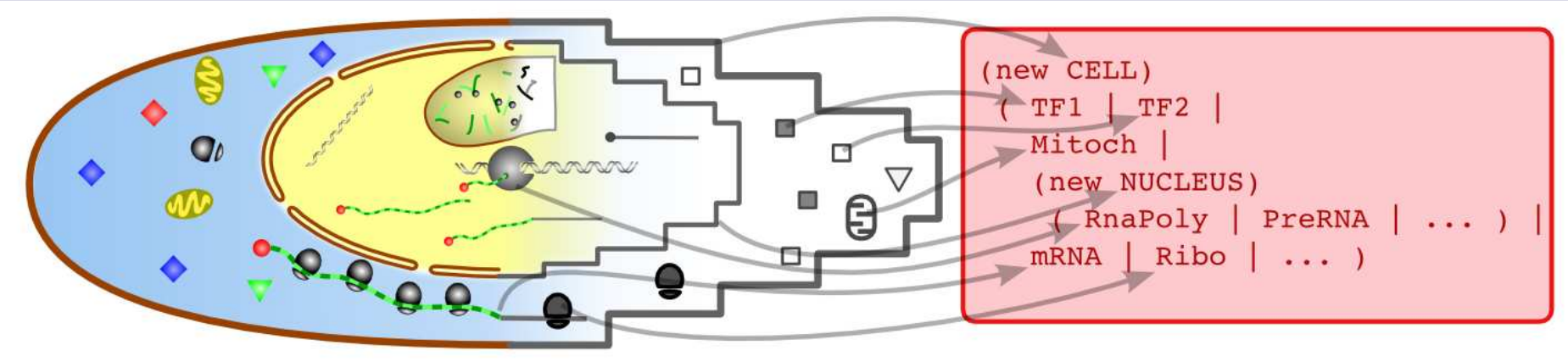
Stochastic Simulation of Biological Systems with Dynamical Compartments

Cristian Versari
versari(at)cs.unibo.it

Department of Computer Science
University of Bologna

Workshop on Computational Approaches to Biology
Pisa, October 19, 2007

Biochemical Modelling by Process Calculi



Modelling of biological systems by means of process calculi:

- formal, unambiguous specification;
- compositional description;
- several kind of analysis:
  - systems of ordinary differential equations;
  - model checking;
  - **stochastic simulation**.

Current Languages and Models

Current languages allow to model

- parallel evolution of biological elements and
- their chemical interaction (reaction);
- compartments with static or dynamic, flat or nested structure;

Stochastic simulation applied to

- "simple" models of biochemical pathways with
- (often) mono-compartment or static multi-compartment structure;
- well-stirred solutions in fixed-volume compartments.

What's left?

Formalisation and analysis (simulation) of bio-systems will deal with:

- more detailed (and complex) kinetic models;
- not well-stirred conditions;
- physical parameters like
  - temperature,
  - pressure;
- physical properties of biological elements like
  - spatial position,
  - tridimensional shape,
  - electric charge,
  - mechanical characteristics (resistance, elasticity, ...);
- multiple compartments with
- dynamical structure,
- varying volumes;
- ...

Simulation is Expensive

Stochastic simulation is computationally expensive:

- current models are "simple" w.r.t. biological reality;
- each run of a simulation may require *hours* for simulating few *(milli)seconds* of evolution of the system.

Remark

Long simulations of detailed systems by complex kinetic models may require **months**!

Reducing Simulation Time

Problem

How to reduce simulation time?

- by using a faster CPU;
- by **parallelisation** of the simulation.

Reducing Simulation Time

Problem

How to reduce simulation time?

- by using a faster CPU;
- by **parallelisation** of the simulation.

Reducing Simulation Time

Problem

How to reduce simulation time?

- by using a faster CPU;
- by **parallelisation** of the simulation.

Parallelisation Issues

Efficiency of parallel implementations of algorithms for simulating bio-systems would be affected by

- enforced synchrony for the whole system;
- high throughput between nodes.

In particular, parallel implementations of process calculi would deal with

- mobility of elements;
- communication overhead for preserving the semantics, depending on communication primitives;
- bottlenecks, in case of centralised implementations.

A core calculus

Premise

Calculi for biological modelling share

- several primitives for the description and evolution of the system;
- the same basic issues towards parallelisation.

Purpose

It would be convenient to develop a **core calculus**

- provided with parallel implementation;
- on top of which other calculi (as many as possible) may be implemented.

π@: a Conservative Core Calculus

The π@ Calculus [Versari, ESOP'07]

π@ ::= π-Calculus + polyadic synch + priority

π@ (paillette) features

- conservative π-Calculus extension;
- polyadic synchronisation for modeling compartment scoping;
- priority for gaining atomicity of sequences of operations.

π@ Syntax

π@ syntax

$$P \quad ::= \quad \sum_{i \in I} \pi_i.P_i \quad \Big| \quad P \mid Q \quad \Big| \quad !P \quad \Big| \quad (\nu\, x)P$$

$$\pi \quad ::= \quad \tau \quad \Big| \quad \mu_1 @ \cdots @ \mu_n : k(x) \quad \Big| \quad \overline{\mu_1 @ \cdots @ \mu_n : k}\langle x \rangle$$

- each channel is represented by a vector of one or more names $\mu_1, \ldots, \mu_n$;
- each input or output action has a priority k;
- higher priority actions are executed first;
- priority is static.

π@ Semantics

π@ semantics

$$\frac{\tau \notin \bigcup_{i<k} I^i(M)}{\tau : k.P + M \ \rightarrow_k \ P} \qquad \frac{P \ \rightarrow_k \ P'}{(\nu\, x)P \ \rightarrow_k \ (\nu\, x)P'}$$

$$\frac{\tau \notin \bigcup_{i<k} I^i(M \mid N)}{(\mu : k(y).P + M) \mid (\overline{\mu} : k\langle z\rangle.Q + N) \ \rightarrow_k \ P\{z/y\} \mid Q}$$

$$\frac{P \ \rightarrow_k \ P' \qquad \tau \notin \bigcup_{i<k} I^i(P \mid Q)}{P \mid Q \ \rightarrow_k \ P' \mid Q} \qquad \frac{P \equiv Q \quad P \ \rightarrow_k \ P' \quad P' \equiv Q'}{Q \ \rightarrow_k \ Q'}$$

- the only difference from π-Calculus semantics is the side condition in red: no additional rules required;
- the $I^k(P)$ function represents the set of actions of priority k ready to be executed by the process P.

Localisation by means of Polyadic Synchronisation

Polyadic synchronisation: channels are identified by one *or more* names

π-Calculus

$$P \equiv \overline{c}.P'$$

π@

$$P \equiv \overline{c_1@c_2}.P'$$

Compartments may be represented by one of the names of each channel:

$$P \equiv \overline{c@compartment_P}.P' \qquad Q \equiv c@compartment_Q.Q'$$

- P and Q may share free names;
- P and Q may interact iff $compartment_P = compartment_Q$.

Atomicity by means of Priority

Priority: high-priority reactions happen *before* lower-priority ones

Example

$$S \equiv \bar{l}.P_1 \mid l.P_2 \mid \underline{\bar{h}}.Q_1 \mid \underline{h}.Q_2 \quad \not\rightarrow \quad T \equiv P_1 \mid P_2 \mid \underline{\bar{h}}.Q_1 \mid \underline{h}.Q_2$$

$$S \equiv \bar{l}.P_1 \mid l.P_2 \mid \underline{\bar{h}}.Q_1 \mid \underline{h}.Q_2 \quad \rightarrow \quad S_2 \equiv \bar{l}.P_1 \mid l.P_2 \mid Q_1 \mid Q_2$$

$$\rightarrow \quad S_3 \equiv P_1 \mid P_2 \mid Q_1 \mid Q_2$$

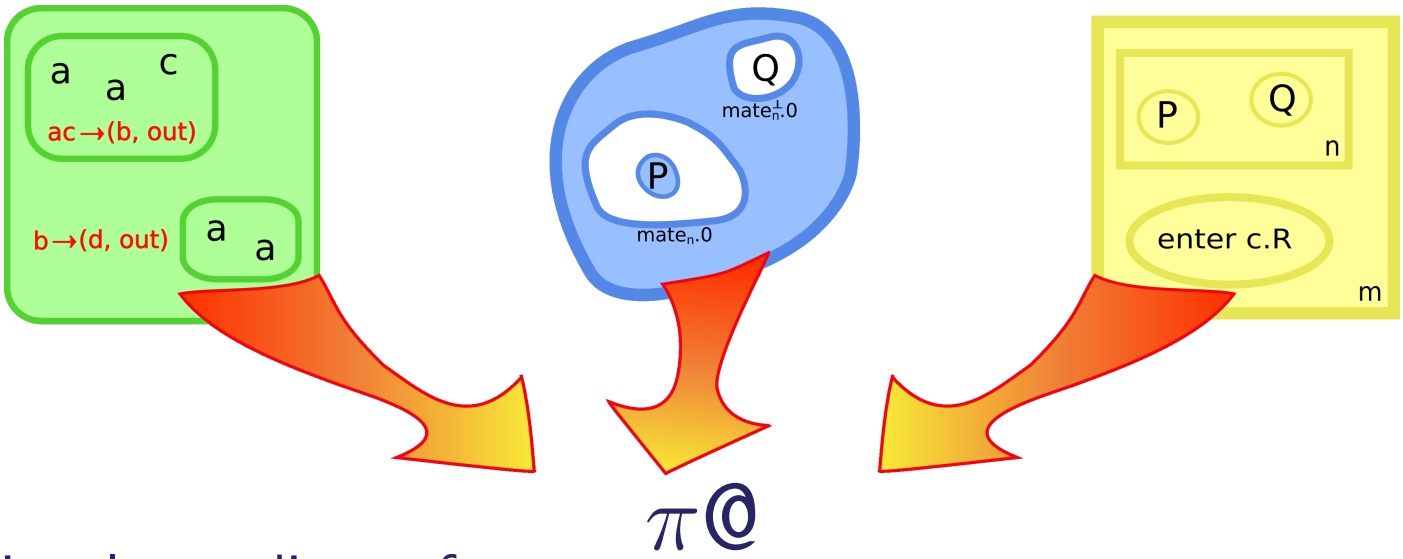
Each atomic sequence of operations may be encoded as *a low priority reaction followed by an unlimited number of high priority reactions*:

$$P_1 \equiv \overline{seq_1}.\underline{\overline{op_{11}}}.\underline{\overline{op_{12}}}.\underline{\overline{op_{13}}} \qquad P_2 \equiv \overline{seq_2}.\underline{\overline{op_{21}}}.\underline{\overline{op_{22}}}.\underline{\overline{op_{23}}}$$

The executions of P_1 and P_2 never overlap

Encodings into $\pi@$

Encodings [Versari, ESOP'07, MECBIC'07]



Compositional encodings of

- Bioambients,
- Brane Calculi,
- some kinds of P systems (with maximal parallelism!)

into $\pi@$ have been provided.

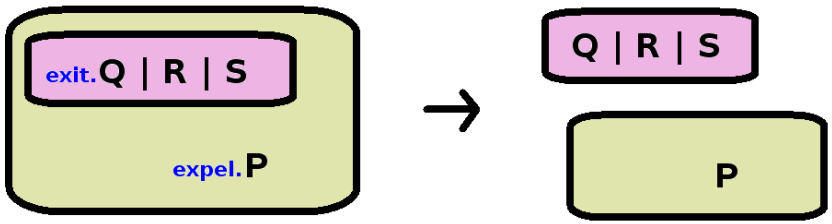
Encodings Examples

Example

Encoding actions

$$[\![\ expel\ n.P\ ]\!]_{c,pc} \equiv \overline{expel@n@c}\langle pc\rangle.[\![\ P\ ]\!]_{c,pc}$$

$$[\![\ exit\ n.Q\ ]\!]_{c,pc} \equiv expel@n@pc(x).\overline{\cdots}\langle\cdots\rangle.[\![\ Q\ ]\!]_{c,x}$$



compartment operations are followed by sequences of higher priority actions updating all involved processes (R and S, in the figure)

Stochastic π@ Calculus

The stochastic π@ calculus [Versari, Busi, CMSB'07]

Sπ@ ::= stochastic π-Calculus + polyadic synch

Sπ@ features:

- conservative extension of the stochastic π-Calculus;
- limited polyadic synchronisation (two names for each channel);
- priority as infinite channel rate;
- channels decorated with informations on *volume* of modelled elements.

Sπ@ Syntax

Sπ@ syntax

$$P ::= \mathbf{0} \mid \sum_{i \in I} \pi_i.P_i \mid P \mid Q \mid !\pi.P \mid (\nu\, x)P$$

$$\pi ::= n@c : v(x) \mid \overline{n}@c : v\langle x \rangle$$

- same operators of the π-Calculus;
- same semantics rules of the stochastic π-Calculus;
- channels composed of two names, *the second one denoting the compartment*;
- I/O actions followed by molecular volume informations.

Modelling with the Stochastic π-Calculus

Example

$$R: \quad S + T \to P + Q \qquad \Longrightarrow$$

$$S \equiv c.P \qquad T \equiv \overline{c}.Q$$

$$S|T \xrightarrow{\text{rate}(c)} P|Q$$

- parallel processes $\Longleftrightarrow$ biochemical elements (e.g. molecules);
- synchronisation/communication $\Longleftrightarrow$ reaction, binding;
- I/O channel = reaction;
- rates r associated with reactions channels c by external function $r = \text{rate}(c)$;
- compartments by restriction.

Compartments by Polyadic Synchronisation in Sπ@

Example

Compartments A, B

$$R: \quad S + T \to P + Q$$

$$\Downarrow$$

$$R_A: \quad S_A + T_A \to P_A + Q_A$$
$$R_B: \quad S_B + T_B \to P_B + Q_B$$

$$\Longrightarrow$$

$$S(x) \equiv c@x.P(x)$$
$$T(x) \equiv \overline{c}@x.Q(x)$$

$$S(a)|T(a) \xrightarrow{rate(c)} P(a)|Q(a)$$
$$S(b)|T(b) \xrightarrow{rate(c)} P(b)|Q(b)$$
$$S(a)|T(b) \not\to$$

- polyadic synchronisation $\Longrightarrow$ each channel denoted by two names:
  - the first name represents the original π-Calculus channel
  - the second one denotes the compartment

Dealing with Volumes

Remark

Stochastic variations of volumes are introduced by expressing the volume of each compartment as the sum of the volumes occupied by each molecule inside it, with

$v(S_j)$ = volume occupied by each molecule of species S_j

- syntax extended to specify the volume of each (type of) molecule;
- volume informations (optional) are appended to I/O actions.

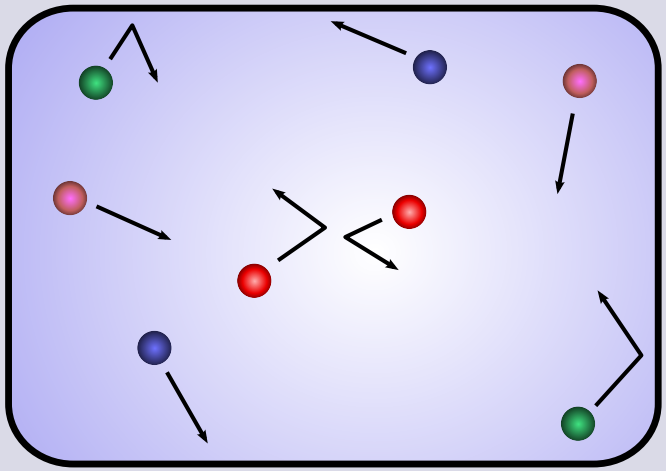
Example

$$R: \quad S + T \to P \qquad v(S) = s \qquad v(T) = t \qquad \Longrightarrow$$

$$S \equiv c:s.P \qquad T \equiv \overline{c}:t.Q$$

$$S|T \xrightarrow{rate(c)} P$$

Gillespie Stochastic Simulation Algorithm



Stochastic Simulation Algorithm (SSA) [Gillespie, JPC'77]

- molecules collide = react
- each step of the algorithm = one molecule pair (active) collision

- one of the most exploited algorithms for stochastic simulation of biochemical systems
- discrete, stochastic, exact
- single, adiabatic compartment
- fixed volume

SSA description

Description

- single compartment of fixed volume V
- N chemical species, M reaction channels
- each channel denoted by a (constant) *reaction rate* (probability) $c_1, \ldots, c_m$
- each step corresponds to the stochastic choice of one of the M reaction (molecular collision between one molecule pair)
- the (variable) number of molecules of each species $X_1, \ldots, X_N$ determine the number of molecular pairs $h_1, \ldots, h_m$ for each reaction (number of combinations for each type of molecular collision = number of ways each reaction may happen)
- rates and molecular combinations influence the *propensity functions* $a_1, \ldots, a_M$, closely related to the probability of each reaction to occur

SSA Propensity functions

Description

- rates and molecular combinations influence the *propensity functions* $a_1, \ldots, a_M$, closely related to the probability of each reaction to occur

Example

Reaction $R: \; S + T \rightarrow P$ Propensity function $a = X_S X_T \, c$

with

- a = propensity function of reaction R
- a directly proportional to
  - the reaction rate c of R
  - the number X_S of molecules of species S
  - the number X_T of molecules of species T

SSA in detail

Algorithm

1. calculate $a_j = h_j c_j \ \forall j$
2. calculate $a_0 = \sum a_j$
3. generate randomly $z_1, z_2 \in [0, 1[$
4. set $\tau = a_0^{-1} \log(z_1^{-1})$
5. find $\mu =$ smallest int s.t. $\sum_{j=1}^{\mu} a_j > z_2 a_0$
6. update $X_1, \ldots, X_N$
7. set $t = t + \tau$
8. go to step 1

- N chemical species
- M reaction channels
- $a_1, \ldots, a_M$ propensity functions
- $c_1, \ldots, c_M$ reaction rates
- $h_1, \ldots, h_M$ molecular combinations for each reaction
- τ time elapsed before next reaction R_μ
- $X_1, \ldots, X_N$ number of molecules of each chemical species
- t current time

Varying Volume

Problem

How to extend the SSA in order to handle varying volumes?

Solution

By unfolding the distinct informations enclosed in the *reaction rate*.

Example

$$R: \; S + T \longrightarrow P \qquad\qquad a = X_S X_T \, c = X_S X_T \, V^{-1} r$$

with a directly proportional to

- the reaction rate (per unit volume) r of R
- the numbers X_S, X_T of molecules of species S, T
- the inverse of the compartment volume V

Multiple Compartments

Problem

How to extend the SSA in order to handle multiple compartments?

Solution

The same type of reaction in distinct compartments represents two distinct reactions: *reaction channel shall be denoted not only by the type of reaction but also by the compartment the reaction happens in*.

Example

Reaction $R : S + T \longrightarrow P$ Compartments C_1, C_2

$$\Longrightarrow \quad R_1 : S_1 + T_1 \longrightarrow P_1 \qquad a_1 = X_{S1} X_{T1} V_1^{-1} r$$

$$R_2 : S_2 + T_2 \longrightarrow P_2 \qquad a_2 = X_{S2} X_{T2} V_2^{-1} r$$

Interaction between Compartments

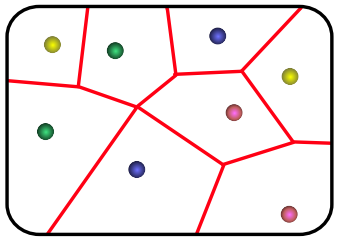
Problem

How to deal with stochastic variations of volumes due to interaction between different compartments (e.g. exchange of molecules)?

Solution

By binding the value of the volume V_k of each compartment C_k to the internal state of C_k, i.e. the number of molecules of each species inside C_k:

$$V_k = \sum_{j=1}^{M} v(S_j) X_j^k$$



- $v(S_j)$ = volume occupied by each molecule of type S_j
- X_j^k = number of molecules of species S_j inside compartment C_k

Multi-compartment SSA (MSSA)

Algorithm

0 calculate $V_k = \sum v(S_i) X_i^k \ \forall k$

1 calculate $a_j^k = V_k^{-1} h_j^k r_j \ \forall k, j$

2 calculate $a_0 = \sum a_j^k$

3 generate randomly $z_1, z_2 \in [0, 1[$

4 set $\tau = a_0^{-1} \log(z_1^{-1})$

5 find (μ, ψ) smallest int s.t. $\sum_{k=1}^{\psi} \sum_{j=1}^{\mu} a_j^k > z_2 a_0$

6 update $X_1^1, \ldots, X_N^K$

7 set $t = t + \tau$

8 go to step 0

- a_j^k propensity function of reaction R_j inside compartment C_k
- r_j reaction rate (per unit volume) of reaction R_j
- h_j^k molecular combinations for reaction R_j inside compartment C_k
- τ time elapsed before the next reaction R_μ inside C_ψ
- X_i^k number of molecules of chemical species S_i inside compartment C_k
- t current time

Computational Complexity

Algorithm

0 calculate $V_k = \sum v(S_i) X_i^k \ \forall k$

1 calculate $a_j^k = V_k^{-1} h_j^k r_j \ \forall k, j$

2 calculate $a_0 = \sum a_j^k$

3 generate randomly $z_1, z_2 \in [0, 1[$

4 set $\tau = a_0^{-1} \log(z_1^{-1})$

5 find (μ, ψ) smallest int s.t. $\sum_{k=1}^{\psi} \sum_{j=1}^{\mu} a_j^k > z_2 a_0$

6 update $X_1^1, \ldots, X_N^K$

7 set $t = t + \tau$

8 go to step 0

Computational complexity for each step:

- 0 $O(N \cdot C)$ (chemical species times compartments)
- 1 $O(M \cdot C)$ (reactions channels times compartments)
- 2 $O(C)$

3,4 constant

5 $O(M \cdot C)$

6 $O(N)$

7,8 constant

Enhancement of the MSSA

Example

System composed of:

- $10 = N$ chemical species;
- $15 = M$ react. channels;
- $100 = C$ compartments;
- species S, T, P reactant in 3,1,2 reactions respectively.

  Reaction $R^5 : S^5 + T^5 \rightarrow P^5$

After the execution of R only

- 1 compartment volume,
- 3 species concentration,
- ≤ 6 propensity functions

changed.

After every computation step only few

- volume values,
- species concentration,
- propensity functions

change.

Enhancements

- update only changed values;
- binary (instead of linear) search for next reaction;

$\Longrightarrow$ logarithmic (instead of linear) complexity.

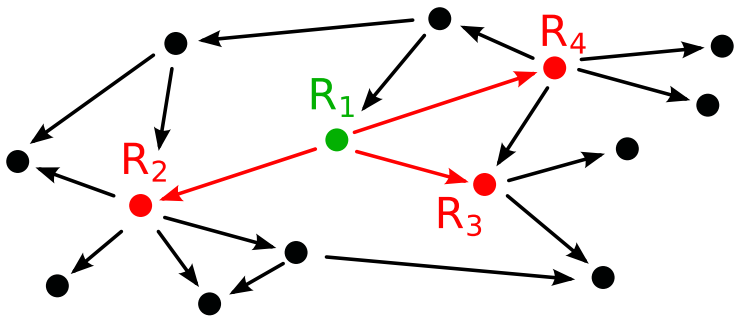
Data structures

Definition (Dependency Graph)

The *dependency graph* G of a chemical system is a directed graph where

- each vertex (in the number of $C \cdot M$) corresponds to one of the propensity functions of the system;
- an edge exists between vertexes V_1 and V_2 iff the firing (in the right compartment) of the reaction associated to V_1 influences the propensity function associated to V_2.

Example



Data structures

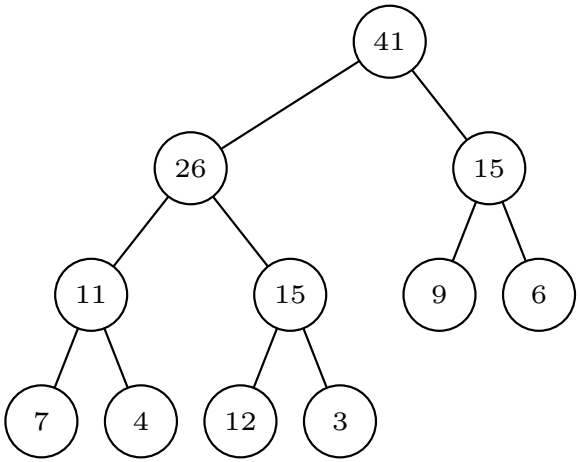
Definition (Non-Cumulative Binary Search Tree)

A *Non-Cumulative Binary Search Tree* (NCBST) is a binary tree

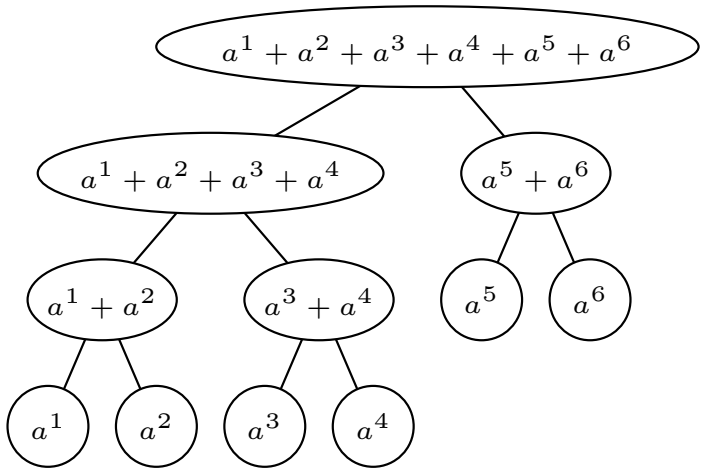
- complete (all the levels are full, except for the last);
- where the value associated to each node is equal to the sum of the values of its offspring.

Example

(a)



(b)



Sπ@ Features

Pros

Sπ@:

- simple (very close to π-Calculus syntax);
- conservative (almost same π-Calculus semantics);
- concise (reactions are specified once, additional informations on compartments and volumes are specified only if required);
- little implementation effort as extension of SPiM [Phillips, Cardelli, BioConcur'04];
- compartments with dynamical structure;
- cross-compartment elements are straightforwardly and consistently specified;
- almost unlimited compartment semantics (able to encode Bioambients, Brane Calculi, Projective Brane, ...).

MSSA Benefits

Pros

Multi-compartment stochastic simulation algorithm:

- low computational complexity (logarithmic)
- deal with variable volumes and multiple compartments with capability of exchanging molecules between compartments;
- exact stochastic simulation of many biochemical phenomena (molecular diffusion, osmosis, ...);
- can be exploited for many other calculi and formalisms (e.g. Bioambients, Brane, Betabinders, P Systems, ...).

Parallelisation issues

Remark

Stochastic process calculi (in particular the π-Calculus) embed the notion of priority as *extremal probability (infinite rate)*.

Question

May priority be implemented also in a *parallel system*?

- what kind of priority can be implemented?
- which communication primitives are required?
- what the performance gain (or loss!)?

Models of Parallel Systems

Process algebras can be exploited for the modelling of different kinds of parallel systems:

- the π-**Calculus** for non-prioritised systems with point-to-point communication
- the **b**π-**Calculus** [Ene, Muntean, FCT'99] for non-prioritised systems with broadcast-based communication
- **CPG** (CCS with priority guards) [Phillips, CONCUR'01] for systems with local priority
- **FAP** (a finite fragment of asynchronous CCS with global priority) for systems with global priority

Critical configurations

Question

Given a CPG or FAP system, may it be encoded into π- or bπ-Calculus by preserving its parallelism?

Answer

Yes.

We should provide a proper encoding function.

Answer

No.

We should provide some critical problem that can be solved in CPG or FAP but not in bπ or π:

- leader election
- last man standing

Last man standing

Example

$$System \equiv \underbrace{P \mid P \mid \cdots \mid P}_{n \text{ times}}$$

- $n = 1 \implies System \equiv P \to \overline{yes}$
- $n > 1 \implies System \equiv P \mid P \mid \cdots \to \overline{no}$

Last man standing:

- the system is composed of $n \geq 1$ peer processes
- the system (i.e. each process) must realise if $n = 1$ or $n > 1$ in a distributed way

Problem

How to detect the presence of other processes without deadlock?

Results [Versari, Busi, Gorrieri, CONCUR'07]

Lemma

- The last man standing problem cannot be solved in π.
- The last man standing problem cannot be solved in bπ.

Lemma

- The last man standing problem can be solved in CPG.
- The last man standing problem can be solved in FAP.

Separation Results

Theorem

- There exists no parallel encoding of FAP or CPG into π or bπ.
- There exists no parallel encoding of FAP into π.

Question

May priority be implemented / exploited also in parallel systems?

Answer

Not without some kind of centralised coordination ($\Longrightarrow$ bottleneck), even if we are provided with powerful communication primitives like broadcasting.

Conclusion

π@ / Sπ@

- simple
  $\Longrightarrow$ easy to use and implement;
- conservative
  $\Longrightarrow$ preserve theoretical results already obtained;
- deal with variable-volume, dynamical compartments
  $\Longrightarrow$ closer to biological scenario;
- may encode almost unlimited compartment semantics
  $\Longrightarrow$ low-level implementation of higher-abstraction languages;
- parallelisation only by centralised implementation
  $\Longrightarrow$ limited performance gain.

Future Work

Future work

- implementation of MSSA and Sπ@
- MSSA with tau-leaping (approximated simulation but faster)
- parallel implementation of Sπ@ with performance analysis
- inclusion of other physical properties (temperature, pressure, ...)

The end

Thank you.

C. Ene and T. Muntean.
Expressiveness of point-to-point versus broadcast communications.
In G. Ciobanu and G. Paun, editors, *FCT*, volume 1684 of *Lecture Notes in Computer Science*, pages 258–268. Springer, 1999.

D. T. Gillespie.
Exact stochastic simulation of coupled chemical reactions.
J. Phys. Chem., 81(25):2340–2361, 1977.

R. Milner.
Communicating and mobile systems: the π-calculus.
Cambridge University Press, New York, NY, USA, 1999.

A. Phillips and L. Cardelli.
A correct abstract machine for the stochastic pi-calculus.
In *Bioconcur'04*. ENTCS, August 2004.

I. Phillips.
Ccs with priority guards.
In K. G. Larsen and M. Nielsen, editors, *CONCUR*, volume 2154 of *Lecture Notes in Computer Science*, pages 305–320. Springer, 2001.

C. Versari.
A core calculus for a comparative analysis of bio-inspired calculi.
In R. D. Nicola, editor, *ESOP*, volume 4421 of *Lecture Notes in Computer Science*, pages 411–425. Springer, 2007.

C. Versari.
Encoding catalytic p systems in pi@.
Electr. Notes Theor. Comput. Sci., 171(2):171–186, 2007.

C. Versari and N. Busi.
Stochastic simulation of biological systems with dynamical compartment structure.
In M. Calder and S. Gilmore, editors, *CMSB*, volume 4695 of *Lecture Notes in Computer Science*, pages 80–95. Springer, 2007.


C. Versari, N. Busi, and R. Gorrieri.
On the expressive power of global and local priority in process calculi.
In L. Caires and V. T. Vasconcelos, editors, *CONCUR*, volume 4703 of *Lecture Notes in Computer Science*, pages 241–255. Springer, 2007.