



# An introduction to approaches in Computational Systems Biology

Linda Brodo  
(Università di Sassari)



# Plan



- Stochastic approaches
  - Simulation
  - Steady state analysis
- Process algebras and ODEs
- Level of abstractions and process algebras
- Logical approaches
  - Probabilistic model checking
  - Model checking and learning rules
  - Pathway Logic
- Static approaches

# Stochastic models



## Continuous simulations

(high concentrations – continuous evolution - deterministic behaviour)

## Stochastic simulations

(few hundreds of components – discrete evolutions – non-deterministic behaviour)

## Biological systems:

- With identical conditions, the same cellular process, in different cells, may evolve along distinct pathways
- At macroscopic level the organism behaves deterministically, at microscopic level the processes are inherently random.

# Intrinsic stochasticity



## Chemical Master Equation

$$P(X_1, \dots, X_N; t + dt) = P(X_1, \dots, X_N; t) \left[ 1 - \sum_{\mu=1}^M a_{\mu} dt \right] + \sum_{\mu=1}^M B_{\mu} dt$$


$$\frac{\partial}{\partial t} P(X_1, \dots, X_N; t) = \sum_{\mu=1}^M [B_{\mu} - a_{\mu} P(X_1, \dots, X_N; t)]$$

$a_{\mu} dt$  = probability that an  $R_{\mu}$  reaction will occur in volume  $V$  in  $(t, t + dt)$ , given that the system is in a state  $\bar{X}$  at time  $t$

$B_{\mu} dt$  = probability that the system is one  $R_j$  reaction removed from the  $X$  and undergoes the  $R_j$  reaction in time  $(t, t + dt)$ .

*describes the transition of a system from one state to another state  
using probabilistic methods*

# Gillespie algorithm 1/2

  
 $c_\mu$  = average prob. per unit time that a particular set of reactant molecules will both collide and react according to the  $R_\mu$  reaction expression.

$h_\mu$  = number of distinct  $R_\mu$  molecular reactant combinations available in state  $\bar{X}$

$$P(\tau, \mu) = \begin{cases} a_\mu \exp(-a_0\tau) & \text{if } 0 \leq \tau < \infty \text{ and} \\ & \mu = 1, \dots, M \\ 0 & \text{otherwise} \end{cases}$$

reaction probability density function

$$a_\mu \equiv h_\mu c_\mu \quad (\mu = 1, \dots, M) \qquad a_0 \equiv \sum_{\nu=1}^M a_\nu \equiv \sum_{\nu=1}^M h_\nu c_\nu$$

# Gillespie algorithm 2/2



The derived algorithm:

1. Initialise  $M$  reaction constants  $c_1, \dots, c_M$ ;  $N$  molecular numbers  $[X_1], \dots, [X_N]$ ;  $t := 0$ ; and a random number generator
2. Calculate  $a_\mu$  for  $\mu = 1, \dots, M$
3. Generate two random numbers  $r_1$  and  $r_2$  using a unit-interval uniform random number generator and calculate  $(\mu, \tau)$  according to

$$\sum_{\nu=1}^{\mu-1} a_\nu < r_2 a_0 \leq \sum_{\nu=1}^{\mu} a_\nu \Rightarrow \mathcal{P}_1(\mu) = \frac{a_\mu}{a_0}$$

$$\tau = \left(\frac{1}{a_0}\right) * \ln \frac{1}{r_1} \Rightarrow \mathcal{P}_2(\tau) = a_0 \exp(-a_0 \tau)$$

4. Set  $t := t + \tau$   
change/update the numbers of molecules to reflect the execution of reaction  $R_\mu$   
go to 2.

# $\pi$ -calculus and its biochemical interpretation



Syntax  $\pi ::= ?x(y) \mid !x(y)$

$P ::= 0 \mid \sum_{i \in I} \pi_i.P \mid P_1 \parallel P_2 \mid P_1 + P_2 \mid (\text{new } x)P \mid !P$

Reaction rules

$(\dots + !x(z).Q) \mid (\dots + ?x(y).P) \text{ ----> } Q \mid P\{z/y\}$

$P \text{ ----> } P'$

$P \text{ ----> } P'$

$Q \equiv P \quad P \text{ ----> } P' \quad P' \equiv Q'$

$\frac{}{P \mid Q \text{ ----> } P' \mid Q}$

$\frac{}{(\text{new } x) P \text{ ----> } (\text{new } x) P'}$

$\frac{}{Q \text{ ----> } Q'}$

molecules

processes

interaction capabilities

common names

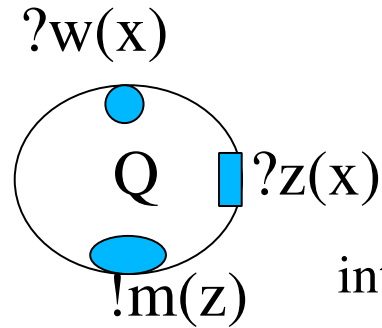
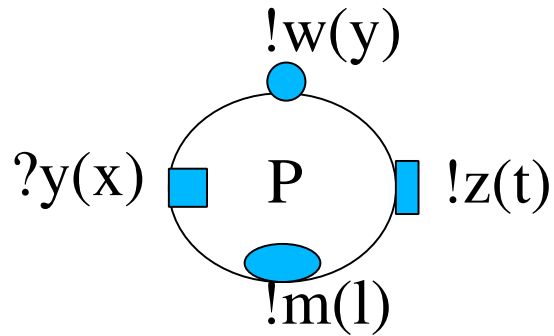
interaction

communication

compartmentalization

bound names

# $\pi$ -calculus and its biochemical interpretation



molecules  $\leftrightarrow$  processes

interaction capabilities  $\leftrightarrow$  common names

RTK ::= (new backbone) (Extra | Transmem | Intra) domains  $\leftrightarrow$  sub-processes

compartmentalization  $\leftrightarrow$  private channels

Active\_kinase ::= !phosp\_site(p\_tyr).Kinase

Mod\_Bind\_domain ::= ?phosp\_site(tyr).!tyr(t)

modifications of motifs  $\leftrightarrow$  channels passing

Regev et al. *The  $\pi$ -calculus as an abstraction for biomolecular systems*, 2004

# $\pi$ -calculus and its biochemical interpretation



$$((\bar{x}\langle z \rangle, r_b).Q) \mid ((x(y), r_b).P) \xrightarrow{x, r_b \cdot 1 \cdot 1} Q \mid P\{z/y\}$$

$$\frac{P \xrightarrow{x, r_b \cdot r_0 \cdot r_1} P'}{P \mid Q \xrightarrow{x, r_b \cdot r'_0 \cdot r'_1} P' \mid Q} \quad \begin{cases} r'_0 = r_0 + In_x(Q) \\ r'_1 = r_1 + Out_x(Q) \end{cases}$$

$$\frac{P \xrightarrow{x, r_b \cdot r_0 \cdot r_1} P'}{(\nu x)P \xrightarrow{x, r_b \cdot r_0 \cdot r_1} (\nu x)P'}$$

$$\frac{Q \equiv P \quad P \xrightarrow{x, r_b \cdot r_0 \cdot r_1} P' \quad P' \equiv Q'}{Q \xrightarrow{x, r_b \cdot r_0 \cdot r_1} Q'}$$

Stochastic rate constant

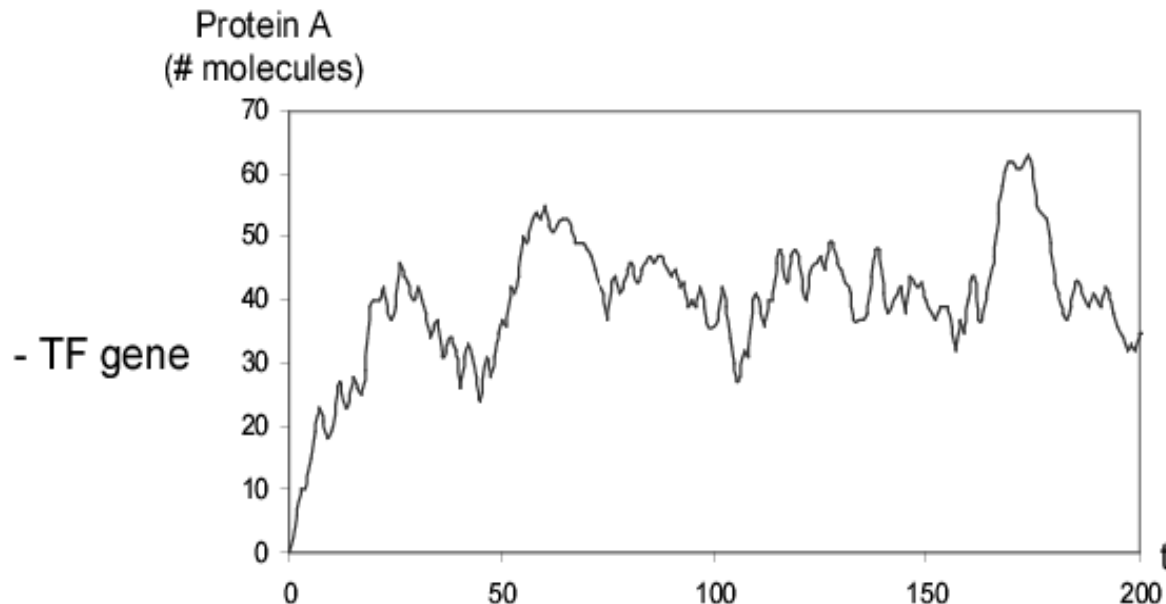
$$\alpha_\mu = r_b * (r_0 * r_1) = c_\mu * h_\mu$$

# Tool: BioSpi 2.0



Each prefix is associated with a base rate and type (instantaneous, symmetric or asymmetric).

$$a_{\mu} = \begin{cases} r_{\mu} \times (\#A) \times (\#B) & \text{for bimolecular reactions} \\ & \text{(homodimerization excluded)} \\ r_{\mu} \times \frac{(\#A) \times (\#A - 1)}{2} & \text{for homodimerization} \\ r_{\mu} \times \#A & \text{for monomolecular reaction} \end{cases}$$



example of BioSpi results for regulation of gene expression by positive feedback

Priami et al. *Application of a stochastic name-passing calculus to representation and simulation of molecular processes*, 2001.

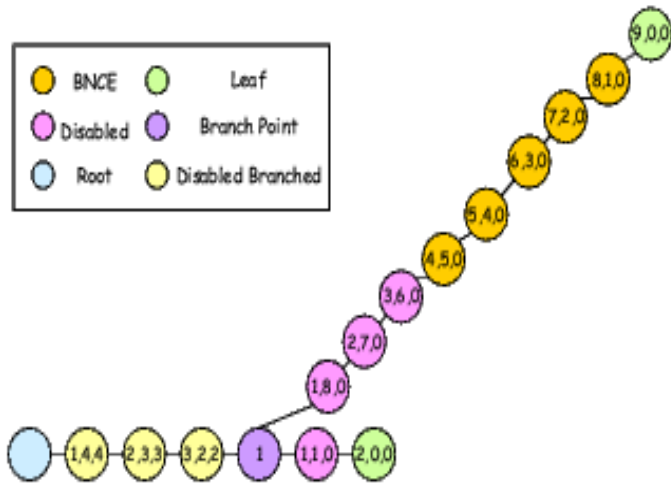
<http://www.wisdom.weizmann.ac.il/~biospi/>

# Tool: BioSpi 2.0



BioSpi 2.0 also traces all the actions executed in a complete run

B



```
.Root_Glucose.comm(.UDP_Glucose.to_root!)
Disabled_Branched_Glucose.comm(.UDP_Glucose.to_root!, .UDP_Glucose.to_root!, 1, 4,
4, global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
Disabled_Branched_Glucose.comm(.UDP_Glucose.to_root!,.UDP_Glucose.to_root!, 2, 3,
3, global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
Disabled_Branched_Glucose.comm(.UDP_Glucose.to_root!,.UDP_Glucose.to_root!, 3, 2,
2, global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
.Branch_Point.comm(.UDP_Glucose.to_root!, BCE_Glucose.new_to_root!,
.UDP_Glucose.to_root!)
Disabled_Glucose.comm(BCE_Glucose.new_to_root!,.UDP_Glucose.to_root!, 1, 8, 0,
global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
Disabled_Glucose.comm(.UDP_Glucose.to_root!,.UDP_Glucose.to_root!, 2, 7, 0,
global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
Disabled_Glucose.comm(.UDP_Glucose.to_root!,.UDP_Glucose.to_root!, 3, 6, 0,
global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
BNCE_Glucose.comm(.UDP_Glucose.to_root!, .UDP_Glucose.to_root!, 4, 5, 0,
global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
BNCE_Glucose.comm(.UDP_Glucose.to_root!, .UDP_Glucose.to_root!, 5, 4, 0,
global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
BNCE_Glucose.comm(.UDP_Glucose.to_root!, .UDP_Glucose.to_root!, 6, 3, 0,
global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)
BNCE_Glucose.comm(.UDP_Glucose.to_root!, .UDP_Glucose.to_root!, 7, 2, 0,
global.branch(1)!, global.cleave(1)!, global.glycogen(1)!)

```

(in the current implementation send/receive pair is selected in a “top-of-the-queue” fashion which is not really random)

# Tool: SPiM

An improved performance stochastic simulator.

Processes as parallel composition of sums:

$$\sum_i \pi_i . P_i \mid Q$$

~~$$(!x(n).P + !x(n).(R \mid Q)) \mid x(m).Q$$~~

- (1)  $Q \equiv P \wedge P \xrightarrow{r} P' \wedge P' \equiv Q' \Rightarrow Q \xrightarrow{r} Q'$
- (2)  $P \xrightarrow{r} P' \Rightarrow \nu x P \xrightarrow{r} \nu x P'$
- (3)  $P \xrightarrow{r} P' \Rightarrow P \mid Q \xrightarrow{r} P' \mid Q$
- (4)  $(x\langle n \rangle . P + \Sigma) \mid (x\langle m \rangle . Q + \Sigma') \xrightarrow{\text{rate}(x)} P \mid Q_{\{n/m\}}$

**Definition 2.2** *Reduction in SPi*

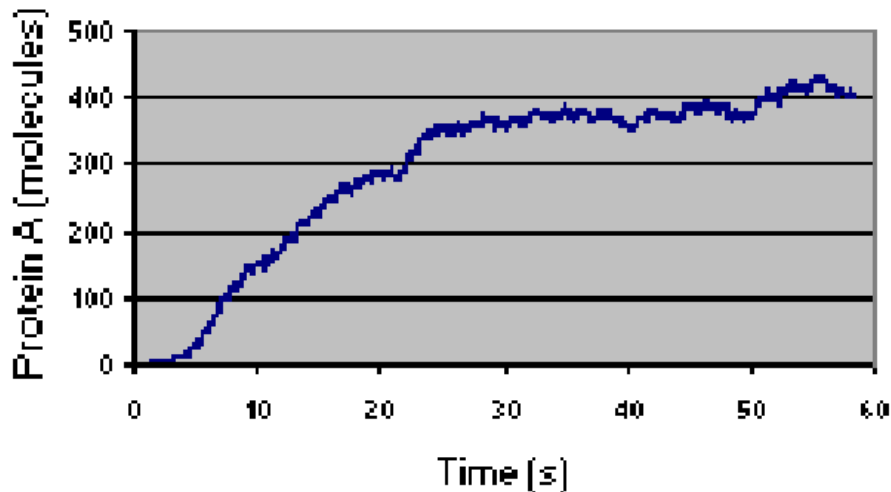
Stochastic rate constant

$$\alpha_\mu = c_x * h_\mu \longrightarrow \text{Act}_x(A) = (\text{In}_x(A) * \text{Out}_x(A)) - \text{Mix}_x(A)$$

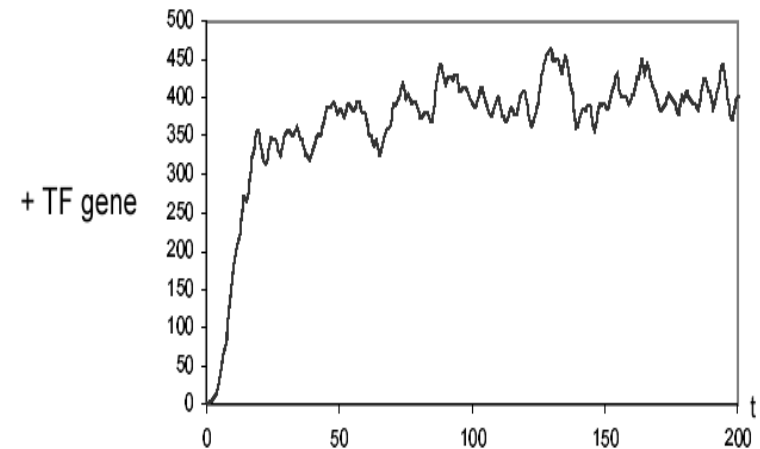
# Example: BioSpi - SPiM



Example of regulation of a gene expression by positive feedback. A protein A can bind with a protein TF by sending private names *unbind*, *send*, *remove*. Now A can send a protein *tail* to TF. Once unbound from A TF increase the promotion of DNA\_A or DNA\_TF, which are then transcribed into RNA\_A or RNA\_TF, which in turn can be translated into A or TF.



SPiM



BioSpi 1.0

# Continuous Time Markov Chain 1/2

(analysis of the steady states)



two models centered on: reagents – pathway.

reagents: levels (high-low) of proteins

pathway: cascade of reactions (*more compact formal definition*)

The two transition systems are (timing aware) bisimilar!

“we believe that any pathway has reagent/pathway-centric models bisimilar”

Stochastic rates of reactions are taken from the mathematical model:

Cho et al. *Mathematical modeling of the influence of RFIP on the ERK Signaling pathway*, 2003.

Hillston et al. *Modelling the influence of RKIP on the ERK signalling pathway using the stochastic process algebra PEPA*, 2004.

# Continuous Time Markov Chain 2/2



An attempt to establish two levels of abstraction:

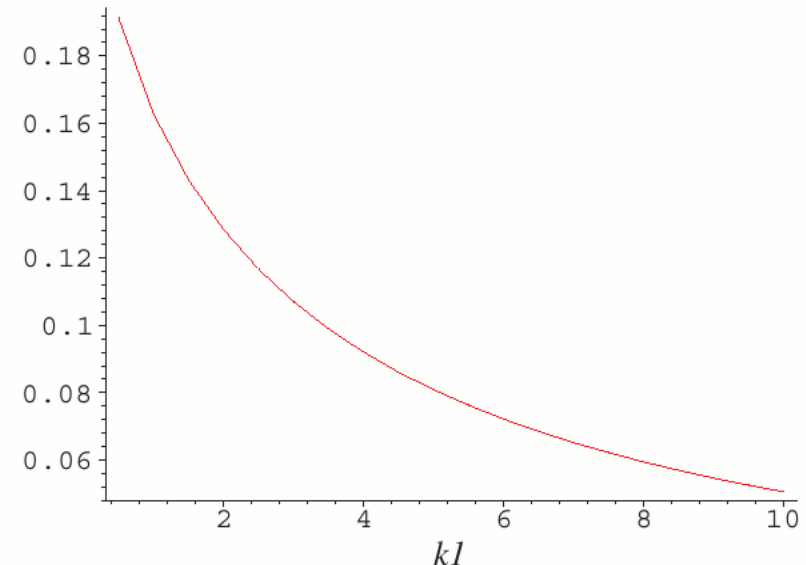
$$\begin{aligned} \text{Raf-1}_H^* &\stackrel{\text{def}}{=} (k1react, k_1). \text{Raf-1}_L^* + (k2react, k_{12}). \text{Raf-1}_L^* && \text{reagent-centric} \\ \text{Raf-1}_L^* &\stackrel{\text{def}}{=} (k5product, k_5). \text{Raf-1}_H^* + (k2react, k_2). \text{Raf-1}_H^* \\ &\quad + (k13react, k_{13}). \text{Raf-1}_H^* + (k14product, k_{14}). \text{Raf-1}_H^* \end{aligned}$$

pathway-centric

$$\begin{aligned} \text{Pathway}_{10} &\stackrel{\text{def}}{=} (k9react, k_9). \text{Pathway}_{11} \\ \text{Pathway}_{11} &\stackrel{\text{def}}{=} (k11product, k_{11}). \text{Pathway}_{10} + (k10react, k_{10}). \text{Pathway}_{10} \end{aligned}$$

Computation of the steady state by standard techniques (PEPA Workbench)

*Throughput of k14product*



Result: throughput (rate x probability)

**Fig. 8.** Plotting the effect of  $k1$  on  $k14product$

# Plan



- Stochastic approaches
  - Simulation
  - Steady state analysis
- Process algebras and ODEs
- Level of abstractions and process algebras
- Logical approaches
  - Probabilistic model checking
  - Model checking and learning rules
  - Pathway Logic
- Static approaches

# From Process Algebras to ODEs (automatically)



Number of molecules must be finite!

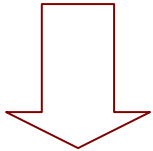
PEPA model

$$A_H \stackrel{def}{=} (k1react, k1).A_L + (k5react, k5).A_L$$

$$A_L \stackrel{def}{=} (k2react, k2).A_H + (k4react, k4).A_H$$

$$A/X_H \stackrel{def}{=} (k2react, k2).A/X_L + (k3react, k3).A/X_L$$

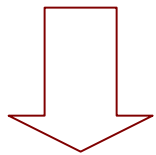
$$A/X_L \stackrel{def}{=} (k1react, k1).A/X_H$$



Activity matrix

(*stoichiometric matrix*)

<i>reactants</i>	<i>k1react</i>	<i>k2react</i>	<i>k3react</i>	<i>k4react</i>	<i>k5react</i>	<i>k6react</i>	<i>concentration variables</i>
<i>A</i>	-1	+1	0	+1	-1	0	<i>m</i> <sub>1</sub>
<i>X</i>	-1	+1	0	0	0	+1	<i>m</i> <sub>2</sub>
<i>A/X</i>	+1	-1	-1	0	0	0	<i>m</i> <sub>3</sub>
<i>B</i>	0	0	+1	-1	+1	0	<i>m</i> <sub>4</sub>
<i>Y</i>	0	0	+1	0	0	-1	<i>m</i> <sub>5</sub>



(*Standard technique*)

ODEs

$$\frac{dm_1(t)}{dt} = -k1m_1(t)m_2(t) + k2m_3(t) + k4m_4(t) - k5m_1(t)$$

$$\frac{dm_2(t)}{dt} = -k1m_1(t)m_2(t) + k2m_3(t) + k6m_5(t)$$

Hillston et al. *Automatically deriving ODEs from process algebra models of signalling pathway*, 2005.

# From $\pi$ -calculus to ODEs



Chemical Parametric form

$$X \equiv \tau^r . P \dots$$

(CPF) -subset  $\pi$ -calculus

$$(X \equiv ?n_{(r)}.P + \dots), (Y \equiv !n_{(r)}.Q + \dots)$$

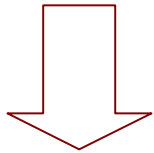


$$(X \equiv ?n_{(r)}.P + \dots \equiv !n_{(r)}.Q + \dots)$$

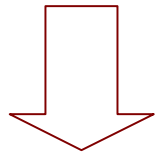
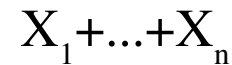
Chemical ground form

$$(X_1 | \dots | X_n)$$

(CGF)



Chemistry



ODEs

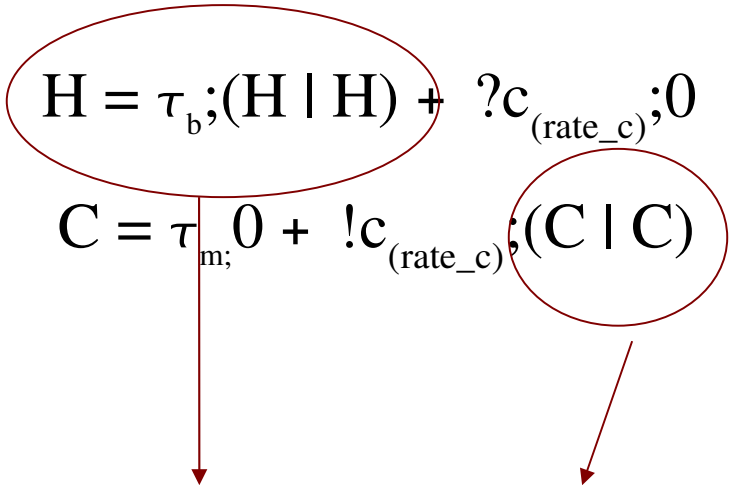
$$[A]^\bullet = -k_1[A][B] - k_2[A][C]$$

$[A]^\bullet =$  *the derivative of number of molecules as a function in time*

# PEPA vs. $\pi$ -calculus



PEPA cannot handle Lotka-Volterra model



- v1:  $C \xrightarrow{m} 0$
- v2:  $H \xrightarrow{b} H+H$
- v3:  $H + C \xrightarrow{p} C+C$

Generation of unbound number  
of molecules of the same type

Unbound  $\pi$ -calculus process to ODEs

# Utility of the translation PAs - ODEs



- Validate P.A. models
- Gain compositionality
- The discrete model and the network structure are lost during translation
- To do: extend the calculus with restriction

# Levels of abstractions



- Biochemical level: **entities** --> molecules ; **interactions** --> association between complementary domains (inactive;active-free;active-bound);
- Compartment level: **entities** --> membranes ; **interactions** --> motion of components (endocytosis; exocytosis; mitosis; meiosis);
- Metabolism level: **entities** --> molecules ; **interactions** --> modifications of molecular components (looking up at molecular modifications at a more abstract view);

Prandi et al. *Process calculi in a biological context*, 2005.

# Levels of abstractions (?)



- Biochemical level:  $\pi$ -calculus, PEPA,  $\kappa$ -calculus
- Compartment level:  $\pi$ -calculus, Bioambients, Brane-calculi
- Metabolic level:  $\pi$ -calculus (VICE: a virtual cell), PEPA (previous example), Petri-nets

Nagasaki et al. *Bio-calculus: its concept and molecular interaction*. 2002

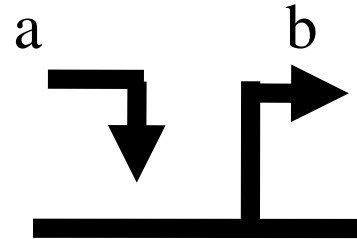
*...we decided that each biological phenomenon is defined with the unique expression similar to that of biology... Then we found that is almost impossible to define such a model for at least two reasons. Firstly, almost all phenomena are far from being completely understood... Secondly, there exists a huge number of parameters....*

# Compositional approach for Gene Networks 1/2

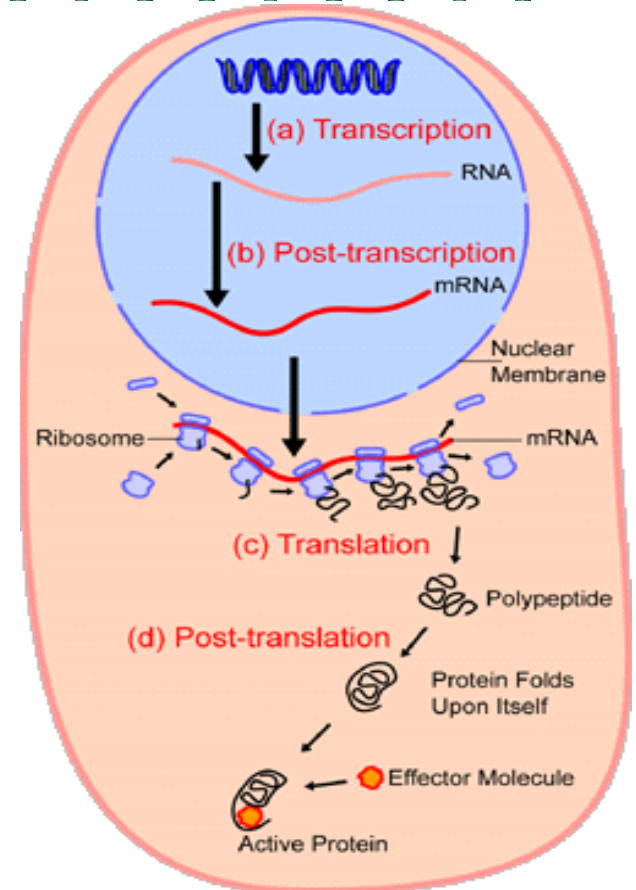


Gene regulatory network

Gene with positive feedback:

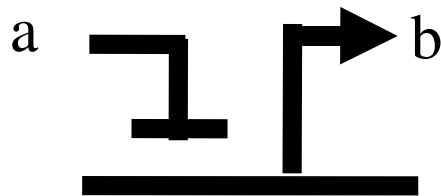


$$POS(a,b) = \lambda_a \cdot \tau_\eta \cdot (TR(b) \mid POS(b)) + \tau_\eta \cdot (TR(b) \mid POS(a,b))$$



Gene with negative feedback:

$$NEG(a,b) = \lambda_a \cdot \tau_\eta \cdot (NEG(a,b)) + \tau_\xi \cdot (TR(b) \mid NEG(a,b))$$



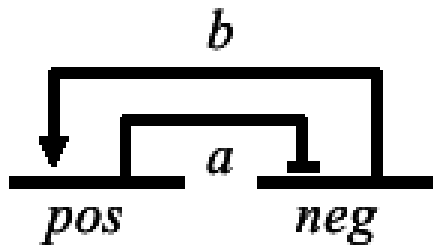
Transcriptional factor:

$$TR(b) = \lambda_b \cdot TR(b) + \tau_\delta \cdot 0$$

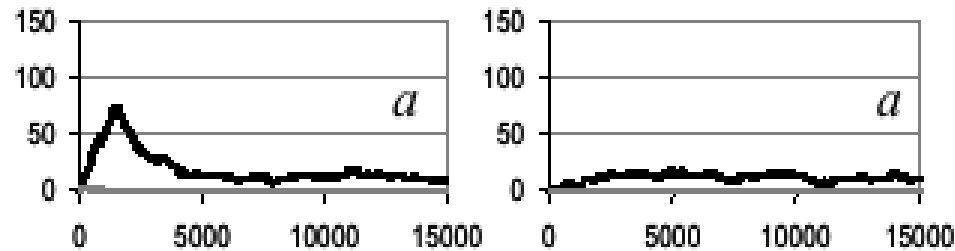
# Compositional approach for Gene Networks 2/2

## Simulation with SPIM

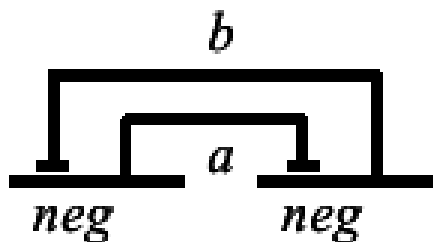
$pos(b,a) \mid neg(a,b)$



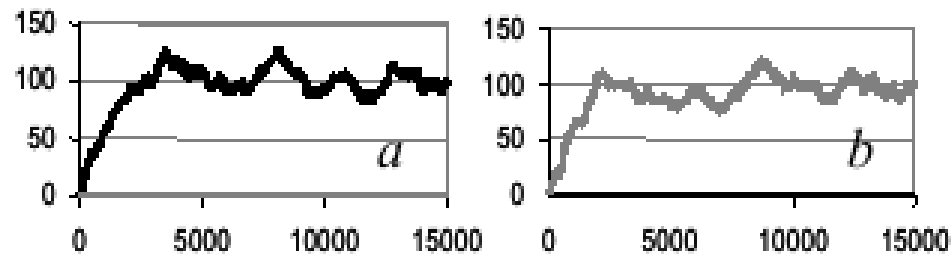
**Monostable**



$neg(b,a) \mid neg(a,b)$

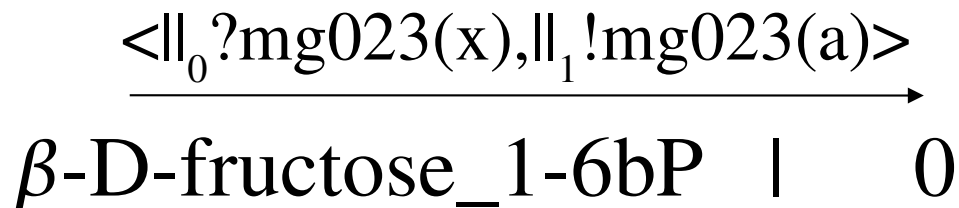
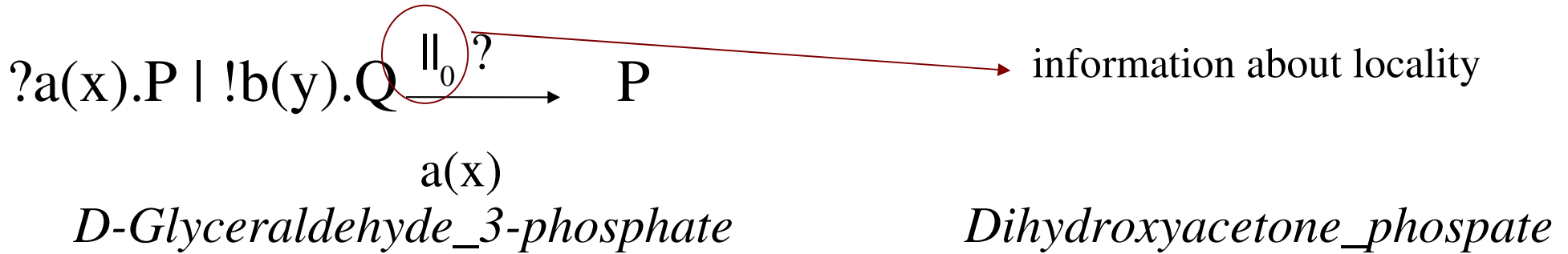


**Bistable**



# VICE: $\pi$ -calculus at metabolic level

Proved labels for  $\pi$ -calculus transition system:

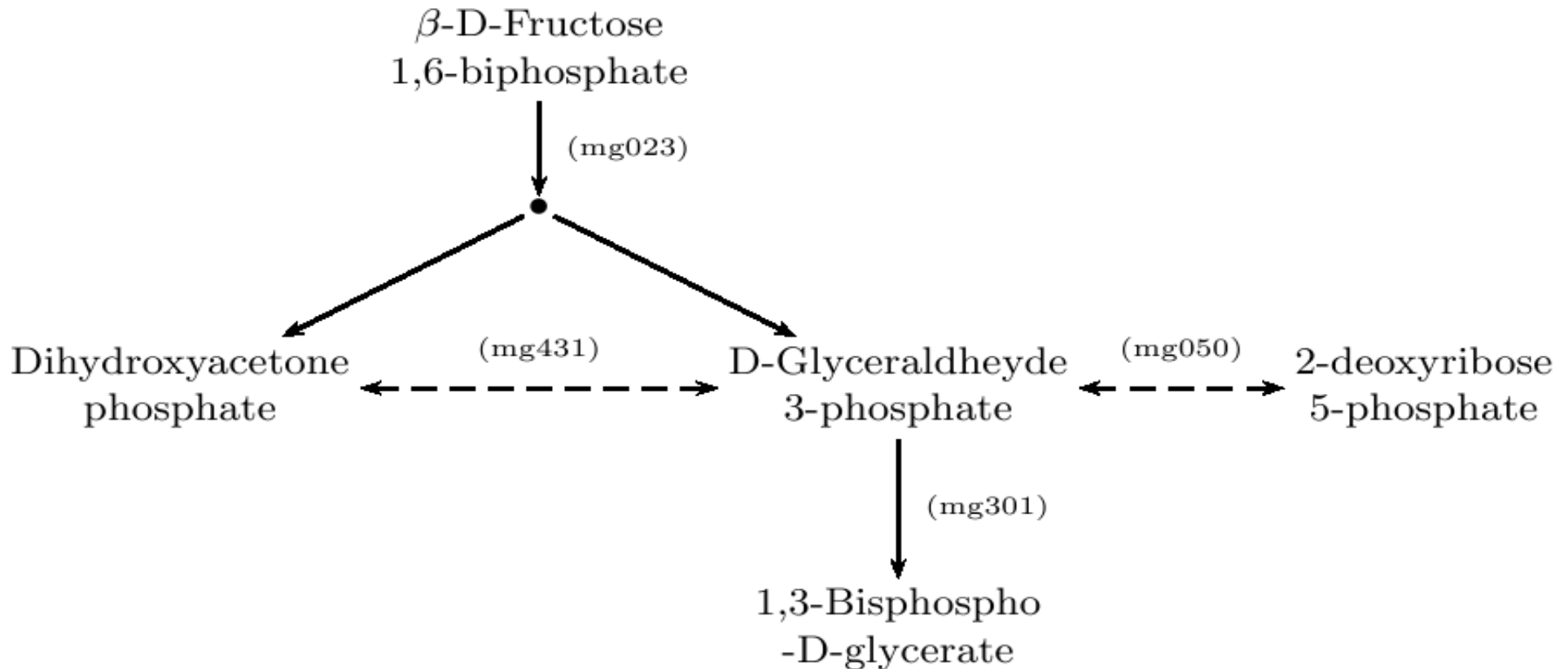


mg023  
is an  
enzyme

reaction rates are on the base of: “Michaelis-Menten” kinetics.

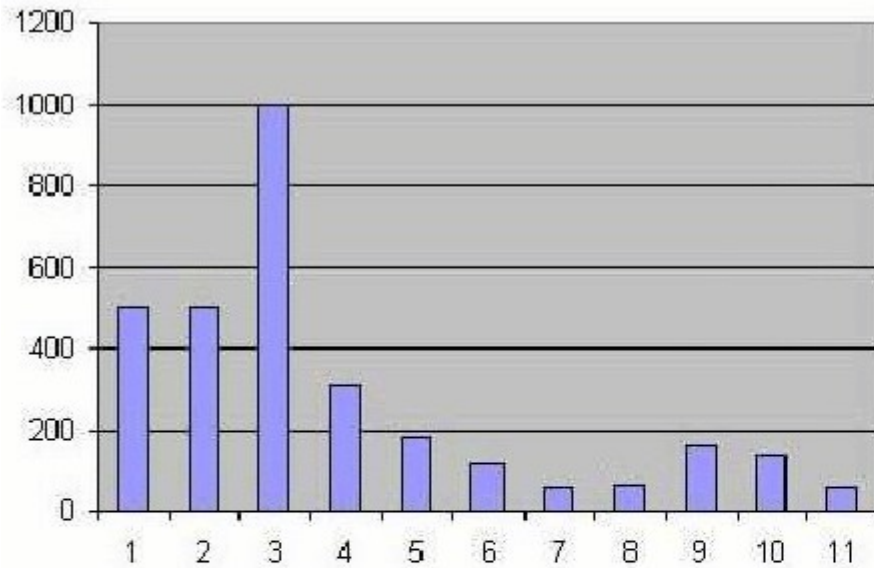
*quantities of the reactants, effects due to the context operators.*

# VICE: $\pi$ -calculus at the metabolic level

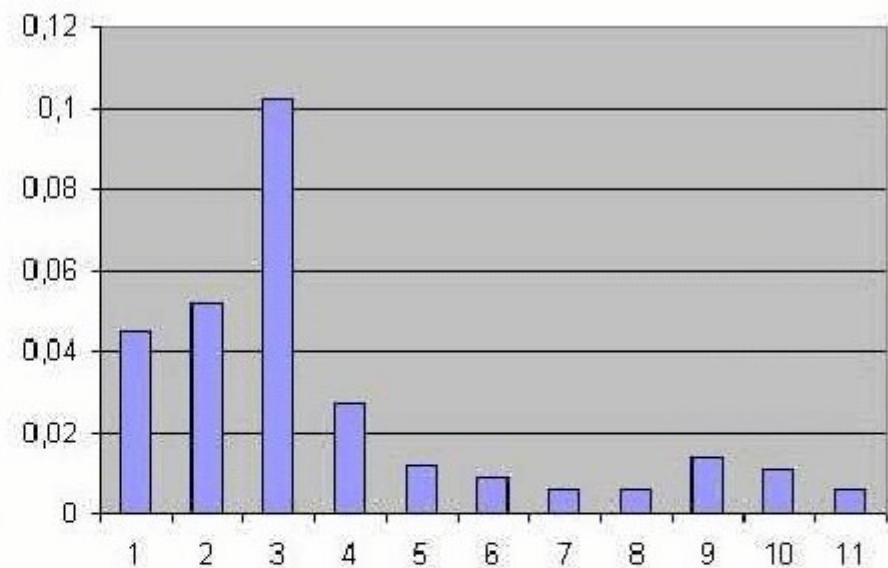


Portion of the Glycolysis Pathway

# VICE: $\pi$ -calculus at the metabolic level



(a)



(b)


1 mg111	5 mg300	9 compl. pyr. dehydrogenase
2 mg215	6 mg430	10 mg299
3 mg023	7 mg407	11 mg357
4 mg031	8 mg216	

# Plan



- Stochastic aspects
  - Stochastic approaches
    - Simulation
    - Steady state analysis
- Level of abstractions and process algebras
- Logical approaches
  - Probabilistic model checking
  - Model checking and learning rules
  - Pathway Logic
- Static approaches

# Probabilistic Model Checking (PRISM)



```
module A
  a : [0..1] init 1;

  [bind] a=1 → r1 : (a'=0);
  [rel]   a=0 → r2 : (a'=1);
  []     a=1 → r3 : (a'=0);
endmodule
```

Stochastic rates taken  
from literature

Future work: the comparison with a  
stochastic  $\pi$ -calculus model !

*What is the probability that the protein A is bound to the protein B at time instant T? ( $\mathcal{P}_{=?}[\text{true } \mathcal{U}^{[T,T]} ab=1]$ );*

*What is the probability that the protein A degrades before binding to the protein B? ( $\mathcal{P}_{=?}[ab=0 \mathcal{U} (a=0 \wedge ab=0)]$ );*

Heath et al. *Probabilistic model checking of complex biological pathway*, 2006.

# Probabilistic Model Checking (PRISM)



```
const int N = 3;
const double R = 1/N;

module RAF1Process
  RAF1: [0..N] init N;
  [bind] (RAF1>0) -> RAF1*R: (RAF1' = RAF1 - 1);
endmodule

module RKIPProcess
  RKIP: [0..N] init N;
  [bind] (RKIP>0) -> RKIP*R: (RKIP' = RKIP - 1);
endmodule
```

The ERK pathway, the same modeled with PEPA – no comparisons are made.

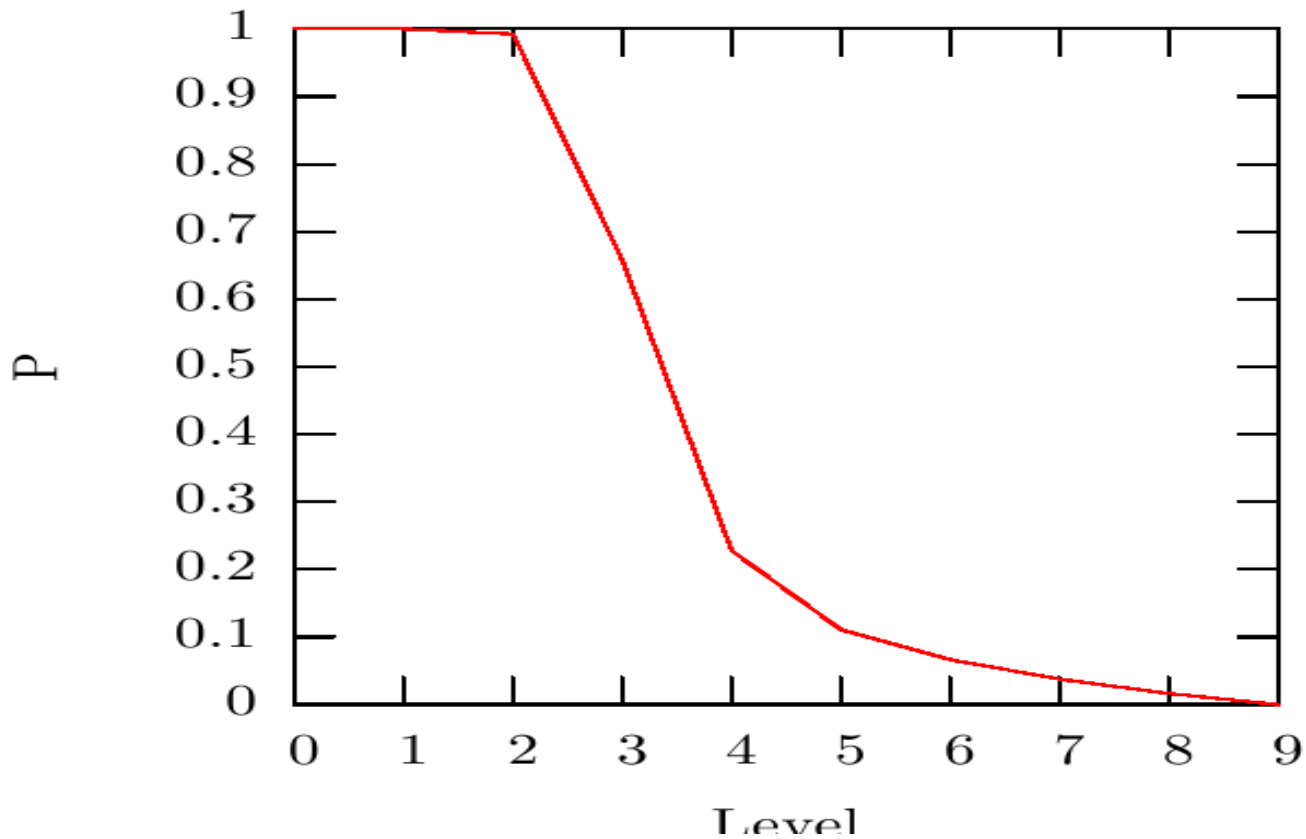
stochastic rates depend on number of molecules

In the steady state, the probability of that condition at varying the parameter  $C$

$$S_{=?}[(RAF1 \geq C - 1) \wedge (RAF1 \leq C + 1)].$$

The results are compared with a ODE system: continuous behaviour vs. stochastic behaviour

# Probabilistic Model Checking (PRISM)



stability of Raf-1\* in steady state

Calder et al. *Analysis of signalling pathways using the PRISM model checker*, 2005.

# BIOCHAM: Model Checking and learning rules 1/3



## Biochemical Abstract Machine (first, qualitative approach)

*Example of syntax*

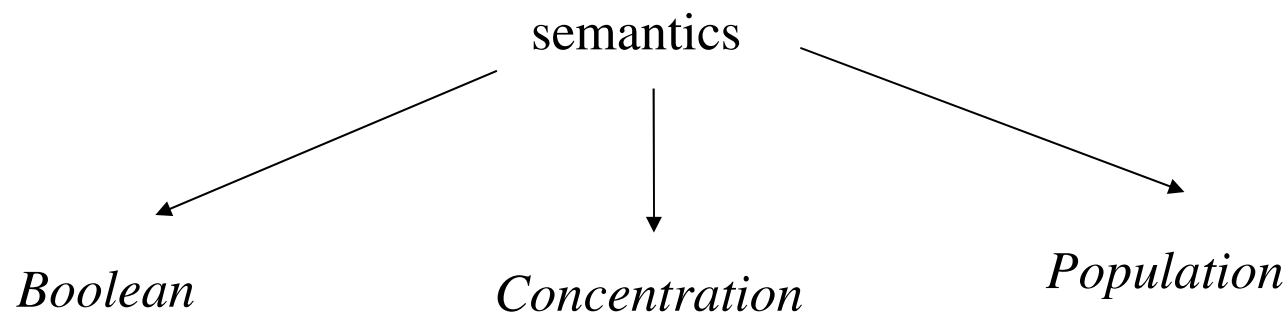
$A + C \Rightarrow C + B$       catalyzed reaction

$A \Leftrightarrow B$                       reversible reaction

$A \Rightarrow \_$                               protein degradation

$A \sim \{p\}$                               denotes a phosphorylated form of the compound A

$A:: \text{nucleus} \Rightarrow A:: \text{cytoplasm}$     for the transport of A from nucleus to cytoplasm



Fages et al. *Machine Learning Biochemical Networks from Temporal Logic Constraints*, 2005

# BIOCHAM: semantics 2/3

## Boolean semantics

- boolean variables represent the presence or the absence of objects;
- non-determinist semantics  $A + B \Rightarrow C + D$  (collects 4 possible reactions)
- based on a Kripke structure  $K=(S,R)$ ,  $S$  set of states,  $R \subseteq S \times S$ ;

performance  
problems

## Concentration semantics

- reaction rules are interpreted by a set of ODE

$$E = \{ e_i \text{ for } S_i \Rightarrow S'_i \}_{i=1..n} \quad dx_k/dt = \sum_{i=1}^n r_i(x_k) * e_i - \sum_{j=1}^n l_j(x_i) * e_j$$

- the evolution of the system is deterministic, is computed by numerical integration algorithms, result is a time serie describyng the temporal evolution of the system;

## Population semantics

- rules describe CTMC; at each step  $N$ (number of object) and  $\tau_i$  (transition rate) following

[Gibson et al. *Probabilistic Model of a Prokaryotic gene and its regulation*, 2000]

Pisa, July 21, 2006

# BIOCHAM: query language 3/3



**Boolean semantics** CTL `reachable(P)` as  $EF(P)$ ; `steady(P)` as  $EG(P)$ ; `stable(P)` as  $AG(P)$ ;  
`checkpoint(Q,P)` as  $!E(!Q \cup P)$ ; `oscill(P)` as ... ; `loop(P)` as ...

**Concentration semantics** LTL deals with numerical quantities

$F([A] > 10)$

$G([A] + [B] < [C])$

**Population semantics** PCTL (PLTL is really implemented for efficiency problems)

$A(\psi \cup \psi')$  becomes  $P_{\geq 1}(\psi \cup \psi')$

---

applying learning techniques

commands as

**`learn_one_deletion(reaction_pattern,spec_CTL)`**

initial model can be modified to satisfy the given formula

it only works for CTL and TLT (for computational problems)

# Pathway logic (just an example..)



- Based on rewrite logic
- Models with different levels of detail
- Dynamically generated pathways using search and model-checking
- Transformation to Petri nets for analysis and visualization

Example of a rewrite rule:

```
{CM | cm:Soup {cyto:Soup [Erk1 - act] {NM | nm:Soup {nuc:Soup}}}}  
=>  
{CM | cm:Soup {cyto:Soup {NM | nm:Soup {nuc:Soup [Erk1 -  
act]}}}} .
```

- System partially specified: *Soup* is a variable!
- Each variables has a type for being matched with the right components.
- Compartments, membranes, active sites of proteins.

# Calculus of looping sequences (rewrite rules)



Syntax ::=  $a \mid \varepsilon \mid T \cdot T \mid (T)^L \mid T|T \mid T \parallel T$

example of terms:  $(a \cdot b \cdot c)^L \quad ((c \cdot d \cdot e)^L \cdot a \cdot b)^L$

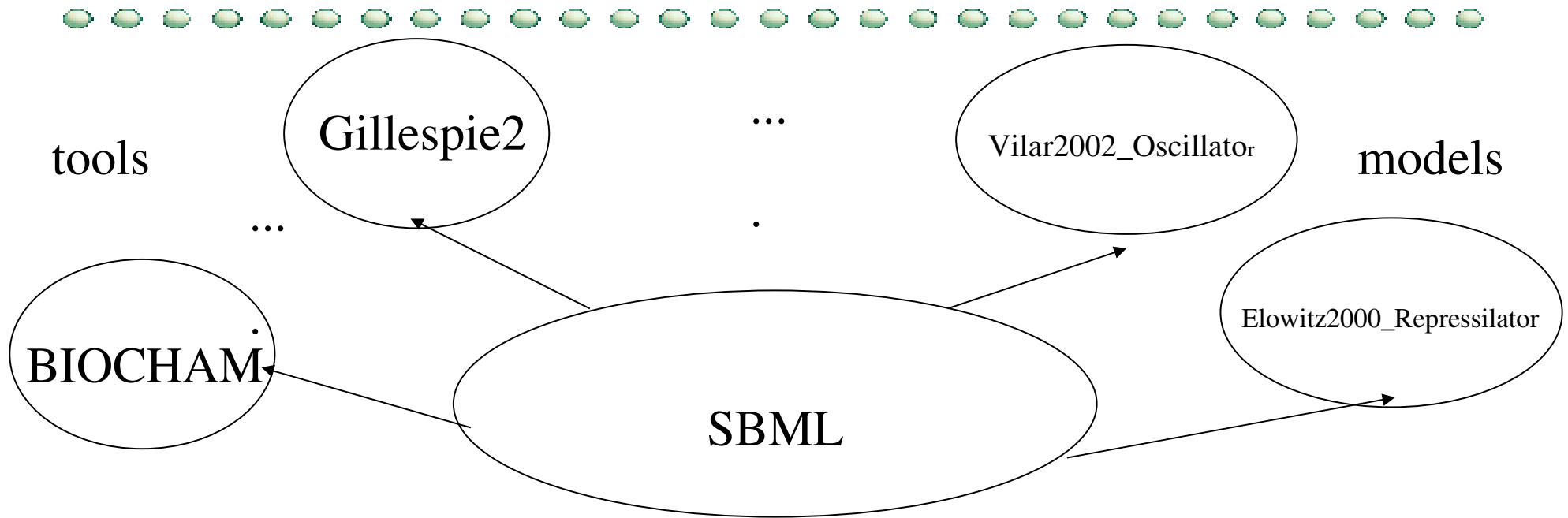
Membrane  $(a \cdot b \cdot c)$

Containment operator  $(m \cdot \dots \cdot m) \mid \text{DNA}$

Reaction Semantics (an example)

$(m \cdot \dots \cdot m)^L \mid (\text{DNA}_b \mid X) \longrightarrow (m \cdot \dots \cdot m)^L \mid (\text{DNA}_b \mid \text{DNA}_{b_l} \mid X) \quad [\text{occ}(\text{DNA}_b, X) = 0]$

# Unifying different views

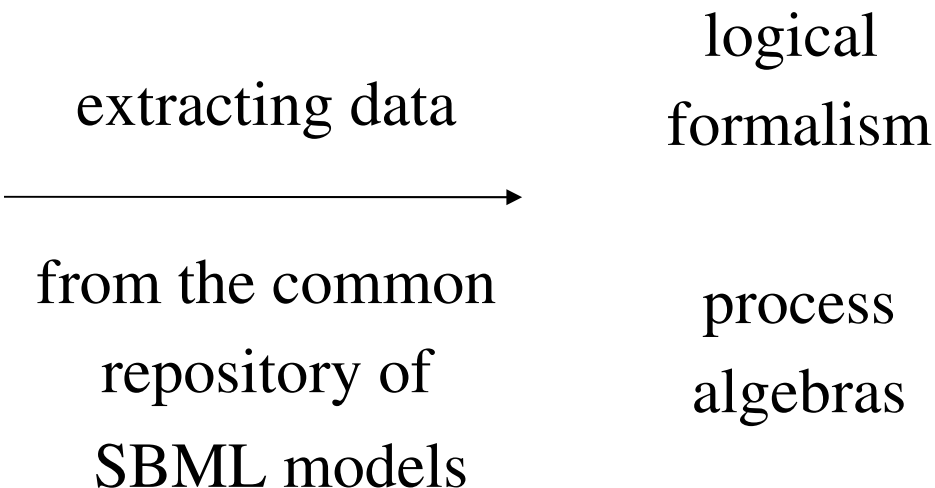


- Biological models, useful numerical information
- Comparing result with working tools

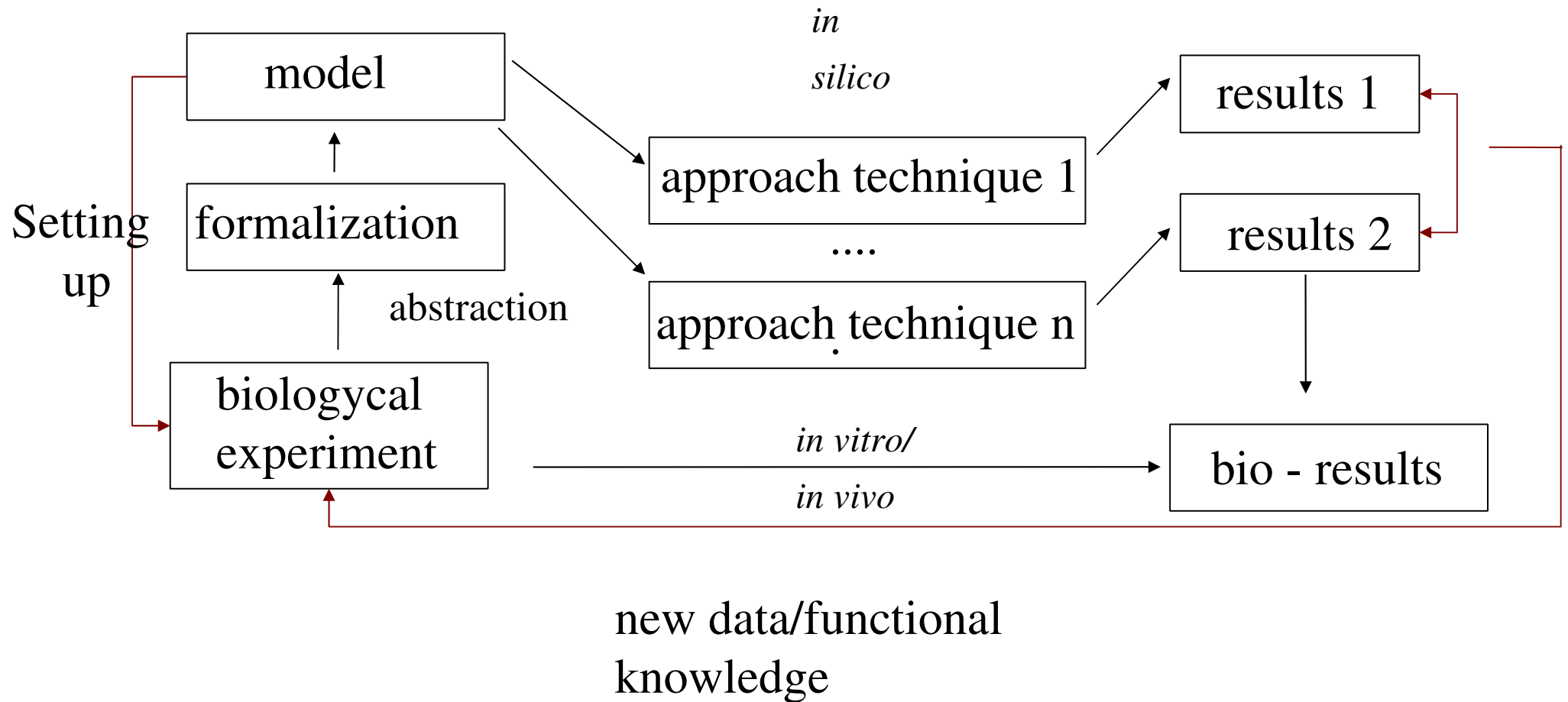
# Some works in this direction



Rate constant	Reaction
k1 = 1	X -> EmptySet
k1 = 1	Y -> EmptySet
k1 = 1	Z -> EmptySet
K = 1 (hill khalf)	PX l-> Y
K = 1 (hill khalf)	PY l-> Z
K = 1 (hill khalf)	PZ l-> X
n = 2.1 (hill nhill)	PX l-> Y
n = 2.1 (hill nhill)	PY l-> Z
n = 2.1 (hill nhill)	PZ l-> X
{alpha0, alpha} = {0, 250} (hill basalRate)	PX l-> Y
{alpha0, alpha} = {0, 250} (hill basalRate)	PY l-> Z
{alpha0, alpha} = {0, 250} (hill basalRate)	PZ l-> X
alpha1 = 0 (hill vmax)	PX l-> Y
alpha1 = 0 (hill vmax)	PY l-> Z
alpha1 = 0 (hill vmax)	PZ l-> X
beta = 5	PX -> EmptySet
beta = 5	PY -> EmptySet
beta = 5	PZ -> EmptySet
beta = 5	X + EmptySet -> PX + X
beta = 5	Y + EmptySet -> PY + Y
beta = 5	Z + EmptySet -> PZ + Z



# Modeling, analysing, querying



# Statical approaches: Control Flow Analysis 1/2



For Bioambients an over-approximation about:

nesting of ambients  $I \subseteq \text{Ambient} \times (\text{Ambient} \cup \text{Cap})$

relevant name bindings  $\mathcal{R} \subseteq \text{Names} \times \text{Names}$

judgement

$(I, \mathcal{R}) \quad *P$

the process  $P$ , within ambient  $*$ , evolves with respect to  $(I, \mathcal{R})$

# Statical approaches: Control Flow Analysis (2/2)

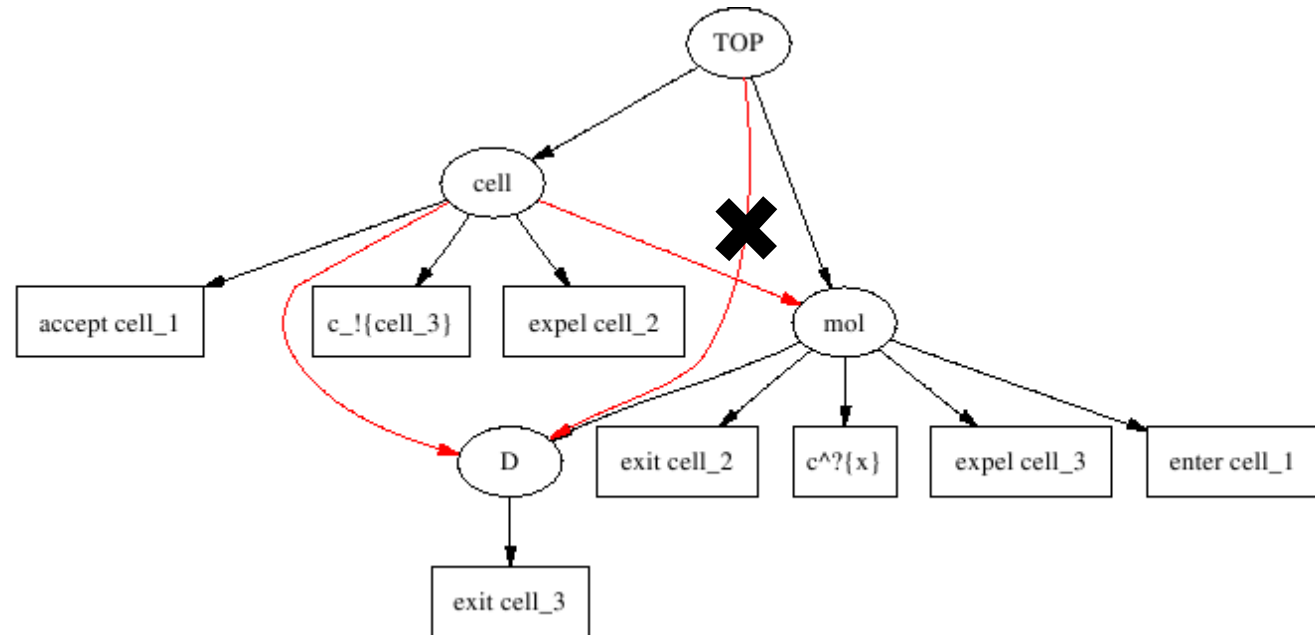


## Context-depending analysis

Focusing on pathway (work in progress)

father-son relationship for *exposed ambient* and *capabilities*

*origin* and *target* configurati  
for a transition



Nielson&Nielson et al. *Control Flow Analysis for Bioambients*, 2003.  
Nielson&Nielson et al. *Static Analysis for Systems Biology*, 2004

# Statical approaches: Type inference 1/3



Three type systems for:

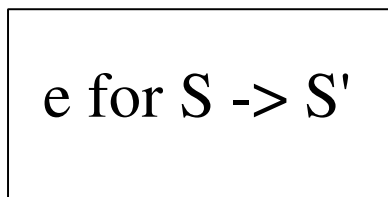
- protein functions
- activation/inhibition effect (in a reaction model)
- location topology

*abstract  
interpretation*

*concrete domain  
C*

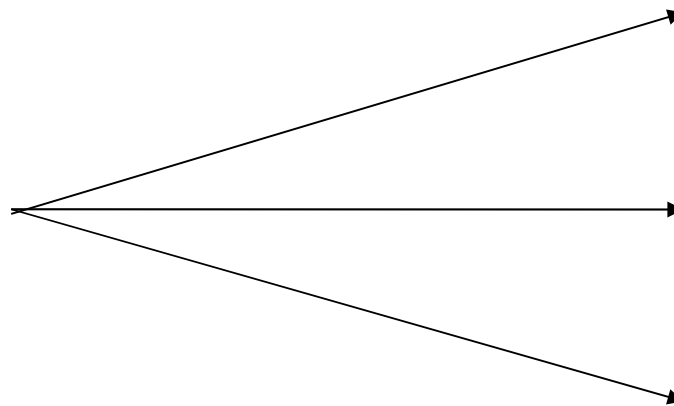
*abstract domains  
D*

Reaction set:



SBML-BIOCHAM  
conventions

*Galois connections*



kinase / phosphatase:

$\{k : \text{Mol} \rightarrow \{\text{true}, \text{false}\}\} \cup$   
 $\{\text{ph} : \text{Mol} \rightarrow \{\text{true}, \text{false}\}\}$

$\mathcal{P}(\{A \text{ activates } B \mid A, B \in \text{Mol}\}) \cup$

$\mathcal{P}(\{A \text{ inhibits } B \mid A, B \in \text{Mol}\})$

topology

Mol x Mol

# Statical approaches: Type inference (2/3)



*Abstract interpretation* is not strictly required

formal framework that allows multiple abstraction levels

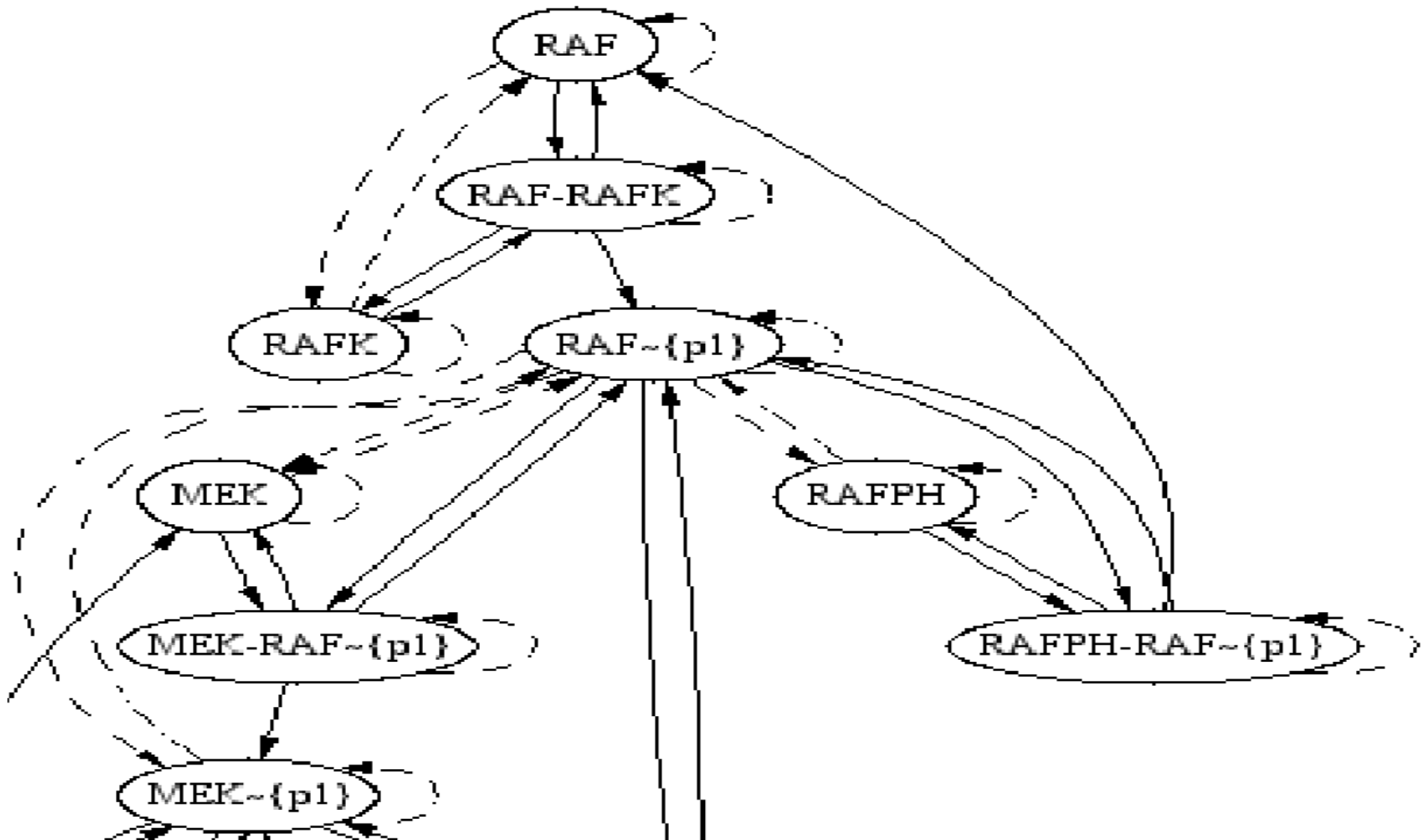
For each type system:

- type inference: given  $x \in C$ , compute  $\alpha(x)$
- type checking: given  $x \in C$  and a typing  $y \in A$ , determine  $x \sqsubseteq_C \mathcal{Y}(y)$

Type system for activation inhibitory influences:

$$\alpha(A + C = C + B) = \{C \text{ inhibits } A, A \text{ inhibits } A, A \text{ activates } B, \\ C \text{ activates } B\}$$

# Statical approaches: Type inference (3/3)



# Static approaches



- Formalization of biological meaningful properties
- Studying the impact that these properties have at “run-time”
- Reviewing language primitives for catching the very basic interactions

# Conclusions



- **A lot to find out** (Find all genes whose expression is directly or indirectly affected by a given compound. Show which pathways may be affected when one or more gene/proteins are turned off or missing.)
- **Connecting levels of abstractions**
- **Working on the same models for comparing results**
- **Connecting methodologies:**
  - comparing results;
  - labelled transition system for inspecting “biologically relevant” properties;
  - testing biological properties (membrane nesting – molecular neighbourhood).