Multiset Rewriting and P Systems

Computational Models for Complex Systems

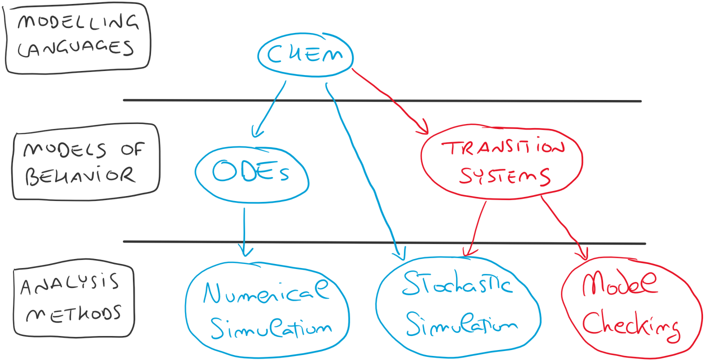
Paolo Milazzo

Dipartimento di Informatica, Università di Pisa
http://pages.di.unipi.it/milazzo
milazzo@di.unipi.it

Laurea Magistrale in Informatica
A.Y. 2019/2020

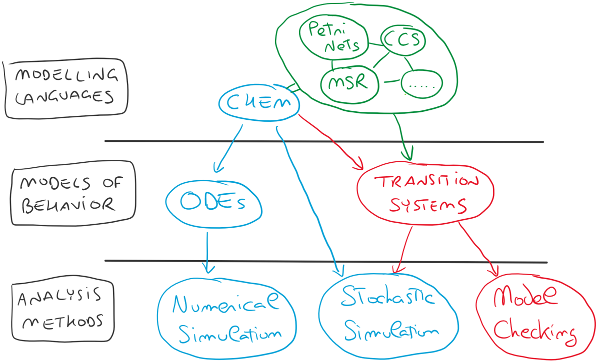
Roadmap

Where do we are?



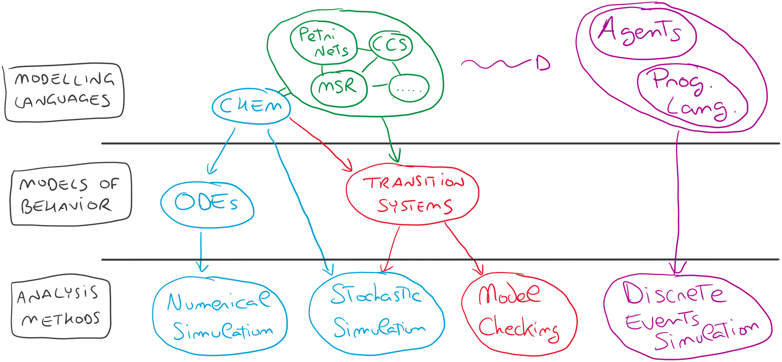
Roadmap

Next step:



Roadmap

Then...



Chemical Reactions in PRISM input language

We have seen that chemical reactions can be expressed in terms of the input language of the PRISM model checking tool

Lotka/Volterra reactions:

$$\begin{cases} \dot{V} = 10V - 0.1VP \\ \dot{P} = -10P + 0.1VP \end{cases}$$

$$V \xrightarrow{10} 2V$$
$$V + P \xrightarrow{0.1} 2P$$
$$P \xrightarrow{10}$$

```
ctmc

const double k1 = 10;
const double k2 = 0.1;
const double k3 = 10;

const MAX = 1000;

module lotka

  v : [0..MAX] init 100;
  p : [0..MAX] init 100;

 [] v>0 & v<MAX -> k1*v : (v'=v+1);
 [] v>0 & p>0 & p<MAX
        -> k2*v*p : (v'=v-1) & (p'=p+1);
 [] p>0 -> k3*p : (p'=p-1);

endmodule
```

Chemical Reactions in PRISM input language

The PRISM input language is a way to specify the Transition System (DTMC or CTMC) describing the behavior of the chemical reactions.

However:

- it is specific for the PRISM tool

It would be better to have a general specification of the Transition System in order not to be restricted to the PRISM tool for its analysis

Chemical solutions as Multisets

A formal specification of chemical reactions and of their behavior can be given in terms of MultiSet Rewriting (MSR)

A multiset is a variant of the mathematical notion of set in which elements can be repeated (more than one occurrence can be present)

- for example: $\{A, A, A, B, B, C, C, C\}$

A chemical solution can be seen as a multiset of symbols representing molecules

Representing Multisets

Given a support set Σ, the mathematical representation of a multiset M over Σ is usually given

- as a set of pairs in $M \subseteq \Sigma \times \mathbb{N}$
- or as a mapping $M : \Sigma \to \mathbb{N}$

Given $\Sigma = A, B, C$, the multiset $M = \{A, A, A, B, B, C, C, C\}$ over Σ is usually represented

- either as $M = \{(A, 3), (B, 2), (C, 3)\}$
- or as $M(A) = 3, M(B) = 2, M(C) = 3$

These representations actually correspond to the representation of chemical solutions we considered in PRISM

- one non-negative integer variable for each molecule
- `A=3; B=2; C=3;`

Representing Multisets as strings

Another possibile representation for multisets (inspired by formal language theory) is as strings

In this case the support set Σ is considered as an alphabet, and multisets correspond to strings over such an alphabet

Given $\Sigma = \{A, B, C\}$ the multiset $M = \{(A, 3), (B, 2), (C, 3)\}$ can be represented as the string

- $M = AAABBCCC$

In a string representing a multiset, the order of the symbols does not matter, so string permutations result in equivalent representations

- $M = AAABBCCC = ABCABCAC = CCCBBAAA = \ldots$

Representing Multisets as strings

Some notes about the string representation:

Given a support set/alphabet Σ, the set of all possible multisets

- represented as set of pairs is $\Sigma \times \mathbb{N}$,
- represented as string is Σ^* (the Kleene closure of the alphabet) that is $\bigcup_{i \in \mathbb{N}} \Sigma^i$, where Σ^i is the set of all strings of length i.

A usual shorthand for multisets represented as string is based on the use of exponents:

- $AAABBCCC = A^3 B^2 C^3$

Multiset union can be expressed as string concatenation:

- $\{(A,3),(B,2),(C,3)\} \cup \{(A,2),(B,1)\} = \{(A,5),(B,3),(C,3)\}$
- $AAABBCCC \cup AAB = AAABBCCCAAB$

Representing reactions as rewriting rules

The idea for the formalization of chemical reactions is to consider a set of reactions essentially as a formal grammar, or better as a set of (multiset) rewriting rules

Definition: Multiset rewriting rule

A multiset rewriting rule is a pair (u, v) with $u, v \in \Sigma^*$, usually denoted $u \mapsto w$

A multiset rewriting rule can be applied to a multiset $w \in \Sigma^*$ such that $u \subseteq w$, obtaining as result the multiset in which u has been replaced by v, namely:

- the application of $u \mapsto v$ to w gives $(w \setminus u) \cup v$
- for example, the application of $AB \mapsto C$ to $A^3B^2C^3$ gives A^2BC^4

MultiSet Rewriting (MSR)

Definition: MultiSet Rewriting (MSR)

A MultiSet Rewriting system is a pair $S = \langle \Sigma, \mathcal{R} \rangle$ where Σ is an alphabet of symbols and $\mathcal{R}$ is a set of multiset rewriting rules

For example: $S = \langle \{A, B, C\}, \{AB \mapsto C, C \mapsto AB\} \rangle$

Now, given a multiset in Σ^*, we can use the mechanism of rewriting rule application to compute traces of the multiset rewriting system.

$A^3B^2C^3 \rightarrow A^2B^1C^4 \rightarrow AC^5 \rightarrow A^2BC^4 \rightarrow \ldots$

The dynamics of MSR: Interleaving semantics

The behavior of a MSR system can also be described as a Transition System

- we can define an (interleaving) semantics for MSR defining inference rules incorporating the mechanism of rewriting rule application

Definition: Interleaving semantics of MSR

The interleaving semantics of a MSR system $\langle \Sigma, \mathcal{R} \rangle$ is the Transition System $(\Sigma^*, \rightarrow)$ where $\rightarrow \subseteq \Sigma^* \times \Sigma^*$ is the least transition relation satisfying the following inference rule:

$$\frac{u \mapsto v \in \mathcal{R}}{uw \rightarrow vw}$$

Stochastic MSR: Syntax and semantics

Stochastic rates can be incorporated in MSR

- This requires to extend both the syntax and the semantics of MSR

Definition: Stochastic multiset rewriting rule

Given an alphabet Σ, a stochastic multiset rewriting rule is a tuple (u, v, r) where $u, v \in \Sigma^*$ and $r \in \mathbb{R}^+$, usually denoted $u \stackrel{r}{\mapsto} v$

Definition: Stochastic MSR

A Stochastic MSR system is a pair $S = \langle \Sigma, \mathcal{R} \rangle$ where Σ is an alphabet of symbols and $\mathcal{R}$ is a set of stochastic multiset rewriting rules

Stochastic MSR: Syntax and semantics

Definition: Semantics of Stochastic MSR

The semantics of stochastic MSR system is the Continuous Time Markov Chain $(\Sigma^*, \rightarrow)$ where $\rightarrow \subseteq \Sigma^* \times \mathbb{R}^+ \times \Sigma^*$ is the least stochastic transition relation satisfying the following inference rule:

$$\frac{u \overset{r}{\mapsto} v \in \mathcal{R}}{uw \xrightarrow{r \cdot f(u,uw)} vw}$$

where $f(u, uw)$ gives the number of instances of u in uw.

MSR: Altenative semantics

Now that we have a language, we can define variants...

In particular, we can consider forms of parallelism in the application of rewriting rules:

- simple parallelism: one or more rewrite rules are applied at each step
- maximal parallelism: as many rules as possibile are applied at each step

These forms of parallelism can be more suitable to model other kinds of system

MSR: Parallel and maxiamally parallel semantics

Definition: Parallel semantics of MSR

The parallel semantics of a MSR system $\langle \Sigma, \mathcal{R} \rangle$ is the Transition System $(\Sigma^*, \rightarrow)$ where $\rightarrow \subseteq \Sigma^* \times \Sigma^*$ is the least transition relation satisfying the following inference rules:

$$\frac{u \mapsto v \in \mathcal{R}}{u \rightarrow v} \qquad \frac{w \rightarrow w'}{wu \rightarrow w'u} \qquad \frac{w_1 \rightarrow w_1' \qquad w_2 \rightarrow w_2'}{w_1 w_2 \rightarrow w_1' w_2'}$$

Definition: Maximally parallel semantics of MSR

The maximally parallel semantics of a MSR system $\langle \Sigma, \mathcal{R} \rangle$ is the Transition System $(\Sigma^*, \Rightarrow)$ where $\Rightarrow \subseteq \Sigma^* \times \Sigma^*$ is the least transition relation satisfying the following inference rules (with $\rightarrow \subseteq \Sigma^* \times \Sigma^*$ auxiliary transition relation):

$$\frac{u \mapsto v \in \mathcal{R}}{u \rightarrow v} \qquad \frac{w_1 \rightarrow w_1' \qquad w_2 \rightarrow w_2'}{w_1 w_2 \rightarrow w_1' w_2'} \qquad \frac{w \rightarrow w' \qquad u \not\rightarrow}{wu \Rightarrow w'u}$$

Multiset languages

What happens if we ignore the sequential ordering of symbols in the words of a language?

A language becomes a set of multisets of terminal symbols:

- it is called multiset language
- it is generated by a multiset grammar
- Example: $\{\varnothing, \{a, b\}, \{a, a, b, b\}, \dots\} = a^n b^n$

Ignoring the ordering of symbols has a high cost in terms of expressiveness

- Context free multiset languages = Regular multiset languages
  - e.g. $a^n b^n = (ab)^n$
- Multiset languages given by general grammars can be accepted by an automaton that is weaker than Turing machines

Multiset languages and maximal parallelism

The maximal parallelism of P systems, used in the context of multiset languages, gives more expressive power to grammars

- general multiset grammar rules applied with maximal parallelism are again able to generate any recursively enumerable language
- a Turing-complete form of automaton is necessary to accept such languages

P Systems

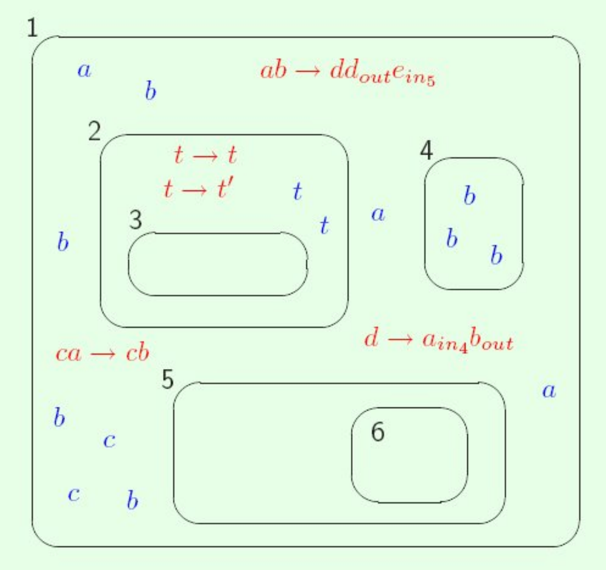
Membrane Systems (or P Systems) are the main class of computing models studied in Membrane Computing

P Systems are distributed computing devices inspired by the structure and the functioning of a living cells.

The key elements of P Systems are:

- Membranes (that create compartments used to distribute computations)
- Multisets (abstractions of chemical solutions that are used as data)
- Evolution (rewriting) rules (abstractions of chemical reactions that are used as programs)

An Example of P System



Formal definition of P Systems

A *P System* Π is given by

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

where:

- V is an *alphabet* whose elements are called *objects*;
- $\mu \subset \mathbb{N} \times \mathbb{N}$ is a *membrane structure*, such that $(i, j) \in \mu$ denotes that the membrane labeled by j is contained in the membrane labeled by i;
- w_i with $1 \leq i \leq n$ are strings from V^* representing multisets over V associated with the membranes $1, 2, \ldots, n$ of μ;
- R_i with $1 \leq i \leq n$ are finite sets of *evolution rules* associated with the membranes $1, 2, \ldots, n$ of μ.

Evolution rules

An evolution rule $u \to v$ consists of a multiset of objects u (representing reactants) and a multiset of messages v (representing products). A message may have one of the following forms:

- a_{here}, meaning that object a remains in the same membrane;
- a_{out}, meaning that object a is sent out of the membrane;
- a_{in_l}, meaning that object a is sent into the child membrane l.

The subscript *here* is often omitted.

Evolution rules can be classified into:

- non-cooperative rules: the left-hand side consists of a single object (e.g. $a \to b^2 d_{out}$)
- cooperative rules: the left-hand side can be any multiset of objects (e.g. $a^2 b \to b^2 d_{out}$)
  - a particular case of cooperative rules are catalytic rules, namely rules of the form $ca \to cb^2$ where c belongs to a special set of objects called catalysts.

Maximal parallelism

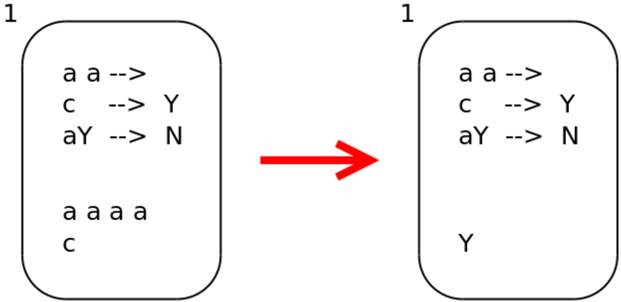
Evolution rules are applied with maximal parallelism:

- More than one rule can be applied (on different objects) in the same computation step
- Each rule can be applied more than once in the same step (on different objects)
- Maximality means that:

  > *A multiset of instances of evolution rules is chosen non–deterministically such that no other rule can be applied to the system obtained by removing all the objects necessary to apply the chosen instances of rules.*

A Simple Example (1)

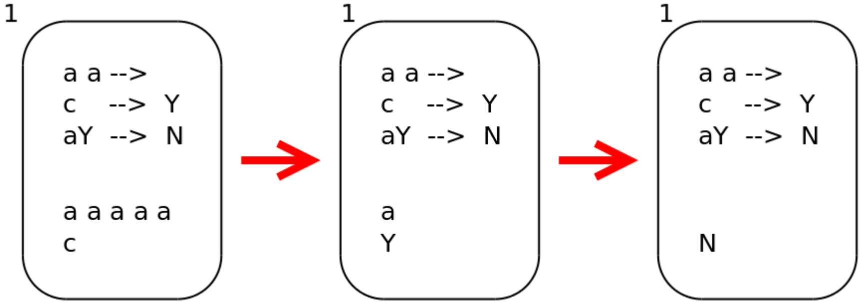
A P System testing whether *n* is even or odd.



INPUT: *n* copies of *a*
OUTPUT: *Y* if *n* is even, *N* otherwise

A Simple Example (2)

A P System testing whether *n* is even or odd.



INPUT: *n* copies of *a*
OUTPUT: *Y* if *n* is even, *N* otherwise

Variants of P Systems

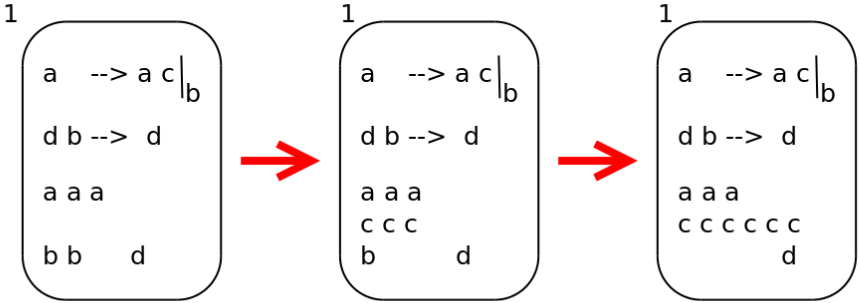
Programming P Systems (for non trivial examples) is very difficult since evolution rules are very basic.

Variants of P Systems obtained by considering different types of evolution rules:

- with rule priorities;
- with promoters and inhibitors;
- with dissolution of membranes;
- symport/antiport rules;
- with active membranes;
- ....

Further Examples of P Systems (1)

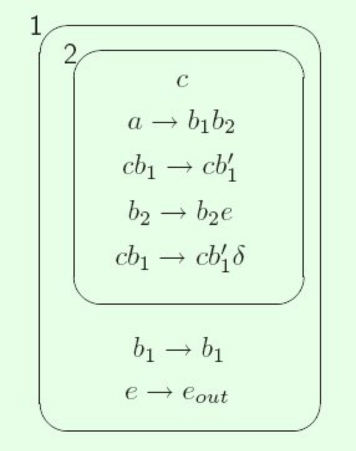
A P System computing $n \times m$ (with a promoter)



INPUT: n copies of a and m copies of b
OUTPUT: $n \times m$ copies of c

Further Examples of P Systems (1)

A P System computing n^2 (with a dissolving rule)



INPUT: n copies of a in membrane 2

OUTPUT: n^2 copies of e sent out of membrane 1

Turing Completeness

P Systems are Turing-complete (or Universal)

- i.e. any Turing machine can be translated into an equivalent P System
- i.e. can be used to compute any computable function

P Systems are massively parallel computational devices. Sometimes such a parallelism can lead to very efficient computations

- e.g. checking whether a string contains as many instances of “a” as of “b” can be done in just two steps (slight variation of the first example)

Actually, suitable variants of P Systems can solve NP-complete problems in polynomial time...

P Systems with active membranes

Active membrane means that evolution rules can change the membrane structure of a P System

- In particular, there can be membrane division rules
- $[u]_i \rightarrow [v]_j[z]_k$

It has been proved that with membrane division it is possible to solve NP-complete problems in polynomial time.

Roughly, the idea is the following:

- Every possible solution is encoded inside a different membrane in a linear number of steps by means of membrane division rules
- Each membrane checks whether the solution it contains is correct (in polynomial time, by def.)

P Systems as Models of Biological Systems

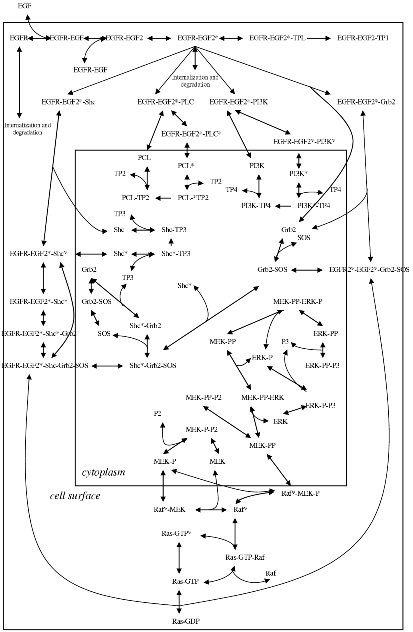
P Systems have a simple notation that capture the essential elements of cellular processes:

- (bio)chemical reactions (i.e. evolution rules)
- membranes

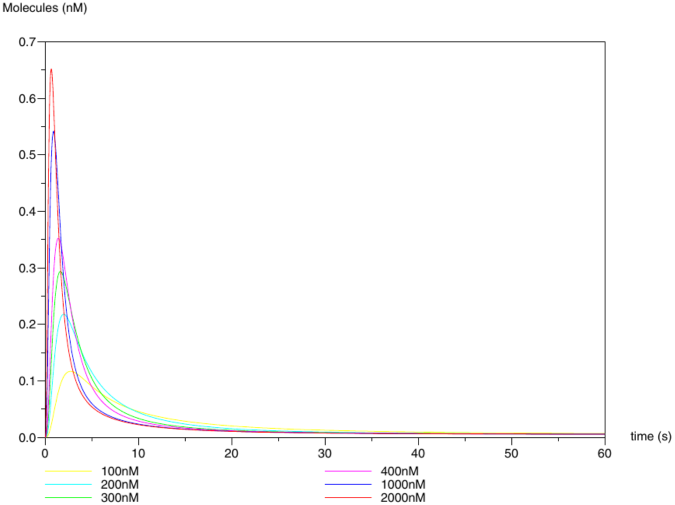
In order to properly describe the dynamics of cellular processes, quantitative extensions of P Systems have to be considered

- evolution rules have to be associated with reaction kinetics information
- translations of P Systems models into ordinary differential equations (ODEs) have been defined
- stochastic simulators for P Systems have been developed

P Systems model (A. Paun et al., 2006)



Results of ODE based simulations



Receptor autophosphorylation for different environmental EGF concentrations

P Systems Models of Populations and Ecosystems (1)

The simplicity of the P Systems notation suggested their application also to the modelling of other kinds of system

- In particular, population dynamics and ecosystems

In facts:

- Individuals of a population can be modelled by means of objects
- Actions and interactions can be modelled by means of evolution rules
- The morphology of the population territory can be modelled by means of membranes
- Independence of individuals agrees with maximal parallelism (e.g. reproductive seasons)

P Systems Models of Populations and Ecosystems (2)

The modelling of populations by means of P Systems has some analogies with the Individual Based Modelling approach

- The dynamics of the population emerges from the interactions among individuals
- Similar to agent-based approaches in computer science

Examples of evolution rules:

- Assume M, F to be individuals (male, female), O to be a offspring and P to be a predator
- Mating and birth: $MF \rightarrow MFO$
- Growth: $O \rightarrow F$
- Death: $F \rightarrow$
- Predation: $PM \rightarrow P$