ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume 21, Number 3, 2018, 288–297

The Unique Satisfiability Problem from a Membrane Computing Perspective

David ORELLANA-MARTÍN[1], Luis VALENCIA-CABRERA[1], Agustín RISCOS-NÚÑEZ[1], and Mario J. PÉREZ-JIMÉNEZ[1]

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

Abstract. Complexity class **DP** is the class of "differences" of any two languages in **NP**. It verifies that $\mathbf{NP} \cup \textbf{co-NP} \subseteq \mathbf{DP} \subseteq \mathbf{P}^{\mathbf{NP}}$, where $\mathbf{P}^{\mathbf{NP}}$ is the second level of the polynomial hierarchy, specifically, it is the class of languages decidable by a deterministic polynomial-time Turing machine having access to an **NP** oracle. The *unique sastifiability problem* (UNIQUE SAT) is a well known **DP** problem which has been proved to be **co-NP**-hard. In this paper, a uniform and polynomial time solution for the UNIQUE SAT problem is given by a family of polarizationless P systems with active membranes and division rules only for elementary membranes, without dissolution rules but using minimal cooperation and minimal production in object evolution rules.

Key-words: Complexity class **DP**; polarizationless P systems with active membranes; cooperative rules; UNIQUE SAT problem.

1. Preliminaries

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

A *multiset* over an alphabet Γ is an ordered pair (Γ, f) where f is a mapping from Γ onto the set of natural numbers $\mathbb{N}$. The *support* of a multiset $m = (\Gamma, f)$ is defined as $supp(m) = \{x \in \Gamma \mid f(x) > 0\}$. A multiset is finite (respectively, empty) if its support is a finite (respectively,

empty) set. We denote by $\emptyset$ the empty multiset. Let $m_1 = (\Gamma, f_1)$, $m_2 = (\Gamma, f_2)$ be multisets over Γ, then the union of m_1 and m_2, denoted by $m_1 + m_2$, is the multiset (Γ, g), where $g(x) = f_1(x) + f_2(x)$ for each $x \in \Gamma$.

1.1. Decision problems and languages

Roughly speaking, a decision problem X is one whose solution/answer is either "yes" or "no". This can be formally defined by an ordered pair (I_X, θ_X), where I_X is a language over a finite alphabet Σ_X and θ_X is a total Boolean function over I_X. The elements of I_X are called *instances* of the problem X. Each decision problem X has associated a language L_X over the alphabet Σ_X as follows: $L_X = \{u \in \Sigma_X \mid \theta_X(u) = 1\}$, that is, L_X is the set of inputs for which the answer is affirmative. Conversely, every language L over an alphabet Σ has associated a decision problem $X_L = (I_{X_L}, \theta_{X_L})$ as follows: $I_{X_L} = \Sigma^*$ and $\theta_{X_L}(u) = 1$ if and only if $u \in L$. Then, given a decision problem X we have $X_{L_X} = X$, and given a language L over an alphabet Σ we have $L_{X_L} = L$.

The complement problem $\overline{X}$ of a decision problem $X = (I_X, \theta_X)$ is the decision problem $(I_X, \neg\theta_X)$, that is for each instance the answer of $\overline{X}$ is "yes" if and only if the answer of X is "no".

1.2. The Cantor pairing function

The Cantor pairing function encode pairs of natural numbers by single natural numbers and it is defined as follows: for each $n, p \in \mathbb{N}$

$$\langle n, p\rangle = \frac{(n+p)(n+p+1)}{2} + n$$

The Cantor pairing function is a primitive recursive bijective function from $\mathbb{N} \times \mathbb{N}$ onto $\mathbb{N}$. Then, for each $t \in \mathbb{N}$ there exist unique natural numbers $n, p \in \mathbb{N}$ such that $t = \langle n, p\rangle$.

2. Minimal cooperation and minimal production in polarizationless P system with active membranes

Membrane Computing is an emergent branch of Natural Computing providing distributed parallel non deterministic computing models whose computational devices are called *membrane systems*, containing processing units called *compartments*. This computing paradigm is inspired by some basic biological features, by the structure and functioning of the living cells, as well as from the cooperation of cells in tissues, organs, and organisms. There exist basically two ways to consider computational devices: cell-like membrane systems and tissue-like membrane systems. The first one, using the biological membranes arranged hierarchically, inspired from the structure of the cell, and the second one using the biological membranes placed in the nodes of a graph, inspired from the cell inter-communication in tissues. We refer the interested reader to [9,11] for more details of these models.

P systems with active membranes were first introduced by Gh. Păun [10], introducing electrical charges (polarizations) associated to membranes, but the rules are non-cooperative and there are no priorities. Nevertheless, the class of all problems solvable in a uniform way in polynomial

time by means of families of P systems with active membranes which use division for elementary and non-elementary membranes coincides with **PSPACE** [13]. Consequently, the usual framework of P systems with active membranes for solving decision problems is too powerful from the computational complexity point of view.

Polarizationless P systems with active membranes were initially studied in [1, 2]. However, this initial approach proposed to replace polarizations by a somehow equivalent or even more powerful ingredient: the ability to change the label of the membranes.

Throughout this paper, P systems with active membranes but no electrical charges nor membrane relabelling are considered. When dissolution rules are forbidden in this framework, only problems in class **P** can be solved in an efficient way [5]. In order to get the *presumable efficiency* (ability to solve **NP**-complete problems in a uniform way in polynomial time) of these systems, minimal cooperation and minimal production in object evolution rules are allowed.

Definition 1. *A polarizationless P system with active membranes, without dissolution, with division rules for elementary membranes and making use of minimal cooperation and minimal production in object evolution rules of degree $q \geq 1$, is a tuple $\Pi = (\Gamma, H, \mu, \mathcal{M}_1, \ldots, \mathcal{M}_q, \mathcal{R}, i_{out})$, where:*

- Γ *is a finite alphabet whose elements are called objects.*
- H *is a finite alphabet such that $H \cap \Gamma = \emptyset$ whose elements are called labels.*
- μ *is a labelled rooted tree (called membrane structure) consisting of q nodes injectively labelled by elements of H (the root of μ is labelled by r_μ).*
- $\mathcal{M}_1, \ldots, \mathcal{M}_q$ *are multisets over* Γ.
- $\mathcal{R}$ *is a finite set of rules, of the following forms:*
  - $\star$ $[\, a \rightarrow c\,]_h$ *or* $[\, a\, b \rightarrow c\,]_h$ *, where* $h \in H$, $a, b, c \in \Gamma$ (object evolution rules).
  - $\star$ $a\,[\ \ ]_h \rightarrow [\, b\,]_h$, *where* $h \in H \setminus \{r_\mu\}$, $a, b \in \Gamma$ (send–in communication rules).
  - $\star$ $[\, a\,]_h \rightarrow b\,[\ \ ]_h$, *where* $h \in H$, $a, b \in \Gamma$ (send–out communication rules).
  - $\star$ $[\, a\,]_h \rightarrow [\, b\,]_h\ [\, c\,]_h$, *where* $h \in H \setminus \{i_{out}, r_\mu\}$, $a, b, c \in \Gamma$ *and h is the label of an elementary membrane* μ (division rules for elementary membranes).

– $i_{out} \in H \cup \{env\}$, *where env is the label of the environment (if $i_{out} \in H$ then i_{out} is the label of a leaf of μ).*

The semantics of this kind of P systems follows the usual principles of P systems with active membranes [10]. We denote by $\mathcal{DAM}^0(mcmp, +c, -d, -n)$ the class of all recognizer polarizationless P system with active membranes, without dissolution, with division rules for elementary membranes and which makes use of minimal cooperation and minimal production in objects evolution rules,

Recognizer membrane systems were introduced in [12] and they provide a natural framework to solve decision problems. This kind of systems are characterized by the following features:

– the working alphabet Γ has two distinguished objects `yes` and `no`;

- there exists an input alphabet Σ strictly contained in Γ;
- there exists an input compartment labelled by $i_{in} \in H$;
- $i_{out} = env$, that is, the output "*zone*" is the enviroment;
- the initial contents of the compartments are multisets over $\Gamma \setminus \Sigma$;
- for each multiset m over Σ the initial configuration of Π with input multiset m is $\mathcal{M}_1, \ldots, \mathcal{M}_{i_{in}} + m, \ldots, \mathcal{M}_q$, that is, the input multiset m is added to the contents of the input compartment;
- all computations of the system $\Pi + m$ halt, where $\Pi + m$ denotes the membrane system Π with input multiset m; and
- for each computation of $\Pi + m$, either object `yes` or object `no` (but not both) must have been released into the environment, and only at the last step of the computation.

3. The complexity class DP

The class **DP** was introduced by C.H. Papadimitriou and M. Yannakis in 1982 [6], as the class of "differences" of any two decision problems in **NP**. Roughly speaking, class **DP** tries to capture the complexity of problems that most likely can not be decided by any non-deterministic Turing machine working in polynomial time.

Formally, a language L is in the class **DP** if and only if there are two languages L_1 and L_2 such that $L_1, L_2 \in \mathbf{NP}$ and $L = L_1 \setminus L_2$. Equivalently, a language L is in the class **DP** if and only if there are two languages L_1 and L_2 such that $L_1 \in \mathbf{NP}$, $L_2 \in$ **co-NP**, and $L = L_1 \cap L_2$. It is easy to proof that $\mathbf{NP} \cup$ **co-NP** $\subseteq \mathbf{DP}$. The class **DP** can defined in terms of decision problems as follows: a decision problem X is in class **DP** if and only if there are two decision problems $X_1 = (I_{X_1}, \theta_{X_1})$ and $X_2 = (I_{X_2}, \theta_{X_2})$ such that $X_1 \in \mathbf{NP}$, $X_2 \in$ **co-NP**, and $L_X = L_{X_1} \cap L_{X_2}$.

`UNIQUE SAT` is the problem of deciding whether a Boolean formula given in conjunctive normal form has exactly one satisfying truth assignment. This problem is in class **DP** because `UNIQUE SAT`$= \mathtt{SAT} \cap X$, being X the **co-NP** problem of deciding whether a Boolean formula given in conjunctive normal form has not two different satisfying truth assignment. `UNIQUE SAT` is a **co-NP** hard problem [3]. Indeed, let us recall that `UNSAT` is the problem of deciding whether a given a Boolean formula in conjunctive normal form is unsatisfiable. It is well known that `UNSAT` is a **co-NP**-complete problem. The function f from the set of instances of `UNSAT` to the set of instances of `UNSAT` defined by

$$f(\varphi(x_1, \ldots, x_n)) = [\varphi(x_1, \ldots, x_n) \wedge z] \vee [(\neg z) \wedge x_1 \wedge \cdots \wedge x_n]$$

being z a propositional variable not in the set $\{x_1, \ldots, x_n\}$, is a polynomial-time reduction from `UNSAT` to `UNIQUE SAT`. Then, `UNIQUE SAT` is a **co-NP** hard problem but it is unknown what is its precise complexity. However, `UNIQUE SAT` is a **DP**-complete problem under *randomized polynomial-time reductions* (see [16] for details).

4. A solution to UNIQUE−SAT by membrane systems

In this section a uniform polynomial-time solution to the UNIQUE SAT problem is provided by means of a family of recognizer polarizationless P systems with active membranes, without dissolution rules, with minimal cooperation and minimal production in objects evolution rules which use only division for elementary membranes, that is, by a family of systems $\mathcal{DAM}^0(mcmp, +c, -d, -n)$. For that purpose, the solution to the SAT problem given in [14] and to the #SAT given in [15] are adapted, basically, in what concerns to the output stage.

The family $\mathbf{\Pi} = \{\Pi(t) \mid t \in \mathbb{N}\}$ is defined in such a manner that a system $\Pi(t)$ will process any Boolean formula φ in conjunctive normal form (CNF) with n variables and p clauses (so that $t = \langle n, p\rangle$) provided that the appropriate input multiset $cod(\varphi)$ is supplied to the system (through the corresponding input membrane). Then, system $\Pi(t)$ will answer how many truth assignments make true the input formula φ.

Thus, for each $n, p \in \mathbb{N}$, we consider the recognizer P system

$$\Pi(\langle n, p\rangle) = (\Gamma, \Sigma, H, \mu, \mathcal{M}_1, \mathcal{M}_2, \mathcal{R}, i_{in})$$

from $\mathcal{DAM}^0(mcmp, +c, -d, -n)$, defined as follows:

- Working alphabet

$$\begin{aligned}\Gamma = \quad & \Sigma \cup \{\beta, \natural, \texttt{yes}, \texttt{yes}', \texttt{no}, \gamma_0\} \cup \{\alpha_i \mid 0 \leq i \leq n+2p+1\} \cup \\ & \{\delta_i \mid 0 \leq i \leq n+2p+3\} \cup \{a_{i,j} \mid 0 \leq i \leq n-1,\, 0 \leq j \leq i\} \cup \\ & \{b_{i,k} \mid 1 \leq i \leq n, 1 \leq k \leq i\} \cup \{c_j \mid 1 \leq j \leq p\} \cup \{d_j \mid 2 \leq j \leq p\} \cup \\ & \{t_{i,k}, f_{i,k} \mid 1 \leq i \leq n, i \leq k \leq n+p-1\} \cup \\ & \{T_{i,k}, F_{i,k} \mid 1 \leq i \leq n, 0 \leq k \leq n-1\} \cup \{T_i, F_i \mid 1 \leq i \leq n\} \cup \\ & \{x_{i,j,k}, \overline{x}_{i,j,k}, x^*_{i,j,k} \mid 1 \leq i \leq n, 1 \leq j \leq p, 0 \leq k \leq n+p\}.\end{aligned}$$

- Input alphabet $\Sigma = \{x_{i,j,0}, \overline{x}_{i,j,0}, x^*_{i,j,0} \mid 1 \leq i \leq n, 1 \leq j \leq p\}$.
- $H = \{1, 2\}$.
- Membrane structure: $\mu = [\,[\quad]_2\,]_1$, that is, $\mu = (V, E)$ where $V = \{1, 2\}$ and $E = \{(1, 2)\}$.
- Initial multisets: $\mathcal{M}_1 = \{\,\alpha_0, \delta_0\}$, $\mathcal{M}_2 = \{\,\beta, b_{i,1}, T^p_{i,0}, F^p_{i,0} \mid 1 \leq i \leq n\,\}$.
- The set of rules $\mathcal{R}$ consists of the following rules:

  R.1 Rules for a general counter.

  $[\,\alpha_k \longrightarrow \alpha_{k+1}\,]_1$ for $0 \leq k \leq n+2p$

  $[\,\delta_k \longrightarrow \delta_{k+1}\,]_1$ for $0 \leq k \leq n+2p+2$

  R.2 Rules to generate all truth assignments.

  $[\,b_{i,i}\,]_2 \longrightarrow [\,t_{i,i}\,]_2\,[\,f_{i,i}\,]_2$, for $1 \leq i \leq n$

  $[b_{i,k} \longrightarrow b_{i,k+1}\,]_2$, for $2 \leq i \leq n, 1 \leq k \leq i-1$

R.3 Rules to generate suitable objects in order to start the next stage.

$$\left.\begin{array}{l}[t_{i,k} \longrightarrow t_{i,k+1}\,]_2\\ [f_{i,k} \longrightarrow f_{i,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n-1, i \leq k \leq n-1$$

$$\left.\begin{array}{l}[T_{i,k} \longrightarrow T_{i,k+1}\,]_2\\ [F_{i,k} \longrightarrow F_{i,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n, 0 \leq k \leq n-2$$

$$\left.\begin{array}{l}[T_{i,n-1} \longrightarrow T_i\,]_2\\ [F_{i,n-1} \longrightarrow F_i\,]_2\end{array}\right\} 1 \leq i \leq n$$

R.4 Rules to produce exactly p copies of each truth assignment.

$$\left.\begin{array}{l}[t_{i,k}\, F_i \longrightarrow t_{i,k+1}\,]_2\\ [f_{i,k}\, T_i \longrightarrow f_{i,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n, n \leq k \leq n+p-2$$

$$\left.\begin{array}{l}[t_{i,n+p-1}\, F_i \longrightarrow \natural\,]_2\\ [f_{i,n+p-1}\, T_i \longrightarrow \natural\,]_2\end{array}\right\} 1 \leq i \leq n$$

R.5 Rules to prepare the input formula for check clauses:

$$\left.\begin{array}{l}[\, x_{i,j,k} \longrightarrow x_{i,j,k+1}\,]_2\\ [\, \overline{x}_{i,j,k} \longrightarrow \overline{x}_{i,j,k+1}\,]_2\\ [\, x^*_{i,j,k} \longrightarrow x^*_{i,j,k+1}\,]_2\end{array}\right\} 1 \leq i \leq n,\ 1 \leq j \leq p,\ 0 \leq k \leq n+p-1$$

R.6 Rules for the first checking stage.

$$\left.\begin{array}{l}[T_i\, x_{i,j,n+p} \longrightarrow c_j\,]_2\\ [T_i\, \overline{x}_{i,j,n+p} \longrightarrow \natural\,]_2\\ [T_i\, x^*_{i,j,n+p} \longrightarrow \natural\,]_2\\ [F_i\, x_{i,j,n+p} \longrightarrow \natural\,]_2\\ [F_i\, \overline{x}_{i,j,n+p} \longrightarrow c_j\,]_2\\ [F_i\, x^*_{i,j,n+p} \longrightarrow \natural\,]_2\end{array}\right\} 1 \leq i \leq n, 1 \leq j \leq p$$

R.7 Rules for the second checking stage.

$$[c_1\, c_2 \longrightarrow d_2\,]_2$$

$$[d_j\, c_{j+1} \longrightarrow d_{j+1}\,]_2\,, \text{ for } 2 \leq j \leq p-1$$

R.8 Rules to prepare objects in the skin membrane.

$$[\,\beta\, d_p \longrightarrow \gamma_0\,]_2$$

$$[\,\gamma_0\,]_2 \longrightarrow \gamma_0\,[\,]_2\,, \text{ for } 0 \leq i \leq n-1$$

R.9 Rules to produce the output.

$$[\,\alpha_{n+2p+1}\,\gamma_0 \longrightarrow \texttt{yes}'\,]_1$$

$$[\,\texttt{yes}'\,\gamma_0 \longrightarrow \texttt{no}\,]_1$$

$$[\,\delta_{n+2p+3}\,\texttt{yes}' \longrightarrow \texttt{yes}\,]_1$$

$$[\,\alpha_{n+2p+1}\,\delta_{n+2p+3} \longrightarrow \texttt{no}\,]_1$$

$$[\,\texttt{yes}\,]_1 \longrightarrow \texttt{yes}\,[\quad]_1$$

$$[\,\texttt{no}\,]_1 \longrightarrow \texttt{no}\,[\quad]_1$$

– The input membrane is the membrane labelled by 2 ($i_{in} = 2$) and the output region is the environment.

4.1. An overview of the computation

It is easy to check that for each pair of natural numbers n, p, the system $\Pi(\langle n, p\rangle)$, previously defined, is deterministic.

We consider the polynomial encoding (cod, s) from UNIQUE SAT in $\mathbf{\Pi}$ defined as follows: let φ be a Boolean formula in conjunctive normal form. Let $Var(\varphi) = \{x_1, \ldots, x_n\}$ be the set of propositional variables and $\{C_1, \ldots, C_p\}$ the set of clauses of φ. Let us assume that the number of variables and the number of clauses of the input formula φ, are greater than or equal to 2. Then, we define $s(\varphi) = \langle n, p\rangle$ and

$$cod(\varphi) = \{x_{i,j,0} \mid \ x_i \in C_j\} \ \cup \ \{\overline{x}_{i,j,0} \mid \neg x_i \in C_j\} \ \cup \ \{x^*_{i,j,0} \mid x_i \notin C_j, \neg x_i \notin C_j\}$$

Notice that we can represent this multiset as a matrix, in such a way that the j-th row ($1 \leq j \leq p$) encodes the j-th clause C_j of φ, and the columns ($1 \leq i \leq n$) are associated with variables. We denote by $cod_k(\varphi)$ the multiset $cod(\varphi)$ with the third index of all objects $x, \overline{x}, x^*$ being k.

The Boolean formula φ will be processed by the system $\Pi(s(\varphi))$ with input multiset $cod(\varphi)$. Next, we informally describe how that system works.

The solution proposed consists of the following stages:

- **Generation stage**: by applying division rules from **R.2**, all truth assignments for the variables $\{x_1, \ldots, x_n\}$ associated with φ are produced. This stage takes exactly n computation steps and at the i-th step, $1 \leq i \leq n$, of this stage, division rule is triggered by an object $b_{i,i}$ over each membrane labelled by 2, producing two new membranes with all its remaining contents replicated in the new membranes labelled by 2. Simultaneously to these divisions, objects $t_{i,k}, f_{i,k}, T_{i,k}, F_{i,k}$ (by applying rules from **R.3**) and objects $x_{i,j,k}, \overline{x}_{i,j,k}, x*_{i,j,k}$ (by applying rules from **R.5**) evolve during this stage in such a manner that at configuration $\mathcal{C}_n$ the following holds:

  (a) There is a membrane labelled by 1 which contains n copies of object α_n.

  (b) There are 2^n membranes labelled by 2 such that each of them contains: a copy of object β, the set $cod_n(\varphi)$, the multiset $\{T^p_i, F^p_i \mid 1 \leq i \leq n\}$; and a different subset $\{r_{1,n}, \ldots, r_{n,n}\}$, being $r \in \{t, f\}$.

- **Production of enough copies for each truth assignment**: in this stage p copies (p is the number of clauses of φ) of each truth assignment are produced, in order to allow the checking of the literal associated with each variable in each clause. By using minimal cooperation and minimal production (applying rules from **R.4**), objects $t_{i,k}$ (respectively, objects $f_{i,k}$) are used to remove all copies of F_i (respectively, T_i). This stage takes exactly p steps, and at configuration $\mathcal{C}_{n+p}$ the following holds:

  (a) The root membrane (labelled by 1) contains n copies of object α_{n+p}.

  (b) There are 2^n membranes labelled by 2 such that each of them contains: a copy of object β, n copies of the garbage object $\natural$, the set $cod_{n+p}(\varphi)$, and a different multiset $\{R^p_1, \ldots, R^p_n\}$, being $R \in \{T, F\}$.

- **First Checking stage**: by applying rules from **R.6**, we check whether or not each clause of the input formula φ is satisfied by the truth assignments generated in the previous stage, encoded by each membrane labelled by 2. This stage takes exactly one computation step, and at configuration $\mathcal{C}_{n+p+1}$:

(a) The root membrane (labelled by 1) contains n copies of object α_{n+p+1}.

(b) There are 2^n membranes labelled by 2 such that each of them contains: a copy of object β, many copies of the garbage object $\natural$ (which they will not evolve in the rest of the computation), and copies of objects c_j whose presence means that clause C_j is true for the truth assignment encoded by that membrane.

- **Second Checking stage**: by applying rules from **R.7**, we check whether or not all clauses of the input formula φ are satisfied by some truth assignment encoded by a membrane labelled by 2. This stage takes exactly $p-1$ steps and at configuration $\mathcal{C}_{n+2p}$ the following holds:

  (a) The root membrane (labelled by 1) contains n copies of object α_{n+2p}.

  (b) There are 2^n membranes labelled by 2 such that each of them contains: a copy of object β, many copies of the garbage object $\natural$ (which they will not evolve in the rest of the computation), and copies of objects d_j and c_j, in such manner that the truth assignment encoded by such membrane makes true φ if and only if contains some object d_p.

- **Output stage.**

  Affirmative answer. If there exists exactly one input assignment that makes true the input formula, at configuration $\mathcal{C}_{n+2p+1}$ there will be exactly one object γ_0 at the skin membrane. Then, by applying the first rule from **R.9**, an object $\texttt{yes}'$ will be produced, while δ_{n+2p+1} evolves into δ_{n+2p+2}. In the next step, only object δ_{n+2p+2} can evolve into δ_{n+2p+3}, since there are no objects that can interact with object $\texttt{yes}'$. Finally, rule $[\, \delta_{n+2p+3}\, \texttt{yes}' \longrightarrow \texttt{yes}\,]_1$ will be fired, and will produce an object $\texttt{yes}$ in the skin membrane, and it will be released to the environment in the last step of the computation. It takes 4 computation steps.

  Negative answer. There are two possibilities:

  ⋆ Case 1: No truth assignment exists making the input formula true. In this case, object γ_0 will not appear in the skin membrane at configuration $\mathcal{C}_{n+2p+1}$. Then, in the next two steps, object δ_{n+2p+1} will change to δ_{n+2p+2} and δ_{n+2p+3}. These two objects will interact by means of the rule $[\, \alpha_{n+2p+1}\, \delta_{n+2p+3} \longrightarrow \texttt{no}\,]_1$, and object $\texttt{no}$ just created will be sent out to the environment and the system will halt, taking 4 steps of computation.

  ⋆ Case 2: More than one truth assignment make this formula true. In this case, more than one object γ_0 will appear in the membrane labelled by 1 at configuration $\mathcal{C}_{n+2p+1}$. As in the affirmative answer case, objects α_{n+2p+1} and γ_0, selected non-deterministically from the existent ones, will create an object $\texttt{yes}'$. Besides this, object δ_{n+2p+1} evolves into δ_{n+2p+2}. But in the next step, by the presence of another object γ_0, rule $[\, \texttt{yes}'\, \gamma_0 \longrightarrow \texttt{no}\,]_1$ will be fired, and will produce an object $\texttt{no}$. This object will not interact with the recently evolved object δ_{n+2p+3}, that will be released into the environment in the last step on the computation, and then the system halts. This case takes 3 steps.

5. Main result

In this section, the most important result of this work is presented; that is: the family of polarizationless P systems from $\mathcal{DAM}^0(mcmp, +c, -d, -n)$, designed in the previous section, solves the UNIQUE SAT problem in a uniform way in polynomial time.

Theorem 1. UNIQUE SAT $\in \mathbf{PMC}_{\mathcal{DAM}^0(mcmp,+c,-d,-n)}$.

Proof. The family $\mathbf{\Pi} = \{\Pi(t) \mid t \in \mathbb{N}\}$ defined in the previous section verifies the following:

(a) Every system $\Pi(t)$ of the family $\mathbf{\Pi}$ belongs to $\mathcal{DAM}^0(mcmp, +c, -d, -n)$.

(b) The family $\mathbf{\Pi}$ is polynomially uniform by Turing machines because for each $n, p \in \mathbb{N}$, the amount of resources needed to build the system $\Pi(\langle n, p\rangle)$ is of a polynomial order in n and p. Indeed,

- The size of the alphabet is of the order $O(n^2 \cdot p^2)$.
- The initial number of membranes is $2 \in \Theta(1)$.
- The initial number of objects in membranes is $2np + 2n + 1 \in \Theta(n \cdot p)$.
- The number of rules is of the order $O(n^2 \cdot p^2)$.
- The maximal number of objects involved in any rule is $3 \in \Theta(1)$.

(c) The pair (cod, s) of polynomial-time computable functions defined fulfills the following: for each input formula φ of the UNIQUE SAT problem, $s(\varphi)$ is a natural number, $cod(\varphi)$ is an input multiset of the system $\Pi(s(\varphi))$, and for each $t \in \mathbb{N}$, $s^{-1}(t)$ is a finite set.

(d) The family $\mathbf{\Pi}$ is polynomially bounded. Indeed, for each input formula φ of the UNIQUE SAT problem, the number of computation steps of the system $\Pi(s(\varphi)) + cod(\varphi)$ is the following: (a) at most $n + 2p + 4$ in the case of the input formula has not or has exactly one truth satisfying truth assignment; and (b) $n + 2p + 3$ in the case that there are more than one truth assignments making true φ.

(e) The family $\mathbf{\Pi}$ is sound and complete with regard to (X, cod, s). This can be informally deduced from the overview of the computations previously described.

Therefore, the family $\mathbf{\Pi}$ of P systems designed in the previous section solves the UNIQUE SAT problem in a uniform way in polynomial time.

□

6. Conclusions

Polarizationless P systems with active membranes and no dissolution rules are *non efficient* computing models, in the sense that only problems in class **P** can be solved in an efficient way by these P systems. In this paper, a uniform and polynomial time solution to the UNIQUE SAT problem is provided by means of a family of polarizationless P systems when *minimal cooperation* (the length of the left-hand side of a rule is at most 2) and *minimal production* (the length of the right-hand side of a rule is 1) are considered in object evolution rules.

US is the class of problems that have a unique solution for each instance. It is easy to prove that UNIQUE SAT is complete for this complexity class [3]. A very interesting open problem

in the framework of computational complexity theory is to know if a problem from **NP** can be reduced in polynomial time to UNIQUE SAT, demonstrating not only that **NP** is a subclass of **US**, but that $\mathbf{DP} = \mathbf{US}$. Using the framework of *Membrane Computing* can be useful as we can extract some ideas comparing the solutions developed for SAT (in [14]) and UNIQUE SAT (in this work) in the same framework, that is, $\mathcal{DAM}^0(mcmp, +c, -d, -n)$.

Acknowledgements. This work was partially supported by Project TIN2017-89842-P of the *Ministerio de Economía y Competitividad* of Spain and by Grant number 61320106005 of the *National Natural Science Foundation of China*.

References

[1] A. Alhazov, L. Pan. Polarizationless P systems with active membranes. *Grammars*, **7** (2004), 141-159.

[2] A. ALHAZOV, L. PAN, Gh. PĂUN, *Trading polarizations for labels in P systems with active membranes*, Acta Informaticae, **41**(2-3, 111–144, 2004.

[3] A. BLASS, Y. GUREVICH, *On the Unique Satisfiability Problem*, Information and Control, **55**, 80–88, 1982

[4] T. H. CORMEN, C. E. LEISERSON, R. L. RIVEST, *An Introduction to Algorithms*. The MIT Press, Cambridge, Massachussets, 1994.

[5] M. A. GUTIÉRREZ, M. J. PÉREZ-JIMÉNEZ, A. RISCOS-NÚÑEZ, F. J. ROMERO, *On the power of dissolution in P systems with active membranes*, Lecture Notes in Computer Science, **3850**, 224–240, 2006.

[6] C. H. PAPADIMITRIOU, M. YANNAKIS, *The complexity of facets (and some facets of complexity)*, Proceedings of the 24th ACM Symposium on the Theory of Computing, pp. 229–234, 1982.

[7] C. H. PAPADIMITRIOU, *Computational complexity*, Addison-Wesley Publishing Company, USA 1994.

[8] Gh. PĂUN, *Computing with membranes*, Journal of Computer and Systems Science, **61**(1), 108-143, 2000.

[9] Gh. PĂUN, *Membrane Computing. An introduction*, Springer-Verlag, Berlin, 2002.

[10] Gh. PĂUN, *P systems with active membranes: attacking* **NP***–complete problems*, Journal of Automata, Languages and Combinatorics, **6**, 75–90, 2001.

[11] Gh. PĂUN, G. ROZENBERG, A. SALOMAA (eds.). *The Oxford Handbook of Membrane Computing*, Oxford University Press, Oxford, 2010.

[12] M. J. PÉREZ-JIMÉNEZ, Á. ROMERO-JIMÉNEZ, F. SANCHO-CAPARRINI, *Complexity classes in models of cellular computing with membranes*, Natural Computing, **2**(3), 265–285, 2003.

[13] P. SOSíK, A. RODRíGUEZ–PATÓN, *P systems with active membranes characterize PSPACE*, In Ch. Mao, T. Yokomori (eds.), Lecture Notes in Computer Science, **4287**, 33–46, 2006.

[14] L. VALENCIA-CABRERA, D. ORELLANA-MARTíN, M. Á. MARTíNEZ-DEL-AMOR, A. RISCOS-NÚÑEZ, M. J. PÉREZ-JIMÉNEZ, *Reaching efficiency through collaboration in membrane systems: Dissolution, polarization and cooperation*, Theoretical Computer Science, **701**, 226–234, 2017.

[15] L. VALENCIA-CABRERA, D. ORELLANA-MARTíN, A. RISCOS-NÚÑEZ, M. J. PÉREZ-JIMÉNEZ, *Counting Membrane Systems*, Lecture Notes in Computer Science, **10725**, 74–87, 2017.

[16] L. G. VALIANT, V. V. VAZIRANI, **NP** *is as easy as detecting unique solutions*, Theoretical Computer Science, **47**, 85–93, 1986.