1, z_i = 0$ and $x_i = 0$ if and only if $y_i = 0, z_i = 1$.

For example, consider the formula $\phi = c_1 \wedge c_2 \wedge c_3 \wedge c_4$ with $c_1 = x_1 \vee \neg x_2 \vee x_3$, $c_2 = \neg x_1 \vee \neg x_2 \vee \neg x_3$, $c_3 = x_1 \vee \neg x_2 \vee \neg x_3$, $c_4 = \neg x_1 \vee x_2 \vee x_3$. The corresponding instance of DP_R is: find positive integers y_i, z_i, $i = \overline{1,3}$ positive integers such that $\sum_{i=\overline{1,3}} y_i + z_i \geq 3$, $y_i + z_i \leq 1$, and

$$\begin{cases} z_1 + y_2 + z_3 \leq 2 \\ y_1 + y_2 + y_3 \leq 2 \\ z_1 + y_2 + y_3 \leq 2 \\ y_1 + z_2 + z_3 \leq 2 \end{cases}$$

We notice that $y_i + z_i = 1$, and that a solution is $y_1 = 0, y_2 = 0, z_3 = 0$, together with the corresponding values for z_1, z_2, y_3. This means that we consider the assignment $x_1 = 0, x_2 = 0, x_3 = 1$ for which the formula ϕ is satisfiable.

We now prove that a formula ϕ is satisfiable if and only if there is a vector of positive integers $(y_1, \ldots, y_r, z_1, \ldots, z_r)$ which is a solution for the above instance of DP_R. First, suppose there is a satisfying assignment for ϕ. If $x_i = 1$ we set $y_i = 1, z_i = 0$, and if $x_i = 0$ we set $y_i = 0, z_i = 1$. Thus we have $y_i + z_i \leq 1$, for all $i = \overline{1,r}$, and also $\sum_{i=\overline{1,r}} y_i + z_i \geq r$. Consider now one of the inequalities

$$\sum_{i=\overline{1,r}} q_{i,a} y_i + \sum_{i=\overline{1,r}} l_{i,a} z_i \leq 2$$

We notice that it contains in its left hand side exactly three variables with coefficient 1, one for each literal appearing in C_a. If the literal with value 1 in C_a is x_j, then its corresponding variable is z_j which is 0. If the literal with value 1 in C_a is $\neg x_j$, then its corresponding variable is y_j which is 0. Thus there are at most two terms equal to 1, meaning that the inequality is satisfied.

Now suppose there is a solution $(y_1, \ldots, y_r, z_1, \ldots, z_r)$ for the DP_R instance. Since $y_i + z_i \leq 1$ for all $i = \overline{1,r}$ and $\sum_{i=\overline{1,r}} y_i + z_i \geq r$, it follows that $y_i + z_i = 1$ for all i. We consider the assignment $x_i = 1$ if $y_i = 1, z_i = 0$ and $x_i = 0$ if $y_i = 0, z_i = 1$. As previously noted, the inequality corresponding to a clause c_a has exactly three variables, each with coefficient 1, in its left hand side. Thus at least one of them must be equal to 0. If that variable is z_j, it means that the literal x_j with assignment $x_j = 1$ appears in C_a. If that variable is y_j, it means that the literal $\neg x_j$ with assignment $x_j = 0$ appears in C_a. Thus ϕ is satisfied. □

We can also consider the problem $1DP_R$ obtained from DP_R by restricting the possible values of the variables to 0 or 1. This corresponds to requesting that in a membrane a rule can be applied at most once. Then exactly the same reduction can be made from $3CNFSAT$ to $1DP_R$ thus placing $1DP_R$ in the category of **NP**-complete problems.

3 Certain Subproblems

We denote by DP_R^k the problem obtained from DP_R by considering $m = k$ fixed. A similar notation is used for DP_O^k.

We start by looking at the case of a P system which has only simple rules, i.e. rules which have only one type of object in their right hand side. Then DP_R^1 describes the decision version of the problem of finding a multiset of simple rules which is $maxR$-applicable: given $d, t, k_{i,1}, C_1$ find x_i such that $\sum_{i=\overline{1,d}} x_i \geq t$ and $\sum_{i=\overline{1,d}} x_i \cdot k_{i,1} \leq C_1$.

Proposition 2. *DP_R^1 is in **P**.*

Proof. Note that all $k_{i,1} \neq 0$ by definition, since rules always have a non-empty left hand side. Let j be chosen such that $k_{j,1} = min_{i=\overline{1,d}}\{k_{i,1}\}$. A solution is given by setting $x_j = \left[\frac{C}{k_{j,1}}\right]$ (the integer part of $\frac{C}{k_{j,1}}$) and $x_i = 0, i \neq j$. □

On a side note, consider the problem $1DP_R^1$ obtained by restricting the possible values of x_i to 0 or 1. This problem is in **P**, as can be seen by following this algorithm. First we renumber the coefficients $k_{i,1}$ (together with the variables x_i) such that $k_{1,1} \leq k_{2,1} \leq \ldots \leq k_{d,1}$. Then we set $s_1 = k_{1,1}$, $s_{i+1} = s_i + k_{i+1,1}$. If $s_d \leq C_1$ then the maximum value for $\sum_i x_i$ is d. Otherwise, there exists an unique j such that $s_j \leq C_1 < s_{j+1}$. Therefore the maximum value for $\sum_i x_i$ is j, since however we choose $j+1$ different coefficients $k_{r_1,1}, k_{r_2,1}, \ldots k_{r_{j+1},1}$ randomly, their sum will be greater than s_{j+1}.

We now consider that the membrane whose $maxR$ evolution we are studying has only two types of objects, i.e. $\#O = 2$. The corresponding decision problem is DP_R^2.

Proposition 3. *DP_R^2 is **NP**-complete.*

To prove this result we consider the following auxiliary problem AP:
For s, r, k positive integers, are there positive integers $x_1, \ldots, x_s$ such that

$$\sum_{i=\overline{1,s}} x_i = r, \sum_{i=\overline{1,s}} k_i x_i = k.$$

Note that if we restrict this problem by imposing the condition that all $x_i \in \{0, 1\}$, then we obtain a subproblem of the subset sum problem, namely: given a set S of positive integers $S = \{k_i \mid i = \overline{1, s}\}$, does exist a subset of S with r elements such that the sum of its elements equals k? This provides a strong hint that AP is **NP**-complete. The proof of Proposition 3 is based on constructing a polynomial-time reduction from $X3C$ to AP, and another one from AP to DP_R^2.

Proof. First, note that both DP_R^2 and AP are in **NP**. This can be easily proved by constructing a Turing machine similar to the one used in the proof of Proposition 1. Secondly, we give a a polynomial-time reduction from $X3C$ to AP. The *exact cover by 3-sets* problem ($X3C$) asks if, given a set X with $3q$ elements and a collection C of 3-element subsets of X, there is a subcollection C' of C which is an *exact cover* for X, i.e. any element of X belongs to exactly one element of C' [4].

In order to reduce $X3C$ to AP, we do the following. Let l be the number of elements of C, and consider indexing the elements of C by $c_1, \ldots, c_l$. For each c_i we consider a variable x_i in the AP problem, thus setting $s = l$. To construct the coefficients k_i, we employ the notations $e_{ij} = \#c_i \cap c_j$ for $i, j = \overline{1, l}$, and $M = 3q+1$. We set $s = l$, $r = q$, $k_i = \sum_{j=\overline{1,l}} e_{ij} \cdot M^{l-j}$ and $k = \sum_{j=\overline{1,l}} 3 \cdot M^{l-j}$. For a solution C' to $X3C$ we set $x_i = 1$ whenever $c_i \in C'$, and $x_i = 0$ otherwise. We prove that this yields a solution of the constructed instance of AP and moreover, that any solution of the instance has $x_i \in \{0, 1\}$ and provides a solution of $X3C$.

Example. Consider the problem $X3C$ for $X = \{1, \ldots, 9\}$ and $c_1 = \{1, 2, 3\}$, $c_2 = \{1, 3, 4\}$, $c_3 = \{4, 5, 6\}$, $c_4 = \{1, 6, 8\}$, $c_5 = \{4, 7, 9\}$, $c_6 = \{7, 8, 9\}$. Then $M = 10$ and the coefficients k_i are written in base 10 such that they have a digit for each variable x_j:

	x_1	x_2	x_3	x_4	x_5	x_6
$\mathbf{k_1}$	**3**	**2**	**0**	**1**	**0**	**0**
k_2	2	3	1	1	1	0
$\mathbf{k_3}$	**0**	**1**	**3**	**1**	**1**	**0**
k_4	1	1	1	3	0	1
k_5	0	1	1	0	3	2
$\mathbf{k_6}$	**0**	**0**	**0**	**1**	**2**	**3**
k	3	3	3	3	3	3

An exact cover of X is c_1, c_3, c_6. Looking at this example, we see why any solution to AP has all $x_i \in \{0, 1\}$: all coefficients have at least a digit equal to 3 and the basis M is chosen such that, when adding coefficients, no carries can occur from lower digits to higher digits.

We first prove that a solution C' for $X3C$ provides a solution for AP. Let $I = \{i \,|\, c_i \in C'\}$. Since C' is an exact cover for X, it follows that I has q elements and that $e_{ij} = 0, i, j \in I, i \neq j$. Moreover, if $j \notin I$ we have that $c_j = c_j \cap (\cup_{i \in I} c_i) = \cup (c_j \cap c_i)$, thus $\sum_{i \in I} e_{ij} = 3$. Since $x_i = 1, i \in I$ and $x_i = 0, i \notin I$ it follows that indeed $\sum_{i=\overline{1,m}} x_i = q$. We also have

$$\sum_{i=\overline{1,l}} k_i x_i = \sum_{i \in I} (\sum_{j=\overline{1,l}} e_{ij} M^{l-j}) =$$

$$= \sum_{i \in I} (e_{ii} M^{l-i} + \sum_{j \notin I} e_{ij} M^{l-j}) = \sum_{i \in I} 3 \cdot M^{l-i} + \sum_{j \notin I} (\sum_{i \in I} e_{ij}) M^{l-j}$$

Using the previous observation, we obtain that the term of the second sum is $3 \cdot M^{l-j}$, thus $\sum_{i=\overline{1,m}} k_i x_i = k$.

Secondly, consider a solution $(x_i)_{i=\overline{1,s}}$ for the instance of AP with s, r, k_i, k as above. Let $I = \{i \,|\, x_i = 1\}$. We prove that if $j \notin I$ then $x_j = 0$ and that $e_{ij} = 0$ for $i, j \in I, i \neq j$. This is sufficient to prove that $C' = \{c_i \,|\, i \in I\}$ is an exact cover, since it follows from the above statement that C' has exactly q elements and $c \cap c' = \emptyset$ for all $c, c' \in C', c \neq c'$. We have

$$\sum_{i=\overline{1,l}} 3 \cdot M^{l-j} = k = \sum_{i=\overline{1,l}} k_i x_i = \sum_{j=\overline{1,l}} (\sum_{i=\overline{1,l}} e_{ij} x_i) M^{l-j} \tag{1}$$

Since $\sum_{i=\overline{1,l}} e_{ij} x_i \leq \sum_{i=\overline{1,l}} 3 x_i = 3q < M$, the two sides of equation (1) represent two decompositions in base M of the same number k. Therefore we have $\sum_{i=\overline{1,l}} e_{ij} x_i = 3$, for any $j = \overline{1,l}$. For $i = j$ we get $e_{ii} x_i = 3 x_i \leq 3$, i.e. all $x_i \in \{0, 1\}$. Thus $3 = \sum_{i \in I} e_{ij}$; considering $j \in I$ we obtain that $3 = 3 + \sum_{i \in I, i \neq j} e_{ij}$, namely that $e_{ij} = 0, i, j \in I, i \neq j$, concluding the second part of the reduction.

We still have left to show that AP reduces to DP_R^2. We recall the data of DP_R^2: given $d, t, C_1, C_2, k_{i,1}, k_{i,2}$ for $i = \overline{1,d}$, do exist positive integers $x_1, \ldots, x_d$ such that

$$\begin{cases} \sum_{i=\overline{1,d}} x_i \geq t \\ \sum_{i=\overline{1,d}} k_{i,1} x_i \leq C_1 \\ \sum_{i=\overline{1,d}} k_{i,2} x_i \leq C_2 \ ? \end{cases} \tag{2}$$

The reduction is as follows: let $K = max_{i=\overline{1,d}} k_i$ and set $d = s, t = r, k_{i,1} = k_i$, $k_{i,2} = K - k_i, C_1 = k$ and $C_2 = Kr - k$. If $x_1, \ldots, x_s$ represent a solution for the instance of AP, it clearly is a solution for this instance of DP_R^2. Reversely, if $x_1, \ldots, x_s$ represent a solution for this instance of DP_R^2, we add the last two inequalities of (2), obtaining $\sum_{i=\overline{1,s}} K \cdot x_i \leq Kr$. Since $\sum_{i=\overline{1,d}} x_i \geq t$, we obtain that $\sum_{i=\overline{1,s}} x_i = r$ and also that $\sum_{i=\overline{1,s}} k_i x_i = k$. □

We compare these results with those for DP_O and its analogous subproblems. Both DP_R and DP_O are **NP**-complete, yet we obtain significant differences when restricting to the case of P systems with simple rules. Namely, while DP_O^1 is **NP**-complete, DP_O^1 is in **P**. When we employ cooperative rules with a fixed maximum number k of objects in the left hand side, the decision problems thus obtained, DP_O^k and DP_R^k, are all **NP**-complete.

4 Conclusion

The most investigated way of applying the rules in a P system is the maximal parallelism (*maxP* case). Two other strategies of applying the rules are also possible. One strategy is to maximize the number of objects consumed in each membrane (*maxO* case), and the other is to maximize the number of rules applied in each membrane (*maxR* case).

The *maxO* strategy was explicitly considered in [1] and [2] where it is proved that the problem of finding a multiset of rules which consume a maximal number of objects is **NP**-complete for both so called simple P systems and cooperative P systems.

In this paper we consider the *maxR* strategy, and study the complexity of finding the multiset of rules in a P system in such a way to have a maximal number of rules applied. We prove that the decision version of this problem is **NP**-complete. However, in contrast to the results for *maxO* strategy, the problem for P systems with simple rules is in **P**.

Together with the results presented in [1, 2, 3], this paper provides the possibility of studying complexity and computability for new classes of P systems. It also facilitates a complexity comparison between various classes of P systems.

Acknowledgements

This work has been partially supported by research grants CNCSIS IDEI 402/2007 and CNMP D1/1052/2007.

References

1. G. Ciobanu, A. Resios. Computational Complexity of Simple P Systems. *Fundamenta Informaticae* vol.87, 49-59, 2008.
2. G. Ciobanu, A. Resios. Complexity of Evolution in Maximum Cooperative P Systems. *Natural Computing*, 2009 (to appear).
3. G. Ciobanu, S. Marcus, Gh. Păun. New Strategies of Using the Rules of a P System in a Maximal Way. *Romanian Journal of Information Science and Technology* vol.12, 21-37, 2009.
4. M.R. Garey, D.S. Johnson. *Computers and Intractability: A Guide to the Theory of **NP**-Completeness*. W.H.Freeman & Co. 1979.
5. C.H. Papadimitriou, K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, 1998.
6. Gh. Păun. *Computing with Membranes: An Introduction*, Springer, 2002.