Continuous Valuations of Temporal Logic Specifications with applications to Parameter Optimization and Robustness Measures

François Fages

Inria Saclay,

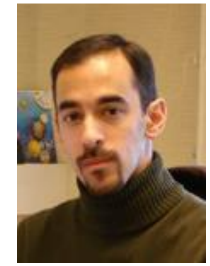
EPI Lifeware (formerly EPI Contraintes)

France

http://lifeware.inria.fr/

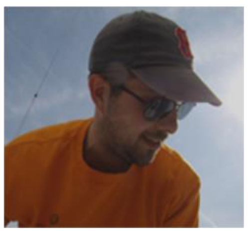
Implemented in the Biochemical Abstract Machine (BIOCHAM v3.7 next v4.0)

Lifeware Group

Research Scientists

Grégory Batt (Inria CR1, co-head)
François Fages (Inria DR1, head)
Jakob Ruess (Inria CR2, sept. 2016)
Sylvain Soliman (Inria CR1)




Associates

Pascal Hersen (CNRS MSC lab, Researcher)
Denis Thieffry (ENS Paris, Professor)
Thierry Martinez (SED, Inria Paris)

PostDoc

Chiara Fracassi (Inria, CNRS MSC lab)


PhD students

François Bertaux (AMX, Ecole Polytechnique, with EPI Bang)
Katherine Chiang (National Taiwan University)
Xavier Duportet (Inria, MIT, startup Pasteur Institute)
Steven Gay (Inria, Post doc Univ. Louvain la Neuve, Belgium)
Artemis Llamosi (CNRS MSC lab, Inria)
Jean-Baptiste Lugagne (Inria, CORDI-C, CNRS MSC lab)
Jonas Sénizergues (Inria)
Pauline Traynard (Inria, Ecole Polytechnique, with ENS)


Lifeware: hardware-software of the living

- How to compute with biochemical reactions ?
  - Analog/digital computation
  - Compositionality and robustness of biochemical circuits
  - Programming artificial vesicles - Reprogramming living cells
- How to analyze natural cell processes as programs ?
  - Cell signaling, cell cycle, circadian clock, gene expression, …
  - Temporal logic specification of the behaviour, parameter inference, robustness
  - Beyond describing, understanding natural circuits and their evolution
- How to control cell processes?
  - Microfluidic platform in an image analysis-model calibration-action loop
  - Optimal experimental design
- How to reason with cell populations ?
  - Cell-to-cell variability analysis and control
  - Model of extrinsic/intrinsic noise

How to Compute with Biochemical Reactions ?

- Binding, complexation: $A + B \rightarrow C$

  $cdk1 + cycB \rightarrow cdk1cycB$

- Unbinding, decomplexation: $A \rightarrow B + C$

- Transformation, phosphorylation, transport: $A \rightarrow B$ $(A + E \rightarrow C \rightarrow B + E)$

  $cdk1cycB \rightarrow cdk1cycBp$

- Gene expression, synthesis: $A \rightarrow A + B$

  $E2Fa \rightarrow E2Fa + RNAcycA$

- Degradation: $A \rightarrow \_$

How to Compute with Biochemical Reactions ?

- Binding, complexation: $A + B \xrightarrow{k.A.B} C$

  $cdk1 + cycB \rightarrow cdk1cycB$

Time matters

- Unbinding, decomplexation: $A \xrightarrow{k.A} B + C$

- Transformation, phosphorylation, transport: $A \xrightarrow{k.A} B$ or $A \xrightarrow{v.A/(k+A)} B$

  $cdk1cycB \rightarrow cdk1cycBp$

- Gene expression, synthesis: $A \xrightarrow{v.A^n/(k+A^n)} A + B$

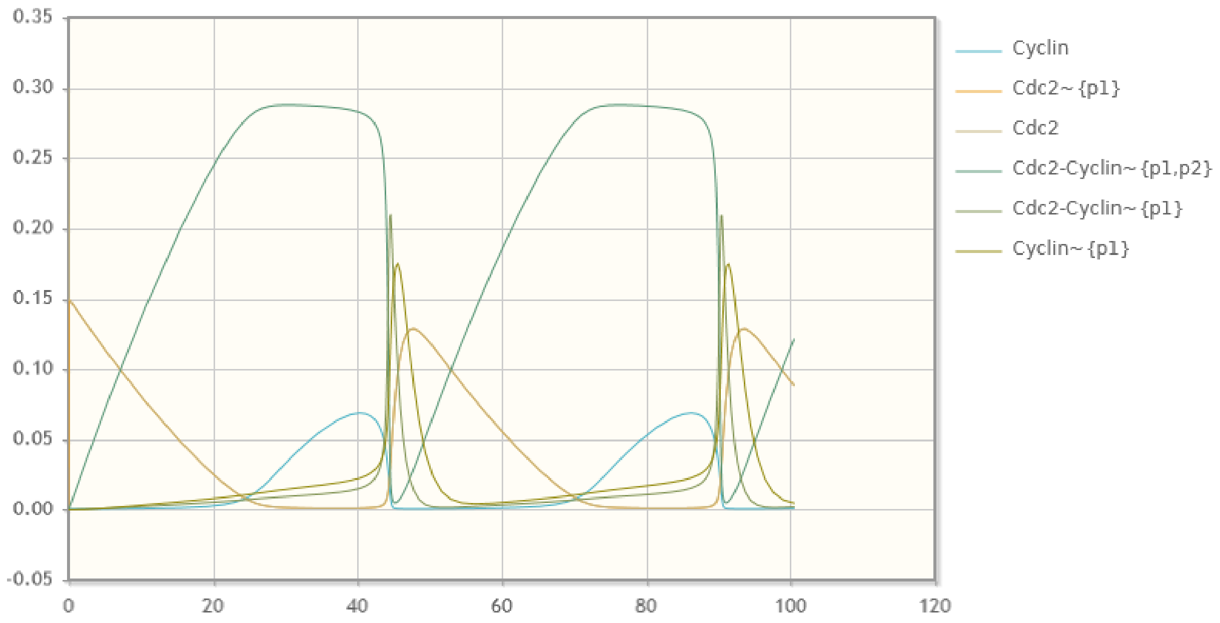
  $E2Fa \rightarrow E2Fa + RNAcycA$

- Degradation: $A \xrightarrow{k.A} \_$ or $A \xrightarrow{v.A/(k+A)} \_$

Semantics of Reactions $A+B \xrightarrow{f(A,B)} C$

Continuous semantics: concentrations, continuous time evolution

Ordinary differential equations (ODE) $\frac{dAi}{dt} = \sum_{r=1}^{n} f_r \times \delta_r(Ai)$



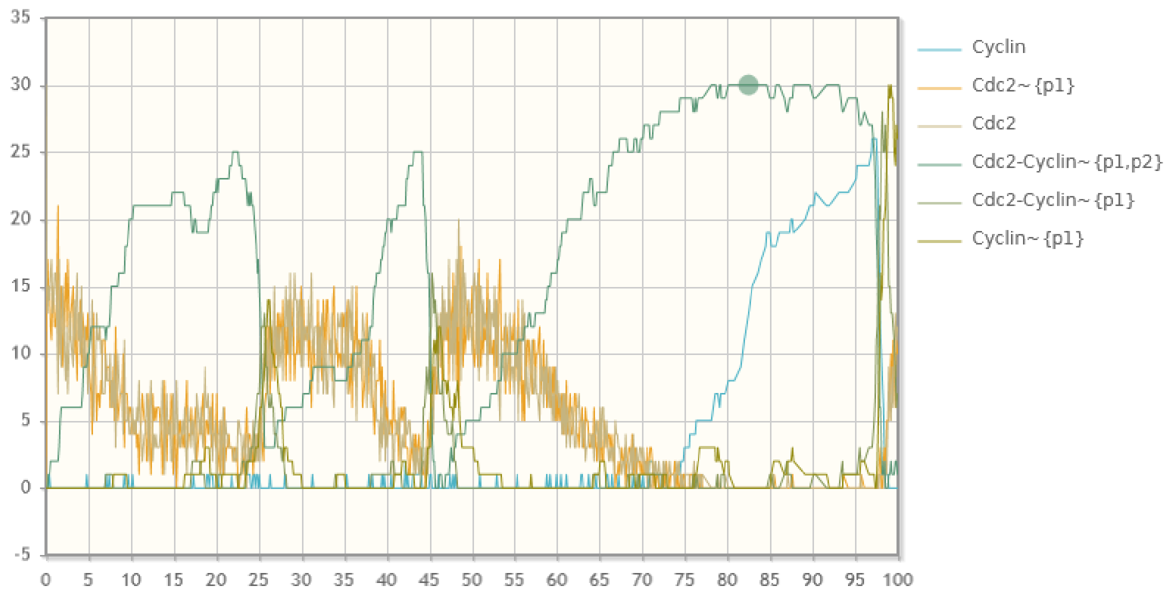
Semantics of Reactions $A+B \xrightarrow{f(A,B)} C$

Continuous semantics: concentrations, continuous time evolution

Ordinary differential equations (ODE) $$\frac{dAi}{dt} = \sum_{r=1}^{n} f_r \times \delta_r(Ai)$$

Stochastic semantics: numbers of molecules, probability and time of transition

Continuous Time Markov Chain (CTMC) $$A, B \xrightarrow{p(S_i),\, t(Si)} C{+}{+}, A{-}{-}, B{-}{-}$$



Semantics of Reactions $A+B \xrightarrow{f(A,B)} C$

Continuous semantics: concentrations, continuous time evolution

Ordinary differential equations (ODE) $\frac{dAi}{dt} = \sum_{r=1}^{n} f_r \times \delta_r(Ai)$

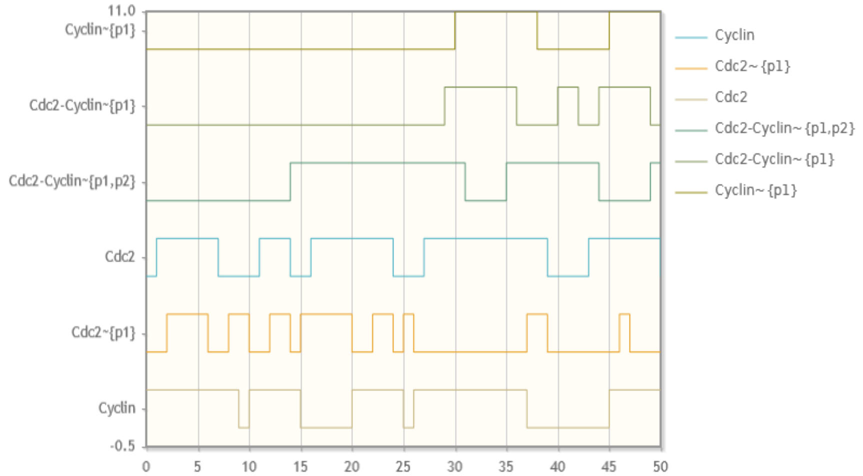
Stochastic semantics: numbers of molecules, probability and time of transition

Continuous Time Markov Chain (CTMC) $A , B \xrightarrow{p(S_i),\, t(Si)} C{+}{+}, A{-}{-}, B{-}{-}$

Petri net semantics: numbers of molecules $A , B \rightarrow C{+}{+}, A{-}{-}, B{-}{-}$

Multiset rewriting

CHAM [Berry Boudol 90] [Banatre Le Metayer 86]



Semantics of Reactions $A+B \xrightarrow{f(A,B)} C$

Continuous semantics: concentrations, continuous time evolution

Ordinary differential equations (ODE) $\frac{dAi}{dt} = \sum_{r=1}^{n} f_r \times \delta_r(Ai)$

Stochastic semantics: numbers of molecules, probability and time of transition

Continuous Time Markov Chain (CTMC) $A , B \xrightarrow{p(S_i),\, t(Si)} C{+}{+},\ A{-}{-},\ B{-}{-}$

Petri net semantics: numbers of molecules $A , B \rightarrow C{+}{+},\ A{-}{-},\ B{-}{-}$

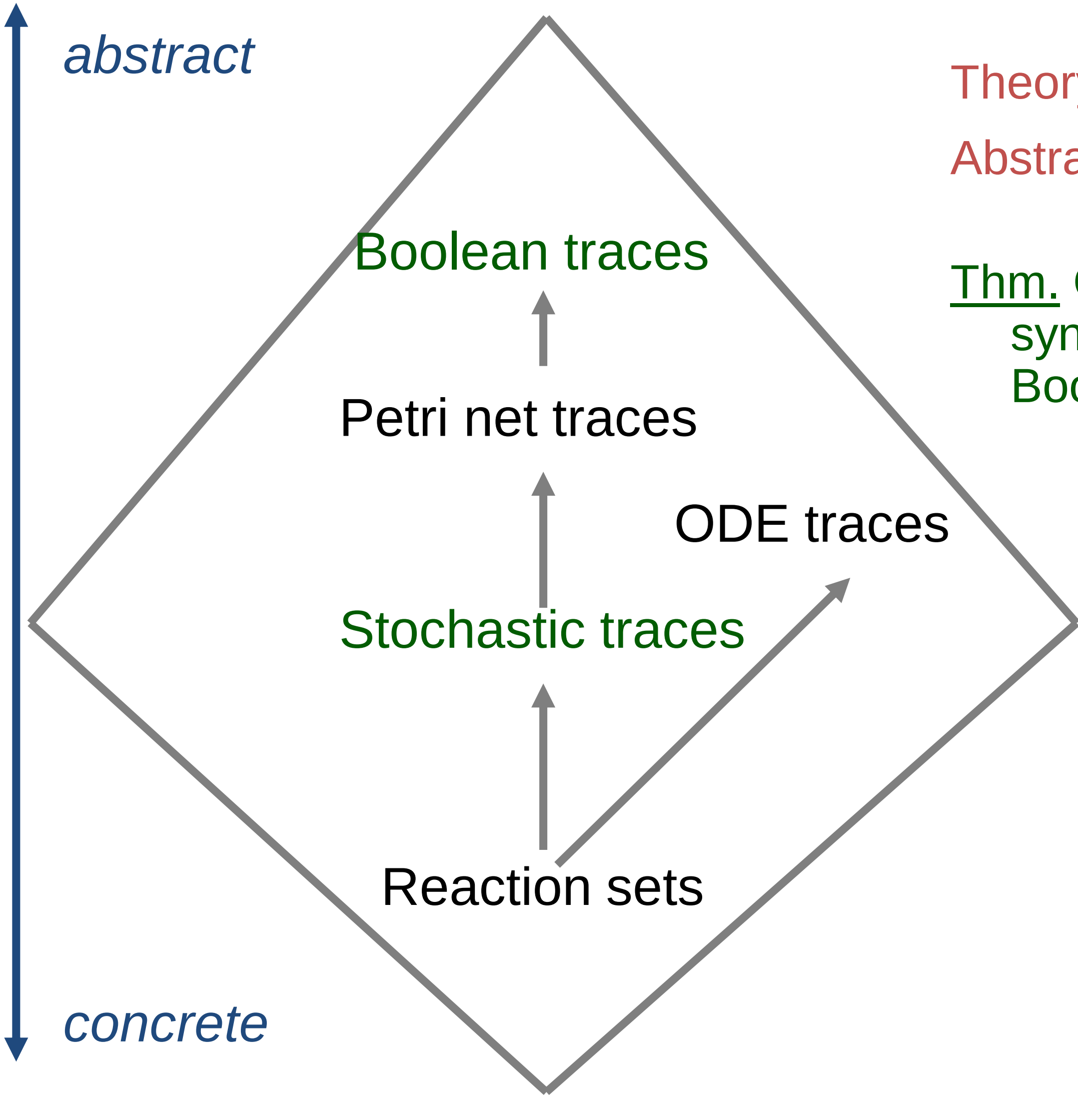
Multiset rewriting

CHAM [Berry Boudol 90] [Banatre Le Metayer 86]

Boolean semantics: presence/absence $A \wedge B \rightarrow C \wedge A/\neg A \wedge B/\neg B$

Asynchronous transition system

Abstraction Relationships



Theory of abstract Interpretation

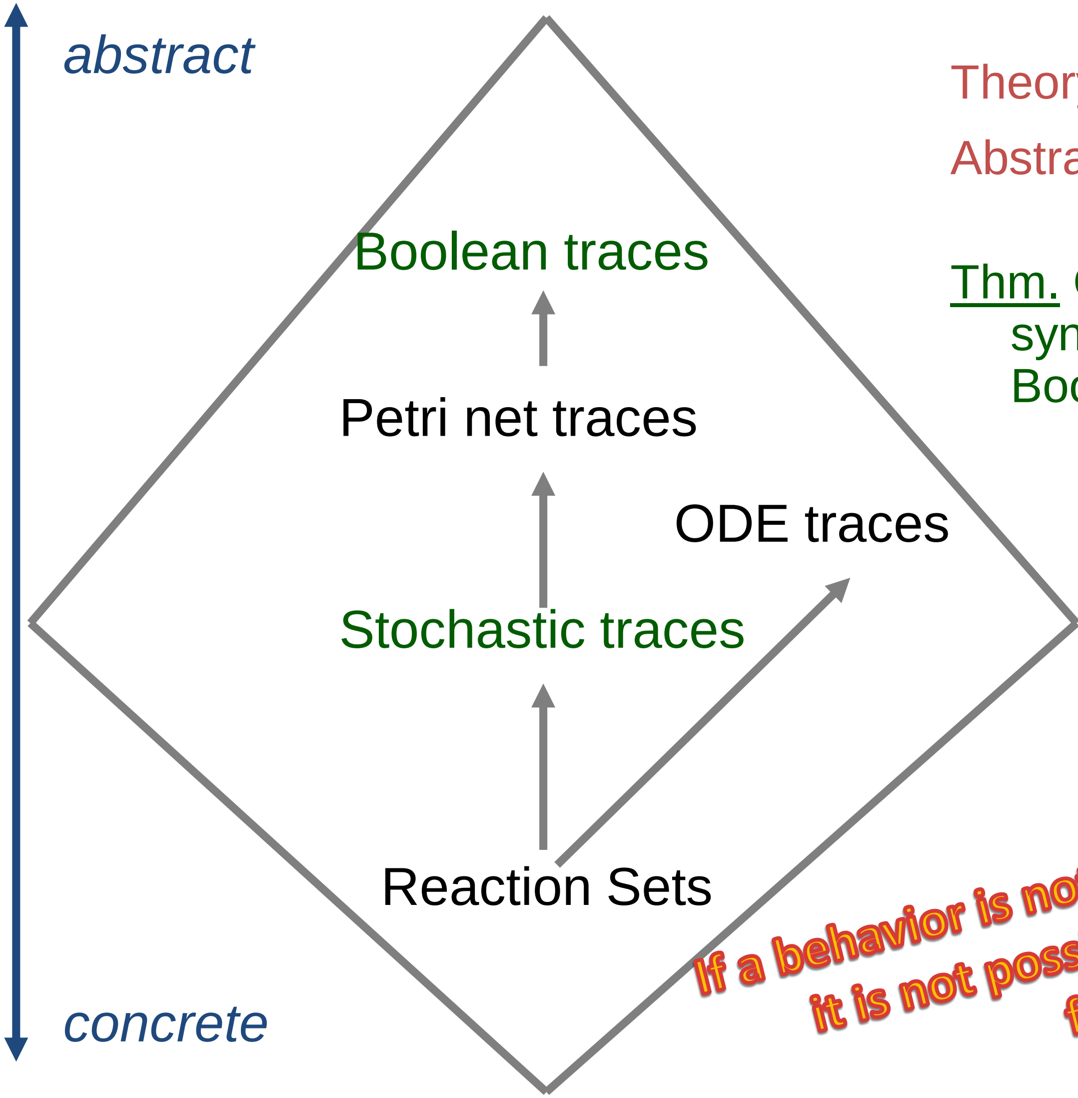
Abstractions as Galois connections

[Cousot Cousot POPL'77]

Thm. Galois connections between the syntactical, stochastic, Petri Net and Boolean trace semantics

[FF Soliman CMSB'06,TCS'08]

Abstraction Relationships



Theory of abstract Interpretation

Abstractions as Galois connections

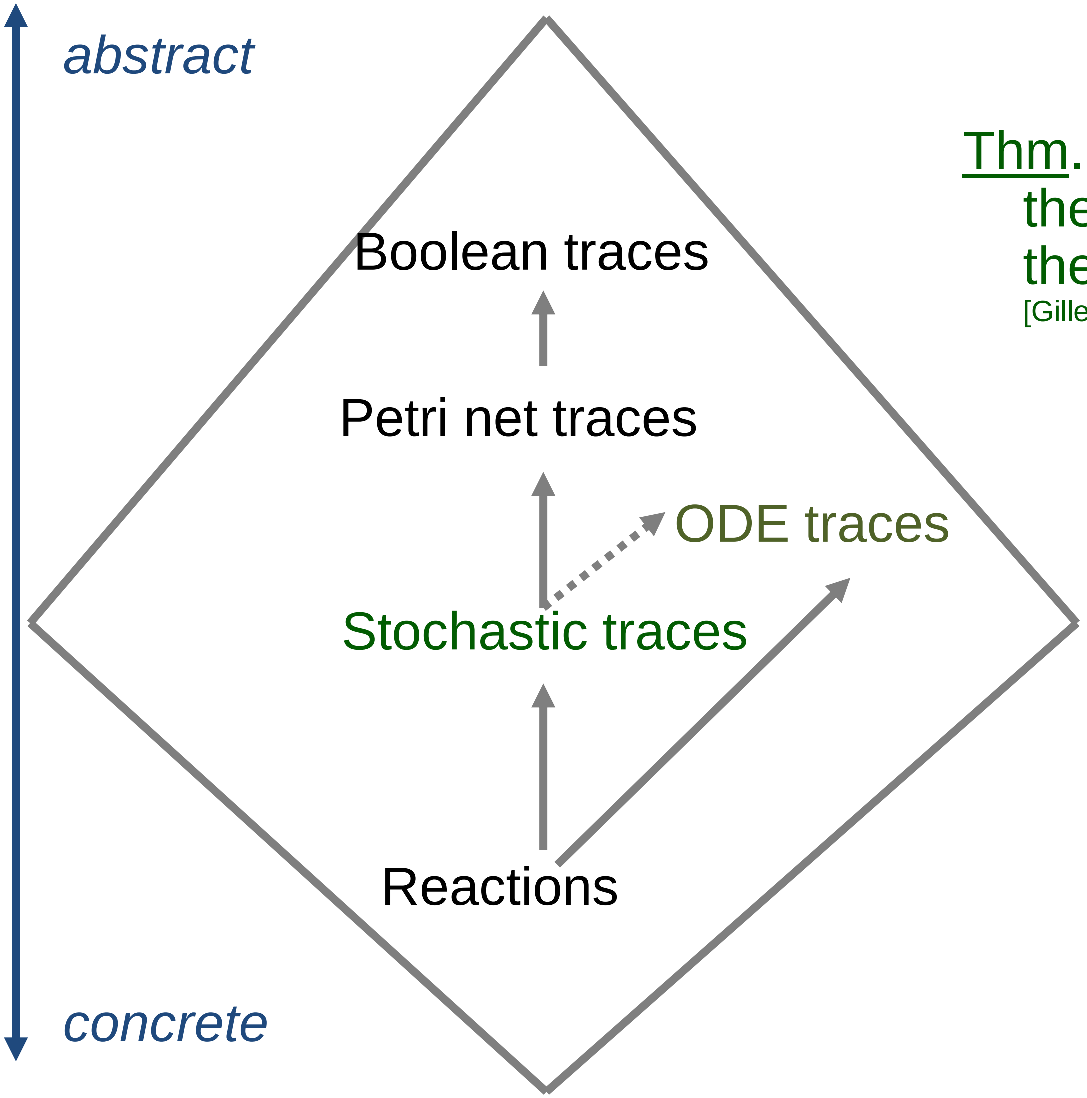
[Cousot Cousot POPL'77]

Thm. Galois connections between the syntactical, stochastic, Petri Net and Boolean trace semantics

[FF Soliman CMSB'06,TCS'08]

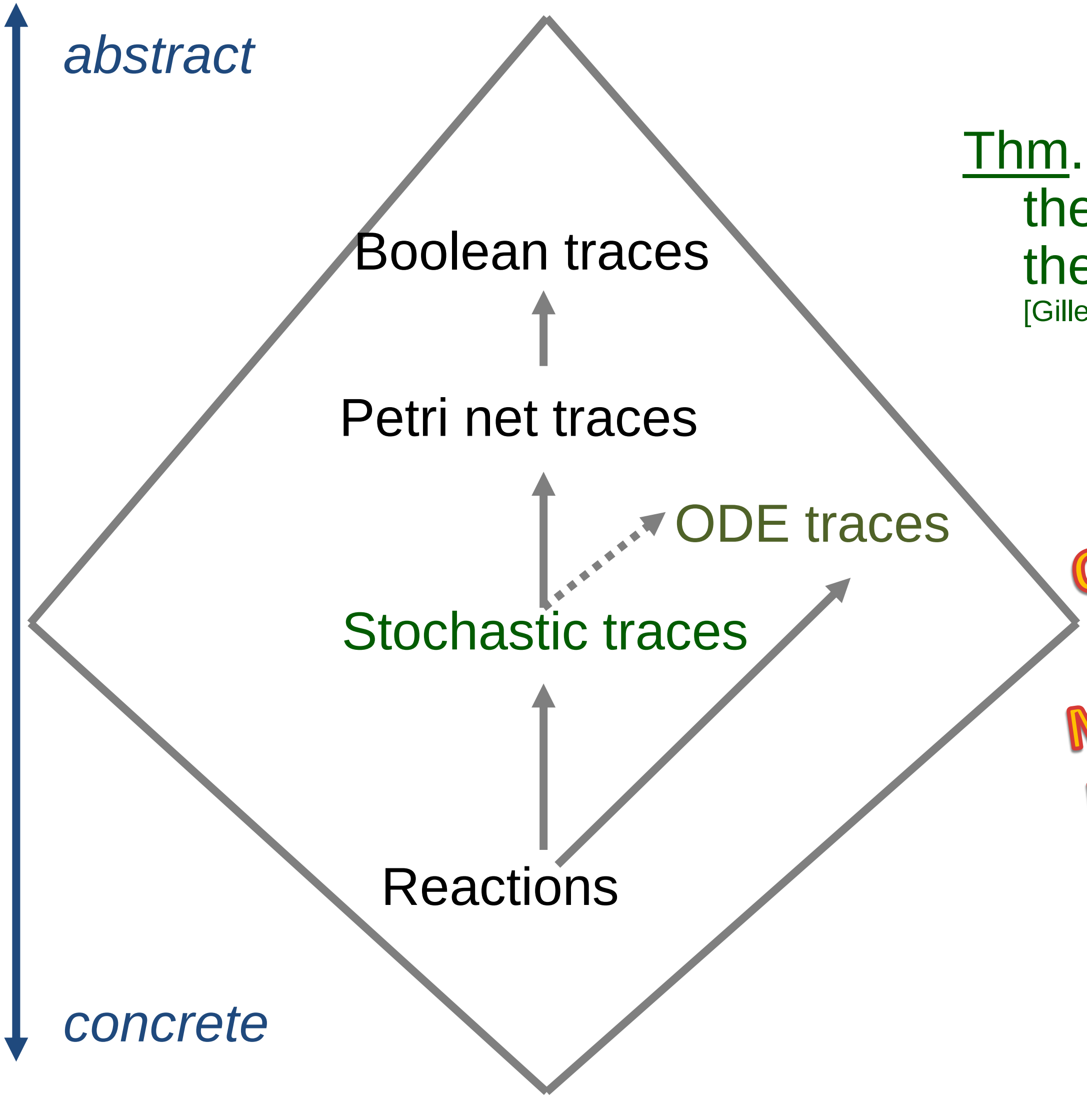
If a behavior is not possible in the Boolean semantics it is not possible in the stochastic semantics for any reaction rates

Abstraction Relationships



<u>Thm</u>. Under large number conditions the ODE semantics approximates the *mean* stochastic behavior [Gillespie 71]

Abstraction Relationships

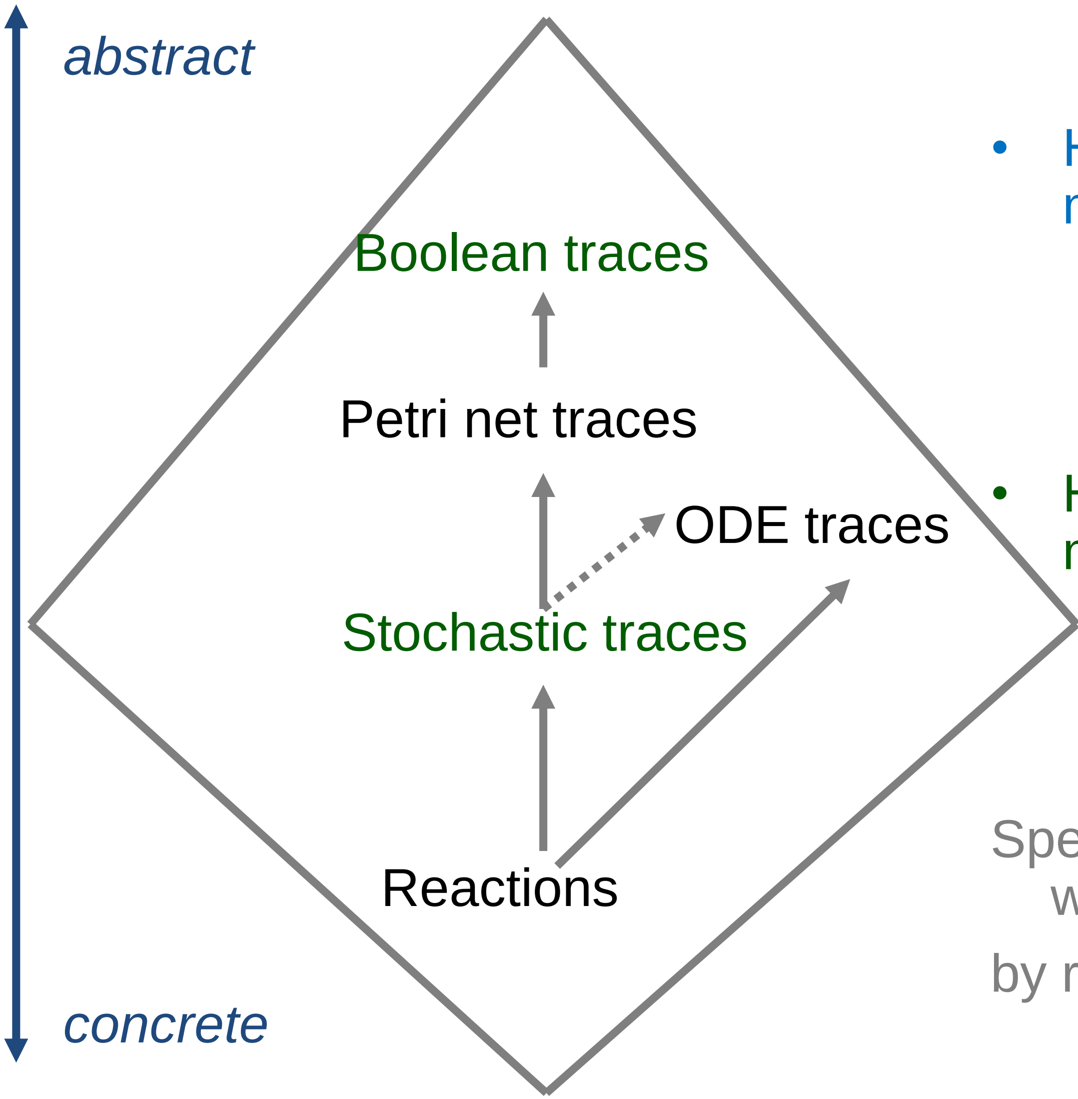


Thm. Under large number conditions the ODE semantics approximates the *mean* stochastic behavior [Gillespie 71]

Hot topic:
higher order moments
ODE for mean, variance,...

Model cell-to-cell variability
Intrinsic and extrinsic noise

Hybrid Models and Hybrid Simulations



- Hybrid Boolean-continuous models (hybrid automata)

  Boolean gene expression + continuous protein activation

- Hybrid stochastic-continuous models (CTMC+ODE)

  Stochastic gene expression + continuous protein activation

Specification of hybrid simulators with *dynamic partitioning*

by reactions+events in SBML

[Chiang FF Huang Soliman 15 ACM TOMACS]

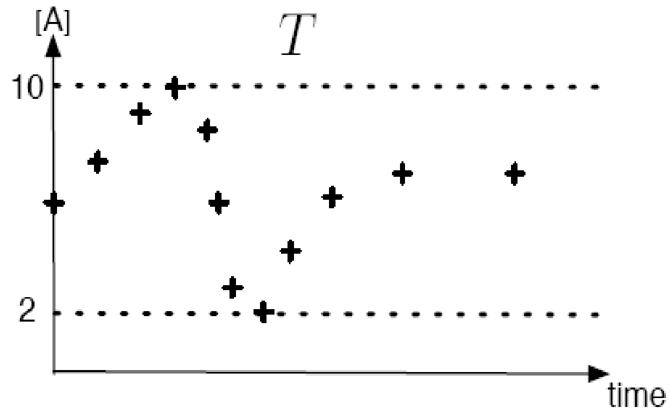
Quantitative Temporal Logic Specifications

- Formalization of (imprecise) behaviors observed experimentally
  - Quantitative temporal logic constraints FO-LTL(Rlin) [A. Rizk 2011 Thesis]
  - Stability **G**φ; Reachability **F**φ , thresholds **F**([A]>0.1),
  - Peaks of concentration ∃V **F**([A]<V ∧ **X**([A]=V ∧ **X**([A]<V))
  - Amplitude, periods and phases as distance between peaks [Traynard Fages Soliman 14 CMSB]

- Model verification
  - Boolean symbolic model-checking [Chabrier Chiaverini Danos FF Schachter 04 TCS]
  - FO-LTL(Rlin) constraint solving [FF Rizk 08 TCS]
  - Continuous satisfaction degree of FO-LTL formulae [Rizk Batt FF Soliman 11 TCS]
  - Parameter sensitivity, robustness measures [Rizk Batt FF Soliman 09 Bioinformatics]

- Model synthesis (parameter inference)
  - Evolutionary search algoritm CMA-ES [Hansen 01] maximize satisfaction degree FO-LTL
  - FO-LTL satisfaction → dynamical model → quantitative predictions, control
  - FO-LTL unsatisfaction → model structure revision → contributions to biology

Model-Checking Generalized to Constraint Solving



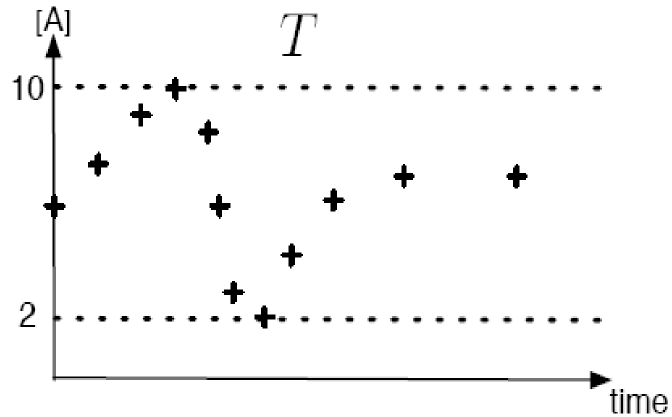
$LTL(\mathbb{R})$

$\Phi = F([A] \geq 7 \wedge F([A] \leq 0))$

Model-checking

the formula is false

Model-Checking Generalized to Constraint Solving



$LTL(\mathbb{R})$

$\Phi = F([A] \geq 7 \wedge F([A] \leq 0))$

Model-checking

the formula is false

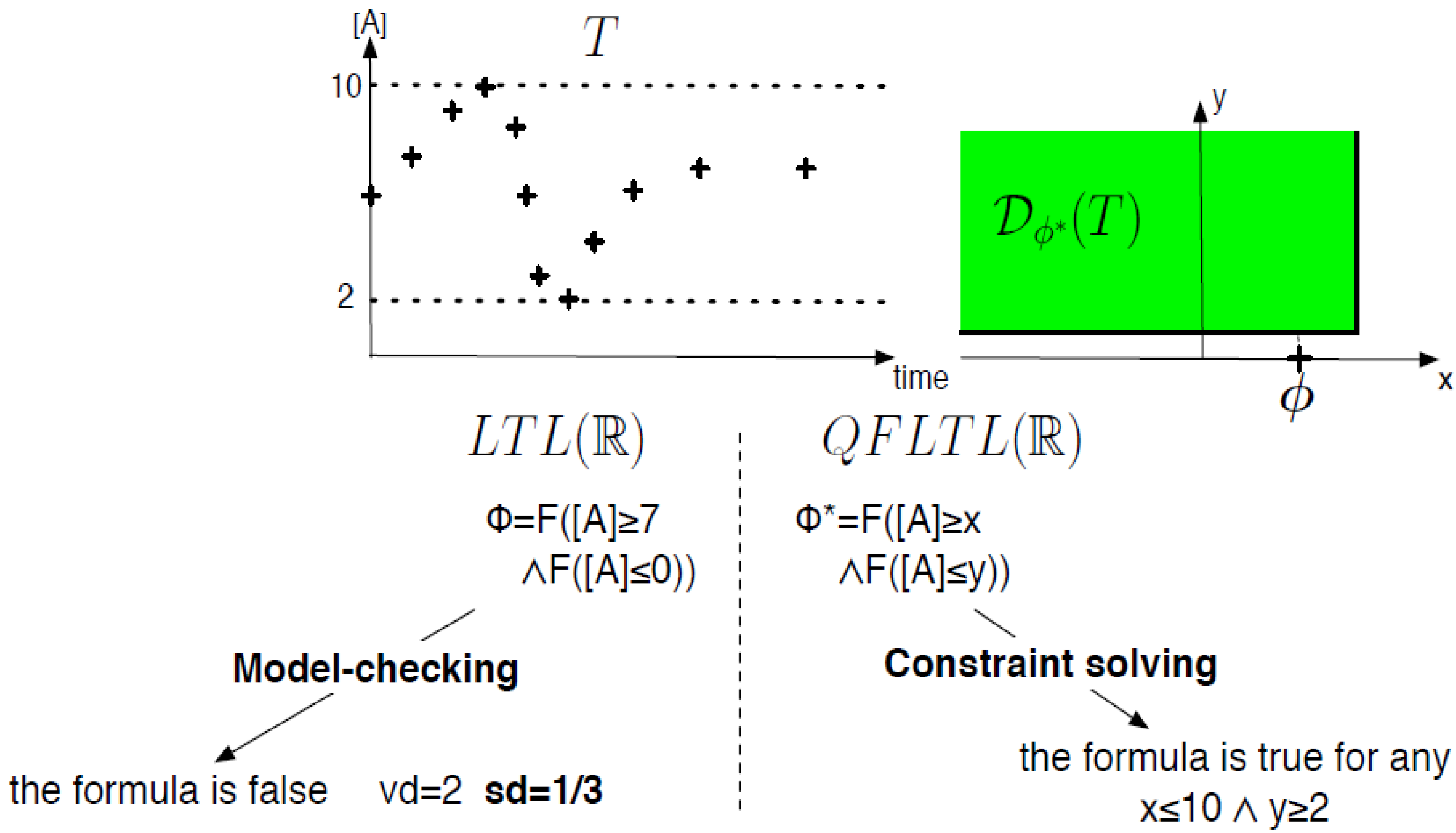
$QFLTL(\mathbb{R})$

$\Phi^* = F([A] \geq x \wedge F([A] \leq y))$

Constraint solving

the formula is true for any $x \leq 10 \wedge y \geq 2$

Model-Checking Generalized to Constraint Solving

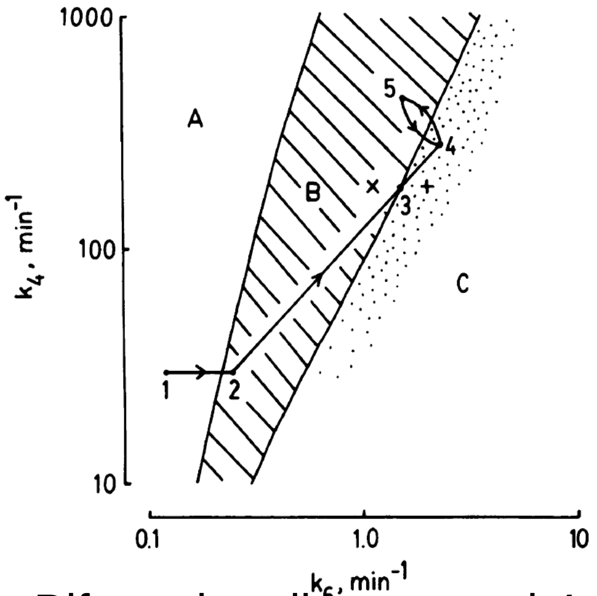


Validity domain $\mathcal{D}_{\phi^*}(T)$ for the **free variables** in ϕ^* [Fages Rizk CMSB'07]

Violation degree $vd(T,\phi) = \text{distance}(val(\phi), \mathcal{D}_{\phi^*}(T))$

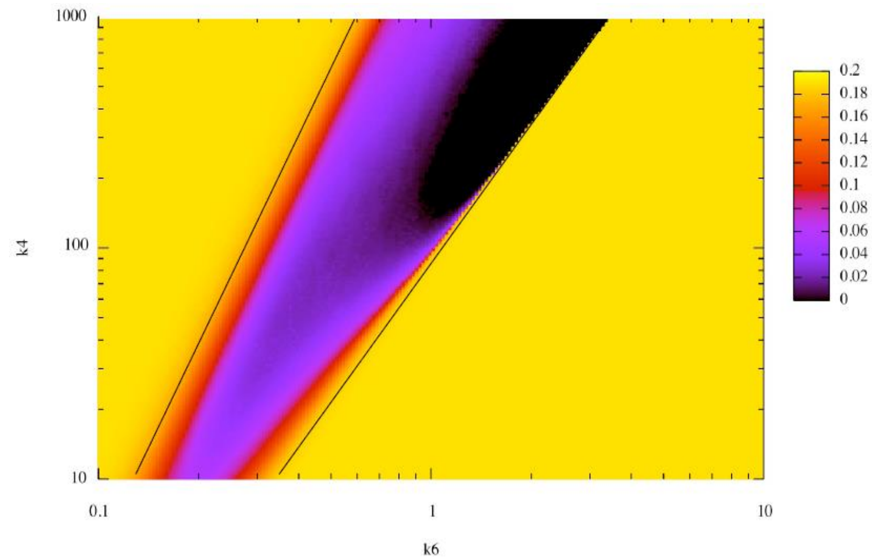
Satisfaction degree $sd(T,\phi) = \frac{1}{1+vd(T,\phi)} \in [0,1]$

FO-LTL(R_{lin}) Continuous Satisfaction Degree in [0,1]



Bifurcation diagram on k4, k6

[Tyson 91]



Continuous satisfaction degree in [0,1]
of the LTL(R) formula for oscillation
with amplitude constraint [Rizk Batt FF Soliman CMSB 08]

- **Parameter search** under LTL(R) constraints in high dimension (100 parameters) by continuous optimization (evolutionary algorithm CMA-ES)
- **Robustness** and **sensitivity** analyses w.r.t. LTL(R) specification

Robustness Measure Definition

Robustness defined with respect to :

- a biological system
- a functionality property D_a
- a set P of perturbations

- General notion of robustness proposed in [Kitano MSB 07]:

$$\mathcal{R}_{a,P} = \int_{p \in P} D_a(p)\ prob(p)\ dp$$

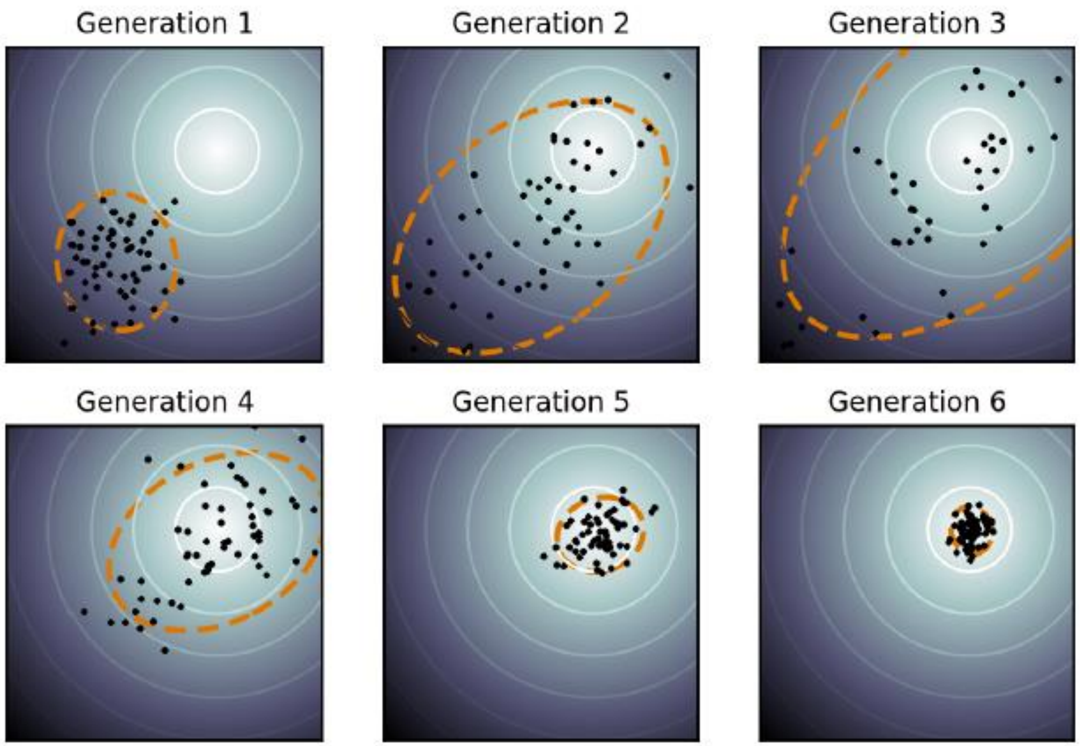
- Our computational measure of robustness w.r.t. LTL($\mathbb{R}$) spec: Given an ODE model with initial conditions, a TL formula ϕ and a set of perturbations P (on initial conditions or parameters),

$$\mathcal{R}_{\phi,P} = \sum_{p \in P} sd(T(p), \phi)\ prob(p)$$

where $T(p)$ is the trace obtained by numerical integration of the ODE for perturbation p

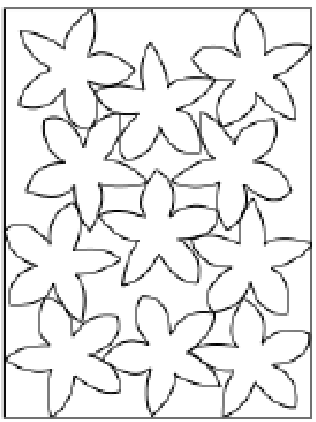
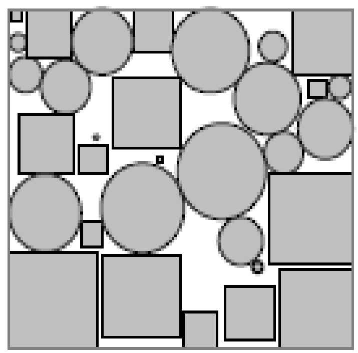
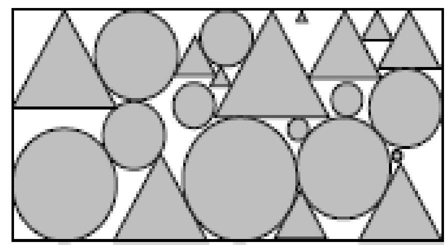
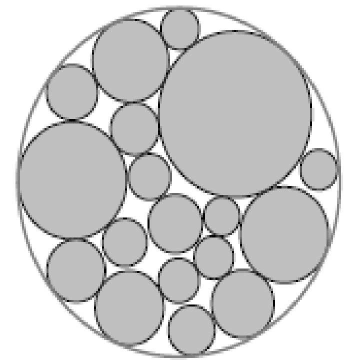
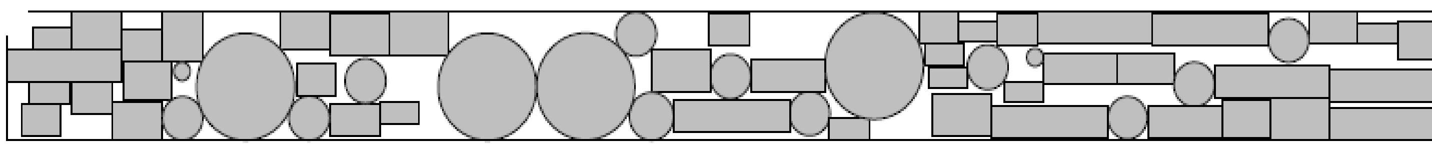
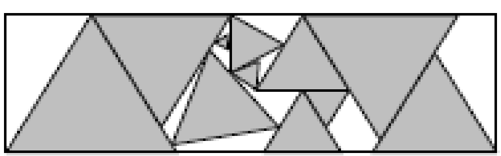
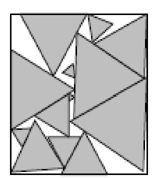
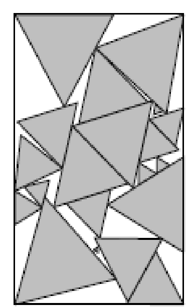
Covariance Matrix Adaptation Evolutionary Strategy

- CMA-ES maximizes a black box fitness function ($sd(\phi)$) in continuous domain ($k_i's$) [Hansen Osermeier 01, Hansen 08]
- CMA-ES uses a probabilistic neighborhood and updates information in covariance matrix at each move

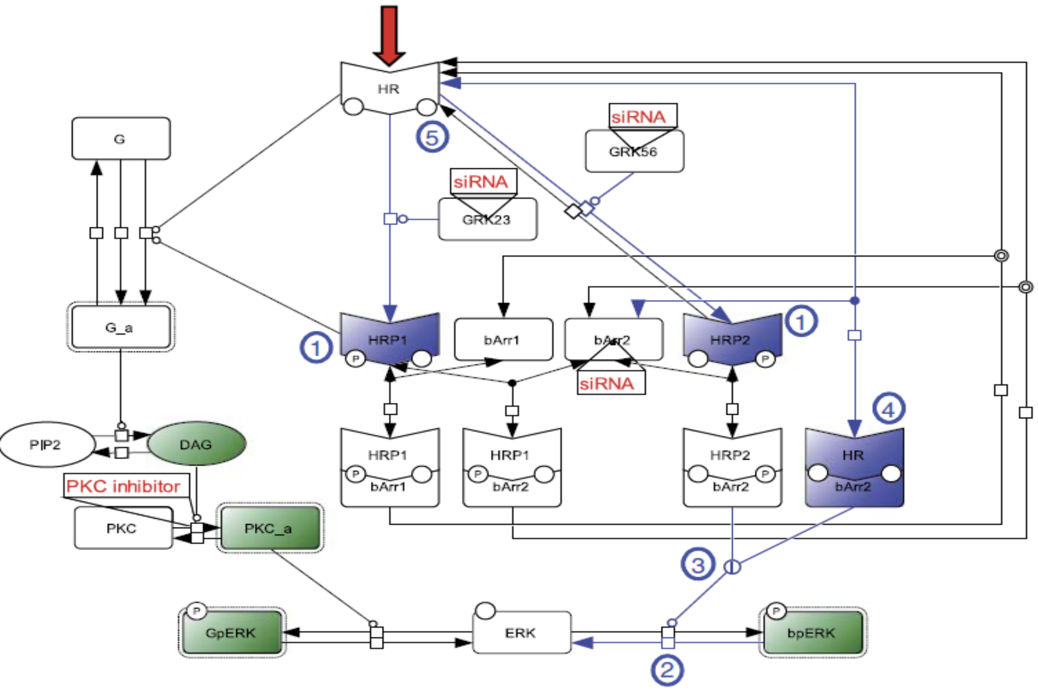


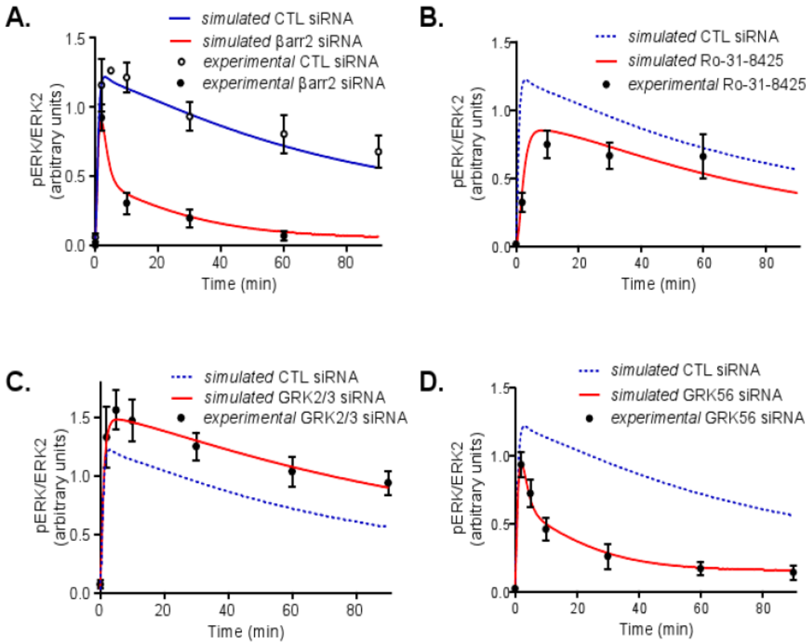
Packing of Complex Shapes with MiniZinc-CMAES

From simple shapes to continuous rotations and complex shapes defined by Bézier curves

Success Story in GPCR Signaling



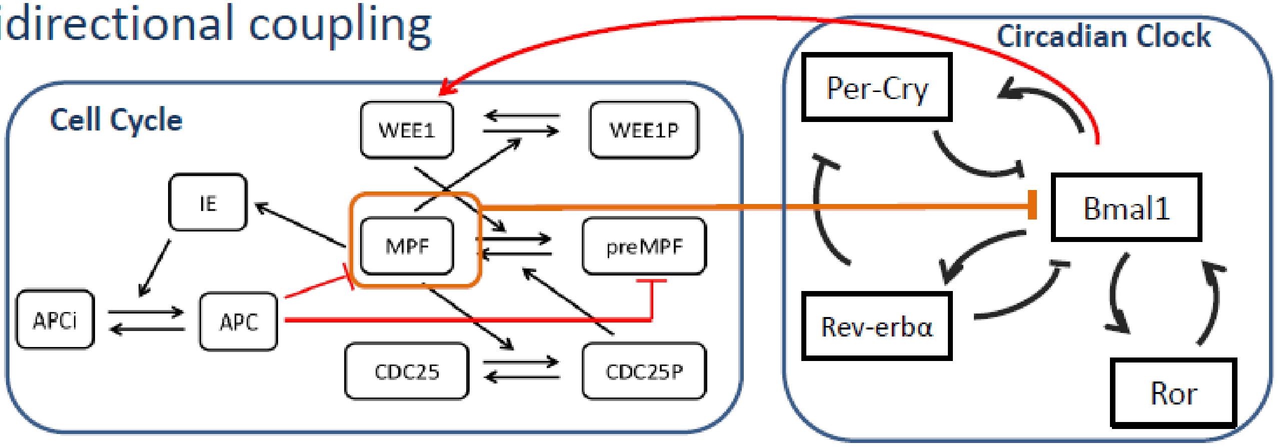


- Reduced model with 4 observables, 4 mutations, known interactions
- *Failure to find satisfying parameter values* using quantitative temporal logic in BIOCHAM
- Revision of the model structure for 3 interactions, experimentally verified *a posteriori*

[D. Heitzler, …, FF , R. Lefkowitz, E. Reiter 2012 *Molecular Systems Biology 8(590)*]

Cell Cycle and Circadian Clock Coupling

Bidirectional coupling



- Influence of circadian clock on cell cycle: time gating for Mitosis through Wee1
- Influence of cell cycle on circadian clock ?
  - Acceleration of the clock observed in fibroblasts in cells with fast cell cycle
  - Hypothesis of selective regulation of clock genes
  - Model-based prediction of up-regulation of RevErb around mitosis

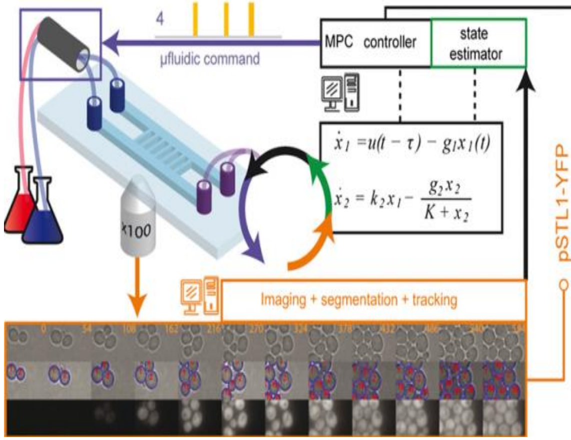
Parameters	First set	Second set
Synthesis coefficient for *Per*	0.66	2.40
Synthesis coefficient for *Cry*	2.30	0.67
Synthesis coefficient for *RevErb-α*	1.04	1.92
Synthesis coefficient for *Ror*	2.1	1.51
Synthesis coefficient for *Bmal1*	0	0.78
Duration	2.97h	2.81h

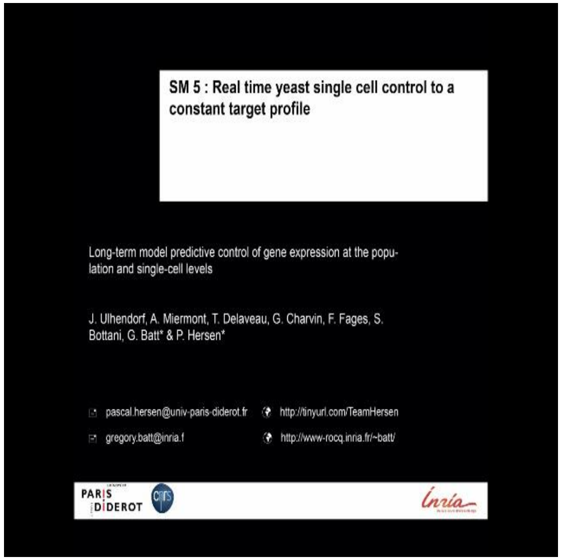
Model-based Control of Gene Expression in Yeast

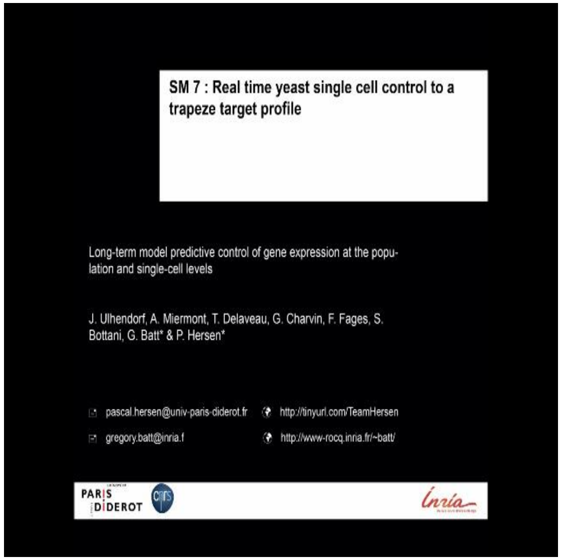
Perception – learning – action loop on a microfluidic device:

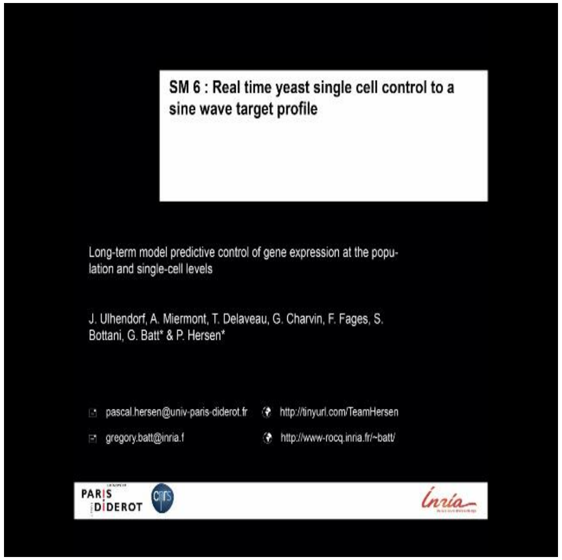
1. Microscope, image analysis (cell tracking or population)
2. Model calibration (kinetic parameter optimization)
3. Osmotic pressure control (parameter optimization)

[Uhlendorf … Batt Hersen PNAS 109(35) 2012]



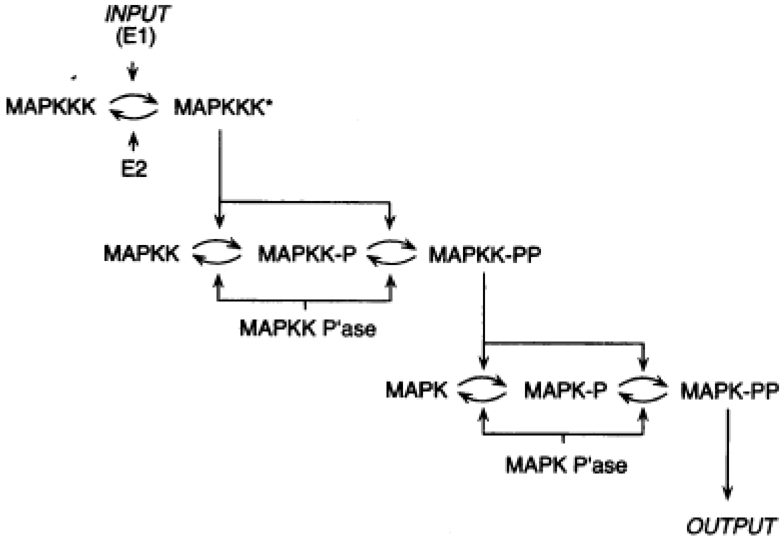


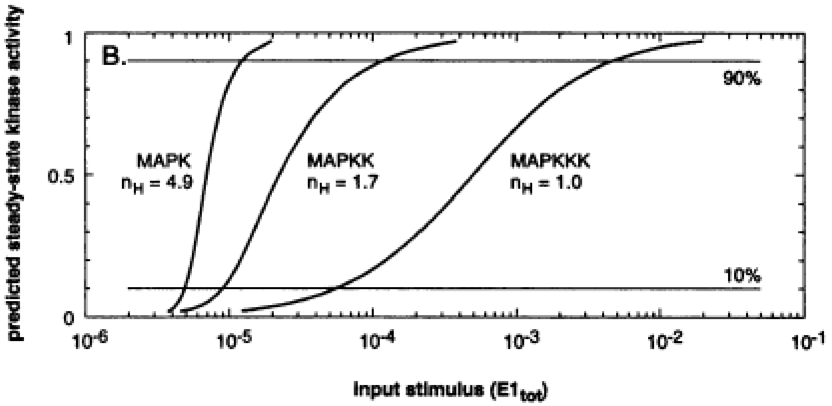




Beyond Describing Natural Circuits
Understanding them ? Why those structures ?

- Analog/Digital Computations
- MAPK signaling = Analog / Digital converter





- How to implement analog circuits with biochemical reactions ?
- How to program with biochemical reactions ?

Analog Arithmetic with Reactions ?

- Infering reaction systems from ODEs [FF Gay Soliman 15 TCS]
- Compute y=f(X)
  1. dy/dt = k*f(X) – k*y   at steady state we will have f(X)=y
  2. Two reactions: k*f(X) for X => X+y   k*y for y =>_
- Multiplication z=x*y
  1. x*y for x+y => x+y+z
  2. z for z => _
- Addition z=x+y
  1. x for x => x+z
  2. y for y => y+z
  3. z for z => _
- Integral z = ∫ x dt
  1. x for x => x+z

General Purpose Analog Computer (Shannon 41)

Logical Preconditions on Reactions?

- Conjunction $X \xrightarrow{A_\theta \wedge B_\theta} Y$

$$X + A + B \rightarrow Y + A + B$$

- Disjunction $X \xrightarrow{A_\theta \vee B_\theta} Y$

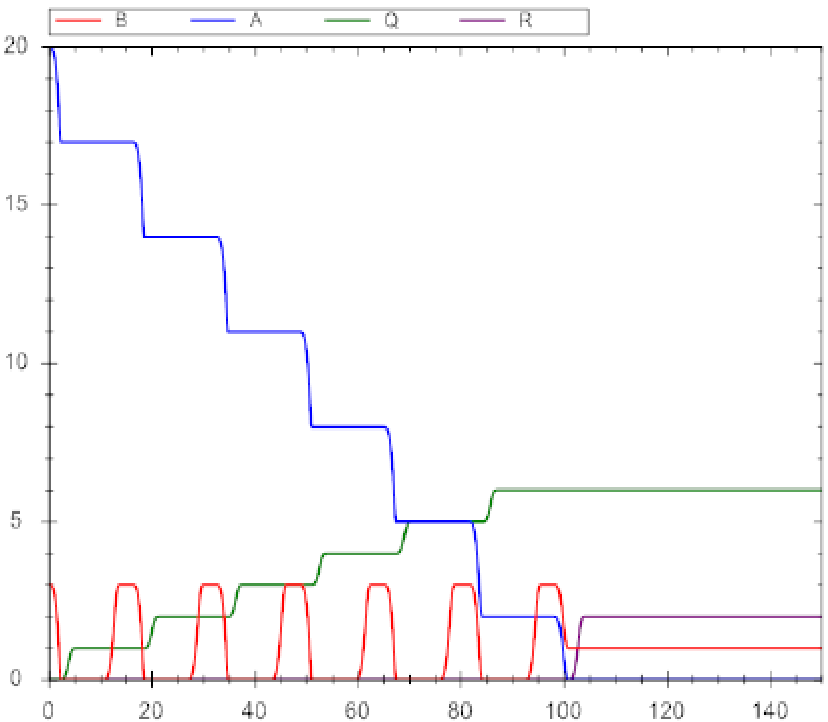
$$X + A \rightarrow Y + A$$
$$X + B \rightarrow Y + B$$

- Negation

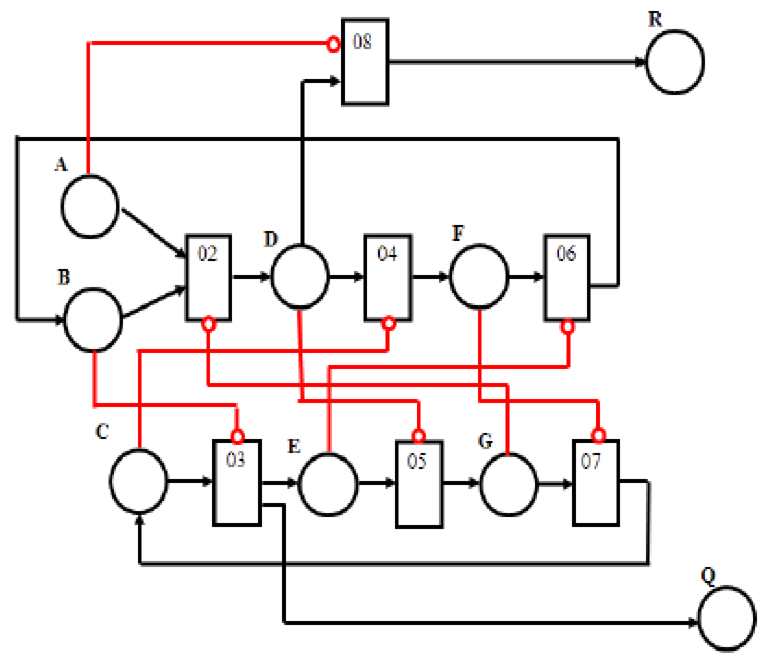
$$\emptyset \xrightarrow{r_s} A'$$
$$A + A' \xrightarrow{r_f} A$$
$$(2A' \xrightarrow{r_f} A')$$

C Compiler into Reactions [Jiang et al 2012, 2013]

$Division(A, B)$
begin
01 while $A \geq B$
02 $A := A - B$
03 $Q := Q + 1$
04 $R := A$
end

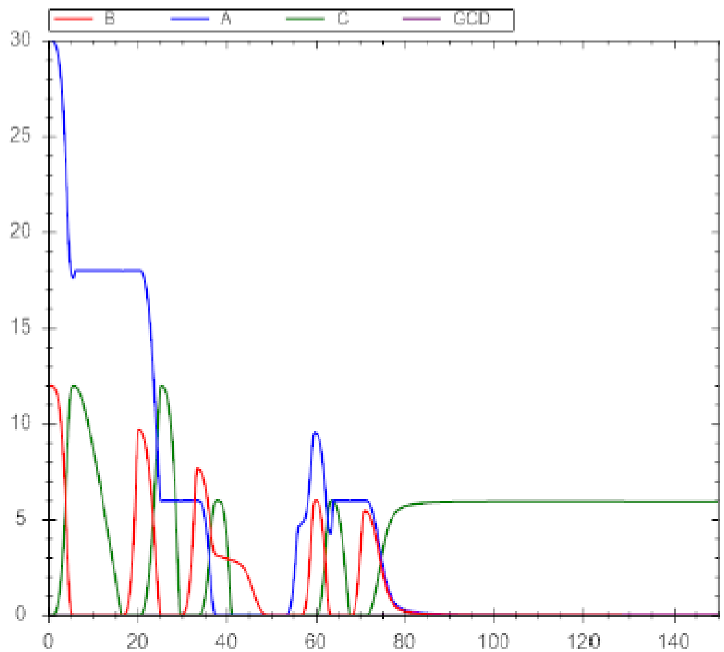


Main Reactions		Preconditions
01	while $[A] \geq [B]$	
02	$(A + B \to D)$	$\neg G_\theta$
03	$C \to Q + E$	$A_\theta \wedge \neg B_\theta$
04	$D \to F$	$\neg C_\theta$
05	$E \to G$	$\neg D_\theta$
06	$F \to B$	$\neg E_\theta$
07	$G \to C$	$\neg F_\theta$
08	$D \to R$	$\neg A_\theta$

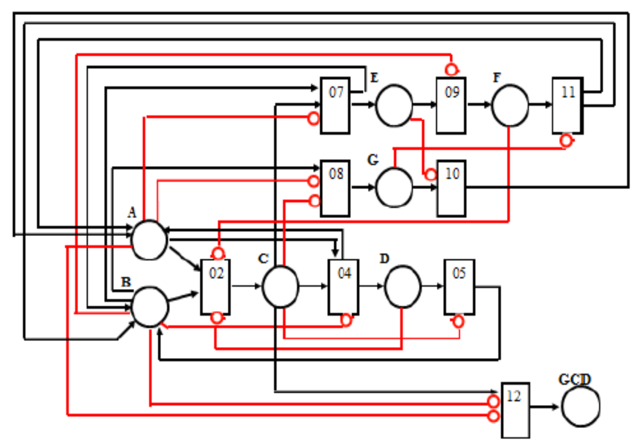


C Compiler into Reactions [Jiang et al 2012, 2013]

```
GreatestCommonDivisor(A, B)
begin
    while A ≠ B
        if A > B
            A := A − B
        else if B > A
            swap(A, B)
    GCD := A
end
```



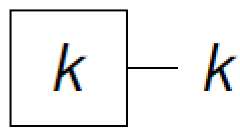
	Main Reactions	Preconditions
01	while $[A] \neq [B]$	
02	$(A + B \rightarrow C)$	$\neg D_\theta \wedge \neg F_\theta$
03	if $[A] > [B]$	
04	$C \rightarrow D$	$A_\theta \wedge \neg B_\theta$
05	$D \rightarrow B$	$\neg C_\theta$
06	else if $[B] > [A]$	
07	$C \rightarrow E$	$\neg A_\theta \wedge B_\theta$
08	$B \rightarrow G$	$\neg C_\theta \wedge \neg A_\theta$
09	$E \rightarrow F$	$\neg B_\theta$
10	$G \rightarrow A$	$\neg E_\theta$
11	$F \rightarrow A + B$	$\neg G_\theta$
12	$C \rightarrow GCD$	$\neg A_\theta \wedge \neg B_\theta$



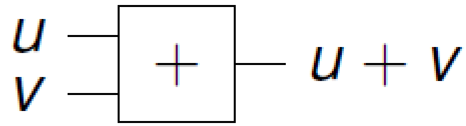
General Purpose Analog Computer [Shannon 41]

idealization of an analog computer: Differential Analyzer

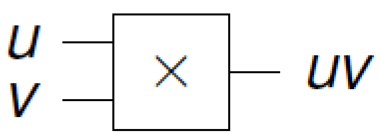
circuit built from:



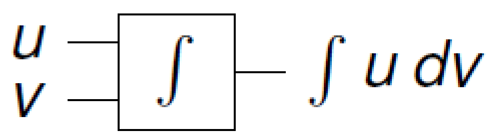
A constant unit



An adder unit



An multiplier unit



An integrator unit

Church-Turing Thesis for Analog Computation

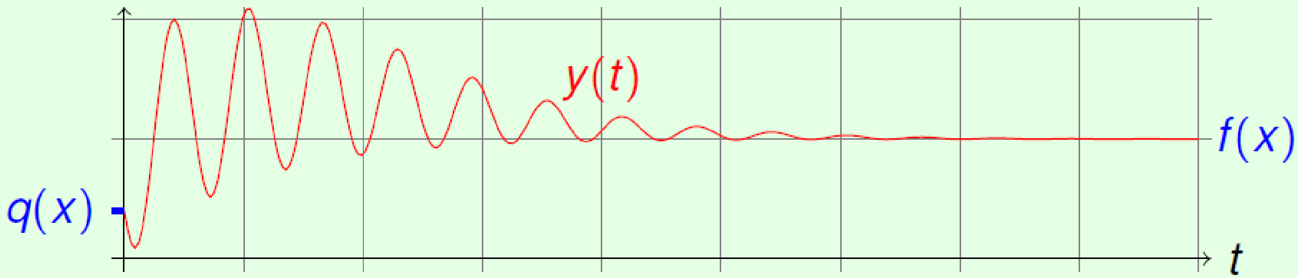
Definition

f is **computable** by a GPAC iff $\exists p, q$ polynomials s.t. $\forall x \in \mathbb{R}$, the solution $y = (y_1, \ldots, y_d)$ of:

$$\begin{cases} y' &= p(y) \\ y(t_0) &= q(x) \end{cases}$$

satisfies $f(x) = \lim_{t \to \infty} y_1(t)$.

Example



Theorem (Bournez, Campagnolo, Graça, Hainry)

f is GPAC-computable functions iff it is computable (in the sense of Computable Analysis).

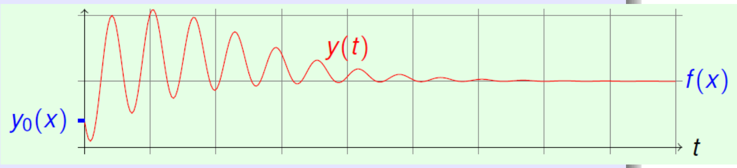
Purely Analog Characterization of Ptime !

[Pouly Bournez Graca 2015]

Definition

f is **poly-computable** by a GPAC iff $\exists p, q$ polynomials s.t. $\forall x \in \mathbb{R}$, the solution $y = (y_1, \ldots, y_d)$ of:

$$\begin{cases} y'(t) = p(y(t)) \\ y(t_0) = q(x) \end{cases}$$



satisfies that:

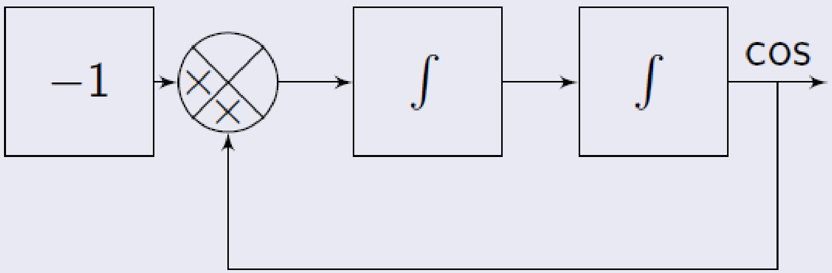
- $\|f(x) - y_1(t)\| \leqslant e^{-\mu}$ when $t \geqslant \text{poly}(\|x\|, \mu)$
- $\|y(t)\| \leqslant \text{poly}(\|x\|, t)$

Theorem

f is poly-computable if and only if it is computable in polytime in the sense of Computable Analysis.

Cosine Function Graph Generation

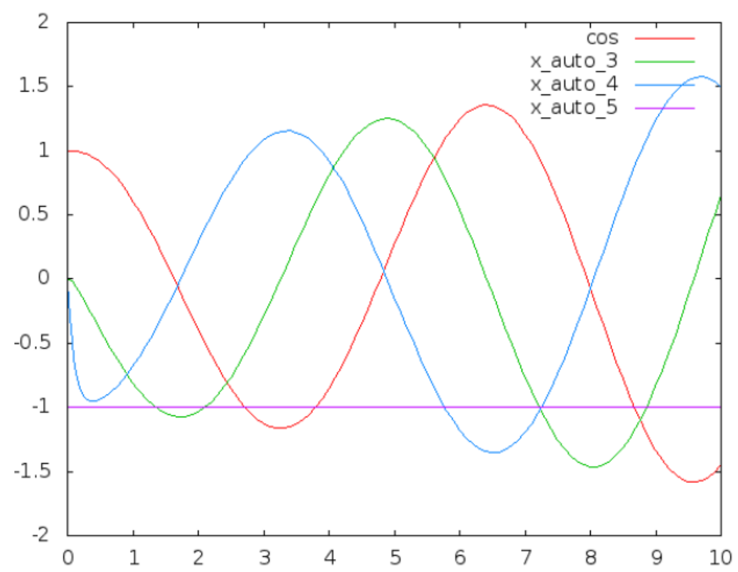
Blocks representation



Example in BIOCHAM

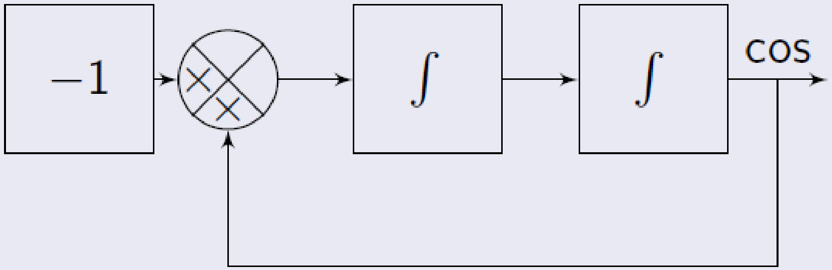
```
    compile_wgpac(cos10 ::
                  integral integral -1*cos10, 10).
    present(cos10).


[0] 10*[x_auto_2]*[cos]for _=[x_auto_2+cos]=>x_auto_1
[1] 10*[x_auto_1]for x_auto_1=>_
[2] _=[x_auto_1]=>x_auto_0
[3] _=[x_auto_0]=>cos
```



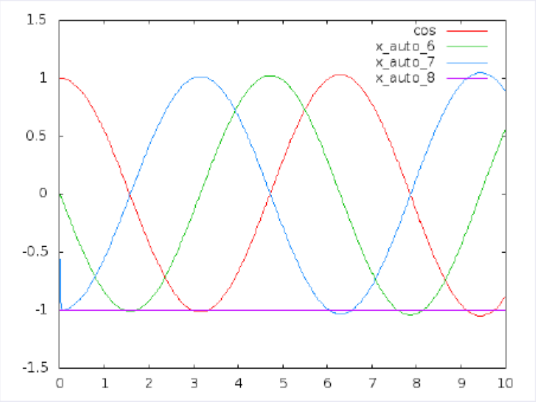
Cosine Function Graph Generation

Blocks representation



Example in BIOCHAM

```
compile_wgpac(cos100 ::
                  integral integral -1*cos100, 100).
present(cos100).
```



Linear Time Invariant Systems

Definition: Laplace transform

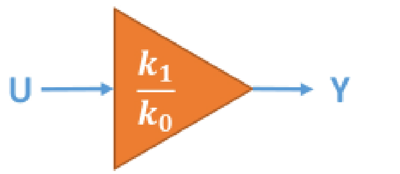
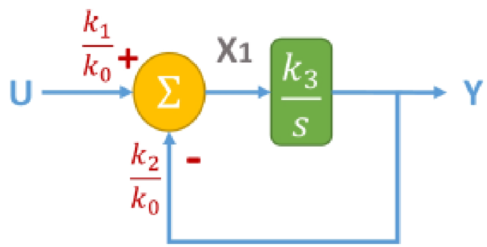
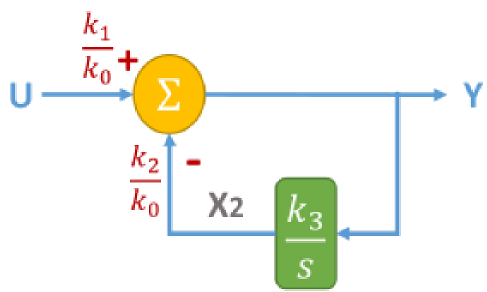
$$\forall s \in \mathbf{R}_+, \quad \mathcal{L}f(s) = \int_0^{+\infty} e^{-st} f(t) \mathrm{d}t$$

$\rightarrow$ Laplace transform of linear and time invariant systems are rational fractions of $\mathbf{R}(s)$.

Definition: Transfer function

The transfer function of a LTI system with input U and output Y is the rational fraction H such that $Y(s) = H(s)U(s)$.
It is said to be *strictly proper* when its degree is negative.

Transfer Function of Reaction Impl. [Jiang et al 2015]

Block Diagram	Normal Transfer Function	CRN Transfer Function
	$\frac{k_1}{k_0}$	$\frac{k_1}{s+k_0}$
	$\frac{k_1 k_3}{k_0 s + k_2 k_3}$	$\frac{k_1 k_3}{s^2 + k_0 s + k_2 k_3}$
	$\frac{k_1 s}{k_0 s + k_2 k_3}$	$\frac{k_1 s}{s^2 + k_0 s + k_2 k_3}$

Compiling Transfer Functions into Reactions

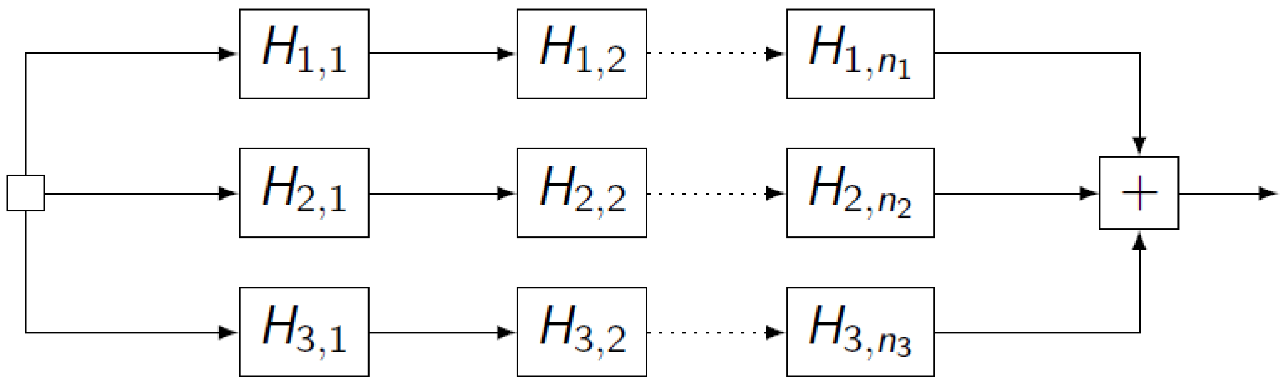
Compiler principle:

1. 'break' the transfer function into simple functions, this step is performed by computing the partial fraction expansion of H ;
2. each of these simple functions consist of a chemical network with input u and output y_i ;
3. recombine the individual outputs y_i to get y.

Compiling Transfer Functions into Reactions

- Write $H = \sum H_i$ where H_i are simple functions, that is either of the form $\frac{a}{(s+\alpha)^n}$ or $\frac{a}{(s^2+\beta s+\gamma)^m}$ or $\frac{bs}{(s^2+\beta s+\gamma)^m}$.
- Each of these functions can in turn be writen a product of elementary functions, that is either of the form $\frac{a}{s+\alpha}$ or $\frac{a}{s^2+\beta s+\gamma}$ or $\frac{bs}{s^2+\beta s+\gamma}$, denoted respectively by $(1,0)$, $(2,0)$ and $(2,1)$.
- Product corresponds to series composition of modules ;
- Sum is performed by parallel computation.

Final Summing Block



Remark: The summing node is chemically implemented through the reactions $y_i \xrightarrow{k} y_i + y$ for each local output y_i and $y \xrightarrow{k} \emptyset$.
Therefore one has

$$Y(s) = \frac{k}{s+k} \sum Y_i(s)$$

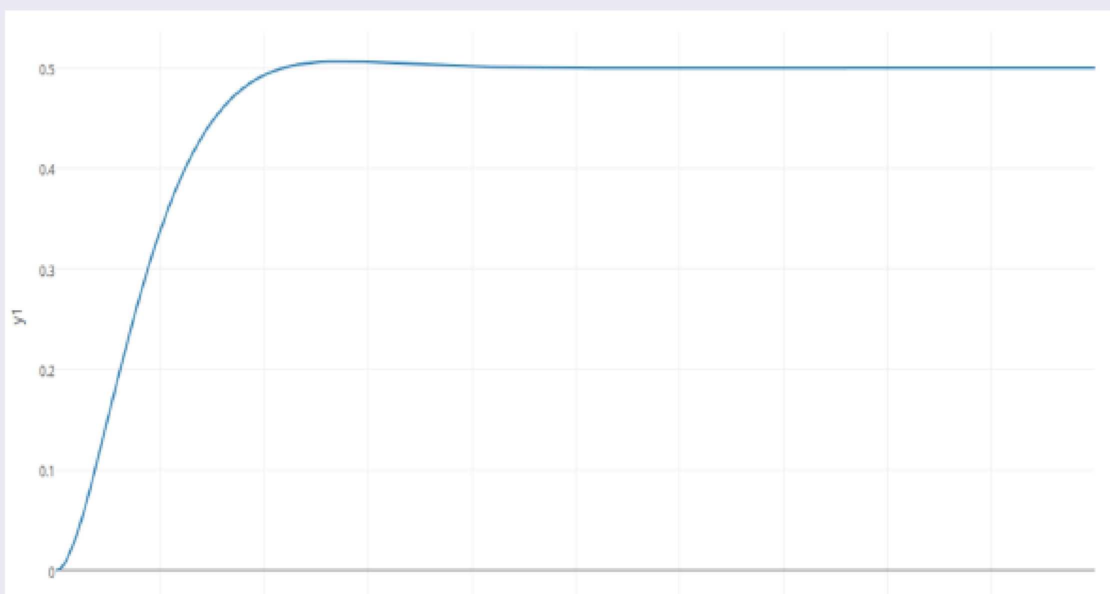
This unwanted factor can be compensated by computing $\hat{H} = \frac{s+k}{k} H$ instead of H.

Example: a first-order filter $\frac{1}{s+2}$, 'naive' implementation

Biocham

```
compile_wgpac([y1 :: integral x1,
               x1 :: u1 + (-2)*y1],
               10).
present(u1).
```

Illustration of time response: $y_1(t)$

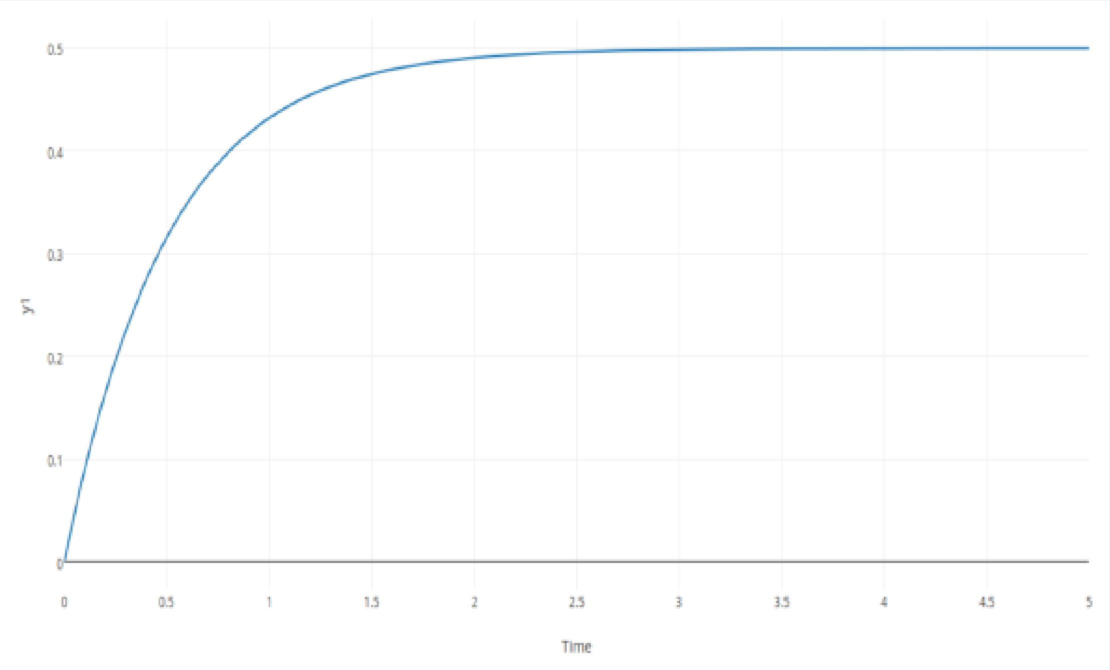


Example: a first-order filter $\frac{1}{s+2}$

Biocham

```
compile_transfer_function(1/(s+2), u1, y1).
present(u1).
```

Illustration of time response: $y_1(t)$



Comparison of the created systems

naive

```
[0] _=[x1]=>y1
[1] 10*[x_auto_4]*[y1]for _=[x_auto_4+y1]=>x_auto_3
[2] 10*[x_auto_3]for x_auto_3=>_
[3] 10*[u1]for _=[u1]=>x1
[4] 10*[x_auto_3]for _=[x_auto_3]=>x1
[5] 10*[x1]for x1=>_
```

transfer function specific

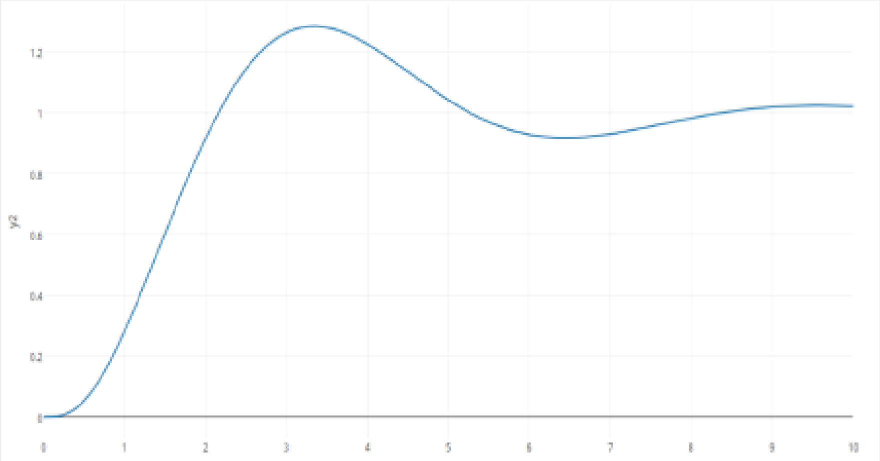
```
[0] _=[u1]=>y1
[1] 2*[y1]for y1=>_
```

Example: a second-order filter $\frac{1}{1+s+s^2}$, 'naive' implementation

Biocham

```
compile_wgpac([y1 :: integral x1, x1 :: u1 + (-1)*y1,
               y2 :: integral y1, u1 :: u2 + (-1)*y2],
               10).
present(u2).
```

Illustration of time response: $y_2(t)$

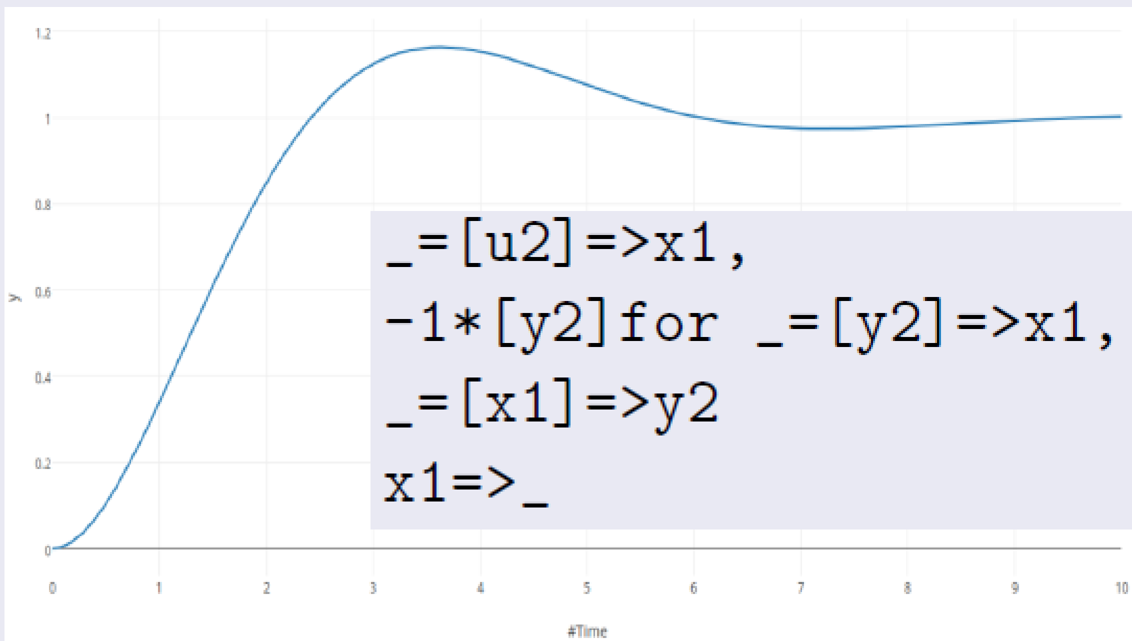


Example: a second-order filter $\frac{1}{1+s+s^2}$

Biocham

```
compile_transfer_function(1/(s*s+s+1), u2, y2).
present(u2).
```

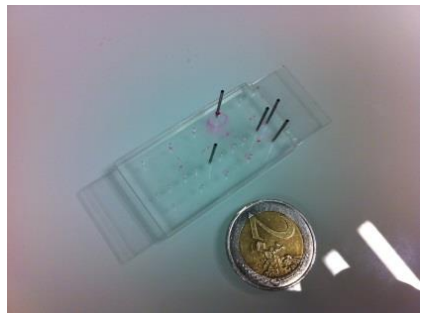
Illustration of time response: $y_2(t)$

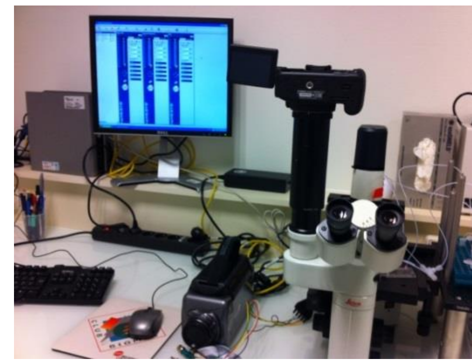



```
_=[u2]=>x1,
-1*[y2]for _=[y2]=>x1,
_=[x1]=>y2
x1=>_
```

Enzymatic Computation in Non-Living Vesicles

- Biosensor design and implementation in non-living vesicles
  [Franck Molina lab CNRS Sys2Diag Montpellier]





- Implementation of linear I/O systems, PI controllers and simple programs ?
  → Issue of approximation and compositionality
  → Issue of reaction code optimization (number of species and reactions)

- Comparison of synthetic programs with natural programs
  → Multiple functions of a circuit ?
  → Evolution history ? Evolution capacity ?

Thank you !

Et désolé si j'étais à l'ouest