Formal Cell Biology in BIOCHAM

François Fages

Constraint Programming project-team,
INRIA Paris-Rocquencourt

To deal with the complexity of biological systems, investigate

- Programming Theory Concepts
- Formal Methods of Circuit and Program Verification
- Automated Reasoning Tools

Software Implementation in the Biochemical Abstract Machine BIOCHAM

modeling environment available at **`http://contraintes.inria.fr/BIOCHAM`**

Systems Biology ?

"Systems Biology aims at systems-level understanding which requires a set of principles and methodologies that links the behaviors of molecules to systems characteristics and functions."

H. Kitano, ICSB 2000

- Analyze (post-)genomic data produced with high-throughput technologies
- Databases and ontologies like SwissProt, GO, KEGG, BioCyc, etc.
- Systems Biology Markup Language (SBML) : exchange format for reaction models
- Integrate heterogeneous data about a specific problem
- Understand and Predict behaviors or interactions in large networks of genes and proteins.

Issue of Abstraction in Systems Biology

Models are built in Systems Biology with two contradictory perspectives :

Issue of Abstraction in Systems Biology

Models are built in Systems Biology with two contradictory perspectives :

1) Models for representing knowledge : the more concrete the better

François Fages

Bertinoro, 3 June 08

Issue of Abstraction in Systems Biology

Models are built in Systems Biology with two contradictory perspectives :

1) Models for representing knowledge : the more concrete the better

2) Models for making predictions : the more abstract the better !

Issue of Abstraction in Systems Biology

Models are built in Systems Biology with two contradictory perspectives :

1) Models for representing knowledge : the more concrete the better

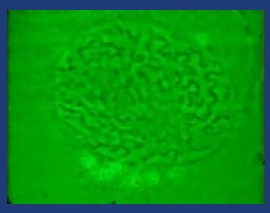
2) Models for making predictions : the more abstract the better !

These perspectives can be reconciled by organizing formalisms and models into a hierarchy of abstractions.

To understand a system is not to know everything about it but to know abstraction levels that are sufficient for answering questions about it

Semantics of Living Processes ?

Formally, “the” behavior of a system depends on *our choice* of observables.

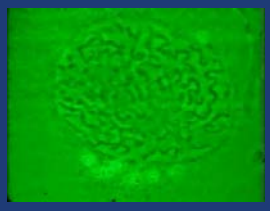


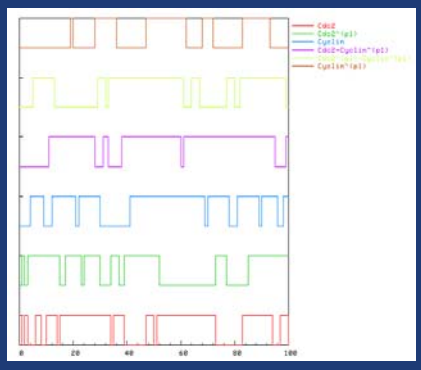
Mitosis movie [Lodish et al. 03]

Boolean Semantics

Formally, “the” behavior of a system depends on *our choice* of observables.

Presence/absence of molecules
Boolean transitions

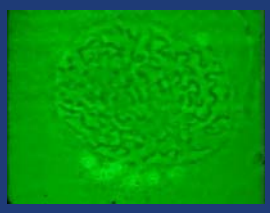


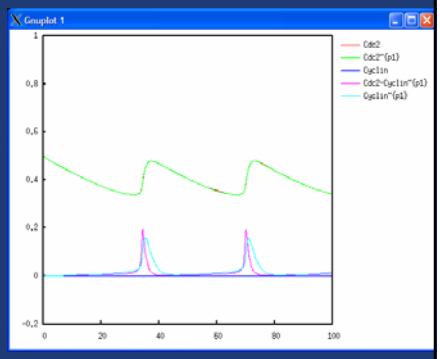


Continuous Differential Semantics

Formally, “the” behavior of a system depends on *our choice* of observables.

Concentrations of molecules
Rates of reactions

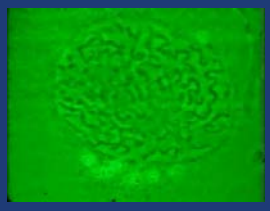


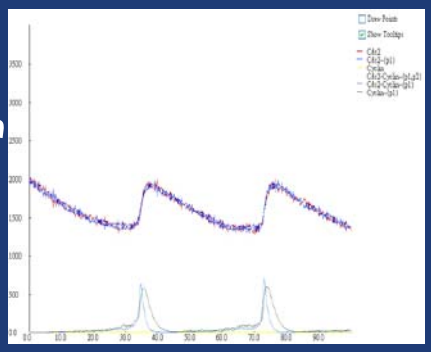


Stochastic Semantics

Formally, “the” behavior of a system depends on *our choice* of observables.

Numbers of molecules
Probabilities of reaction

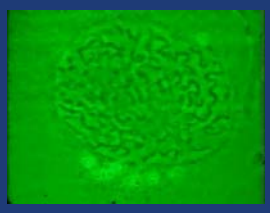




Temporal Logic Semantics

Formally, “the” behavior of a system depends on *our choice* of observables.

Presence/absence of molecules
Temporal logic formulas

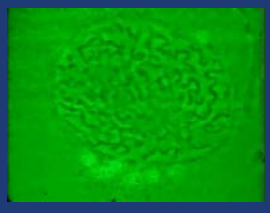


F x
F (x ^ F (¬ x ^ y))
FG (x v y)
…

Constraint Temporal Logic Semantics

Formally, “the” behavior of a system depends on *our choice* of observables.

Concentrations of molecules
Constraint LTL temporal formulas



F (x >0.2)
F (x >0.2 ^ F (**x<0.1** ^ y>0.2))
FG (x>0.2 v y>0.2)
…

A Logical Paradigm for Systems Biology

Biological process model = Transition System
Biological property = Temporal Logic Formula
Biological validation = Model-checking

[Lincoln et al. PSB'02] [Chabrier Fages CMSB'03] [Bernot et al. TCS'04] …

Model:	*BIOCHAM*	Biological Properties:
- Boolean	- simulation	- Temporal logic CTL
- Differential	- query evaluation	- LTL with constraints
- Stochastic	- rule learning	- PCTL with constraints
(SBML)	- parameter search	

Types: static analyses

Outline of the Talk

1. Abstract machines: *Rule-based Models* of biochemical systems
   1. Syntax of molecules, compartments and reactions
   2. Hierarchy of semantics: stochastic, differential, discrete, boolean
   3. Cell cycle control models
2. Abstract behaviors: *Temporal Logic* formalization of biological properties
   1. Computation Tree Logic CTL for the boolean semantics
   2. Linear Time Logic with constraints LTL(R) for the differential semantics
   3. Probabilistic PCTL for the stochastic semantics
3. Automated Reasoning Tools
   1. Rule learning from CTL specification
   2. Kinetic parameter inference from LTL(R) specification

L. Calzone, F. Fages, S. Soliman. *Bioinformatics 22.* 2006
L. Calzone, N. Chabrier, F. Fages, S. Soliman. *Trans. Computational System Biology 6* 2006
F. Fages, S. Soliman. *Theoretical Computer Science*. 2008. F. Fages, A. Rizk. *Theor.Comp.Sc.* 2008.

Syntax of proteins

Cyclin dependent kinase 1 (free, inactive) **`Cdk1`**

Complex Cdk1-Cyclin B (low activity) **`Cdk1-CycB`**

Phosphorylated form at site threonine 161 (high activity) mitosis promotion factor **`Cdk1~{thr161}-CycB`**

Elementary Reaction Rules

Complexation: A + B => A-B

`cdk1+cycB => cdk1-cycB`

Decomplexation A-B => A + B

Elementary Reaction Rules

Complexation: A + B => A-B
`cdk1+cycB => cdk1-cycB`

Decomplexation A-B => A + B

Phosphorylation: A =[C]=> A~{p}
`Cdk1-CycB =[Myt1]=> Cdk1~{thr161}-CycB`
`Cdk1~{thr14,tyr15}-CycB =[Cdc25~{Nterm}]=> Cdk1-CycB`

Dephosphorylation A~{p} =[C]=> A

Elementary Reaction Rules

Complexation: A + B => A-B
`cdk1+cycB => cdk1-cycB`

Decomplexation A-B => A + B

Phosphorylation: A =[C]=> A~{p}
`Cdk1-CycB =[Myt1]=> Cdk1~{thr161}-CycB`
`Cdk1~{thr14,tyr15}-CycB =[Cdc25~{Nterm}]=> Cdk1-CycB`

Dephosphorylation A~{p} =[C]=> A

Synthesis: _ =[C]=> A.
`_ =[#E2-E2f13-Dp12]=> CycA`

Degradation: A =[C]=> _.
`cycE =[@UbiPro]=> _`
(not for `cycE-cdk2` which is stable)

Elementary Reaction Rules

Complexation: A + B => A-B *Decomplexation* A-B => A + B

```
cdk1+cycB => cdk1-cycB
```

Phosphorylation: A =[C]=> A~{p} *Dephosphorylation* A~{p} =[C]=> A

```
Cdk1-CycB =[Myt1]=> Cdk1~{thr161}-CycB
Cdk1~{thr14,tyr15}-CycB =[Cdc25~{Nterm}]=> Cdk1-CycB
```

Synthesis: _ =[C]=> A. *Degradation*: A =[C]=> _.

```
_ =[#E2-E2f13-Dp12]=> CycA
```

```
cycE =[@UbiPro]=> _
```

(not for `cycE-cdk2` which is stable)

Transport: A::L1 => A::L2

```
Cdk1~{p}-CycB::cytoplasm => Cdk1~{p}-CycB::nucleus
```


From Syntax to Semantics

```
S ::= _ | molecule + S
R ::= S=>S | kinetics for S=>S
```

Example `k*[A]*[B] for A+B => C`

SBML (Systems Biology Markup Lang.): import/export exchange format

From Syntax to Semantics

```
S ::= _ | molecule + S
R ::= S=>S | kinetics for S=>S
```

Example `k*[A]*[B] for A+B => C`

SBML (Systems Biology Markup Lang.): import/export exchange format

BIOCHAM : three *abstraction levels*

1. Stochastic Semantics: number of molecules
   - Continuous time Markov chain

$$p_{ij} = \frac{\tau_{ij}}{\sum_{(S_i,S_k,\tau_{ik})\in s} \tau_{ik}}$$

From Syntax to Semantics

```
S ::= _ | molecule + S
R ::= S=>S | kinetics for S=>S
```

Example `k*[A]*[B] for A+B => C`

SBML (Systems Biology Markup Lang.): import/export exchange format

BIOCHAM : three *abstraction levels*

1. Stochastic Semantics: number of molecules
   - Continuous time Markov chain

$$p_{ij} = \frac{\tau_{ij}}{\sum_{(S_i,S_k,\tau_{ik})\in s} \tau_{ik}}$$

2. Differential Semantics: concentration
   - Ordinary Differential Equations (hybrid system)

$$dx_k/dt = \sum_{i=1}^{n} r_i(x_k) * e_i - \sum_{j=1}^{n} l_j(x_k) * e_j$$

From Syntax to Semantics

```
S ::= _ | molecule + S
R ::= S=>S | kinetics for S=>S
```

Example `k*[A]*[B] for A+B => C`

SBML (Systems Biology Markup Lang.): import/export exchange format

BIOCHAM : three *abstraction levels*

1. Stochastic Semantics: number of molecules
   - Continuous time Markov chain

$$p_{ij} = \frac{\tau_{ij}}{\sum_{(S_i,S_k,\tau_{ik})\in s} \tau_{ik}}$$

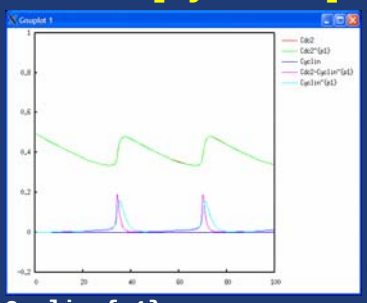
2. Differential Semantics: concentration
   - Ordinary Differential Equations (hybrid system)

$$dx_k/dt = \sum_{i=1}^{n} r_i(x_k) * e_i - \sum_{j=1}^{n} l_j(x_k) * e_j$$

3. Boolean Semantics: presence-absence of molecules
   - Asynchronuous Transition System

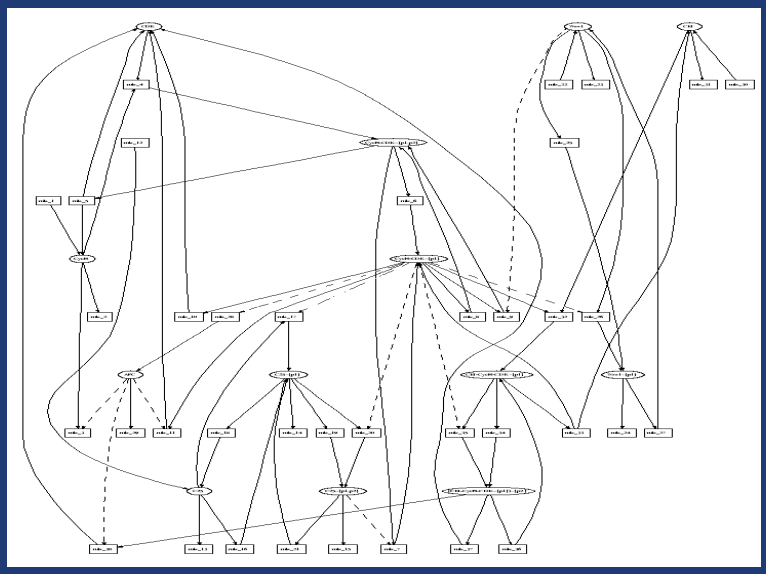
A, B → (A/¬A), (B /¬B), C

Budding Yeast Cell Cycle Control Model [Tyson 91]

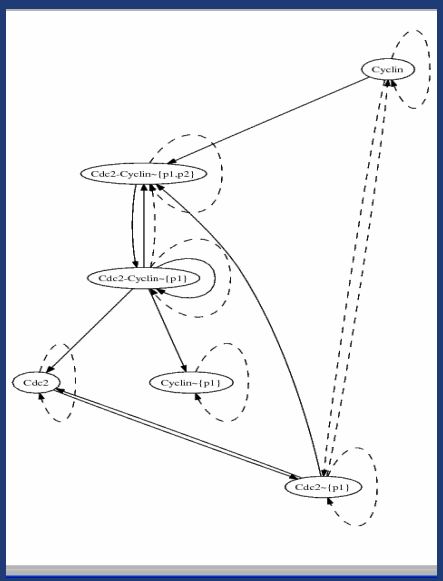


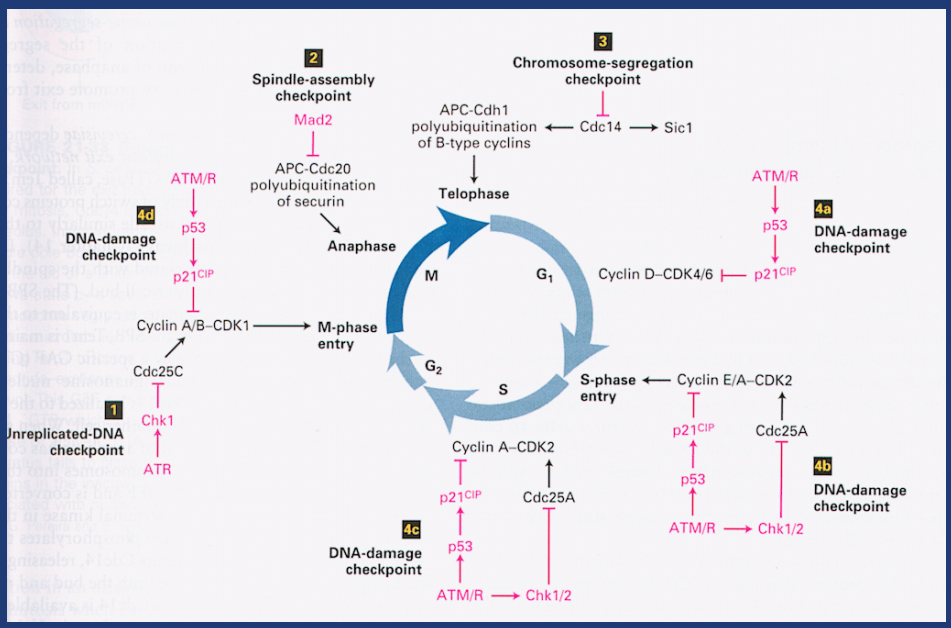
```
MA(k1) for  _ => Cyclin.
MA(k2) for  Cyclin => _.
MA(K7) for  Cyclin~{p1} => _.

MA(k8) for  Cdc2 => Cdc2~{p1}.
MA(k9) for  Cdc2~{p1} =>Cdc2.

MA(k3) for  Cyclin+Cdc2~{p1} => Cdc2~{p1}-Cyclin~{p1}.
MA(k4p) for Cdc2~{p1}-Cyclin~{p1} => Cdc2-Cyclin~{p1}.
k4*[Cdc2-Cyclin~{p1}]^2*[Cdc2~{p1}-Cyclin~{p1}] for
    Cdc2~{p1}-Cyclin~{p1} =[Cdc2-Cyclin~{p1}] => Cdc2-Cyclin~{p1}.

MA(k5) for Cdc2-Cyclin~{p1} => Cdc2~{p1}-Cyclin~{p1}.
MA(k6) for Cdc2-Cyclin~{p1} => Cdc2+Cyclin~{p1}.
```

Reaction Hypergraph

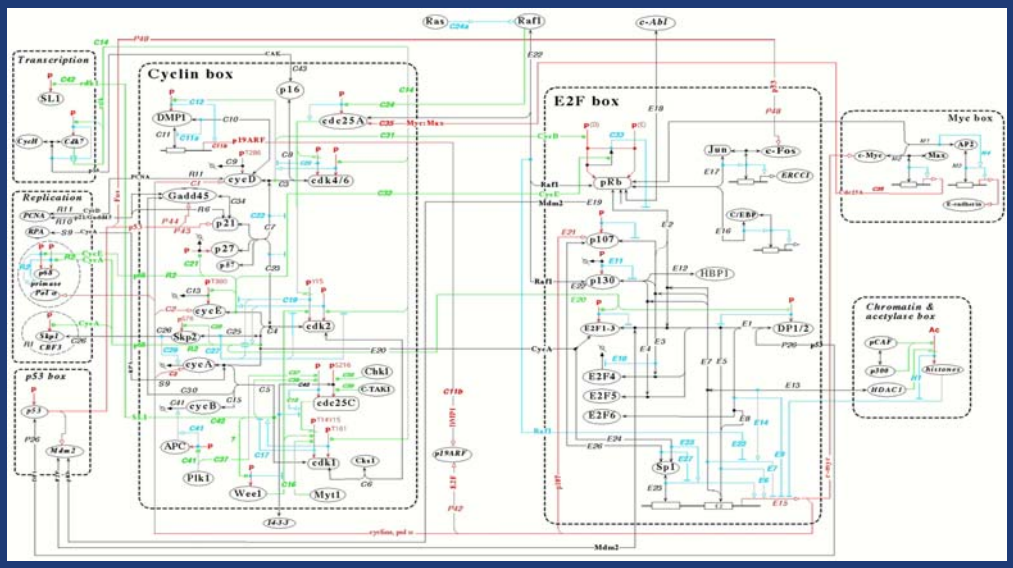


Activation/Inhibition Influence Graph

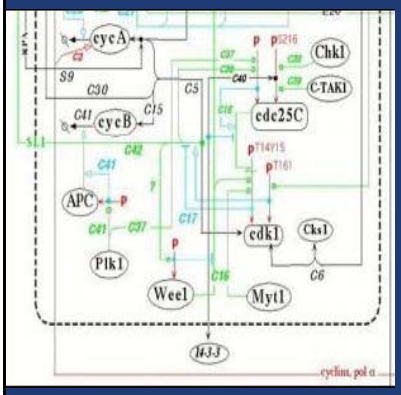




Mammalian Cell Cycle Control Map [Kohn 99]



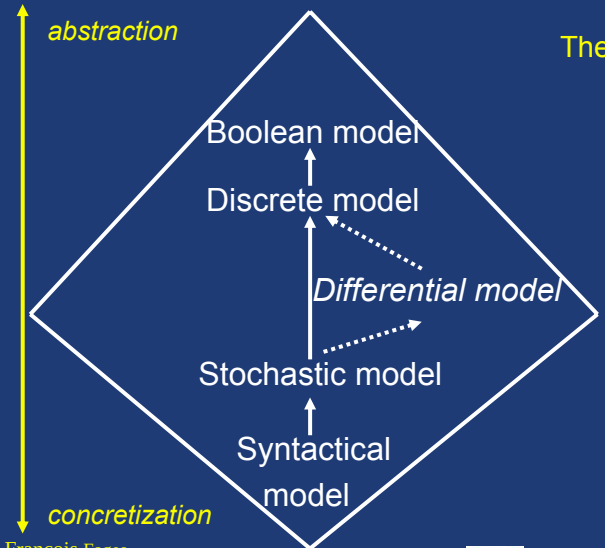
Transcription of Kohn's Map




```
_ =[ E2F13-DP12-gE2 ]=> cycA.
...
cycB =[ APC~{p1} ]=>_.
cdk1~{p1,p2,p3} + cycA => cdk1~{p1,p2,p3}-cycA.
cdk1~{p1,p2,p3} + cycB => cdk1~{p1,p2,p3}-cycB.
...
cdk1~{p1,p3}-cycA =[ Wee1 ]=> cdk1~{p1,p2,p3}-cycA.
cdk1~{p1,p3}-cycB =[ Wee1 ]=> cdk1~{p1,p2,p3}-cycB.
cdk1~{p2,p3}-cycA =[ Myt1 ]=> cdk1~{p1,p2,p3}-cycA.
cdk1~{p2,p3}-cycB =[ Myt1 ]=> cdk1~{p1,p2,p3}-cycB.
...
cdk1~{p1,p2,p3} =[ cdc25C~{p1,p2} ]=> cdk1~{p1,p3}.
cdk1~{p1,p2,p3}-cycA =[ cdc25C~{p1,p2} ]=> cdk1~{p1,p3}-cycA.
cdk1~{p1,p2,p3}-cycB =[ cdc25C~{p1,p2} ]=> cdk1~{p1,p3}-cycB.
```

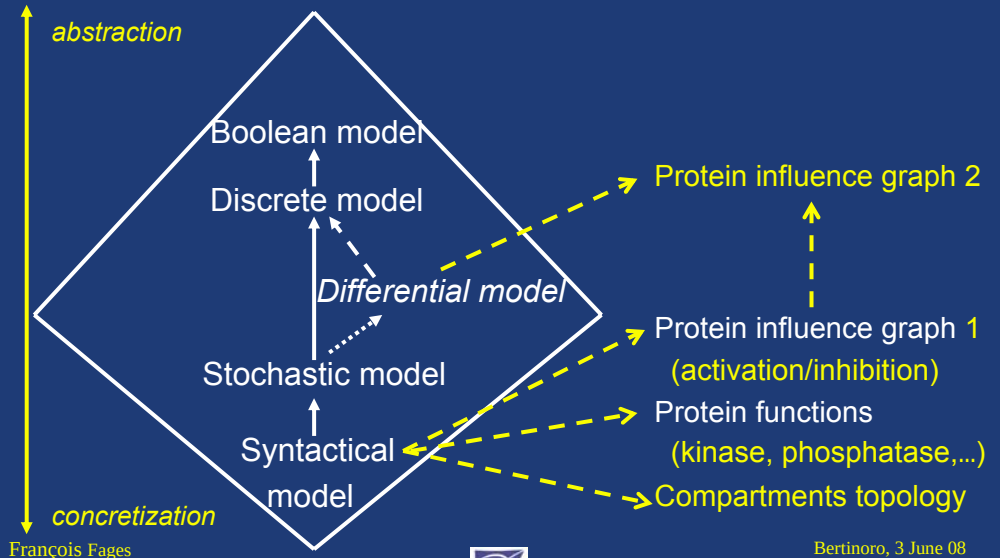
165 proteins and genes, 500 variables, 800 rules [Chiaverini Danos 02]

Hierarchy of Semantics

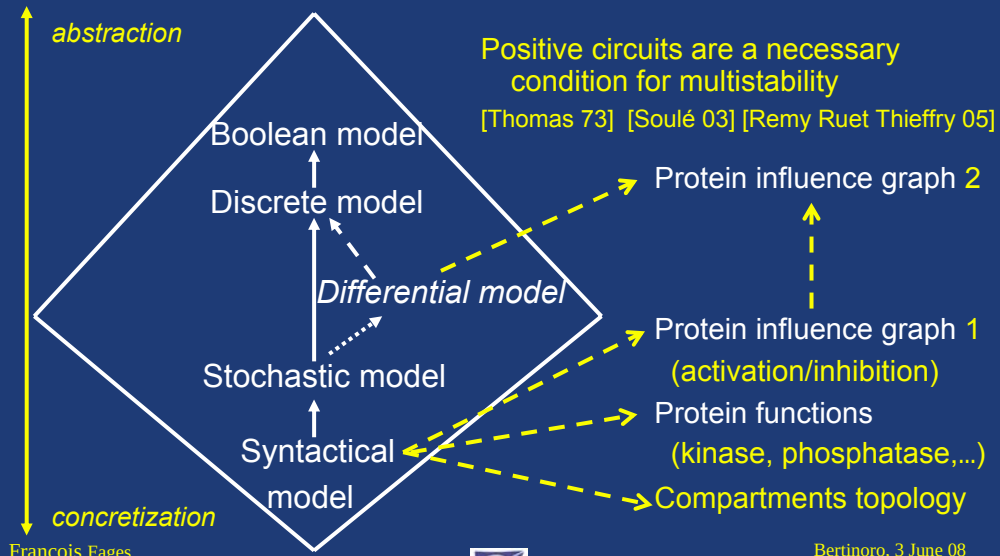


Theory of abstract Interpretation
[Cousot Cousot POPL'77]
[Fages Soliman TCSc'08]

Type Inference / Type Checking



Type Inference / Type Checking



The Theory of Abstract Interpretation

In this setting [Cousot Cousot 77], a domain is a lattice $\mathcal{D}(\sqsubseteq, \bot, \top, \sqcup, \sqcap)$ where $\sqsubseteq$ is the "information loss" ordering.

A Galois connection $\mathcal{C} \rightarrow_{\alpha} \mathcal{A}$ between two lattices $\mathcal{C}$ and $\mathcal{A}$ is defined by two abstraction and concretization functions $\alpha : \mathcal{C} \rightarrow \mathcal{A}$ and $\gamma : \mathcal{A} \rightarrow \mathcal{C}$ that are monotonic:

- $\forall\ x, y \in \mathcal{C}\ x \sqsubseteq_{\mathcal{C}} y\ \Rightarrow\ \alpha(x) \sqsubseteq_{\mathcal{A}} \alpha(y)$,
- $\forall\ x, y \in \mathcal{A}\ x \sqsubseteq_{\mathcal{A}} y\ \Rightarrow\ \gamma(x) \sqsubseteq_{\mathcal{C}} \gamma(y)$,

and are adjoint:

- $\forall c \in \mathcal{C}, \forall y \in \mathcal{A} : x \sqsubseteq_{\mathcal{C}} \gamma(y) \Leftrightarrow \alpha(x) \sqsubseteq_{\mathcal{A}} y$.

If $\gamma \circ \alpha$ is the identity, the abstraction α loses no information, and $\mathcal{C}$ and $\mathcal{A}$ are isomorphic from the information standpoint (although α may be not onto and γ not one-to-one).

Properties of Galois Connections

1. $\gamma \circ \alpha$ is extensive (i.e. $x \sqsubseteq_{\mathcal{C}} \gamma \circ \alpha(x)$) and represents the information lost by the abstraction;
2. $\alpha \circ \gamma$ is contracting (i.e. $\alpha \circ \gamma(y) \sqsubseteq_{\mathcal{A}} y$);
3. $\gamma \circ \alpha$ is the identity *iff* γ is onto *iff* α is one-to-one.
4. α preserves $\sqcup$, and γ preserves $\sqcap$;
5. $\gamma(a) = max\ \alpha^{-1}(\downarrow a) = \sqcup \alpha^{-1}(\downarrow a)$
6. $\alpha(c) = min\ \gamma^{-1}(\uparrow c) = \sqcap \gamma^{-1}(\uparrow c)$

where $\downarrow a = \{b \mid b \sqsubseteq a\}$ and $\uparrow a = \{b \mid a \sqsubseteq b\}$.

It is equivalent in the definition of Galois connections to replace the condition of adjointness by conditions 1 and 2,

or by condition 5 which also entails the monotonicity of γ.

Systems Biology Markup Language SBML Models

Formally, the concrete domain of reaction models is the powerset of all possible reaction rules ordered by set inclusion :

Def. 1 *Given a finite set $\mathcal{M}$ of molecule names, the universe of reactions is the set of rules*

$$\mathcal{R} = \{e \texttt{ for } S{=}{>}S' \mid \begin{array}{l} e \text{ is a kinetic expression,} \\ \text{and } S \text{ and } S' \text{ are solutions of molecules in } \mathcal{M}\}. \end{array}$$

The domain of SBML reaction models is $\mathcal{C}_{\mathcal{R}} = (\mathcal{P}(\mathcal{R}), \subseteq)$.

Stochastic Semantics

For a given volume V_k of the location where the compound x_k resides, a concentration C_k for a molecule is translated into a number of molecules $N_k = \lfloor C_k \times V_k \times N_A \rfloor$, where N_A is Avogadro's number.

The kinetic expression e_i for each reaction i evaluates on numbers of molecules for each compound, instead of concentrations, in a (positive) reaction weight τ_i.

An element s of the domain precisely defines a Markov chain, where the probability p_{ij} of transition from state S_i to S_j is obtained by normalizing the reaction rate $\tau_{i,j} = \sum_{(S_i,S_j,\tau)\in s} \tau$ in

$$p_{ij} = \frac{\tau_{ij}}{\sum_{(S_i,S_k,\tau_{ik})\in s} \tau_{ik}}$$

Stochastic Semantics Domain

Def. 3 *Let a discrete state be a vector of integers of dimension $|\mathcal{M}|$. The universe $\mathcal{S}$ of stochastic transitions is the set of triplets (S_i, S_j, τ_{ij}) where S_i and S_j are discrete states and $\tau_{ij} \in \mathbb{R}^+$.*

The domain of stochastic transitions is $\mathcal{D}_\mathcal{S} = (\mathcal{P}(\mathcal{S}), \subseteq)$.

Discrete states and solutions in reaction rules have the same mathematical structure, and can both be represented by $|\mathcal{M}|$-dimensional vectors of integers.

Proposition 4 *Let $\alpha_{\mathcal{RS}} : \mathcal{C}_\mathcal{R} \to \mathcal{D}_\mathcal{S}$ be the function associating to a reaction model the state transition graph labelled with thte $\tau_{i,j}$'s. Let $\gamma_{\mathcal{RS}}(s) = \cup {\alpha_{\mathcal{RS}}}^{-1}(\downarrow s)$. $\mathcal{C}_\mathcal{R} \underset{\gamma_{\mathcal{RS}}}{\overset{\alpha_{\mathcal{RS}}}{\longleftrightarrow}} \mathcal{D}_\mathcal{S}$ is a Galois connection.*

Discrete Semantics

Def. 6 *The universe $\mathcal{D}$ of discrete transitions is the set of pairs of discrete states. The domain of discrete transitions is $\mathcal{D}_\mathcal{D} = (\mathcal{P}(\mathcal{D}), \subseteq)$.*

The discrete semantics is the classical Petri net semantics of reaction models [RML93ismb,SHK06bmcbi,Chaouiya07bioinfo,GHL07cmsb].

Classical Petri net analysis tools can be used for the analysis of reaction For instance, the elementary mode analysis of metabolic networks [SPM02bioinfo] has been shown in [ZS03insilicobio] to be equivalent to the classical analysis of Petri nets by T-invariants.

Proposition 7 *Let $\alpha_{\mathcal{SD}} : \mathcal{D}_\mathcal{S} \to \mathcal{D}_\mathcal{D}$ be the function associating to a set of stochastic transitions the discrete transitions obtained by projection on the two first components, and $\gamma_{\mathcal{SD}}(d) = \cup {\alpha_{\mathcal{SD}}}^{-1}(\downarrow d)$. $\mathcal{D}_\mathcal{S} \underset{\gamma_{\mathcal{SD}}}{\overset{\alpha_{\mathcal{SD}}}{\longleftrightarrow}} \mathcal{D}_\mathcal{D}$ is a Galois connection.*

Boolean Semantics

Def. 9 *Let a boolean state be a vector of booleans of dimension $|\mathcal{M}|$ indicating the presence of each molecule in the state. The universe $\mathcal{B}$ of boolean transitions is the set of pairs of boolean states.*

The domain of boolean transitions is $\mathcal{D}_{\mathcal{B}} = (\mathcal{P}(\mathcal{B}), \subseteq)$.

Let $\alpha_{\mathcal{NB}} : \mathbb{N}^{|\mathcal{M}|} \to \mathbb{B}^{|\mathcal{M}|}$ be the zero/non-zero abstraction (or threshold abstraction) from the integers to the booleans, and its pointwise extension from discrete states to boolean states.

Proposition 10 *Let $\alpha_{\mathcal{DB}} : \mathcal{D}_{\mathcal{D}} \to \mathcal{D}_{\mathcal{B}}$ be the set extension of $\alpha_{\mathcal{NB}}$. Let $\gamma_{\mathcal{DB}}(b) = \cup {\alpha_{\mathcal{DB}}}^{-1}(\downarrow b)$. $\mathcal{D}_{\mathcal{D}} \underset{\gamma_{\mathcal{DB}}}{\overset{\alpha_{\mathcal{DB}}}{\longleftrightarrow}} \mathcal{D}_{\mathcal{B}}$ is a Galois connection.*

BIOCHAM Boolean Semantics

Given a reaction model R, let us denote by S_{BB} the set of boolean transitions obtained by considering all pssible consumption of reactants.

For instance, a rule like `A+B=>C+D` is interpreted by four boolean transition rules :

- $A \wedge B \longrightarrow A \wedge B \wedge C \wedge D$
- $A \wedge B \longrightarrow \neg A \wedge B \wedge C \wedge D$
- $A \wedge B \longrightarrow A \wedge \neg B \wedge C \wedge D$
- $A \wedge B \longrightarrow \neg A \wedge \neg B \wedge C \wedge D$

Note that in Boolean Petri nets, or in Pathway Logic, complete consumption is assumed.

Proposition 11 *For any reaction model R, $\alpha_{\mathcal{DB}}(\alpha_{\mathcal{SD}}(\alpha_{\mathcal{RS}}(R))) \subseteq S_{BB}$.*

Differential Semantics ?

The differential semantics of reaction models interprets a set of reaction rules $\{e_i \texttt{ for } S_i => S'_i\}_{i=1,...,n}$ over molecular concentration variables $\{x_1, ..., x_m\}$, by the following system of Ordinary Differential Equations (ODE):

$$dx_k/dt = \sum_{i=1}^{n} r_i(x_k) * e_i - \sum_{j=1}^{n} l_j(x_k) * e_j$$

where we recall that $r_i(x_k)$ (resp. l_i) is the stoichiometric coefficient of x_k in the right (resp. left) member of rule i.

- synchronous semantics (evolution of variables in parallel)
- deterministic semantics (average behavior)
- not compatible with the rule set inclusion ordering
- infinite number of molecules
- infinitesimal time steps

Type Checking/Inference by Abstract Interpretation

A type system A for a concrete domain C is a Galois connection $\mathcal{C} \rightarrow_\alpha \mathcal{A}$.

The type inference problem is

NPUT a concrete element $x \in \mathcal{C}$ (e.g. a reaction model)

TPUT its typing $\alpha(x)$ (e.g. the protein functions of the model).

The type checking problem is,

NPUT $x \in \mathcal{C}$ (e.g. a reaction model)
and a typing $y \in \mathcal{A}$ (e.g. a set of protein functions),

TPUT determine whether $x \sqsubseteq_{\mathcal{C}} \gamma(y)$
(i.e. whether the reactions are compatible with the protein functions)
or equivalently $\alpha(x) \sqsubseteq_{\mathcal{A}} y$ (the typing contains the inferred types)

Algorithms in $O(n)$ if the abstractions can be computed rule per rule.

Protein Functions as Types

Abstract domain $\mathcal{A}_\mathcal{F} = \mathcal{P}(\{\text{kinase}(A)|A \in \mathcal{M}\} \cup \{\text{phosphatase}(A)|A \in \mathcal{M}\})$

The typing of reactions by protein functions is defined by the abstraction :

$\alpha_\mathcal{F}$(`A =[B]=> C`) = {`kinase(B)`} if `C` is strictly more phosphorylated than `A`

$\alpha_\mathcal{F}$(`A =[B]=> C`) = {`phosphatase(B)`} if `C` is strictly less phosphorylated

$\alpha_\mathcal{F}$(`A + B => A-B, A-B => C + B`) = { `kinase(B)`}

if `C` is strictly more phosphorylated than `A`

$\alpha_\mathcal{F}$(`A + B => A-B, A-B => C + B`) = { `phosphatase(B)`}

if `C` is strictly less phosphorylated than `A`

Proposition 12 *Let* $\gamma_\mathcal{F}(f) = \cup {\alpha_\mathcal{F}}^{-1}(\downarrow f)$, $\mathcal{C}_\mathcal{R} \underset{\gamma_\mathcal{F}}{\overset{\alpha_\mathcal{F}}{\longleftrightarrow}} \mathcal{A}_\mathcal{F}$ *is a Galois connection.*

Evaluation Results in BIOCHAM

- MAPK model [Levchenko et al. 00]
  the kinase function of `RAFK`, `RAF~{p1}` and `MEK~{p1,p2}` is inferred;
  the phosphatase function of `RAFPH`, `MEKPH` and `MAPKPH` is inferred;
  the kinase function of `MAPK~{p1,p2}` is not visible and not inferred.
- Model of the mammalian cell cycle control after [Kohn 99] 165 proteins and genes, 500 variables and 800 rules. Type inference in $<$ 1sec CPU :
  - No compound is both a kinase and a phosphatase;
  - `cdc25A` and `cdc25C` are the only phosphatases found together with the deacetylase `HDAC1`.
  - The `cdk` are inferred to be kinases only in complexes with cyclins;
  - the acetylases `pCAF`, `p300` are identified to kinases.

Differential Influence Graph as Types

The Jacobian matrix J is formed of the partial derivatives $J_{ij} = \partial \dot{x_i}/\partial x_j$.

The *differential influence graph* is defined by

$$\alpha_{\mathcal{JI}}(R) = \{A \text{ activates } B \mid \partial \dot{x_B}/\partial x_A > 0 \text{ in some point of the space}\}$$
$$\cup \{A \text{ inhibits } B \mid \partial \dot{x_B}/\partial x_A < 0 \text{ in some point of the space}\}$$

Not immediate to compute.

Syntactical Influence Graph

$$\text{Let } \alpha_{\mathcal{RI}}(R) = \{A \text{ activates } B \mid \exists (e_i \text{ for } l_i \Rightarrow r_i) \in x, \; l_i(A) > 0 \text{ and } r_i(B) - l_i(B) > 0\}$$
$$\cup \{A \text{ inhibits } B \mid \exists (e_i \text{ for } l_i \Rightarrow r_i) \in R, \; l_i(A) > 0 \text{ and } r_i(B) - l_i(B) < 0\}$$

We have $\alpha_{\mathcal{RI}}(\{\mathtt{A + B => C}\}) = \{$ A inhibits B, A inhibits A, B inhibits A, B inhibits B, A activates C, B activates C$\}$

$\alpha_{\mathcal{RI}}(\{\mathtt{A = [C] => B}\}) = \{$ C inhibits A, A inhibits A, A activates B, C activates B$\}$

$\alpha_{\mathcal{RI}}(\{\mathtt{A = [B] => \_}\}) = \{$ B inhibits A, A inhibits A$\}$

$\alpha_{\mathcal{RI}}(\{\mathtt{\_ = [B] => A}\}) = \{$ B activates A$\}$

Note that $\alpha_{\mathcal{RI}}$ is computable in $O(n)$ time

(0.2 sec. on the 800 rules of Kohn's map in Biocham)

René Thomas's Conditions Apply on Influence Graphs

[Thomas 81] :

The existence of negative circuits is a necessary condition for oscillations.

The existence of positive circuits is a necessary condition for multistationarity.

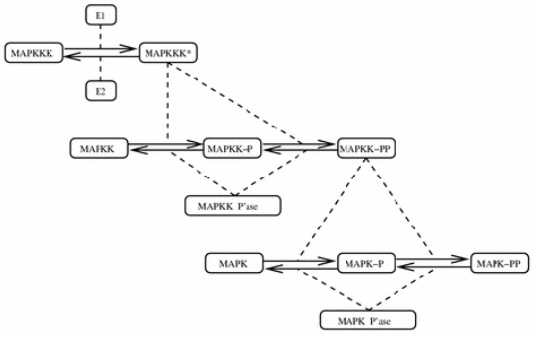
Proved for :

ODE systems [Soulé 03] [Demonjeot et al. 03] ...

boolean networks [Rémy Ruet Thieffry 05]

discrete networks [Richard 06]

MAPK Signalling Cascade

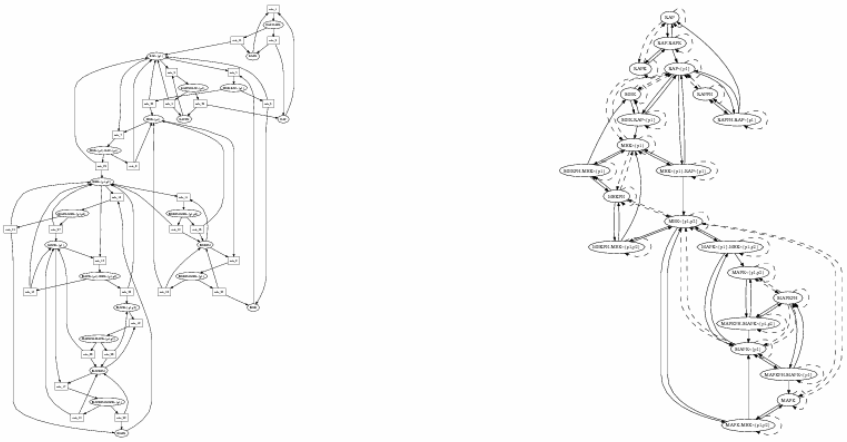


Purely directional cascade of reactions: no negative feedback

oscillations observed [Qiao et al. 07]

violation of René Thomas's rule [Kholodenko et al. 06] !

MAPK Reaction and Influence Graphs in BIOCHAM



Negative feedback by sequestration [Sepulchre et al. 08]

Monotonic Kinetics

In a reaction model $R = \{e_i$ `for` l_i`=>`$r_i \mid i \in I\}$, we say that a kinetic expression e_i is *monotonic* iff for all molecules x_k we have

1. $\partial e_i/\partial x_k \geq 0$ in all points of the space,
2. $l_i(x_k) > 0$ whenever $\partial e_i/\partial x_k > 0$ in some point of the space.

The mass action law kinetics, $e = k * \Pi {x_i}^{l_i}$, Michaelis-Menten and Hill's kinetics $e_i = V_m * {x_s}^n/(K_m + {x_s}^n)$ are monotonic.

Inhibitions with negative Hill kinetics of the form $e_i = V_m/(K_m + {x_s}^n)$ are not monotonic.

Monotonic Kinetics

In a reaction model $R = \{e_i \texttt{ for } l_i \texttt{=>} r_i \mid i \in I\}$, we say that a kinetic expression e_i is *monotonic* iff for all molecules x_k we have

1. $\partial e_i / \partial x_k \geq 0$ in all points of the space,
2. $l_i(x_k) > 0$ whenever $\partial e_i / \partial x_k > 0$ in some point of the space.

The mass action law kinetics, $e = k * \Pi {x_i}^{l_i}$, Michaelis-Menten and Hill's kinetics $e_i = V_m * {x_s}^n / (K_m + {x_s}^n)$ are monotonic.

Inhibitions with negative Hill kinetics of the form $e_i = V_m / (K_m + {x_s}^n)$ are not monotonic.

Over-approximation Theorem

Theorem 1 *For any reaction model R with monotonic kinetics, $\alpha_{\mathcal{JI}}(R) \subseteq \alpha_{\mathcal{RI}}(R)$.*

PROOF: If (A activates B) $\in \alpha_{\mathcal{JI}}(R)$ then $\partial \dot{B} / \partial A > 0$. Hence there exists a term in the differential semantics, of the form $(r_i(B) - l_i(B)) * e_i$ with $\partial e_i / \partial A$ of the same sign as $r_i(B) - l_i(B)$.

Let us suppose that $r_i(B) - l_i(B) > 0$ then $\partial e_i / \partial A > 0$ and since e_i is monotonic we get that $l_i(A) > 0$ and thus that (A activates B) $\in \alpha_{\mathcal{RI}}(R)$. If on the contrary $r_i(B) - l_i(B) < 0$ then $\partial e_i / \partial A < 0$, which is not possible for a monotonic kinetics.

Similarly for (A inhibits B). □

Strict inclusion for $R = \{k_1 * A \texttt{ for } A \texttt{=>} B.\ k_2 * A \texttt{ for } \_ = [A] \texttt{=>} A\}$ as $\dot{A} = (k_2 - k_1) * A$ can be made always positive, null or negative.

Corollary 2 *$\alpha_{\mathcal{JI}}(R, IC) \subseteq \alpha_{\mathcal{RI}}(R)$ in the phase space w.r.t. some initial conditions IC.*

Strongly Monotonic Kinetics

In a reaction model $R = \{e_i \texttt{ for } l_i \texttt{=>} r_i \mid i \in I\}$, a kinetic expression e_i is *strongly monotonic* iff for all molecules x_k we have

1. $\partial e_i / \partial x_k \geq 0$ in all points of the space,
2. $l_i(x_k) > 0$ iff there exists a point in the space s.t. $\partial e_i / \partial x_k > 0$

Proposition 3 *Mass action law, Michaelis Menten, and Hill kinetics are strongly monotonic.*

Lemma 4 *Let R be a reaction model with strongly monotonic kinetics.*

If (A activates B) is in $\alpha_{\mathcal{RI}}(R)$ but (A inhibits B) is not in $\alpha_{\mathcal{RI}}(R)$

then (A activates B) is in $\alpha_{\mathcal{JI}}(R)$.

If (A inhibits B) is in $\alpha_{\mathcal{RI}}(R)$ but (A activates B) is not in $\alpha_{\mathcal{RI}}(R)$

then (A inhibits B) is in $\alpha_{\mathcal{JI}}(R)$.

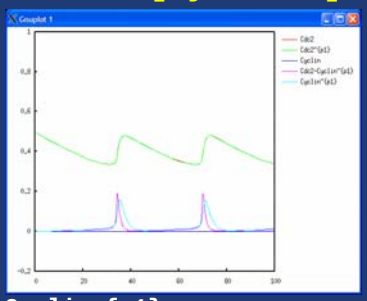
Equivalence Theorem

Theorem 5 *Let R be a reaction model with strongly monotonic kinetics and where no molecule is at the same time an activator and an inhibitor of the same target molecule, then $\alpha_{\mathcal{RI}}(R) = \alpha_{\mathcal{JI}}(R)$.*

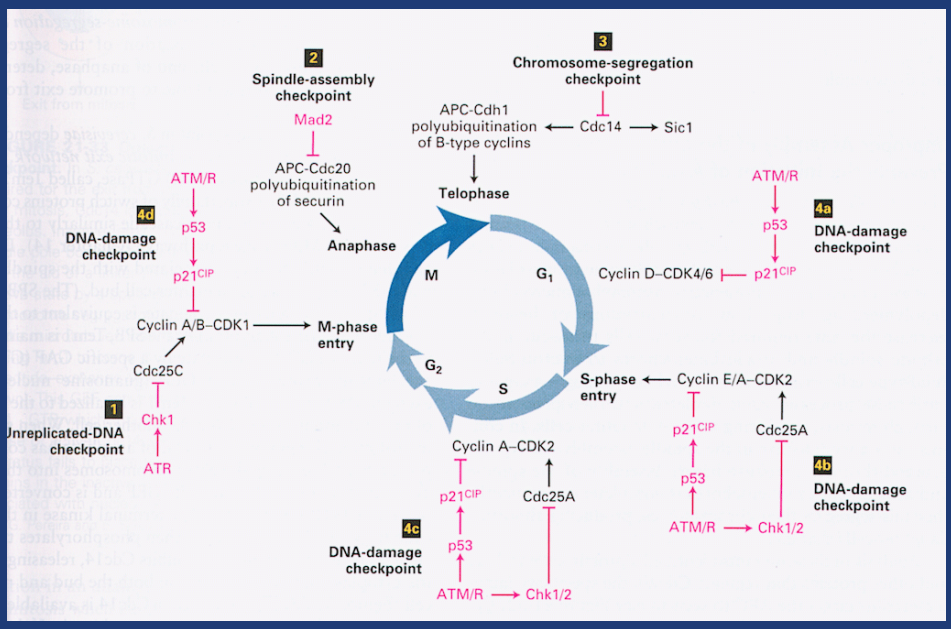
Corollary 6 *The graph of differential influences of a reaction model of n rules with strongly monotonic kinetics is computable in time $O(n)$ if no molecule is at the same time an activator and an inhibitor.*

Corollary 7 *The graph of differential influences of a reaction model is independent of the kinetic expressions as long as they are strongly monotonic, if no molecule is at the same time an activator and an inhibitor.*

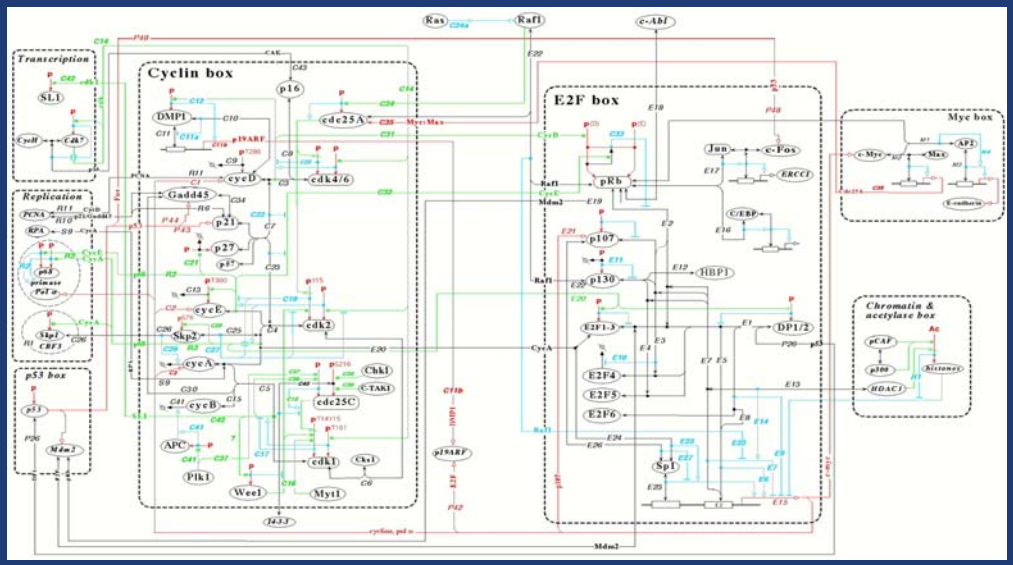
Budding Yeast Cell Cycle Control Model [Tyson 91]



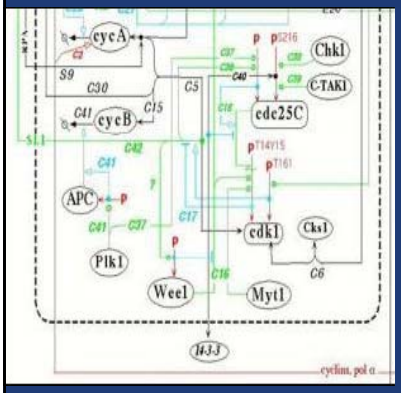
```
MA(k1) for  _ => Cyclin.
MA(k2) for  Cyclin => _.
MA(K7) for  Cyclin~{p1} => _.

MA(k8) for  Cdc2 => Cdc2~{p1}.
MA(k9) for  Cdc2~{p1} =>Cdc2.

MA(k3) for  Cyclin+Cdc2~{p1} => Cdc2~{p1}-Cyclin~{p1}.
MA(k4p) for Cdc2~{p1}-Cyclin~{p1} => Cdc2-Cyclin~{p1}.
k4*[Cdc2-Cyclin~{p1}]^2*[Cdc2~{p1}-Cyclin~{p1}] for
    Cdc2~{p1}-Cyclin~{p1} =[Cdc2-Cyclin~{p1}] => Cdc2-Cyclin~{p1}.

MA(k5) for Cdc2-Cyclin~{p1} => Cdc2~{p1}-Cyclin~{p1}.
MA(k6) for Cdc2-Cyclin~{p1} => Cdc2+Cyclin~{p1}.
```

Mammalian Cell Cycle Control Map [Kohn 99]



Transcription of Kohn's Map



_ =[E2F13-DP12-gE2]=> cycA.
...
cycB =[APC~{p1}]=>_.
cdk1~{p1,p2,p3} + cycA => cdk1~{p1,p2,p3}-cycA.
cdk1~{p1,p2,p3} + cycB => cdk1~{p1,p2,p3}-cycB.
...
cdk1~{p1,p3}-cycA =[Wee1]=> cdk1~{p1,p2,p3}-cycA.
cdk1~{p1,p3}-cycB =[Wee1]=> cdk1~{p1,p2,p3}-cycB.
cdk1~{p2,p3}-cycA =[Myt1]=> cdk1~{p1,p2,p3}-cycA.
cdk1~{p2,p3}-cycB =[Myt1]=> cdk1~{p1,p2,p3}-cycB.
...
cdk1~{p1,p2,p3} =[cdc25C~{p1,p2}]=> cdk1~{p1,p3}.
cdk1~{p1,p2,p3}-cycA =[cdc25C~{p1,p2}]=> cdk1~{p1,p3}-cycA.
cdk1~{p1,p2,p3}-cycB =[cdc25C~{p1,p2}]=> cdk1~{p1,p3}-cycB.

165 proteins and genes, 500 variables, 800 rules [Chiaverini Danos 02]

Computation Tree Logic CTL Properties

```
Ei(reachable(Cyclin)))
Ei(reachable(!(Cyclin))))
Ai(oscil(Cyclin)))
Ei(reachable(Cdc2~{p1})))
Ei(reachable(!(Cdc2~{p1}))))
Ai(oscil(Cdc2~{p1})))
Ei(reachable(Cdc2-Cyclin~{p1,p2})))
Ei(reachable(!(Cdc2-Cyclin~{p1,p2}))))
Ai(oscil(Cdc2-Cyclin~{p1,p2})))
Ei(reachable(Cdc2-Cyclin~{p1})))
Ei(reachable(!(Cdc2-Cyclin~{p1}))))
Ai(oscil(Cdc2-Cyclin~{p1})))
Ai(AG(!(Cdc2~{p1})->checkpoint(Cdc2,Cdc2~{p1}))))
Ai(AG(!(Cdc2-Cyclin~{p1})->checkpoint(Cdc2-Cyclin~{p1,p2},Cdc2-
  Cyclin~{p1})))
...
```

reachable(P) = EF(P)
steady(P) = EG(P)
stable(P) = AG(P)
checkpoint(P,Q) = !E(!P U Q)
oscil(P) = EG(F P ^ F !P)
~ EG(EF P ^ EF !P)

Cell Cycle Model-Checking (with BDD NuSMV)

biocham: check_reachable(cdk46~{p1,p2}-cycD~{p1}).
Ei(EF(cdk46~{p1,p2}-cycD~{p1})) is true
biocham: check_checkpoint(cdc25C~{p1,p2}, cdk1~{p1,p3}-cycB).
Ai(!(E(!(cdc25C~{p1,p2}) U cdk1~{p1,p3}-cycB))) is true
biocham: nusmv(Ai(AG(!(cdk1~{p1,p2,p3}-cycB) -> checkpoint(Wee1, cdk1~{p1,p2,p3}-cycB)))).
Ai(AG(!(cdk1~{p1,p2,p3}-cycB)->!(E(!(Wee1) U cdk1~{p1,p2,p3}-cycB)))) is false
biocham: why.
-- Loop starts here
cycB-cdk1~{p1,p2,p3} is present
cdk7 is present
cycH is present
cdk1 is present
Myt1 is present
cdc25C~{p1} is present
rule_114 cycB-cdk1~{p1,p2,p3}=[cdc25C~{p1}]=>cycB-cdk1~{p2,p3}.
cycB-cdk1~{p2,p3} is present
cycB-cdk1~{p1,p2,p3} is absent
rule_74 cycB-cdk1~{p2,p3}=[Myt1]=>cycB-cdk1~{p1,p2,p3}.
cycB-cdk1~{p2,p3} is absent
cycB-cdk1~{p1,p2,p3} is present

Mammalian Cell Cycle Control Benchmark

500 variables, 2^{500} states. 800 rules. BIOCHAM NuSMV model-checker time in sec. [Chabrier Chiaverini Danos Fages Schachter TCS 04]

Initial state G2	Query:	Time:
	compiling	29 s
Reachability G1	EF CycE	2 s
Reachability G1	EF CycD	1.9 s
Reachability G1	EF PCNA-CycD	1.7 s
Checkpoint for mitosis complex	¬EF (¬ Cdc25~{Nterm} U Cdk1~{Thr161}-CycB)	2.2 s
Oscillation	EG ((CycA => EF ¬ CycA) ^ (¬ CycA => EF CycA))	31.8 s

Learning Model Revision from Temporal Properties

- *Theory T*: BIOCHAM model
  - molecule declarations
  - reaction rules: complexation, phosphorylation, …
- *Examples φ*: CTL specification of biological properties
  - Reachability
  - Checkpoints
  - Stable states
  - Oscillations
- *Bias R*: Rule pattern
  - Kind of rules to add or delete

Find a revision T' of T such that $T' \models \varphi$

Complexity of Model-checking and Satisfiability

	Model-checking	Satisfiability
	given an explicit Kripke structure K and a formula ϕ, does K,s $\models \phi$?	given a formula ϕ, does there exist a structure K,s such that K,s $\models \phi$?
LTL, LTL(U) :	Pspace complete	Pspace complete
LTL(F) :	NP-complete	NP-complete
CTL :	Ptime	DetExpTime complete
CTL* :	Pspace complete	DetExpExpTime complete

Simple Model of Cell Cycle Control

[Tyson et al. 91] model over 6 variables,
initial state present(cdc2).

```
_=>Cyclin.
Cyclin=>_.
Cyclin+Cdc2~{p1}=>Cdc2-Cyclin~{p1,p2}.
Cdc2-Cyclin~{p1,p2}=>Cdc2-Cyclin~{p1}.
Cdc2-Cyclin~{p1,p2}=[Cdc2-Cyclin~{p1}]=>Cdc2-Cyclin~{p1}.
Cdc2-Cyclin~{p1}=>Cdc2-Cyclin~{p1,p2}.
Cdc2-Cyclin~{p1}=>Cyclin~{p1}+Cdc2.
Cyclin~{p1}=>_.
Cdc2=>Cdc2~{p1}.
Cdc2~{p1}=>Cdc2.
```


(Aut. Generated) CTL Specification of the Model

```
biocham: add_genCTL.
    reachable(Cyclin).
    reachable(!(Cyclin)).
    oscil(Cyclin).
    reachable(Cdc2~{p1}).
    reachable(!(Cdc2~{p1})).
    checkpoint(Cdc2, Cdc2~{p1}).
    oscil(Cdc2).
    …
    reachable(Cyclin~{p1}).
    reachable(!(Cyclin~{p1}))
    oscil(Cyclin~{p1}).
    checkpoint(Cdc2-Cyclin~{p1}, Cyclin~{p1}).
```


Model Compression

```
biocham: reduce_model.
    1: deleting Cyclin=>_
    2: deleting Cdc2-Cyclin~{p1,p2}=[Cdc2-Cyclin~{p1}]=>Cdc2-
    Cyclin~{p1}
    3: deleting Cdc2-Cyclin~{p1}=>Cdc2-Cyclin~{p1,p2}
    4: deleting Cdc2~{p1}=>Cdc2
    After reduction, 6 rules remain corresponding to the bias ? => ?
    Deletion(s):
    Cyclin=>_.
    Cdc2-Cyclin~{p1,p2}=[Cdc2-Cyclin~{p1}]=>Cdc2-Cyclin~{p1}.
    Cdc2-Cyclin~{p1}=>Cdc2-Cyclin~{p1,p2}.
    Cdc2~{p1}=>Cdc2.
```


Theory Revision

```
biocham: delete_rules(Cdc2=>Cdc2~{p1}).
   Cdc2=>Cdc2~{p1}
biocham: revise_model.
   1: adding Cdc2-Cdc2~{p1}=>Cdc2+Cdc2~{p1}
   2: adding Cdc2=>_
   2: backtracking on previous add -> deleting Cdc2=>_
   2: adding Cdc2=[Cyclin]=>_
   2: backtracking on previous add -> deleting Cdc2=[Cyclin]=>_
   2: adding Cdc2=[Cdc2-Cdc2~{p1}]=>_
   3: adding Cdc2=>Cdc2~{p1}
   4: deleting Cdc2=[Cdc2-Cdc2~{p1}]=>_
   5: deleting Cdc2-Cdc2~{p1}=>Cdc2+Cdc2~{p1}
   Modifications found:
    Deletion(s):
    Addition(s): Cdc2=>Cdc2~{p1}.
```


Search for all Solutions

```
biocham: learn_one_addition(elementary_interaction_rules).
   Time: 5.00 s
   Rules tested: 112
   Good rules to be added: 2
   Cdc2=>Cdc2~{p1}
   Cdc2=[Cyclin]=>Cdc2~{p1}
```


Theory Revision Algorithm

General idea of constraint programming: replace a *generate-and-test* algorithm by a *constrain-and-generate* algorithm.

Anticipate whether one has to *add or remove a rule*?

- *Positive* ECTL formula: if false, remains false after removing a rule
  - Reachability, stability
  - *Need to add rules*
- *Negative* ACTL formula: if false, remains false after adding a rule
  - Checkpoints
  - *Need to remove a rule on the path given by the model checker*
- Unclassified CTL formulae
  - oscil(a)= AG((a ⇒ EF¬a)^(¬a ⇒ EFa))

Optimisations

Restrict the search space for rules to add by:

- Considering *type information* on molecular species
  - Kinase(A) B=[A]=>B~{p}. for any B
  - Phosphatase(A) B~{p}=[A]=>B. for any B
  - Kinase(A,B)
  - Phosphatase(A,B)
- Considering the *influence graph* between molecular species
  - Activates(A,B) _=[A]=>B. A+B'=>B. B~{p}=[A]=>B. B'=[A]=>B.
  - Inhibits(A,B) B=[A]=>_. A+B=>A-B. B=[A]=>B~{p}. B=[A]=>B'.
- Considering the topology of *locations*
  - Neighbor(L,L') A:L+…=>B:L'+…

LTL(R) with Constraints for the Differential Semantics

- Constraints over *concentrations* and *derivatives* as FOL formulae over the reals:
  - [M] > 0.2
  - [M]+[P] > [Q]
  - d([M])/dt < 0
- Linear Time Logic LTL operators for time X, F, U, G
  - F([M]>0.2)
  - FG([M]>0.2)
  - F ([M]>2 & F (d([M])/dt<0 & F ([M]<2 & d([M])/dt>0 & F(d([M])/dt<0))))
  - oscil(M,n) defined as at least n alternances of sign of the derivative
  - Period(A,75)= ∃ t ∃v F(T = t & [A] = v & d([A])/dt > 0 & X(d([A])/dt < 0)
    & F(T = t + 75 & [A] = v & d([A])/dt > 0 & X(d([A])/dt < 0)))…

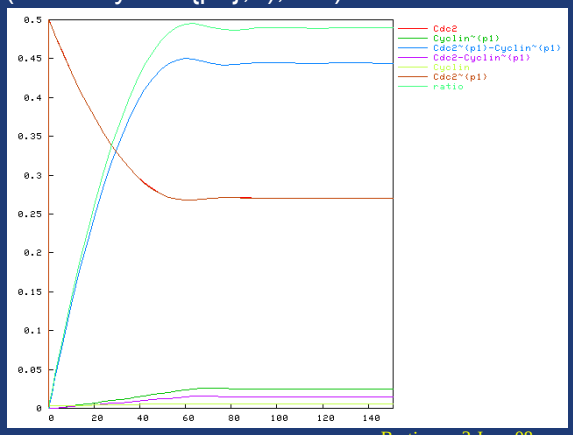
Inferring Parameters from LTL(R) Specification

biocham: learn_parameter([k3,k4],[(0,200),(0,200)],20,
oscil(Cdc2-Cyclin~{p1},3),150).

Inferring Parameters from LTL(R) Specification

biocham: learn_parameter([k3,k4],[(0,200),(0,200)],20,
oscil(Cdc2-Cyclin~{p1},3),150).

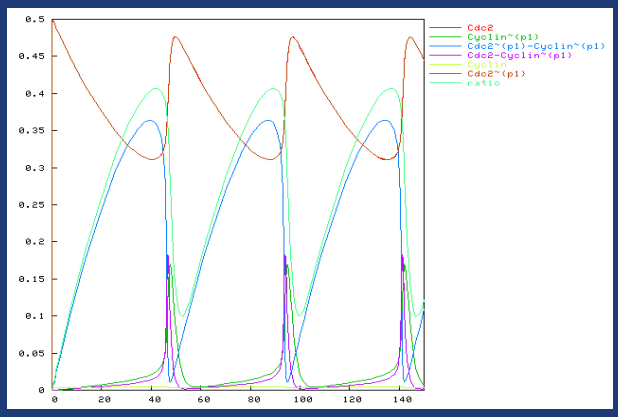
First values found :
parameter(k3,10).
parameter(k4,70).



Inferring Parameters from LTL(R) Specification

biocham: learn_parameter([k3,k4],[(0,200),(0,200)],20,
oscil(Cdc2-Cyclin~{p1},3) & F([Cdc2-Cyclin~{p1}]>0.15), 150).

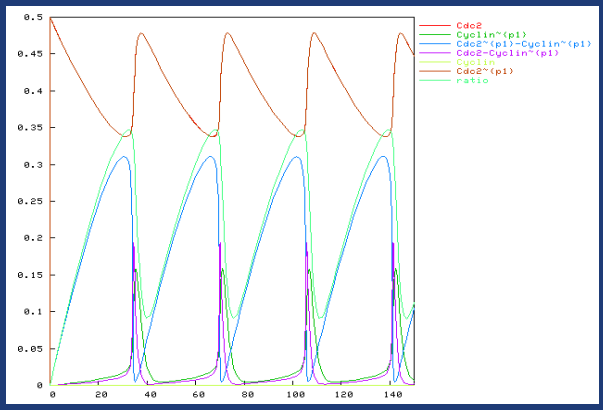
First values found :
parameter(k3,10).
parameter(k4,120).



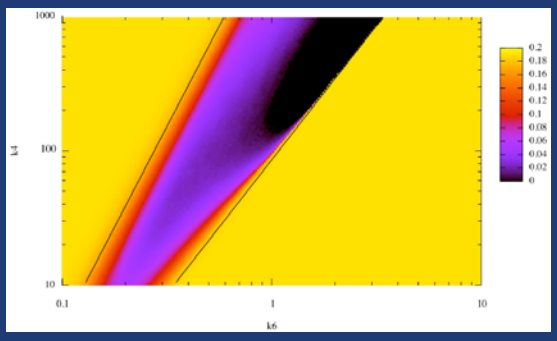
Inferring Parameters from LTL(R) Specification

biocham: learn_parameter([k3,k4],[(0,200),(0,200)],20,
period(Cdc2-Cyclin~{p1},35), 150).

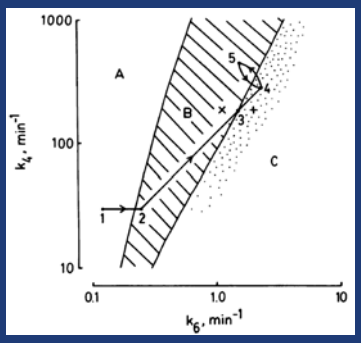
First values found:
parameter(k3,10).
parameter(k4,280).



LTL(R) Satisfaction Degree and Bifurcation Diagram

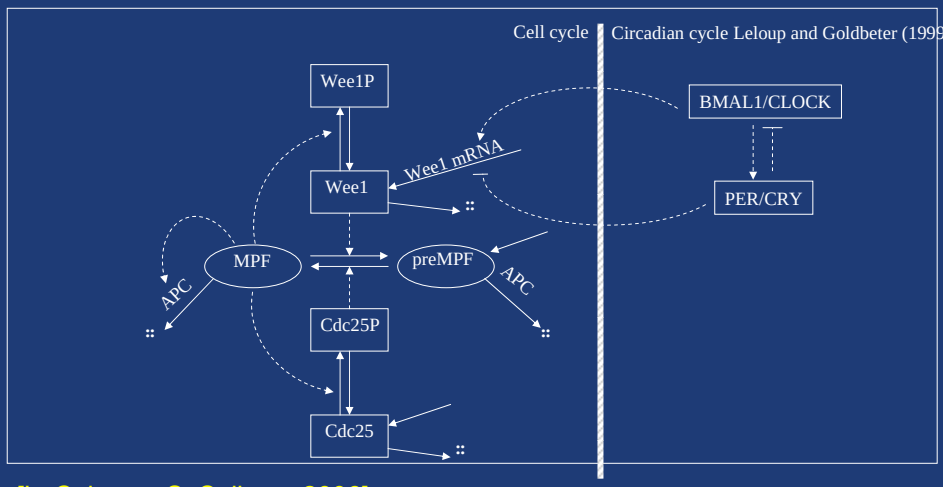


Satisfaction degree of LTL(R) formulas for oscillation with amplitude constraint
[Rizk Batt Fages Soliman 08]

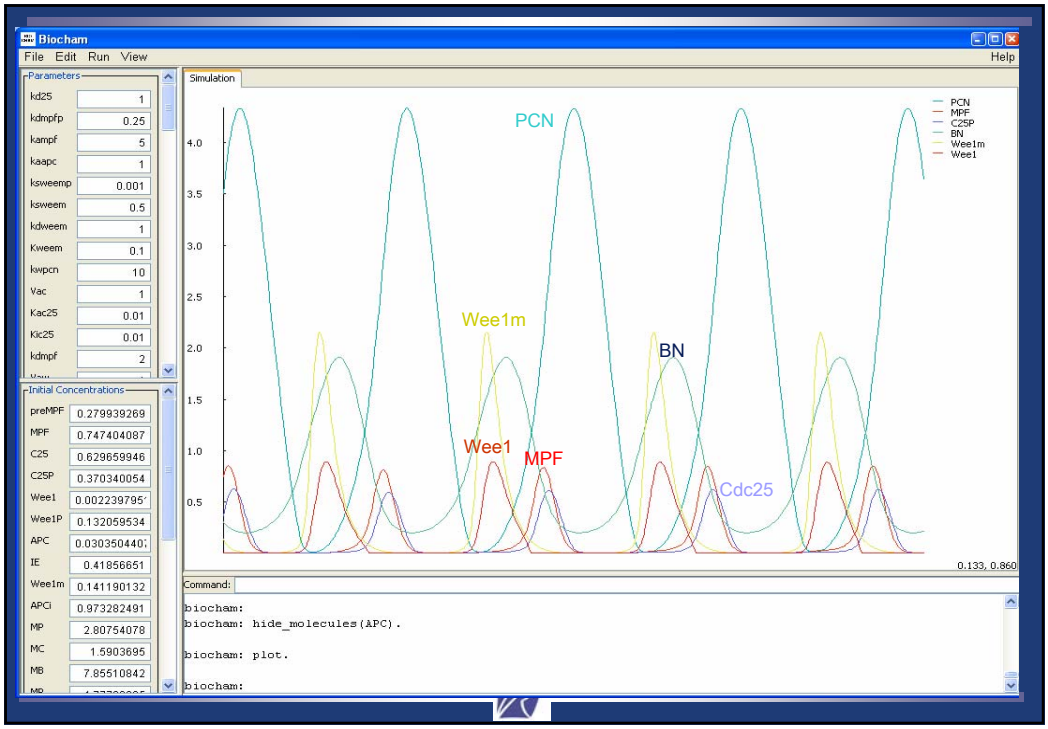


Bifurcation diagram on k4, k6
[Tyson 91]

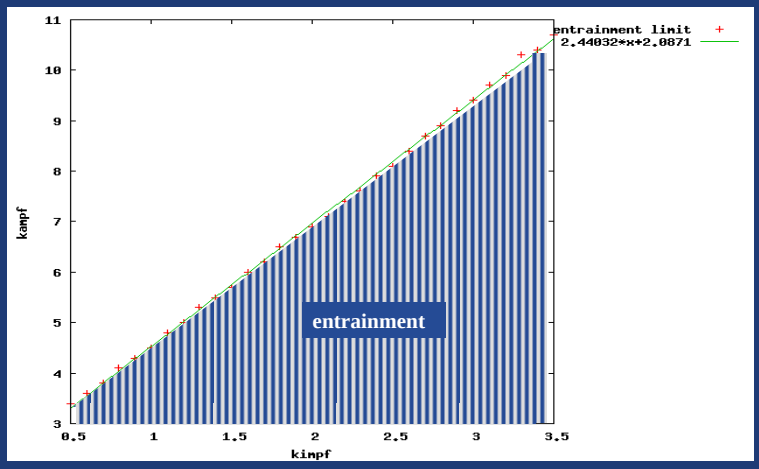
Linking the Cell and Circadian Cycles through Wee1



[L. Calzone, S. Soliman 2006]



Condition on Wee1/Cdc25 for the Entrainment in Period



Entrainment in period constraint expressed in LTL with the period formula

Conclusion

- New focus in Systems Biology: formal methods
  - Beyond diagrammatic notations: formal syntax, semantics, abstract interpretation
  - Beyond curve fitting: formalization of biological experiments in Temporal Logic
  - Model-checking, parameter search from temporal properties
- New focus in Programming Theory: numerical methods
  - Beyond discrete machines: continuous dynamics, hybrid systems
  - Quantitative transition systems
  - Temporal logic with numerical constraints
- "Computer" Science as science of complexity
  - Beyond tools: concepts and methods applicable to other sciences

Collaborations

EU STREP APrIL2 : Stephen Muggleton, IC, Luc de Raedt, U. Freiburg,…

- *Learning in a probabilistic logic setting (finished)*

EU NoE REWERSE : semantic web, François Bry, Münich, R. Backofen,

- *Connecting Biocham to gene and protein ontologies/types (finished)*

EU STREP TEMPO : Cancer chronotherapies, INSERM Villejuif, F. Lévi;

- Coupled models of cell cycle, circadian cycle, cytotoxic drugs.

INRA Tours : E. Reiter, D. Heitzler, INRIA F. Clément

- Models of Angiotensine and FSH signaling.

Evry Epigenomic project, AIV "Frontières du vivant" (ENS, Necker)

- New tools for synthetic biology

Language-based Approaches to Cell Systems Biology

Qualitative models: from *diagrammatic notation* to

- Boolean networks [Kaufman 69, Thomas 73]
- Petri Nets [Reddy 93, Chaouiya 05]
- *Process algebra π–calculus* [Regev-Silverman-Shapiro 99-01, Nagasali et al. 00]
- Bio-ambients [Regev-Panina-Silverman-Cardelli-Shapiro 03]
- Pathway logic [Eker-Knapp-Laderoute-Lincoln-Meseguer-Sonmez 02]
- Reaction rules [Chabrier-Fages 03] [Chabrier-Chiaverini-Danos-Fages-Schachter 04]

Quantitative models: from *ODEs* and *stochastic simulations* to

- Hybrid Petri nets [Hofestadt-Thelen 98, Matsuno et al. 00]
- Hybrid automata [Alur et al. 01, Ghosh-Tomlin 01] HCC [Bockmayr-Courtois 01]
- Stochastic π–calculus [Priami et al. 03] [Cardelli et al. 06]
- Reaction rules with continuous time dynamics [Fages-Soliman-Chabrier 04]

Influence Graphs as Types

$$\begin{aligned}
\text{Let } \alpha_{\mathcal{RI}}(R) = \quad & \{A \text{ activates } B \quad | \ \exists (e_i \text{ for } l_i \Rightarrow r_i) \in x, \\
& \quad l_i(A) > 0 \text{ and } r_i(B) - l_i(B) > 0\} \\
& \cup \{A \text{ inhibits } B \quad | \ \exists (e_i \text{ for } l_i \Rightarrow r_i) \in R, \\
& \quad l_i(A) > 0 \text{ and } r_i(B) - l_i(B) < 0\}
\end{aligned}$$

We have $\alpha_{\mathcal{RI}}(\{\mathtt{A + B => C}\}) = \{$ A inhibits B, A inhibits A, B inhibits A, B inhibits B, A activates C, B activates C$\}$

$\alpha_{\mathcal{RI}}(\{\mathtt{A = [C] => B}\}) = \{$ C inhibits A, A inhibits A, A activates B, C activates B$\}$

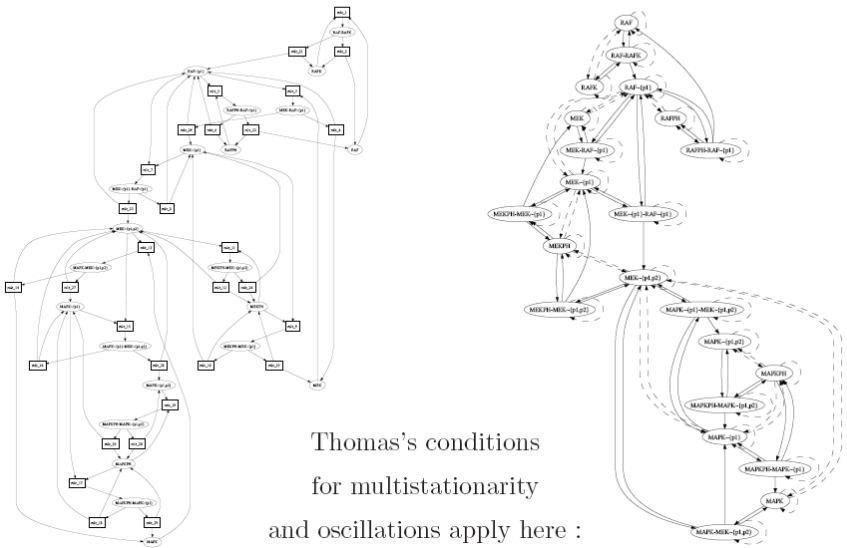
$\alpha_{\mathcal{RI}}(\{\mathtt{A = [B] => \_}\}) = \{$ B inhibits A, A inhibits A$\}$

$\alpha_{\mathcal{RI}}(\{\mathtt{\_ = [B] => A}\}) = \{$ B activates A$\}$

Note that $\alpha_{\mathcal{RI}}$ is computable in $O(n)$ time

(0.2 sec. on the 800 rules of Kohn's map in Biocham)

MAPK model: Reaction Graph $\rightarrow_\alpha$ Influence Graph



Thomas's conditions for multistationarity and oscillations apply here :

Influence Graph Abstraction from the Differential Semantics

Let us denote by β the mapping from $\mathcal{C}_{\mathcal{R}}$ to $\mathcal{D}_{\mathcal{J}}$ that extracts $\dot{x_k}$ and hence the Jacobian from the kinetic expressions in the reaction rules.

Def. 14 *The differential influence abstraction* $\alpha_{\mathcal{JI}} : \mathcal{D}_{\mathcal{J}} \rightarrow \mathcal{A}_{\mathcal{I}}$ *is the function*

$$\alpha_{\mathcal{JI}}(x) = \{A \text{ activates } B \mid \partial \dot{x_B}/\partial x_A > 0 \text{ in some point of the phase space}\}$$
$$\cup \{A \text{ inhibits } B \mid \partial \dot{x_B}/\partial x_A < 0 \text{ in some point of the phase space}\}$$

defined purely from the kinetic expressions... compatibility with the rules ?

Kripke Semantics of CTL*

Kripke structure K=(S,R) where S is a set of states and R⊆SxS is *total*.

s |= ϕ if propositional formula ϕ is true in s,

s |= E ϕ if there is a path π from s such that π |= ϕ,

s |= A ϕ if for every path π from s, π |= ϕ,

π |= ϕ if s |= ϕ where s is the starting state of π,

π |= X ϕ if π^1 |= ϕ,

π |= ϕ1 U ϕ2 iff there exists k ≥ 0 such that π^k |= ϕ2 for all j < k π^j |= ϕ1.

π |= ϕ1 W ϕ2 iff ∀j π^j |= ϕ1 or ∃ k ≥ 0 π^k |= ϕ1∧ ϕ2 and ∀j < k π^j |= ϕ1.

F ϕ = (true U ϕ) π |= F ϕ if there exists k ≥ 0 such that π^k |= ϕ,

G ϕ = (ϕ W false) π |= G ϕ if for every k ≥ 0, π^k |= ϕ

Duality in CTL*

$\neg E\,\phi = A\,\neg\,\phi$

$\neg\,X\,\phi = X\,\neg\,\phi$

$\neg\,(\phi 1\ U\ \phi 2) = \neg\,\phi 2\ W\ \neg\,\phi 1$

$\neg\,F\,\phi = G\,\neg\,\phi$

CTL*(X) : fragment of CTL* without U, W, F, G

CTL*(U) : fragment of CTL* without X

CTL : fragment of CTL* with E, A immediately before X, F, G, U , W

ϕ can be identified to the set of states where it is true $\phi \sim \{s \in S : s \models \phi\}$

LTL : fragment of CTL* without E, A

LTL(U) : fragment of LTL without X

LTL(F) : fragment of LTL without X, U, W

Positive and Negative CTL Formulae

Let K = (S,R,L) and K' = (S,R',L) be two Kripke structures such that $R \subset R'$

Def. An ECTL (positive) formula is a CTL formula with no occurrence of A (nor negative occurrence of E).

Ex. : reachability $EF(\phi)$, steady $EG(\phi)$

Def. An ACTL (negative) formula is a CTL formula with no occurrence of E (nor negative occurrence of A).

Ex. : checkpoint $\neg E(\neg\phi_2 U\ \phi)$, stable $AG(\phi)$

Monotonicity of Positive ECTL Formulae

Let K = (S,R) and K' = (S,R') be two Kripke structures such that R⊂R'.

Proposition For any ECTL formula ϕ, if K',s $\not\models \phi$ then K,s $\not\models \phi$.

Proof We show that K,s $\models \phi$ implies K',s $\models \phi$ by induction on the proof of ϕ

If ϕ is propositionnal, s $\models \phi$ hence K',s $\models \phi$;

If $\phi=\phi1\&\phi2$ *(resp.* $\phi1|\phi2$*)* then by induction *K',*s$\models\phi1$ and *(resp.* or*) K',*s$\models\phi2$.

If $\phi=EX\ \phi1$ then $K,\pi \models X\ \phi1$ for some path π in K, hence in K', so $K,\pi^1\models \phi1$ and by induction $K',\pi^1\models \phi1$ hence K', $\pi \models X\ \phi1$

If $\phi=E(\phi1\ U\ \phi2)$ then $K,\pi \models \phi1\ U\ \phi2$ for some path π in K, hence in K', so there exists k $K,\pi^k\models \phi2$ and for all j<k $K,\pi^j\models \phi1$. By induction $K',\pi^k\models \phi2$ and for all j<k $K',\pi^j\models \phi1$ hence $K,\pi \models \phi1\ U\ \phi2$.

Anti-monotonicity of Negative ECTL Formulae

Let K = (S,R) and K' = (S,R') be two Kripke structures such that R⊂R'.

Proposition For any ACTL formula ϕ, if K,s $\not\models \phi$ then K',s $\not\models \phi$.

Proof If K,s $\not\models \phi$ then K,s $\models \neg\phi$ where $\neg\phi$ is an ECTL formula.
By the previous proposition, K',s $\models \neg\phi$ hence K',s $\not\models \phi$.

Theory Revision Algorithm Rules

Initial state: <(0, 0, 0), (E,U,A), R>

E transition: <(E,U,A), (E∪{e},U,A), R> → <(E∪{e},U,A), (E,U,A),R> if R |= e

E' transition: <(E,U,A), (E ∪{e},U,A), R> → <(E ∪{e},U,A), (E,U,A),R ∪ {r}>
if R |≠ e and ∀ f ∈{e} ∪ E ∪ U ∪ A, K ∪ {r} |= f

Theory Revision Algorithm Rules

Initial state: <(0, 0, 0), (E,U,A), R>

E transition: <(E,U,A), (E∪{e},U,A), R> → <(E∪{e},U,A), (E,U,A),R> if R |= e

E' transition: <(E,U,A), (E ∪{e},U,A), R> → <(E ∪{e},U,A), (E,U,A),R ∪ {r}>
if R |≠ e and ∀ f ∈{e} ∪ E ∪ U ∪ A, K ∪ {r} |= f

U transition: <(E,U,A), (0,U ∪{u},A), R > → <(E,U ∪ {u},A), (0,U,A),R> if R |= u

U' transition: <(E,U,A), (0,U ∪{u},A), R > → <(E,U ∪{u},A), (0,U,A),R ∪ {r}>
if R|≠u and ∀ f ∈ {u} ∪ E ∪ U ∪ A, R ∪ {r} |= f

U" transition: <(E,U,A), (0,U ∪ {u},A), R ∪ Re > → <(E,U ∪{u},A),(0,U,A), R>
if K, si|≠u and ∀ f ∈{u} ∪ E ∪ U ∪ A, R |= f

Theory Revision Algorithm Rules

Initial state: <(0, 0, 0), (E,U,A), R>

E transition: <(E,U,A), (E∪{e},U,A), R> → <(E∪{e},U,A), (E,U,A),R> if R |= e

E' transition: <(E,U,A), (E ∪{e},U,A), R> → <(E ∪{e},U,A), (E,U,A),R ∪ {r}>
if R |≠ e and ∀ f ∈{e} ∪ E ∪ U ∪ A, K ∪ {r} |= f

U transition: <(E,U,A), (0,U ∪{u},A), R > → <(E,U ∪ {u},A), (0,U,A),R> if R |= u

U' transition: <(E,U,A), (0,U ∪{u},A), R > → <(E,U ∪{u},A), (0,U,A),R ∪ {r}>
if R|≠u and ∀ f ∈ {u} ∪ E ∪ U ∪ A, R ∪ {r} |= f

U" transition: <(E,U,A), (0,U ∪ {u},A), R ∪ Re > → <(E,U ∪{u},A),(0,U,A), R>
if K, si|≠u and ∀ f ∈{u} ∪ E ∪ U ∪ A, R |= f

A transition: <(E,U,A), (0, 0,A ∪{a}), R > → <(E,U,A ∪{a}), (Ep,Up,A),R> if R |= a

A' transition: <(E∪Ep,U∪Up,A),(0,0,A∪{a}), R∪Re>→<(E,U,A∪{a}),(Ep,Up,A),R>
if R|≠ a, ∀ f ∈{u} [E ∪ U ∪ A, R |= f and Ep ∪ Up is the set of formulae no longer satisfied after the deletion of the rules in Re.

Termination

Proposition The model revision algorithm terminates.

Proof

The termination of the algorithm is proved by considering the lexicographic ordering over the couple < a, n >
where a is the number of unsatisfied ACTL formulae,
and n is the number of unsatisfied ECTL and UCTL formulae.

Each transition strictly decreases *a*,
or lets *a* unchanged and strictly decreases *n*.

Correctness

Proposition If the terminal configuration is of the form < (E,U,A), (0,0,0), R > then the model R satisfies the initial CTL specification.

Proof

Each transition maintains only true formulae in the satisfied set,

and preserves the complete CTL specification

in the union of the satisfied set and the untreated set.

Incompleteness

Two reasons:

1) The satisfaction of ECTL and UCTL formula is searched by adding only one rule to the model (transition E' and U')

2) The Kripke structure associated to a Biocham set of rules adds loops on terminal states. Hence adding or removing a rule may have an opposite deletion or addition of those loops.