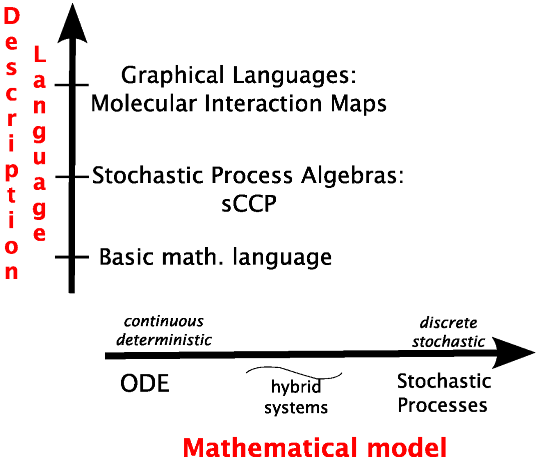
Constraint-based simulation of biological systems described by Molecular Interaction Maps

Luca Bortolussi[1] Simone Fonda[4] Alberto Policriti[2,3]

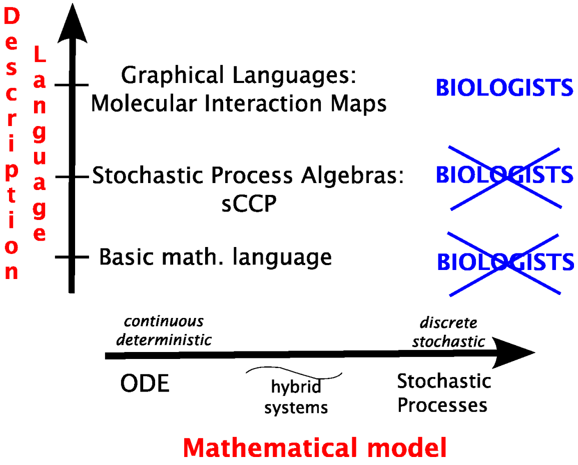
[1]Dipartimento di Matematica ed Informatica
Università degli studi di Trieste
luca@dmi.units.it
[2]Dipartimento di Matematica ed Informatica
Università degli studi di Udine
[3]Istituto di Genomica Applicata
Parco Scientifico Tecnologico, Udine.
[4]Dipartimento di Informatica
Università degli studi di Pisa.

WCB 2007, Porto, 13^{th} September 2007

Views of Computational Systems Biology

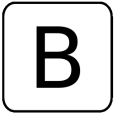


Views of Computational Systems Biology



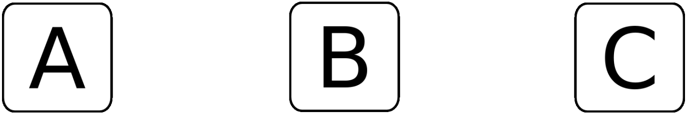
Outline

Molecular Interaction Maps



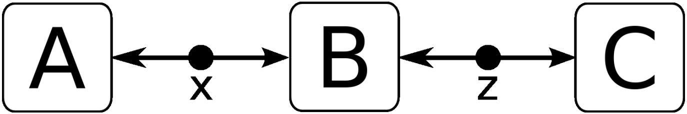
K. W. Kohn et alt. MIM of bioregulatory networks: A general rubric for systems biology. *Mol. Bio. of the Cell*, 2006.

Molecular Interaction Maps



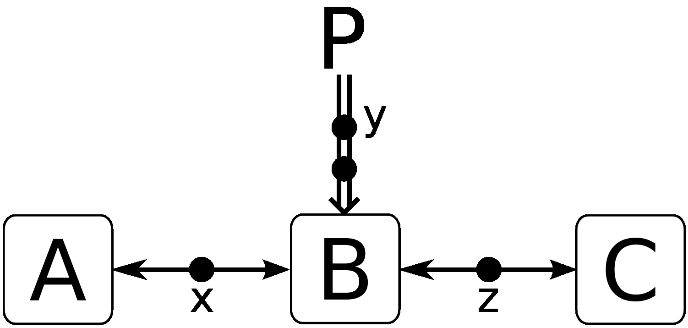
K. W. Kohn et alt. MIM of bioregulatory networks: A general rubric for systems biology. *Mol. Bio. of the Cell*, 2006.

Molecular Interaction Maps



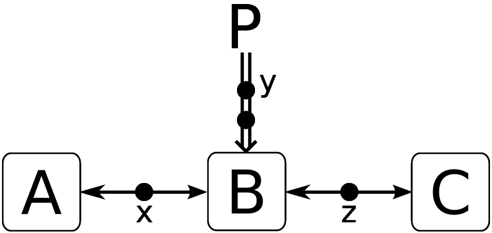
K. W. Kohn et alt. MIM of bioregulatory networks: A general rubric for systems biology. *Mol. Bio. of the Cell*, 2006.

Molecular Interaction Maps



K. W. Kohn et alt. MIM of bioregulatory networks: A general rubric for systems biology. *Mol. Bio. of the Cell*, 2006.

Interpretations



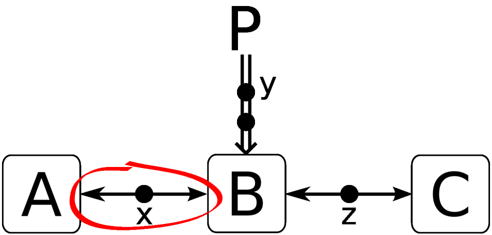
Explicit

A:B
B:C
pB

Combinatorial

pB
A:B
B:C
A:B:C
A:pB
pB:C
A:pB:C

Combinatorial explosion



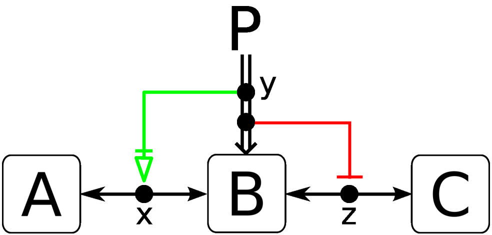
Explicit
1 reaction

A + B → A:B

Combinatorial
4 reactions

A + B → A:B
A + pB → A:pB
A + B:C → A:B:C
A + pB:C → A:pB:C

Contingencies



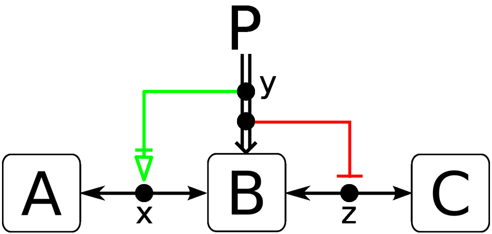
Explicit

pB
B:C
A:pB

Combinatorial

pB
B:C
A:pB

Contingencies



Explicit

pB
B:C
A:pB

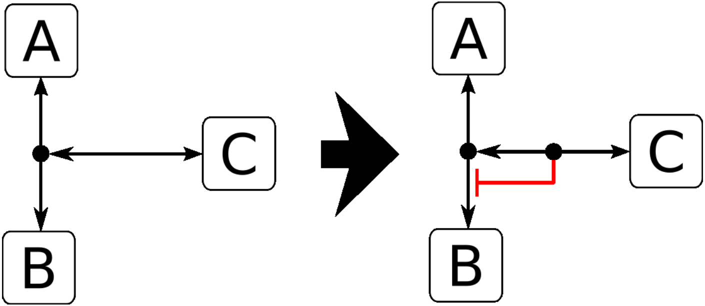
Combinatorial

pB
B:C
A:pB

Ambiguity in MIMs

Interpretation $\neq$ formal semantic

MIMs are inherently ambiguous.
We identified some cases of ambiguity and defined a set of Graph Rewriting Rules to disambiguate them.



Concurrent Constraint Programming

Constraint Store

- In this process algebra, the main object are **constraints**, which are *formulae over an interpreted first order language* (i.e. $X = 10$, $Y > X - 3$).
- Constraints can be added to a "container", the constraint store, but can never be removed.

Agents

Agents can perform two basic operations on this store (asynchronously):

- Add a constraint (**tell** ask)
- Ask if a certain relation is entailed by the current configuration (**ask** instruction)

V. Saraswat, *Concurrent Constraint Programming*, MIT press, 1993

Syntax of CCP

$$Program = Decl.A$$

$$D = \varepsilon \mid Decl.Decl \mid p(x) :- A$$

$$\begin{aligned} A = \;& \mathbf{0} \\ \mid\;& \mathrm{tell(c).A} \\ \mid\;& \mathrm{ask(c_1).A_1 + ask(c_2).A_2} \\ \mid\;& A_1 \parallel A_2 \mid \exists_x A \mid p(x) \end{aligned}$$

Syntax of sCCP

Syntax of Stochastic CCP

$$Program = D.A$$

$$D = \varepsilon \mid D.D \mid p(\overrightarrow{x}) : -A$$

$$\begin{aligned} A &= \mathbf{0} \mid \mathrm{tell}_{\infty}(c).A \mid M \mid \exists_x A \mid A \parallel A \\ M &= \pi.G \mid M + M \\ \pi &= \mathrm{tell}_{\lambda}(c) \mid \mathrm{ask}_{\lambda}(c) \\ G &= \mathbf{0} \mid \mathrm{tell}_{\infty}(c).G \mid p(\overrightarrow{y}) \mid M \mid \exists_x G \mid G \parallel G \end{aligned}$$

L. Bortolussi, *Stochastic Concurrent Constraint Programming*, QAPL, 2006

Stochastic Rates

Rates are functions from the constraint store $\mathcal{C}$ to positive reals:

$$\lambda : \mathcal{C} \longrightarrow \mathbb{R}^{+}.$$

Rates can be thought as speed or duration of communications.

sCCP – technical details

Operational Semantics

Show Details

- There are *two transition relations*, one instantaneous (finite and confluent) and one stochastic.
- Traces are sequences of events with variable time delays among them.

Discrete vs. Continuous Semantics

Show Details

- The operational semantics is *abstract w.r.t. the notion of time*: we can map the labeled transition system into a discrete or a continuous time Markov Chain.

Implementation

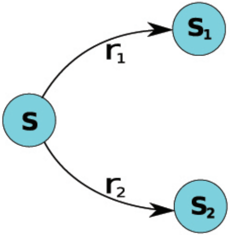
- We have an interpreter written in Prolog, using the *CLP engine of SICStus* to manage the constraint store.
- Efficiency issues.

Stream Variables

- *Quantities varying over time* can be represented in sCCP as unbounded lists.
- Hereafter: special meaning of $X = X + 1$.

Continuous Time Markov Chains

A **Continuous Time Markov Chain** (CTMC) is a direct graph with edges labeled by a real number, called the rate of the transition (representing the speed or the frequency at which the transition occurs).

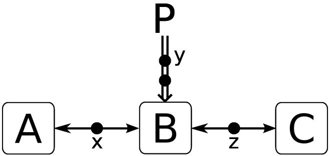


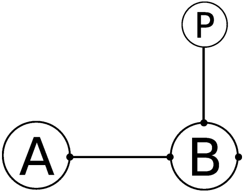
- In each state, we select the next state according to a *probability distribution* obtained normalizing rates (from S to S_1 with prob. $\frac{r_1}{r_1+r_2}$).
- The time spent in a state is given by an exponentially distributed random variable, with rate given by the *sum of outgoing transitions* from the actual node ($r_1 + r_2$).

Encoding — target

Implicit simulation of MIMs

Encoding — overview





- interaction sites = ports (*boolean state*);
- molecules = collection of ports;
- complexes = graphs:
  - *vertices* are molecules;
  - *edges* connect two ports;

Encoding — description

Static description

- Port types (*constraint store*);
- Molecular types (*constraint store*);
- Contingencies (*constraint store*);
- Interactions (*sCCP agents*);

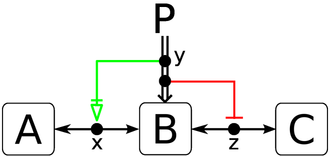
Dynamic description

- Instances of port and molecular types (*constraint store*);
- Complex types (*constraint store*);
- Counters of the number of each port and complex type (*constraint store*).

Encoding — store predicates

- `molecular_type(molecular_type_id, port_list, contingency_list)`
- `node(molecular_type_id, mol_id)`
- `edge([mol_id1, port_id1], [mol_id2, port_id2])`
- `complex_type(complex_id, node_list, edge_list, contingency_list)`
- `complex_number(complex_type_id, Num)`
- `port_number(port_id, Num)`

Encoding — contingencies



Contingencies are logical rules

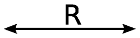
IF (there are some edges)
THEN (inhibit or allow some other ports of edges)

IF (there is y) THEN (inhibit z)
IF (there is y) THEN (allow x)

Encoding — dynamics

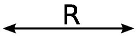
- Each arrow (*interaction capability*) in the MIM is associated to an *sCCP agent*.
- These agents modify the store according to the prescriptions of the MIM.
- There are also other agents, like `port_ managers`, performing minor tasks (e.g. bookkeeping).

Simulation in sCCP



1. choose reaction
   - Interaction agents compete stochastically to determine next reaction
   - reactions act on port (types)
2. choose actual complexes involved
   - Each port type has a port manager agent doing this
3. build product and apply enabled contingencies

Simulation in sCCP

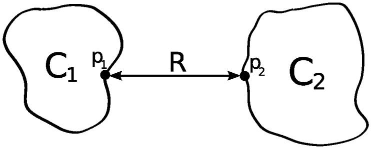


1. choose reaction
   - Interaction agents compete stochastically to determine next reaction
   - reactions act on port (types)
2. choose actual complexes involved
   - Each port type has a port manager agent doing this
3. build product and apply enabled contingencies

Simulation in sCCP

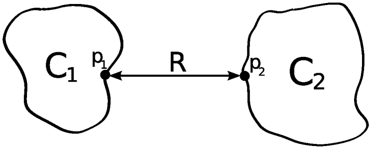
1. choose reaction
   - Interaction agents compete stochastically to determine next reaction
   - reactions act on port (types)
2. choose actual complexes involved
   - Each port type has a port manager agent doing this
3. build product and apply enabled contingencies

Simulation in sCCP



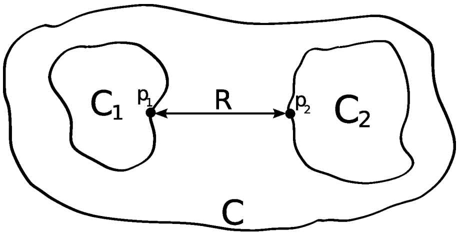
1. choose reaction
   - Interaction agents compete stochastically to determine next reaction
   - reactions act on port (types)
2. choose actual complexes involved
   - Each port type has a port manager agent doing this
3. build product and apply enabled contingencies

Simulation in sCCP



1. choose reaction
   - Interaction agents compete stochastically to determine next reaction
   - reactions act on port (types)
2. choose actual complexes involved
   - Each port type has a port manager agent doing this
3. build product and apply enabled contingencies

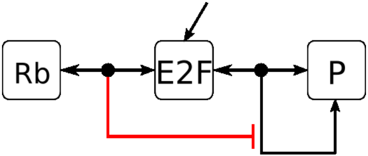
Simulation in sCCP

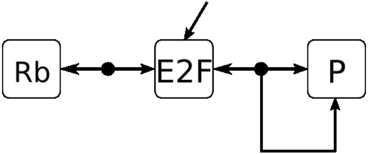


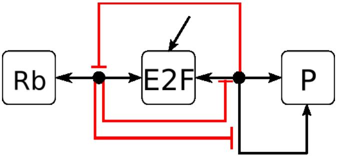
1. choose reaction
   - Interaction agents compete stochastically to determine next reaction
   - reactions act on port (types)
2. choose actual complexes involved
   - Each port type has a port manager agent doing this
3. build product and apply enabled contingencies

A simple example

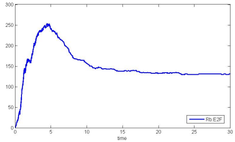
Mammalian G1/S cell cycle phase transition

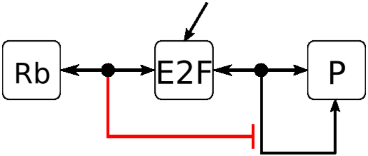


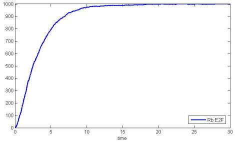


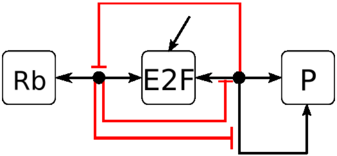


A simple example









Conclusions

- sCCP allows an implicit simulation of MIMs
- the key ingredient is the *use of the constraint store* to represent and manage graph-based representation of complexes
- the encoding is compositional and linear in the size of MIMs
- the stochastic simulation is a natural consequence of the semantics of sCCP
- Future work: a more efficient implementation

Operational Semantic: Instantaneous Transition

Instantaneous Transition

$(IR1)$	$\langle \mathrm{tell}_{\infty}(c), d, V \rangle \longrightarrow \langle \mathbf{0}, d \sqcup c, V \rangle$	
$(IR2)$	$\left\langle p(\vec{x}), d, V \right\rangle \longrightarrow \left\langle A[\vec{x}/\vec{y}], d, V \right\rangle$	if $p(\vec{y}) :- A$
$(IR3)$	$\langle \exists_x A, d, V \rangle \longrightarrow \langle A[y/x], d, V \cup \{y\} \rangle$	with $y \in \mathcal{V}_2 \setminus V$
$(IR4)$	$\dfrac{\langle A_1, d, V \rangle \longrightarrow \langle A_1', d', V' \rangle}{\langle A_1.A_2, d, V \rangle \longrightarrow \langle A_1'.A_2, d', V' \rangle}$	
$(IR5)$	$\dfrac{\langle A_1, d, V \rangle \longrightarrow \langle A_1', d', V' \rangle}{\langle A_1 \parallel A_2, d, V \rangle \longrightarrow \langle A_1' \parallel A_2, d', V' \rangle}$	

Theorem

The instantaneous transition $\longrightarrow$ is confluent and can be applied only a finite number of steps to each configuration $\mathfrak{C}$.

Return

Operational Semantic: Stochastic Transition

Stochastic Transition

(*SR*1) $\langle \text{tell}_\lambda(c), d, V\rangle \Longrightarrow_{(1,\lambda(d))} \langle \mathbf{0}, d \sqcup c, V\rangle$ if $d \sqcup c \neq \mathit{false}$

(*SR*2) $\langle \text{ask}_\lambda(c), d, V\rangle \Longrightarrow_{(1,\lambda(d))} \langle \mathbf{0}, d, V\rangle$ if $d \vdash c$

(*SR*3) $$\frac{\langle \pi, d, V\rangle \Longrightarrow_{(p,\lambda)} \langle \mathbf{0}, d', V\rangle}{\langle \pi.A, d, V\rangle \Longrightarrow_{(p,\lambda)} \overrightarrow{\langle A, d', V\rangle}}$$ with $\pi = \text{ask}$ or $\pi = \text{tell}$

(*SR*4) $$\frac{\langle A_1, d, V\rangle \Longrightarrow_{(p,\lambda)} \overrightarrow{\langle A_1', d', V'\rangle}}{\langle A_1.A_2, d, V\rangle \Longrightarrow_{(p,\lambda)} \overrightarrow{\langle A_1'.A_2, d', V'\rangle}}$$

(*SR*5) $$\frac{\langle M_1, d, V\rangle \Longrightarrow_{(p,\lambda)} \overrightarrow{\langle A_1', d', V'\rangle}}{\langle M_1 + M_2, d, V\rangle \Longrightarrow_{(p',\lambda')} \overrightarrow{\langle A_1', d', V'\rangle}}$$
with $p' = \frac{p\lambda}{\lambda + \text{rate}(\langle M_2, d, V\rangle)}$ and $\lambda' = \lambda + \text{rate}(\langle M_2, d, V\rangle)$

(*SR*6) $$\frac{\langle A_1, d, V\rangle \Longrightarrow_{(p,\lambda)} \overrightarrow{\langle A_1', d', V'\rangle}}{\langle A_1 \parallel A_2, d, V\rangle \Longrightarrow_{(p',\lambda')} \overrightarrow{\langle A_1' \parallel A_2, d', V'\rangle}}$$
with $p' = \frac{p\lambda}{\lambda + \text{rate}(\langle A_2, d, V\rangle)}$ and $\lambda' = \lambda + \text{rate}(\langle A_2, d, V\rangle)$

rate returns the *sum of rates of all active agents*.

Operational Semantic: Stochastic Transition

Theorem

Let $\langle A, d, V \rangle \in \mathfrak{C}$ *be the current configuration. Then the next stochastic transition executes one of the agents prefixed by a guard belonging to the set* $\mathrm{exec}(\langle A, d, V \rangle)$*, call it* $\overline{A}$*. Moreover, the probability of the transition (i.e. the first label in* $\Longrightarrow$*) is*

$$\frac{\mathrm{rate}(\langle \overline{A}, d, V \rangle)}{\mathrm{rate}\left(\mathrm{exec}(\langle A, d, V \rangle)\right)},$$

and the rate associated to the transition (the second label in $\Longrightarrow$*) is*

$$\mathrm{rate}\left(\mathrm{exec}(\langle A, d, V \rangle)\right).$$

◂ Return

Rates

Rates can be interpreted as priorities or as frequencies.

Rates as Priorities

- A rate can represent the *priority of execution* of a process.
- There is a *global scheduler* choosing probabilistically between active processes, according to their priority.
- *Discrete time evolution*.

Rates as Frequencies

- A rate can represent the *frequency* or *speed* of a process.
- The higher the speed, the higher the probability of seeing a certain process executed.
- *Continuous time evolution*.

Return

Discrete and Continuous Time

Discrete time

Discrete time transition can be recovered from stochastic transition $\Longrightarrow_{(p,\lambda)}$ by dropping the second label. Hence we leave only the probability associated to transitions, obtaining a Discrete time Markov Chain.

Continuous Time

Continuous time transition can be recovered from stochastic transition $\Longrightarrow_{(p,\lambda)}$ by multiplying the two labels. Hence we consider the rate associated to the transition, obtaining a Continuous time Markov Chain.

Return

Discrete and Continuous Time Observables

Discrete time I/O observables

$$\mathcal{O}_d\left(\langle A, d\rangle\right) = \left\{(d', p) \mid p = \mathrm{Prob}\left(\langle A, d\rangle \longrightarrow \left\langle \mathbf{0}, d'\right\rangle\right)\right\}.$$

Continuous time I/O observables

$$\mathcal{O}_c(\langle A, d\rangle)(t) = \left\{(d', p) \mid p = \mathrm{Prob}\left(\langle A, d\rangle \longrightarrow \left\langle \mathbf{0}, d'\right\rangle\right)(t)\right\}.$$

Theorem

$$\lim_{t\to\infty} \mathcal{O}_c(\langle A, d\rangle)(t) = \mathcal{O}_d(\langle A, d\rangle).$$

◂ Return