

Counting P systems

David Orellana-Martín

Research Group on Natural Computing
Dept. of Computer Science and Artificial Intelligence
Universidad de Sevilla

Seville, Spain, January 31, 2017



- Membrane systems have been used to solve a large number of problems.

Introduction

- Membrane systems have been used to solve a large number of problems.
- Depending on their own essence, we can talk about “**easier**” or “**harder**” problems.

Introduction

- Membrane systems have been used to solve a large number of problems.
- Depending on their own essence, we can talk about “**easier**” or “**harder**” problems.
- We can switch syntactic (and semantic) ingredients...

Introduction

- Membrane systems have been used to solve a large number of problems.
- Depending on their own essence, we can talk about “**easier**” or “**harder**” problems.
- We can switch syntactic (and semantic) ingredients...
- ... in order to solve different sets of problems.

- Several kinds of membrane systems have been created to solve problems of different natures.

- Several kinds of membrane systems have been created to solve problems of different natures.
 - Recognizer P systems

- Several kinds of membrane systems have been created to solve problems of different natures.
 - Recognizer P systems
 - Computing P systems

Introduction

- Several kinds of membrane systems have been created to solve problems of different natures.
 - Recognizer P systems
 - Computing P systems
 - Generating P systems

- Several kinds of membrane systems have been created to solve problems of different natures.
 - Recognizer P systems
 - Computing P systems
 - Generating P systems
 - Function P systems

Complexity classes

- Classic computational complexity classes have been characterized in the framework of Membrane Computing.

¹L. G. Valiant, The Complexity of Computing the Permanent, *Theoretical Computer Science* 8 (1979), 189–201

Complexity classes

- Classic computational complexity classes have been characterized in the framework of Membrane Computing.
- **P**

¹L. G. Valiant, The Complexity of Computing the Permanent, *Theoretical Computer Science* 8 (1979), 189–201

Complexity classes

- Classic computational complexity classes have been characterized in the framework of Membrane Computing.
- **P, NP**

¹L. G. Valiant, The Complexity of Computing the Permanent, *Theoretical Computer Science* 8 (1979), 189–201

Complexity classes

- Classic computational complexity classes have been characterized in the framework of Membrane Computing.
- **P, NP, PP**

¹L. G. Valiant, The Complexity of Computing the Permanent, *Theoretical Computer Science* 8 (1979), 189–201

Complexity classes

- Classic computational complexity classes have been characterized in the framework of Membrane Computing.
- **P, NP, PP, PSPACE...**
- In **1979**, L. G. Valiant introduced a new complexity class ¹.

¹L. G. Valiant, The Complexity of Computing the Permanent, *Theoretical Computer Science* 8 (1979), 189–201

Complexity classes

- Classic computational complexity classes have been characterized in the framework of Membrane Computing.
- **P, NP, PP, PSPACE...**
- In **1979**, L. G. Valiant introduced a new complexity class ¹.
- Here, we have decision problems, but the answer of the system must not be yes or no...
- ... but the **number of positive computations** we obtain!

¹L. G. Valiant, The Complexity of Computing the Permanent, *Theoretical Computer Science* 8 (1979), 189–201

Counting Turing machines

- Standard non-deterministic Turing Machine with an auxiliary output device that prints in **binary notation** on a special tape the number of accepting computations induced by the input.

Counting Turing machines

- Standard non-deterministic Turing Machine with an auxiliary output device that prints in **binary notation** on a special tape the number of accepting computations induced by the input.
- A counting Turing machine can solve a special kind of problems: **counting** problems.
- The complexity class of these problems is called **#P**.

Counting Turing machines

- Standard non-deterministic Turing Machine with an auxiliary output device that prints in **binary notation** on a special tape the number of accepting computations induced by the input.
- A counting Turing machine can solve a special kind of problems: **counting** problems.
- The complexity class of these problems is called **#P**.
- An example of this kind of problems is **#SAT**.

Counting membrane systems

- A new type of membrane systems is created in order to solve this kind of problems.

Counting membrane systems

- A new type of membrane systems is created in order to solve this kind of problems.
- A **counting membrane system** of degree q is a membrane system

$$\Pi = (\Gamma, \Sigma, C, \mathcal{M}_1, \dots, \mathcal{M}_1, \mathcal{R}, i_{in}, i_{out})$$

where C is a special alphabet called *output alphabet*, that will represent the answer to the problem in binary representation.

Counting membrane systems

- A new type of membrane systems is created in order to solve this kind of problems.
- A **counting membrane system** of degree q is a membrane system

$$\Pi = (\Gamma, \Sigma, C, \mathcal{M}_1, \dots, \mathcal{M}_1, \mathcal{R}, i_{in}, i_{out})$$

where C is a special alphabet called *output alphabet*, that will represent the answer to the problem in binary representation.

- We will denote **PMC_C** the class of counting membrane systems.

Theorem

$$\#\text{SAT} \subseteq \text{PMC}_{\mathcal{DAM}_c^0}(mcmp, +c, -d, -n)$$

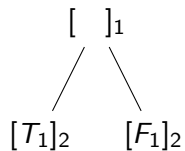
- Based on the solution given in ²...

²L. Valencia-Cabrera, D. Orellana-Martín, A. Riscos-Núñez, M. J. Pérez Jiménez. Reaching efficiency through complicity in membrane systems: dissolution, polarization and cooperation, *Theoretical Computer Science*, submitted 2016

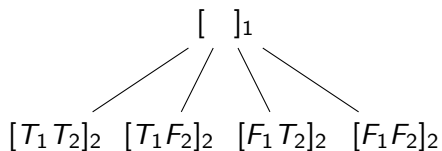
- Generation stage

$$\begin{array}{c} [\quad]_1 \\ | \\ [\quad]_2 \end{array}$$

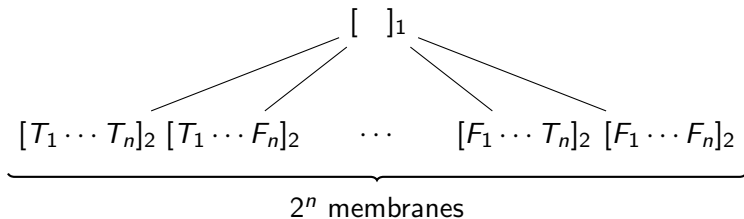
- Generation stage



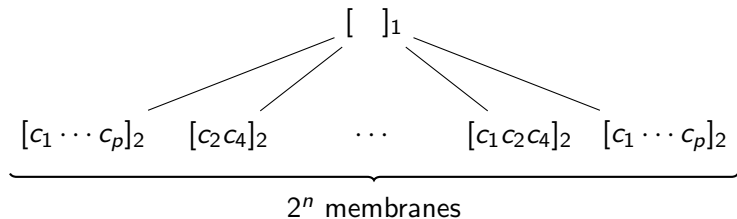
- Generation stage



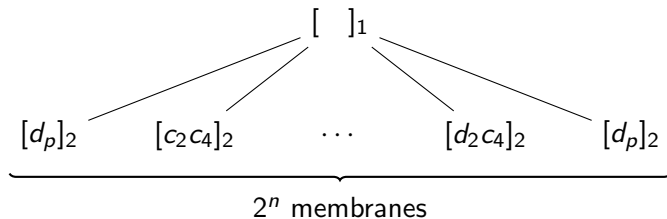
- Generation stage



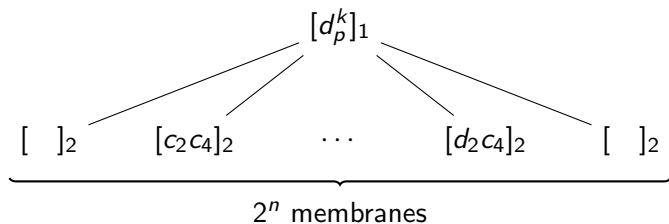
- First checking stage



- Second checking stage



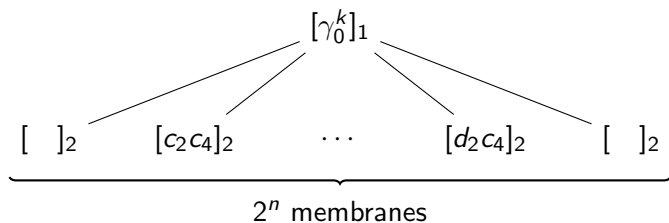
- Second checking stage



$k \equiv$ the number of truth assignments that make true the input formula

- At this point, we would send out:
 - yes if at least an object d_p is in membrane 1.
 - no if there are no objects d_p in membrane 1.
- We change d_p by α_0 .

- Second checking stage



$k \equiv$ the number of truth assignments that make true the input formula

First results

- And we proceed with the next rules:

$$[\gamma_i \gamma_i \rightarrow \gamma_{i+1}]_1$$

- In n steps we will have in membrane 1:

$$\gamma_k \cdots \gamma_0 \rightarrow \alpha_k \cdots \alpha_0, \alpha_i \in \mathcal{C}$$

- We send them out **in polynomial time** with respect of the input.

Corollary

$$\#P \subseteq \text{PMC}_{\text{DAM}_C^0}(mcmp, +c, -d, -n)$$

- If we can solve problems from **NP** in \mathcal{R} then we can solve their counting counterparts from **#P** in \mathcal{R}_c ...

Future work

- If we can solve problems from **NP** in \mathcal{R} then we can solve their counting counterparts from **#P** in \mathcal{R}_c ...
- ... but what happens with non-efficient membrane systems?

Future work

- If we can solve problems from **NP** in \mathcal{R} then we can solve their counting counterparts from **#P** in \mathcal{R}_c ...
- ... but what happens with non-efficient membrane systems?
- `PerfectMatching` \in **P**...

- If we can solve problems from **NP** in \mathcal{R} then we can solve their counting counterparts from **#P** in \mathcal{R}_c ...
- ... but what happens with non-efficient membrane systems?
- $\text{PerfectMatching} \in \mathbf{P}$...
- ... $\#\text{PerfectMatching} \in \#\mathbf{P}$ (Permanent)
- $\mathbf{P} = \text{PMC}_{SAM^0(mcmp,+c,-d,-n)} \dots$

Future work

- If we can solve problems from **NP** in \mathcal{R} then we can solve their counting counterparts from **#P** in \mathcal{R}_C ...
- ... but what happens with non-efficient membrane systems?
- $\text{PerfectMatching} \in \mathbf{P}$...
- ... $\#\text{PerfectMatching} \in \#\mathbf{P}$ (Permanent)
- $\mathbf{P} = \text{PMC}_{\text{SAM}^0(\text{mcmp}, +c, -d, -n)} \dots$
- ... $\#\mathbf{P} \subseteq \text{PMC}_{\text{SAM}_C^0(\text{mcmp}, +c, -d, -n)}$?