ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **15**, Number 1, 2012, 33–49

Solving Common Algorithmic Problem by Recognizer Tissue P Systems

Yunyun NIU[1], Linqiang PAN[1], Mario J. PÉREZ-JIMÉNEZ[2]

[1] Key Laboratory on Image Processing and Intelligent Control
Department of Control Science and Engineering
Huazhong University of Science and Technology
Wuhan 430074, Hubei, China
E-mail: `niuyunyun1003@163.com`,
`lqpan@mail.hust.edu.cn` (Corresponding author)

[2] Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla, 41012 Sevilla, Spain
E-mail: `marper@us.es`

Abstract. Common Algorithmic Problem is an optimization problem, which has the nice property that several other NP-complete problems can be reduced to it in linear time. In this work, we deal with its decision version in the framework of tissue P systems. A tissue P system with cell division is a computing model which has two types of rules: communication and division rules. The ability of cell division allows us to obtain an exponential amount of cells in linear time and to design cellular solutions to computationally hard problems in polynomial time. We here present an effective solution to Common Algorithmic Decision Problem by using a family of recognizer tissue P systems with cell division. Furthermore, a formal verification of this solution is given.

Key-words: membrane computing; tissue P system; cell division; common algorithmic problem

1. Introduction

Membrane Computing is a branch of Natural Computing, which is inspired by the structure and the functioning of living cells [10], as well as the organization of cells in tissues, organs, and other higher order structures. The devices of this model, called P systems, provide distributed parallel and non-deterministic computing models. Since being introduced by Gh. Păun in 1998, Membrane Computing has received important attention from the scientific community. As computer scientists, biologists, formal linguists and complexity theoreticians plug into this area, Membrane Computing has definitely become a rich and exciting realm of cross-disciplinary research. Please refer to [11] for an introduction of Membrane Computing, to [13] for a recent overview, and to [15] for further bibliography.

In last years, many different classes of P systems have been investigated. The most studied variants are the cell-like models of P systems, where membranes are hierarchically arranged in a tree-like structure. Most of them are computationally universal (*i.e.*, able to compute whatever a Turing machine can do), as well as computationally efficient (*i.e.*, able to trade space for time and solve in this way presumably intractable problems in a feasible time) [1–3, 9].

Another interesting class of P system is that of tissue P systems [7], where instead of considering a hierarchical arrangement, membranes are placed in the nodes of a graph. Tissue P systems are abstracted from the intercellular communication and the cooperation between cells in tissues [8]. Here, we focus on a variant of tissue P systems: tissue P system with cell division [12].

Common Algorithmic Problem (`CAP`) [6] is an optimization problem, that can be defined as follows. Let S be a finite set and F be a family of subsets of S. Find the cardinality of a maximal subset of S which does not include any set belonging to F. The sets in F are called forbidden sets. Several other NP–complete problems can be reduced to `CAP` in linear time (using a logarithmic bounded space), this is the case for independent set problem, vertex cover problem, maximum clique problem, satisfiability problem, Hamiltonian path problem and tripartite matching problem [6, 14], so we can say that they are subproblems of `CAP` in the sense of linear time reduction.

In [14], an effective solution to `CAP` was proposed using a family of recognizer P systems with active membranes. However, there is no known way to transform a recognizer P system with active membranes to a tissue P system. Tissue P systems with cell division can solve some NP-complete problems in polynomial time, *e.g.*, the subset sum problem [4], the partition problem [5], and the 3–coloring problem [3]. But it remains open how to compute the reduction of an NP problem to another NP–complete problem by P systems. So, in this work, we give a direct solution to `CAP` in the framework of tissue P systems with cell division.

The paper is organized as follows. Some preliminaries are recalled in section 2 including the definition of recognizer tissue P systems with cell division. A polynomial–time solution to `CAP` is presented in section 3, and the formal verification is also given to prove the solution correctness. Some discussion is presented in section 4.

2. Preliminaries

An *alphabet* Σ is a non empty set, whose elements are called *symbols*. An ordered sequence of symbols is a *string*. 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 strings of length n built with symbols from the alphabet Σ is denoted by Σ^n and $\Sigma^* = \cup_{n\geq 0}\Sigma^n$. A *language* over Σ is a subset of Σ^*.

A *multiset* m over a set A is a pair (A, f), where f is a map from A to the set of natural numbers $\mathbb{N}$. If $m = (A, f)$ is a multiset, then its *support* is defined as $supp(m) = \{x \in A \mid f(x) > 0\}$ and its *size* is defined as $\sum_{x\in A} f(x)$. A multiset is empty (resp. finite) if its support is the empty set (resp. finite).

If $m = (A, f)$ is a finite multiset over A, and $supp(m) = \{a_1, \dots, a_k\}$, then it will be denoted as $m = \{\{a_1^{f(a_1)}, \dots, a_k^{f(a_k)}\}\}$. That is, superscripts indicate the multiplicity of each element, and if $f(x) = 0$ for any $x \in A$, then this element is omitted. If $m_1 = (A, f)$ and $m_2 = (A, g)$ are multisets over A, then the union of m_1 and m_2 is defined as $m_1m_2 = (A, h)$, where $h = f + g$.

A *recognizer tissue P system with cell division* of degree $q \geq 1$ is a tuple of the form

$$\Pi = (\Gamma, \Sigma, \Omega, w_1, \dots, w_q, \mathcal{R}, i_{in}, i_{out}), \text{ where:}$$

- $q \geq 1$ is the initial degree of the system, which contains q cells labeled with $1, 2, \dots, q$; all these q cells are placed in a common environment labeled with 0;
- Γ is the working alphabet, which contains two distinguished objects `yes` and `no`, at least one copy of them occurring in some initial multisets $w_1, \dots, w_q$, but not occurring in Ω;
- Σ is an input alphabet strictly contained in Γ;
- $\Omega \subseteq \Gamma$ is the set of objects occurring in the environment, each one in arbitrarily many copies;
- $w_1, \dots, w_q$ are strings over Γ, describing the multisets of objects located in the cells of the system at the beginning of the computation;
- $\mathcal{R}$ is a finite set of rules of the following forms:
  - (a) Communication rules: $(i, u/v, j)$, for $i, j \in \{0, 1, 2, \dots, q\}, i \neq j, u, v \in \Gamma^*$ ($|u| + |v|$ is called the length of the communication rule $(i, u/v, j)$).
  - (b) Division rules: $[a]_i \to [b]_i[c_i]$, where $i \in \{1, 2, \dots, q\}, a \in \Gamma$ and $b, c \in \Gamma \cup \{\lambda\}$.
- $i_{in} \in \{1, \dots, q\}$ is the input cell;
- $i_{out} \in \{0, 1, \dots, q\}$ indicates the output region, where $i_{out} = 0$ denotes that the output region is the environment.

All computations halt (that is, they always reach a configuration where no further rules can be applied). During a computation of Π, either the object yes or the object no (but not both) must be released into the environment, and only in the last step of the computation.

When a division rule $[a]_i \to [b]_i[c]_i$ is applied, all the objects in the original cells are replicated and the copies of them are placed in each of the new cells, with the exception of the objects a, which is replaced by $b \in \Gamma \cup \{\lambda\}$ in the first new cell and by $c \in \Gamma \cup \{\lambda\}$ in the second one.

When a rule $(i, u/v, j)$ is applied, the objects of the multiset represented by u are sent from region i to region j and simultaneously the objects of the multiset v are sent from region j to region i. For a cell in the system Π, it is possible to have more than one applicable communication rules in a step. These applicable communication rules are used in non-deterministic maximally parallel manner (the system non-deterministically chooses and applies a multiset of communication rules that is maximal, no further rule can be added).

In each step, all cells which can evolve must evolve in a maximally parallel way. This way of applying rules has only one restriction: when a cell is divided, the division rule is the only one which is applied for that cell in that step; the objects inside that cell do not evolve by means of communication rules. The labels of the cells produced by division precisely identify the rules which can be applied to them in the subsequent steps.

A configuration of Π at an instant t is described by the multisets of objects over Γ associated with all the cells present in the system at that moment, and the multiset over $\Gamma - \Omega$ associated with the environment at the instant t. All computations start from the initial configuration and proceed as defined above. A computation $\mathcal{C}$ is called an accepting computation (respectively, rejecting computation) if the object yes (respectively, no) appears in the environment associated to the corresponding halting configuration of $\mathcal{C}$, and only in the last step of the computation.

Definition 1. Let $X = (I_X, \theta_X)$ be a decision problem, 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 (that is, a predicate). The decision problem X is *solvable in polynomial time by a family* $\mathbf{\Pi} = \{\Pi(n) \mid n \in \mathbb{N}\}$ *of recognizer tissue P systems with cell division* 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 $\Pi(s(u))$;
  - 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 $cod(u)$ halts and, moreover, 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.

We denote by $\mathbf{PMC}_{TDC}$ the set of all decision problems which can be solved by means of recognizer tissue P systems with cell division in polynomial time.

3. A Solution to Common Algorithmic Decision Problem

Common Algorithm Decision Problem (CADP) can be defined as follows. *Given S a finite set, F a family of subsets of S, and $k \in N$, we ask if there exists a subset A of S such that $|A| \geq k$, and which does not include any set belonging to F. The sets in F are called forbidden sets.*

We address the solution of this problem via a brute force algorithm, in the framework of recognizer tissue P systems with cell division. Our strategy will consist of the following phases:

- *Generation Stage*: The initial cell, labeled by 2, is divided into two new cells. The division is iterated until we have all possible subsets to the problem (one subset of S for each membrane with label 2). Simultaneously, in the membrane with label 1 there is a counter, and it will determine the moment in which the checking stage starts.

- *Checking Stage*: The system checks whether or not there exists a subset A of S such that A does not include any forbidden set in the family F and $|A| \geq k$.

- *Output Stage*: The system sends to the environment the right answer according to the results of the previous stage.

Let us consider the polynomial time computable function between $\mathbb{N}^3$ and $\mathbb{N}$, $\langle n, m, k\rangle = \langle\langle n, m\rangle, k\rangle$, induced by the pair function $\langle n, m\rangle = ((n + m)(n + m + 1)/2) + n$. We shall construct a family $\Pi = \{\Pi(i) \mid i \in \mathbb{N}\}$ such that each system $\Pi(\langle n, m, k\rangle)$ will solve all instances of CADP with given size paraments: the size n of a finite set S, the size m of the family F of forbidden sets, and the target subset size k.

For each $(n, m, k) \in \mathbb{N}^3$, the system $\Pi(\langle n, m, k\rangle) = (\Gamma(\langle n, m, k\rangle), \Sigma(\langle n, m, k\rangle), \Omega(\langle n, m, k\rangle), w_1, w_2, \mathcal{R}(\langle n, m, k\rangle), i_{in}, i_{out})$ is constructed with the following components:

- $\Gamma(\langle n, m, k\rangle) = \Sigma \cup \{a_j, T_j, F_j, f_j \mid 1 \leq j \leq n\}$

$\cup\{r_i \mid 1 \le i \le m\} \cup \{b_i \mid 1 \le i \le 2n+m+1\}$
$\cup\{F_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}$
$\cup\{d_i \mid 1 \le i \le 2n+mn+2m+k+1\}$
$\cup\{e_i \mid 1 \le i \le 2n+mn+2m+k+3\}$
$\cup\{c_i \mid 1 \le i \le n+1\} \cup \{f, g, \mathtt{yes}, \mathtt{no}\}$.

- $\Sigma(\langle n,m,k\rangle) = \{s_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}$.
- $\Omega(\langle n,m,k\rangle) = \Gamma(\langle n,m,k\rangle) - \{\mathtt{yes}, \mathtt{no}\}$.
- $w_1 = \{\{b_1, c_1, d_1, e_1, g, \ \mathtt{yes}, \ \mathtt{no}\}\}$.
- $w_2 = \{\{f, a_1, a_2, \cdots, a_n\}\}$.
- $\mathcal{R}(\langle n,m,k\rangle)$ is the set of rules:

  1. **Division rule:**
     $r_{1,j} \equiv [a_j]_2 \rightarrow [T_j]_2[F_j]_2$, for $1 \le j \le n$.
  2. **Communication rules:**
     $r_{2,i} \equiv (1, b_i/b_{i+1}^2, 0)$, for $1 \le i \le n$;
     $r_{3,i} \equiv (1, c_i/c_{i+1}^2, 0)$, for $1 \le i \le n$;
     $r_{4,i} \equiv (1, d_i/d_{i+1}^2, 0)$, for $1 \le i \le n$;
     $r_{5,i} \equiv (1, e_i/e_{i+1}, 0)$, for $1 \le i \le 2n + 2m + mn + 2$;
     $r_6 \equiv (1, b_{n+1}c_{n+1}d_{n+1}/f, 2)$;
     $r_{7,j} \equiv (2, c_{n+1}F_j/c_{n+1}F_{1,j}, 0)$, for $1 \le j \le n$;
     $r_{8,ij} \equiv (2, F_{i,j}/f_jF_{i+1,j}, 0)$, for $1 \le i \le m$, $1 \le j \le n$;
     $r_{9,i} \equiv (2, b_i/b_{i+1}, 0)$, for $n+1 \le i \le 2n+m$;
     $r_{10,i} \equiv (2, d_i/d_{i+1}, 0)$, for $n+1 \le i \le 2n+m+mn$;
     $r_{11,ij} \equiv (2, b_{2n+m+1}f_js_{ij}/b_{2n+m+1}r_i, 0)$, for $1 \le i \le m$, $1 \le j \le n$;
     $r_{12,i} \equiv (2, d_{2n+mn+m+i}r_i/d_{2n+mn+m+i+1}, 0)$, for $1 \le i \le m$;
     $r_{13,ij} \equiv (2, d_{2n+mn+2m+i}T_j/d_{2n+mn+2m+i+1}, 0)$, for $1 \le i \le k$, $1 \le j \le n$;
     $r_{14} \equiv (2, d_{2n+mn+2m+k+1}/g\ \mathtt{yes}, 1)$;
     $r_{15} \equiv (2, \mathtt{yes}/\lambda, 0)$;
     $r_{16} \equiv (1, e_{2n+mn+2m+k+3}\ g\ \mathtt{no}/\lambda, 2)$;
     $r_{17} \equiv (2, \mathtt{no}/\lambda, 0)$.

- $i_{in} = 2$ is the input cell.
- $i_{out} = 0$ is the output region (i.e., the environment).

3.1. An Overview of a Computation

First of all we define a polynomial encoding for CADP in $\mathbf{\Pi}$. Let $u = (\{s_1, \cdots, s_n\}, (B_1, \cdots, B_m), \ k)$ be an instance of CADP. Let the size mapping be $s(u) = \langle n,m,k\rangle$ and the encoding of instance be $cod(u) = \{s_{i,j} \mid s_j \in B_i\}$, for a given CADP-instance

$u = (\{s_1, \cdots, s_n\},\ (B_1, \cdots, B_m),\ k)$. Next we informally describe how the system $\Pi(s(u))$ with input $cod(u)$ works.

Let us start with the generation stage. In cells with label 2, the division rules are applied. Cells with label 2 are repeatedly divided, each time expanding one object a_j, corresponding to s_j in the finite set S, into T_j and F_j, corresponding to the existence or absence of s_j in certain subset. In this way, after n steps we get 2^n cells with label 2, each one associated with a subset of S. The object f is duplicated, hence a copy of it will appear in each cell. In parallel with the above operation of dividing cells with label 2, the counters b_i, c_i, d_i, e_i from cell with label 1 grow their subscripts. In each step, the number of copies of objects of the first three types is doubled, hence after n steps we get 2^n copies of b_{n+1}, c_{n+1}, and d_{n+1} in cell with label 1. Object b_i is used to check whether a forbidden set B_i is not included in the corresponding subset A, object c_i is used to multiply the number of copies of f_j, object d_i is used to check whether there exists at least one subset A such that A does not include any forbidden set B_i from the family F and $|A| \geq k$. The object e_i will be used to produce the object `no`, if this will be the case, at the end of the computation.

The checking stage starts when the generation stage is finished after n steps. Note that cells with label 2 cannot divide any more, because the objects a_j were exhausted. At this moment, the content of the cell with label 1 is $\{\{b_{n+1}^{2^n}, c_{n+1}^{2^n}, d_{n+1}^{2^n}, e_{n+1}, g, \mathtt{yes}, \mathtt{no}\}\}$. At step $n+1$, the counters $b_{n+1},\ c_{n+1},\ d_{n+1}$ are brought into cells with label 2, in exchange of f by applying rule r_6. Because we have 2^n copies of each object of these types and 2^n cells with label 2, each one containing exactly one copy of f, due to the maximality of the parallelism of using the rules, each cell 2 gets precisely one copy of each of $b_{n+1},\ c_{n+1},\ d_{n+1}$.

Recall that T_j represents that s_j is in the corresponding subset, while F_j represents that s_j is not in the corresponding subset. The object F_j introduces the object $F_{1,j}$, when c_{n+1} is present. This phase needs at most n steps, because only one copy of c_{n+1} is available in each cell with label 2. Then further m steps are necessary for $F_{1,j}$ to grow its first subscript generating m copies of f_j. (Object f_j represents element s_j from S not in the corresponding subset A. In order to check which forbidden sets are not included in A, it is possible to need one copy of f_j for each forbidden set.) The counters b_i and d_i increase their subscripts, until reaching the value $2n+m+1$. In parallel, object e_i increases its subscript to $2n+m+2$ in cell with label 1.

When object b_{2n+m+1} is present, we apply the rules $r_{11,ij}$ to check which forbidden sets are not included in the corresponding subset of S. The objects r_i represents that the forbidden set B_i is not included in the corresponding subset of S. It takes at most mn steps, because there is only one copy of b_{2n+m+1} in each cell with label 2. After step $2n+m+mn+1$, the rule $r_{12,i}$ is used to check whether there exists a subset A which does not include any forbidden set. If and only if it is positive, the subscript of d_i in the corresponding cell with label 2 grows to $2n+2m+mn+1$. After step $2n+2m+mn+1$, in the cell with label 2 whose corresponding subset of S does not include any forbidden set, the rule $r_{13,ij}$ is used to check whether the cardinality of the corresponding subset is not less than k. If and only if it is still positive, the subscript of d_i in the corresponding cell with label 2 grows to $2n+2m+mn+k+1$.

When the checking stage is done, the subscript of object e_i in cell with label 1 grows to $2n+2m+mn+k+2$. The output stage starts from step $2n+2m+mn+k+2$.

- *Affirmative answer*: If there exists at least one subset of set S which does not include any forbidden set, and its cardinality is not less than k, then there is an object $d_{2n+2m+mn+k+1}$ in the corresponding cell with label 2 as described above. One of the cells with label 2 containing object $d_{2n+2m+mn+k+1}$ gets the objects yes and g in exchange of $d_{2n+2m+mn+k+1}$ by the rule r_{14}. In the next step, the object yes in cell 2 leaves the system by the rule r_{15}, signaling the fact that there exists one subset A of S such that $|A| \geq k$, and A does not include any forbidden set from the family F. At that step, the cell with label 1 contains the object $e_{2n+2m+mn+k+3}$ but not the object g. The computation halts at step $2n + 2m + mn + k + 3$.
- *Negative answer*: In this case, the subscript of counter e_i reaches $2n + 2m + mn + k + 3$ and the object g is still in the cell with label 1. The object no can be moved to the environment by the rules r_{16} and r_{17}, signaling that there is no subset A of S such that $|A| \geq k$, and A does not include any forbidden set from the family F. The computation finishes at step $2n + 2m + mn + k + 4$.

3.2. Formal Verification

In this subsection, we prove that the family built above solves the common algorithmic decision problem in a polynomial time, according to Definition 1. First of all, this definition requires that the defined family is *consistent*, in the sense that all systems of the family must be recognizer tissue P systems with cell division. By construction (types of rules and working alphabet) it is clear that it is a family of tissue P systems with cell division. In order to show that all members in $\mathbf{\Pi}$ are recognizer systems it suffices to check that all the computations halt (this will be deduced from the polynomial boundness), and that either an object yes or an object no is sent out exactly in the last step of the computation (this will be deduced from the soundness and completeness).

3.2.1. Polynomial uniformity of the family

We now show that the family $\mathbf{\Pi} = \{\Pi(\langle n, m, k\rangle) \mid n, m, k \in \mathbb{N}\}$ defined above is polynomially uniform by Turing machines. To this aim we are going to prove that it is possible to build $\Pi(\langle n, m, k\rangle)$ in polynomial time with respect to the size of the instances of the CADP.

It is easy to check that the rules of a system $\Pi(\langle n, m, k\rangle)$ of the family are defined recursively from the values n, m and k. Besides, the necessary resources to build an element of the family are of a polynomial order, as shown below:

- Size of the alphabet: $4mn + 11n + 6m + 2k + 10 \in \Theta(mn)$;
- Initial number of cells: $2 \in \Theta(1)$;
- Initial number of objects: $n + 8 \in \Theta(n)$;

- Number of rules: $4mn + 8n + 5m + k + 7 \in \Theta(mn)$;
- Maximal length of a rule: $5 \in \Theta(1)$.

Therefore, a deterministic Turing machine can build $\Pi(\langle n, m, k\rangle)$ in a polynomial time with respect to n, m and k.

3.2.2. Polynomial boundness of the family

In order to prove that the system $\Pi(s(u))$ with input $cod(u)$ is polynomially bounded, it suffices to find the moment in which any computation halts, or at least, an upper bound for it.

Proposition 1. *The family* $\mathbf{\Pi} = \{\Pi(\langle n, m, k\rangle) \mid n, m, k \in \mathbb{N}\}$ *is polynomially bounded with respect to* $(\texttt{CADP}, cod, s)$.

Proof. We will informally go through the stages of the computation in order to estimate an upper bound for the number of steps. The computation will be checked more in detail when addressing the soundness and completeness proof.

Let $u = (\{s_1, s_2, \cdots, s_n\}, (B_1, B_2, \cdots, B_m), k)$ be an instance of CADP. We shall check what happens during the computation of the system $\Pi(\langle n, m, k\rangle)$ with the input $cod(u)$ in order to find the halting step, or at least, an upper bound for it.

Firstly, the generation stage has exactly n steps, performing all the divisions of the cells of the system. The order in which the divisions are performed is nondeterministically chosen in each computation, but the divisions are carried out in the first n steps in all cases.

After one more step, the objects b_{n+1}, c_{n+1} and d_{n+1} arrive at cells labeled by 2, and then the checking stage starts with the rule $r_{7,j}$. The objects F_j introduce the objects $F_{1,j}$. This phase needs at most n steps. Then we need further m steps for $F_{1,j}$ to grow its first subscript and introduce m copies of f_j for each of the forbidden sets. From step $2n + m + 2$, the rules $r_{11,ij}$ are applied in each cell with label 2 in order to check which forbidden sets are not included in the corresponding subset. This checking needs at most mn steps. When the subscript of d_i grows to $2n+m+mn+1$, the system starts to check whether there exists a subset A that does not include any forbidden set. This process needs at most m steps. Then we need k steps to check whether the cardinality of the subset is not less than k by applying $r_{13,j}$. The checking stage ends at step $2n + 2m + mn + k + 1$.

The last one is the answer stage. The longest case is obtained when the answer is negative. In this case, at step $2n + 2m + mn + k + 2$ only the counter e_i is working and produces the object $e_{2n+2m+mn+k+3}$. In next step the object $e_{2n+2m+mn+k+3}$ works together with object g bringing the object no to a cell with label 2. Finally, at step $2n + 2m + mn + k + 4$, the object no is sent to the environment.

Therefore, there exists a polynomial bound (with respect to n, m and k) on the number of steps of the computation. □

3.2.3. Soundness and completeness of the family

In order to prove the soundness and completeness of the family $\mathbf{\Pi}$ with respect to $(\texttt{CADP}, cod, s)$, we shall prove that for a given instance u of CADP, the system $\Pi(s(u))$ with input $cod(u)$ sends out the object yes if and only if the answer to the problem for the considered instance u is affirmative, or, otherwise, the object no is sent out. In both cases the answer will be sent to the environment at the last step of the computation.

Recall that each cell with label 2 represents a subset of the given set S. For simplification in the following proofs, we introduce the function ψ from $P(S) \times \mathbb{N}$ to Γ as follows:

$$\psi(A, j) = \begin{cases} T_j, & \text{if } s_j \in A, \\ F_j, & \text{if } s_j \notin A, \end{cases}$$

where $A \subseteq S = \{s_1, s_2, \cdots, s_n\}$, $1 \leq j \leq n$.

Given a computation $\mathcal{C}$, we denote by $\mathcal{C}_i$ the configuration at the i-th step. Moreover, $\mathcal{C}_i(j)$ will denote the multiset associated with the cell or the enviroment with label j in such configuration. In what follows, the system means the system $\Pi(s(u))$ with input multiset $cod(u)$.

We start analyzing the generation (i.e., the first n steps of the computation). It has two parallel processes, each of them in one kind of cells with label 1 or 2.

Proposition 2. *If $\mathcal{C}$ is an arbitrary computation of the system, then $\mathcal{C}_i(1) = \{\{b_{i+1}^{2^i}, c_{i+1}^{2^i}, d_{i+1}^{2^i}, e_{i+1}, g, \texttt{yes}, \texttt{no}\}\}$, for all i $(0 \leq i \leq n)$.*

Proof. We shall reason by induction on i.

Base Case. We have $\mathcal{C}_0(1) = \{\{b_1, c_1, d_1, e_1, g, \texttt{yes}, \texttt{no}\}\}$, thus the proposition holds for $i = 0$.

Let $i < n$ and let us suppose the result holds for i. By inductive hypothesis, for all i $(1 \leq i < n)$, we have $\mathcal{C}_i(1) = \{\{b_{i+1}^{2^i}, c_{i+1}^{2^i}, d_{i+1}^{2^i}, e_{i+1}, g, \texttt{yes}, \texttt{no}\}\}$. In this configuration, only the rules $r_{2,i} - r_{5,i}$ can be applied in cell with label 1, and therefore $\mathcal{C}_{i+1}(1) = \{\{b_{i+2}^{2^{i+1}}, c_{i+2}^{2^{i+1}}, d_{i+2}^{2^{i+1}}, e_{i+2}, g, \texttt{yes}, \texttt{no}\}\}$. $\square$

Proposition 3. *If $\mathcal{C}$ is an arbitrary computation of the system, then*

- *for each subset $A \subseteq \{s_1, s_2, \cdots, s_n\}$ there exists only one cell with label 2 in $\mathcal{C}_n$ whose multiset is $\{\{s_{i,j} \mid 1 \leq i \leq m, 1 \leq j \leq n\}\} \cup \{\{f, \psi(A,1), \ldots, \psi(A,n)\}\}$, and the multiset $\{\{\psi(A,1), \ldots, \psi(A,n)\}\}$ does not appear in the other cells with label 2;*
- *there exist exactly 2^n cells with label 2 in configuration $\mathcal{C}_n$, in each one of them there is an object f.*

Proof. In the initial configuration, $\mathcal{C}_0(2) = \{\{f, a_1, a_2, \cdots, a_n\}\}$. In the following n steps, only rule $r_{1,j}$ can be applied to produce cells with label 2 containing the objects $\psi(A,1), \cdots, \psi(A,n)$. More precisely, for each j $(1 \leq j \leq n)$, if $s_j \in A$ then after applying the rule $r_{1,j}$, we take the cell where T_j occurs, and if $s_j \notin A$, we take the cell where F_j occurs. In this way, it is easy to see that the multiset $\{\{\psi(A,1),\cdots,\psi(A,n)\}\}$ does not appear in the other cells with label 2.

The order that objects a_j participate in the division process is non-deterministic. But after n steps no more division rules are applied in any cell with label 2. The rules $r_{1,j}$ ensure that each of the 2^n possible subsets will be represented by one cell with label 2 in the system. □

Proposition 4. *If $\mathcal{C}$ is an arbitrary computation of the system, then*

- $\mathcal{C}_{n+1}(1) = \{\{e_{n+2}, f^{2^n}, g, \texttt{yes}, \texttt{no}\}\}$*;*
- *for each subset $A \subseteq \{s_1, s_2, \cdots, s_n\}$ there exists only one cell with label 2 in $\mathcal{C}_{n+1}$ whose multiset is $\{\{s_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}\} \cup \{\{b_{n+1}, c_{n+1}, d_{n+1}, \psi(A,1), \cdots, \psi(A,n)\}\}$, and the multiset $\{\{\psi(A,1), \cdots, \psi(A,n)\}\}$ does not appear in the other cells with label 2.*

Proof. The multiset $\mathcal{C}_{n+1}(1)$ is obtained from $\mathcal{C}_n(1)$ by the application of the rule r_6. The object e_{n+1} increases its subscript by one by the rule $r_{5,i}$, so one copy of e_{n+2} appears in $\mathcal{C}_{n+1}(1)$. By proposition 3, at step n, there exist exactly 2^n cells with label 2, each of them containing one copy of object f. At step $n+1$, the rule r_6 can be applied, 2^n copies of objects b_{n+1}, c_{n+1} and d_{n+1} in cell with label 1 are traded for f from the cells with label 2. Due to the maximality of the parallel usage of rules, each cell with label 2 gets one copy of objects b_{n+1}, c_{n+1} and d_{n+1}, while the cell with label 1 gets 2^n copies of object f. Therefore, $\mathcal{C}_{n+1}(1) = \{\{e_{n+2}, f^{2^n}, g, \texttt{yes}, \texttt{no}\}\}$; and for each subset $A \subseteq \{s_1, s_2, \cdots, s_n\}$ there exists only one cell with label 2 in $\mathcal{C}_{n+1}$ whose multiset is $\{\{s_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}\} \cup \{b_{n+1}, c_{n+1}, d_{n+1}, \psi(A,1), \cdots, \psi(A,n)\}$. By Proposition 3 and the above proof, it is no difficult to see that the multiset $\{\{\psi(A,1), \cdots, \psi(A,n)\}\}$ does not appear in the other cells with label 2. □

Proposition 5. *Let $\mathcal{C}$ be an arbitrary computation of the system. $\mathcal{C}_{2n+m+1}(1) = \{\{e_{2n+m+2}, f^{2^n}, g, \texttt{yes}, \texttt{no}\}\}$ holds.*

Proof. From step $n+2$ to step $2n+m+1$ of the computation, only rule $r_{5,i}$ is applicable in cell with label 1, yielding $\mathcal{C}_{2n+m+1}(1) = \{\{e_{2n+m+2}, f^{2^n}, g, \texttt{yes}, \texttt{no}\}\}$. □

Proposition 6. *Let $\mathcal{C}$ be an arbitrary computation of the system. For each subset $A \subseteq \{s_1, s_2, \cdots, s_n\}$, there exists only one cell with label 2 in $\mathcal{C}_{2n+m+1}$ whose multiset is $\{\{s_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}\} \cup \{\{T_j \mid 1 \le j \le n, \psi(A,j) = T_j\}\} \cup \{\{F_{m+1,j}, f_j^m \mid 1 \le j \le n, \psi(A,j) = F_j\}\} \cup \{\{c_{n+1}, b_{2n+m+1}, d_{2n+m+1}\}\}$, and the multiset $\{\{T_j \mid 1 \le j \le n, \psi(A,j) = T_j\}\} \cup \{\{F_{m+1,j}, f_j^m \mid 1 \le j \le n, \psi(A,j) = F_j\}\}$ does not appear in the other cells with label 2.*

Proof. Based on Proposition 4, we prove Proposition 6 holds.

For each subset $A \subseteq \{s_1, s_2, \cdots, s_n\}$ there exists only one cell with label 2 in $\mathcal{C}_{n+1}$ whose multiset is $\{\{s_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}\} \cup \{\{b_{n+1}, c_{n+1}, d_{n+1}, \psi(A,1), \cdots, \psi(A,n)\}\}$, and the multiset $\{\{\psi(A,1), \cdots, \psi(A,n)\}\}$ does not appear in the other cells with label 2.

From step $n+2$ to step $2n+m+1$, the objects $s_{i,j}$ $(1 \le i \le m, 1 \le j \le n)$ keep unchanged because no rules can be applied to them.

By the rules $r_{9,i}$ and $r_{10,i}$, the subscripts of objects b_{n+1} and d_{n+1} in $\mathcal{C}_{n+1}$ increase, until reaching the value $2n+m+1$ at step $2n+m+1$. Clearly, the multiplicity of object b_{2n+m+1} is one, and the multiplicity of object d_{2n+m+1} is also one.

By the rules $r_{7,j}$, with the object c_{n+1}, the objects F_j introduce the objects $F_{1,j}$. It takes n steps to complete this process. By the rules $r_{8,ij}$, for each $F_{1,j}$, it takes m steps to introduce m copies of f_j, and the first subscript of $F_{1,j}$ grows to $m+1$. The objects T_j keep unchanged. Therefore, at step $2n+m+1$, the corresponding cell with label 2 has multiset $\{\{s_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}\} \cup \{\{T_j \mid 1 \le j \le n, \psi(A,j) = T_j\}\} \cup \{\{F_{m+1,j}, f_j^m \mid 1 \le j \le n, \psi(A,j) = F_j\}\} \cup \{\{c_{n+1}, b_{2n+m+1}, d_{2n+m+1}\}\}$.

By Proposition 4, in $\mathcal{C}_{n+1}$, the multiset $\{\{\psi(A,1), \ldots, \psi(A,n)\}\}$ appears in only one cell with label 2. By this observation and the above proof, it is not difficult to see that the multiset $\{\{T_j \mid 1 \le j \le n, \psi(A,j) = T_j\}\} \cup \{\{F_{m+1,j}, f_j^m \mid 1 \le j \le n, \psi(A,j) = F_j\}\}$ appears in only one cell with label 2. Therefore, Proposition 6 holds. □

Proposition 7. *Let $\mathcal{C}$ be an arbitrary computation of the system.* $\mathcal{C}_{2n+m+mn+1}(1) = \{\{e_{2n+m+mn+2}, f^{2^n}, g, \texttt{yes}, \texttt{no}\}\}$ *holds.*

Proof. By the rule $r_{5,i}$, the subscript of object e_{2n+m+2} in $\mathcal{C}_{2n+m+1}(1)$ grows to the value $2n+m+mn+2$ at step $2n+m+nm+1$. The objects f^{2^n}, g, $\texttt{yes}$, $\texttt{no}$ keep unchanged. Therefore, Proposition 7 holds. □

Proposition 8. *Let $\mathcal{C}$ be an arbitrary computation of the system. For each subset $A \subseteq \{s_1, s_2, \cdots, s_n\}$, there exists only one cell with label 2 in $\mathcal{C}_{2n+m+mn+1}$ whose multiset is $\{\{T_j \mid 1 \le j \le n, \psi(A,j) = T_j\}\} \cup \{\{F_{m+1,j} \mid 1 \le j \le n, \psi(A,j) = F_j\}\} \cup \{\{r_i \mid s_j \in B_i, s_j \notin A, 1 \le i \le m, 1 \le j \le n\}\} \cup \{\{f_j^{\alpha_j} \mid 1 \le j \le n\}\} \cup \{\{c_{n+1}, b_{2n+m+1}, d_{2n+m+mn+1}\}\}$, where α_j $(0 \le j \le m)$ is the number of copies of f_j which remained in this cell.*

Proof. By Proposition 6, for each subset $A \subseteq \{s_1, s_2, \cdots, s_n\}$, there exists only one cell with label 2 in $\mathcal{C}_{2n+m+1}$ whose multiset is $\{\{s_{i,j} \mid 1 \le i \le m, 1 \le j \le n\}\} \cup \{\{T_j \mid 1 \le j \le n, \psi(A,j) = T_j\}\} \cup \{\{F_{m+1,j}, f_j^m \mid 1 \le j \le n, \psi(A,j) = F_j\}\} \cup \{\{c_{n+1}, b_{2n+m+1}, d_{2n+m+1}\}\}$. In the following, we consider this unique cell with label 2.

The objects T_j, c_{n+1}, b_{2n+m+1} and $F_{m+1,j}$ keep unchanged. By the rule $r_{10,i}$, the subscript of object d_{2n+m+1} in $\mathcal{C}_{2n+m+1}$ increases, until reaching the value $2n+m+mn+1$ at step $2n+m+mn+1$. Clearly, the number of copies of object $d_{2n+m+mn+1}$ is one.

With the presence of b_{2n+m+1} in $\mathcal{C}_{2n+m+1}$ (not appearing in $\mathcal{C}_i$ $(i < 2n+m+1)$), the rule $r_{11,ij}$ can be applied. If object f_j (i.e., $s_j \notin A$) and object $s_{i,j}$ (i.e., $s_j \in B_i$) appear, the rule $r_{11,ij}$ is applied and an object r_i (which means that B_i is not included in A) is introduced into the corresponding cell with label 2. Because we have only one copy of b_{2n+m+1} in each cell with label 2, each forbidden set contains at most n objects, and we have m forbidden sets, it takes at most mn steps for these operations.

Note that at step $2n + m + mn + 1$ it is possible to have some copies of f_j in the corresponding cell with label 2 (the m copies of f_j are not completely consumed by the rule $r_{11,ij}$). Let α_j $(0 \leq j \leq m)$ be the number of copies of f_j which remained in the corresponding cell with label 2.

Therefore, in $\mathcal{C}_{2n+m+mn+1}$, the corresponding cell with label 2 has multiset $\{\{T_j \mid 1 \leq j \leq n, \psi(A,j) = T_j\}\} \cup \{\{F_{m+1,j} \mid 1 \leq j \leq n, \psi(A,j) = F_j\}\} \cup \{\{r_i \mid s_j \in B_i, s_j \notin A, 1 \leq i \leq m, 1 \leq j \leq n\}\} \cup \{\{f_j^{\alpha_j} \mid 1 \leq j \leq n\}\} \cup \{\{c_{n+1}, b_{2n+m+1}, d_{2n+m+mn+1}\}\}$.

By Proposition 6 and the above proof, it is not difficult to see that in $\mathcal{C}_{2n+m+mn+1}$ the cell with the multiset described in Proposition 8 is unique. □

Proposition 9. *If* $\mathcal{C}$ *is an arbitrary computation of the system, then* $\mathcal{C}_{2n+2m+mn+1}(1) = \{\{e_{2n+2m+mn+2}, f^{2^n}, g, \texttt{yes}, \texttt{no}\}\}$ *holds.*

Proof. By the rule $r_{5,i}$, the subscript of object $e_{2n+m+mn+2}$ in $\mathcal{C}_{2n+m+mn+1}(1)$ grows to the value $2n + 2m + mn + 2$ at step $2n + 2m + mn + 1$. The objects f, g, $\texttt{yes}$ and $\texttt{no}$ keep unchanged. Therefore, Proposition 9 holds. □

Proposition 10. *If* $\mathcal{C}$ *is an arbitrary computation of the system, then*

- *there exist exactly* 2^n *cells with label 2 in configuration* $\mathcal{C}_{2n+2m+mn+1}$*;*
- *if* $A \subseteq \{s_1, s_2, \cdots, s_n\}$ *and* A *includes some of the forbidden sets, then in the corresponding cell with label 2 in* $\mathcal{C}_{2n+2m+mn+1}$ *the multiset contains an object* $d_{2n+m+mn+1+\alpha}$*, where* $0 \leq \alpha < m$ *such that all the clauses* $B_1, \cdots, B_\alpha$ *are not included in* A*, but* $B_{\alpha+1}$ *is included in* A*;*
- *if* $A \subseteq \{s_1, s_2, \cdots, s_n\}$ *and* A *does not include any forbidden set, then there exists one cell with label 2 in* $\mathcal{C}_{2n+2m+mn+1}$ *whose associated multiset contains an object* $d_{2n+2m+mn+1}$*.*

Proof. From the configuration $\mathcal{C}_{2n+m+mn+1}$, we start to check whether there exists a subset $A \subseteq \{s_1, s_1, \cdots, s_n\}$ that does not include any forbidden set. Such checking is simultaneous in all 2^n cells with label 2.

Let us consider a subset A of S. By Proposition 8, the object $d_{2n+m+mn+1}$ appears, the rule $r_{12,i}$ can be applied. The forbidden sets are checked in the order from B_1 to B_m. For each forbidden set which is not included in A (that is, the corresponding object r_i appears), we increase by one the subscript of d_i (it is possible to have several copies of r_i, but only one copy of r_i is used by the rule $r_{12,i}$), hence the subscript of d_i reaches the value $2n+2m+mn+1$ if and only if any forbidden set is not included in A. If the sets $B_1, \cdots, B_\alpha$ $(0 \leq \alpha < m)$ are not included in A, but $B_{\alpha+1}$ is included in A, then the subscript of d_i can only reach the value $2n+m+mn+1+\alpha$. Therefore, Proposition 10 holds. □

Proposition 11. *If* $\mathcal{C}$ *is an arbitrary computation of the system, then we have* $\mathcal{C}_{2n+2m+mn+k+1}(1) = \{\{e_{2n+2m+mn+k+2}, f^{2^n}, g, \texttt{yes}, \texttt{no}\}\}$.

Proof. By the rule $r_{5,i}$, the subscript of object $e_{2n+2m+mn+2}$ in $\mathcal{C}_{2n+2m+mn+1}(1)$ grows to the value $2n + 2m + mn + k + 2$ at step $2n + 2m + mn + k + 1$. The objects $f, g, \texttt{yes}, \texttt{no}$ keep unchanged. Therefore, Proposition 11 holds. □

Proposition 12. *If $\mathcal{C}$ is an arbitrary computation of the system, then*

- *there exist exactly 2^n cells with label 2 in configuration $\mathcal{C}_{2n+2m+mn+k+1}$;*
- *if $A \subseteq \{s_1, s_2, \cdots, s_n\}$, $k \in \mathbb{N}$, $0 \leq |A| < k$, and A does not include any forbidden set, then in the corresponding cell with label 2 in $\mathcal{C}_{2n+2m+mn+k+1}$ the multiset contains an object $d_{2n+2m+mn+1+\alpha}$, where $\alpha = |A|$ and $0 \leq \alpha < k$;*
- *if $A \subseteq \{s_1, s_2, \cdots, s_n\}$, $k \in \mathbb{N}$, $|A| \geq k$, and A doesn't include any forbidden set, then there exists a cell with label 2 in $\mathcal{C}_{2n+2m+mn+k+1}$ whose associated multiset contains an object $d_{2n+2m+mn+k+1}$.*

Proof. From the configuration $\mathcal{C}_{2n+2m+mn+1}$, we start to check that, for each $A \subseteq \{s_1, s_1, \cdots, s_n\}$ which does not include any forbidden set, whether the cardinality of A is not less than k. Such checking is simultaneous in all 2^n cells with label 2.

Let us consider a subset A of S. By Proposition 10, the object $d_{2n+2m+mn+1}$ appears, the rule $r_{13,i}$ can be applied. The cardinality of A is equal to the number of objects T_j in the corresponding cell. For each T_j (that is, the element s_j in the subset A), we increase by one the subscript of d_i, hence the subscript of d_i reaches the value $2n + 2m + mn + k + 1$ if and only if the number of s_j in subset A is not less than k. If the number of s_j in subset A is less than k, then the subscript of d_i can only reach the value $2n + 2m + mn + 1 + \alpha$, where $\alpha = |A|$. Therefore, Proposition 12 holds. □

Proposition 13. *Let $\mathcal{C}$ be an arbitrary computation of the system, and suppose that there exists a subset A of S such that $|A| \geq k$ and it does not include any forbidden set from the family F, then*

(a) $\mathcal{C}_{2n+2m+mn+k+2}(1) = \{\{e_{2n+2m+mn+k+3},\ f^{2^n},\ \texttt{no},\ d_{2n+2m+mn+k+1}\}\}$,

(b) $\mathcal{C}_{2n+2m+mn+k+3}(1) = \{\{e_{2n+2m+mn+k+3},\ f^{2^n},\ \texttt{no},\ d_{2n+2m+mn+k+1}\}\}$,

and the object `yes` *appears in* $\mathcal{C}_{2n+2m+mn+k+3}(0)$.

Proof. The configuration of item (a) is obtained by the application of rules $r_{5,i}$ and r_{14} to the previous configuration $\mathcal{C}_{2n+2m+mn+k+1}$. By the rule $r_{5,i}$, the object $e_{2n+2m+mn+k+2}$ in $\mathcal{C}_{2n+2m+mn+k+1}(1)$ grows by one its subscript at step $2n + 2m + mn + k + 2$. By Proposition 12, there exists a cell with label 2 in $\mathcal{C}_{2n+2m+mn+k+1}$ whose multiset contains an object $d_{2n+2m+mn+k+1}$. The object $d_{2n+2m+mn+k+1}$ is moved to the cell with label 1 by the rule r_{14}, where the objects `yes`, g are moved to a cell with label 2 (if there are more than one cells with label 2 whose multisets contain $d_{2n+2m+mn+k+1}$, then the target cell for `yes`, g is non-deterministically chosen among the cells containing object $d_{2n+2m+mn+k+1}$).

The configuration of item (b) is obtained by the application of rule r_{15} to the previous configuration $\mathcal{C}_{2n+2m+mn+k+2}$. At step $2n + 2m + mn + k + 3$, there is no rule applied in cell with label 1. But with the application of r_{15} in cell with label 2, the object `yes` leaves the system to the environment, signaling there exists a subset A of S such that $|A| \geq k$ and it does not include any forbidden set from the family F. The only one copy of object g is consumed by the rule r_{14}, so the rule r_{16} cannot be applied. The object `no` cannot exit into the environment. □

Proposition 14. *If $\mathcal{C}$ is an arbitrary computation of the system, there is no subset A of S such that $|A| \geq k$, and it does not include any forbidden set from the family F, then*

(a) $\mathcal{C}_{2n+2m+mn+k+2}(1) = \{\{e_{2n+2m+mn+k+3}, f^{2^n}, g, \texttt{ yes}, \texttt{no}\}\}$,

(b) $\mathcal{C}_{2n+2m+mn+k+3}(1) = \{\{f^{2^n}, \texttt{yes}\}\}$,

(c) $\mathcal{C}_{2n+2m+mn+k+4}(1) = \{\{f^{2^n}, \texttt{yes}\}\}$,

and the object `no` *appears in* $\mathcal{C}_{2n+2m+mn+k+4}(0)$.

Proof. If there exists no subset A of S such that $|A| \geq k$ and A does not include any forbidden set from the family F, by Proposition 12, all cells with label 2 do not contain object $d_{2n+2m+mn+k+1}$. Of course, the cell with label 1 cannot get object $d_{2n+2m+mn+k+1}$. The configurations of item (a) is obtained by the application of rule $r_{5,i}$ to the previous configuration $\mathcal{C}_{2n+2m+mn+k+1}$. By the rule $r_{5,i}$, the object $e_{2n+2m+mn+k+2}$ in $\mathcal{C}_{2n+2m+mn+k+1}(1)$ grows by one its subscript at step $2n+2m+mn+k+3$. The objects f, g, `yes`, `no` keep unchanged.

At step $2n + 2m + mn + k + 3$, the rule r_{16} can be applied, then the objects $e_{2n+2m+mn+k+3}, g,$ `no` are sent to a cell with label 2. So the configuration of item (b) is obtained by the application of the rule r_{16} to the previous configuration $\mathcal{C}_{2n+2m+mn+k+2}$.

At last, the configuration of item (c) is obtained by the application of rules r_{17} to the previous configuration $\mathcal{C}_{2n+2m+mn+k+3}$. The object `no` leaves the system to the environment signaling that there exists no subset A of S such that $|A| \geq k$ and A does not include any forbidden set from the family F. □

3.3. Main Results

From the discussion in the previous sections and according to the definition of solvability given in Section 2, we have the following result:

Theorem 1. `CADP` $\in \mathbf{PMC}_{TDC}$.

Corollary 1. $\mathbf{NP} \cup \mathbf{co} - \mathbf{NP} \subseteq \mathbf{PMC}_{TDC}$.

Proof. It suffices to make the following observations: the `CADP` is NP–complete, `CADP` $\in \mathbf{PMC}_{TDC}$ and this complexity class is closed under polynomial-time reduction and under complement. □

4. Conclusions

In this work, a family of recognizer tissue P system with cell division is designed to solve the `CADP`. Although the algorithm proposed here is theoretically proved to be efficient for `CADP`, the real implementation of such algorithms is a great challenge.

A solution to CADP by P systems with active membranes was proposed in [14], where four types of rules were applied in those systems: object evolution rules, communication rules, dissolving rules and division rules for elementary membranes; moreover, three charges $+, -, 0$ are used to control the application of these types of rules. The solution to the CADP given in this work is based on tissue P system with cell division, where two kinds of rules are used: communication and division rules. Moreover, no electrical charges are associated with the membranes.

Acknowledgements. The authors wish to acknowledge the support of the project TIN2009-13192 of the Ministerio de Ciencia e Innovación of Spain, cofinanced by FEDER funds, and the "Proyecto de Excelencia con Investigador de Reconocida Valía" of the Junta de Andalucía under grant P08-TIC04200. The first two authors are also supported by National Natural Science Foundation of China (61033003, 91130034 and 30870826), Ph.D. Programs Foundation of Ministry of Education of China (20100142110072), and Natural Science Foundation of Hubei Province (2008CDB113 and 2011CDA027).

References

[1] Alhazov A., Martín-Vide C., Pan L., *Solving a PSPACE-complete problem by P systems with restricted active membranes*, Fundamenta Informaticae, 2003, Vol. **58**(2), pp. 67–77.

[2] Alhazov A., Pérez-Jiménez M. J., *Uniform solution of QSAT using polarizationless active membranes*, Lecture Notes in Computer Science, 2007, Vol. **4664**, pp. 122–133.

[3] Díaz-Pernil D., Gutiérrez-Naranjo M., Pérez-Jiménez M. J., Riscos-Núñez A., *A linear-time tissue P system based solution for the 3-coloring problem*, Electronic Notes in Theoretical Computer Science, 2007, Vol. **171**, pp. 81–93.

[4] Díaz-Pernil D., Gutiérrez-Naranjo M. A., Pérez-Jiménez M. J., Riscos-Núñez A., *Solving subset sum in linear time by using tissue P systems with cell division*, Lecture Notes in Computer Science, 2007, Vol. **4527**, pp. 170–179.

[5] Díaz-Pernil D., Gutiérrez-Naranjo M. A., Pérez-Jiménez M. J., Riscos-Núñez A., *Solving the partition problem by using tissue-like P systems with cell division*, Proceedings of 6th Brainstorming Week on Membrane Computing, Report RGNC 01/08, 2008, Fenix Editoria, pp. 124–134.

[6] Head T., Yamamura M., Gal S., *Aqueous computing: writing on molecules*, Proceedings of the Congress on Evolutionary Computation 1999, IEEE Service Center, Piscataway, NJ, 1999, pp. 1006–1010.

[7] Martín-Vide C., Pazos J., Păun G., Rodríguez Patón A., *A new class of symbolic abstract neural nets: tissue P systems*, Lecture Notes in Computer Science, 2002, Vol. **2387**, pp. 290–299.

[8] Martín-Vide C., Pazos J., Păun G., Rodríguez Patón A., *Tissue P systems*, Theoretical Computer Science, 2003, Vol. **296**, pp. 295–326.

[9] Pan L., Martín-Vide C., *Solving multidimensional 0-1 knapsack problem by P systems with input and active membranes*, Journal of Parallel and Distributed Computing, 2005, Vol. **65**, pp. 1578–1584.

[10] Păun G., *Computing with membranes*, Journal of Computer and System Sciences, 2000, Vol. **61**(1), pp. 108–143.

[11] Păun G., *Membrane Computing. An Introduction*, Springer–Verlag, Berlin, 2002.

[12] Păun G., Pérez-Jiménez M. J., Riscos-Núñez A., *Tissue P systems with cell division*, International Journal of Computers, Communications and Control, 2008, Vol. **3**(3), pp. 295–303.

[13] Păun G., Rozenberg G., Salomaa A. (eds.), *The Oxford Handbook of Membrane Computing*, Oxford University Press, 2010.

[14] Pérez-Jiménez M. J., Romero-Campero F. J., *Attacking the common algorithmic problem by recognizer P system*, Lecture Notes in Computer Science, 2005, Vol. **3354**, pp. 304–315.

[15] P systems web page `http://ppage.psystems.eu/`