Biological Systems as Reactive Systems

Luca Cardelli

Microsoft Research
Cambridge UK

Munich 2005-04-06
50 Years of Molecular Cell Biology

- Genes are made of DNA
  - Store digital information as sequences of 4 different nucleotides
  - Direct protein assembly through RNA and the Genetic Code

- Proteins (>10000) are made of amino acids
  - Process signals
  - Activate genes
  - Move materials
  - Catalyze reactions to produce substances
  - Control energy production and consumption

- Bootstrapping still a mystery
  - DNA, RNA, proteins, membranes are today interdependent. Not clear who came first
  - Separation of tasks happened a long time ago
  - Not understood, not essential
Towards **Systems Biology**

- Biologists now understand many of the cellular components
  - A whole team of biologists will typically study a single protein for years
  - When each component and each reaction is understood, the system is understood (?)

- But this has not led to understand how “the system” works
  - Behavior comes from complex chains of interactions between components
  - Predictive biology and pharmacology still rare
  - Synthetic biology still unreliable

- New approach: try to understand “the system”
  - Experimentally: massive data gathering and data mining (e.g. Genome projects)
  - Conceptually: modeling and analyzing networks (i.e. interactions) of components

- What kind of a system?
  - Just beyond the basic chemistry of energy and materials processing...
  - Built right out of digital information (DNA)
  - Based on information processing for both survival and evolution

- Can we fix it when it breaks?
  - Really becomes: How is information structured and processed?
Storing Processes

- Today we represent, store, search, and analyze:
  - Gene sequence data
  - Protein structure data
  - Metabolic network data
  - Signalling pathway data
  - ...

- How can we represent, store, and analyze biological processes?
  - Scalable, precise, dynamic, highly structured, maintainable representations for systems biology.
  - Not just huge lists of chemical reactions or differential equations.

- In computing...
  - There are well-established scalable representations of dynamic reactive processes.
  - They look more or less like little, mathematically based, programming languages.
Structural Architecture

Eukaryotic Cell
(10~100 trillion in human body)

Membranes everywhere

H.Lodish et al.
Molecular Cell Biology
fourth edition p.1
Abstract Machines of Systems Biology

The “hardware” (biochemistry) is fairly well understood. But what is the “software” that runs on these machines?
Reactive Systems

- **Modeling biological systems**
  - Not as continuous systems (often highly nonlinear)
  - But as discrete reactive systems; abstract machines with:
    - *States* represent situations
    - Event-driven *transitions* between states represent dynamics
  - The adequacy of describing (discrete) complex systems as reactive systems has been argued convincingly [Harel]

- **Many biological systems exhibit features of reactive systems:**
  - Deep layering of abstractions
  - Complex composition of simple components
  - Discrete transitions between states
  - Digital coding and processing of information
  - Reactive information-driven behavior
  - High degree of concurrency and nondeterminism
  - “Emergent behavior” not obvious from part list
**Chemistry vs. π-calculus**

A process calculus (chemistry, or SBML)

\[
\begin{align*}
\text{Na} + \text{Cl} & \rightarrow_{k_1} \text{Na}^+ + \text{Cl}^- \\
\text{Na}^+ + \text{Cl}^- & \rightarrow_{k_2} \text{Na} + \text{Cl}
\end{align*}
\]

This Petri-Net-like graphical representation degenerates into spaghetti diagrams: precise and dynamic, but not scalable, structured, or maintainable.

A compositional graphical representation, and the corresponding calculus.

\[
\begin{align*}
\text{Na} &= !r_{k_1}, ?s_{k_2}; \text{Na} \\
\text{Cl} &= ?r_{k_1}, !s_{k_2}; \text{Cl}
\end{align*}
\]

A different process calculus (π)
Methods

- **Model Construction** *(writing things down precisely)*
  - Formalizing the notations used in systems biology.
  - Formulating description languages.
  - Studying their kinetics (semantics).

- **Model Validation** *(using models for postdiction and prediction)*
  - Simulation from compositional descriptions
    - Stochastic: quantitative concurrent semantics.
    - Hybrid: discrete transitions between continuously evolving states.
  - “Program” Analysis
    - Control flow analysis
    - Causality analysis
  - Modelchecking
    - Standard, Quantitative, Probabilistic
Basic Modeling Guidelines

- Regev-Shapiro: “Molecules as Computation”:

<table>
<thead>
<tr>
<th>Molecule</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interaction capability</td>
<td>Channel</td>
</tr>
<tr>
<td>Interaction</td>
<td>Communication</td>
</tr>
<tr>
<td>Modification (of chemical components)</td>
<td>State change (state-transition systems)</td>
</tr>
</tbody>
</table>

Cellular Abstractions: Cells as Computation
Regev&Shapiro NATURE vol 419, 2002-09-26, 343

- They chose $\pi$-calculus and adapted it with stochastic features
  - To match the stochastic aspects of (bio)chemistry
  - Many probabilistic process calculi predate them, but only Hillston (CSP) and Priami ($\pi$) had already studied stochastic calculi.
\(\pi\)-calculus Executive Summary

- **It’s for:**
  - The modular description of concurrent, nondeterministic systems
  - Study of such systems based on their descriptions

- **It’s got:**
  - Processes
  - Channels
  - A minimalistic syntax (it’s a language and also a model)

- **You can:**
  - Fork new processes
  - Create new channels
  - Do I/O over channels (synchronous and asynchronous)
    including passing channels over channels
  - Make nondeterministic choices
  - Define processes recursively

- **That’s it.**
  - Except for extensive model theory and metatheory.
  - Cannot pass processes over channels
    (simulated by passing channels to them)
  - Cannot define procedures
    (simulated by supplying reply channels)
\( \pi \)-calculus

**Syntax**

\[
\pi ::= x(y) \quad \text{receive } y \text{ along } x \\
\overline{x}(y) \quad \text{send } y \text{ along } x
\]

\[P ::= 0 \mid \sum_{i \in I} \pi_i.P_i \mid [x = y]P \mid P_1|P_2 \mid (\text{new } x)P \mid 1P\]

**Structural congruence**

**Renaming of bound variables**

\[
x(y).P = x(z).(\{z/y\}P) \quad \text{if } z \notin FN(P)
\]

\[(\text{new } y).P = (\text{new } z).(\{z/y\}P) \quad \text{if } z \notin FN(P)\]

**Structural congruence laws**

\[
\begin{align*}
P|Q &= Q|P & \text{commutativity of parallel composition} \\
(P|Q)|R &= P|(Q|R) & \text{associativity of parallel composition} \\
P + Q &= Q + P & \text{commutativity of summation} \\
(P + Q) + R &= P + (Q + R) & \text{associativity of summation} \\
(\text{new } x)0 &= 0 & \text{restriction of inert processes} \\
(\text{new } x)(\text{new } y)P &= (\text{new } y)(\text{new } x)P & \text{polyadic restriction} \\
((\text{new } x)P)|Q &= (\text{new } x)(P|Q) \quad \text{if } x \notin FN(Q) & \text{scope extrusion} \\
!P &= P!P & \text{replication}
\end{align*}
\]

**Reaction rules**

\[
(\cdots + \overline{x}(z).Q)(\cdots + x(y).P) \rightarrow Q|P\{z/y\} \quad \text{communication (COMM)}
\]

\[
\frac{P \rightarrow P'}{P|Q \rightarrow P'|Q} \quad \text{reaction under parallel composition (PAR)}
\]

\[
\frac{P \rightarrow P'}{(\text{new } x)P \rightarrow (\text{new } x)P'} \quad \text{reaction under restriction (RES)}
\]

\[
\frac{Q \equiv P \quad P \rightarrow P' \quad P' \equiv Q'}{Q \rightarrow Q'} \quad \text{structural congruence (STRUCT)}
\]
Stochastic $\pi$-calculus Executive Summary

- A simple variant of $\pi$-calculus:
  - Channels have stochastic “firing” rates with exponential distribution.
  - Nondeterministic choice becomes stochastic race.
  - Cuts down to CTMCs (Continuous Time Markov Chains) in the finite case (not always). Then, standard analytical tools are applicable.
  - Can be given friendly automata-like scalable graphical syntax (work in progress: Andrew Phillips).
  - Is directly executable (e.g. via the Gillespie algorithm from physical chemistry).
  - Is analyzable (large body of literature, at least in the non-stochastic case).

Figure 2. Regulating Gene Expression by Positive Feedback [6]

Figure 3. Protein A molecules v.s. time in presence (left) and absence (right) of $TF A$. Phillips, L. Cardelli. BioConcur’04.
Importance of Stochastic Effects

- **A deterministic system:**
  - May get “stuck in a fixpoint”.
  - And hence never oscillate.

- **A similar stochastic system:**
  - May be “thrown off the fixpoint” by stochastic noise, entering a long orbit that will later bring it back to the fixpoint.
  - And hence oscillate.

Surprisingly enough, we have found that parameter values that give rise to a stable steady state in the deterministic limit continue to produce reliable oscillations in the stochastic case, as shown in Fig. 5. Therefore, the presence of noise not only changes the behavior of the system by adding more disorder but can also lead to marked qualitative differences.

---

Mechanisms of noise-resistance in genetic oscillators
José M. G. Vilar, Hao Yuan Kueh, Naama Barkai, Stanislas Leibler
PNAS April 30, 2002 vol. 99 no. 9 p.5991

Fig. 5. Time evolution of $\dot{R}$ for the deterministic Eq. (1) (a) and stochastic (b) versions of the model. The values of the parameters are as in the caption of Fig. 1, except that now we set $\lambda = 0.05$ h$^{-1}$. For these parameter values, $\tau < 0$, so that the fixed point is stable.

Fig. 6. Phase portrait as in Fig. 4 but for a situation in which the system falls into the stable fixed point $(R_0, C_0)$. The dotted arrow to the left of the fixed point illustrates a perturbation that would initiate a single sweep of the (former) oscillatory trajectory.
Gene Networks
The Gene Machine

The “Central Dogma” of Molecular Biology

DNA  messenger RNA  PROTEIN  SYSTEMS

regulation

transcription  translation  interaction

4-letter digital code  4-letter digital code  20-letter digital code  50,000 (?) shapes

Lactose Operon

Gene space

Metabolic space

Protein space

Taken from Pedro Mendes

2005-04-06
Regulation of a gene (positive and negative) influences transcription. The regulatory region has precise DNA sequences, but not meant for coding proteins: meant for binding regulators.

Transcription produces molecules (RNA or, through RNA, proteins) that bind to regulatory region of other genes (or that are end-products).

**Human (and mammalian) Genome Size**
- **3Gbp** (Giga base pairs) **750MB @ 4bp/Byte (CD)**
  - Non-repetitive: **1Gbp 250MB**
  - In genes: **320Mbp 80MB**
  - Coding: **160Mbp 40MB**
  - Protein-coding genes: **30,000-40,000**

**M.Genitalium** (smallest true organism)
- **580,073bp 145KB (eBook)**

**E.Coli** (bacteria): **4Mbp 1MB (floppy)**

**Yeast** (eukarya): **12Mbp 3MB (MP3 song)**

**Wheat** **17Gbp 4.25GB (DVD)**
**Gene Composition**

Is a shorthand for:

Under the assumptions [Kim & Tidor]
1) The solution is well-stirred
   (no spatial dependence on concentrations or rates).
2) There is no regulation cross-talk.
3) Control of expression is at transcription level only
   (no RNA-RNA or RNA-protein effects)
4) Transcriptions and translation rates monotonically
   affect mRNA and protein concentrations resp.

**Ex: Bistable Switch**

**Ex: Oscillator**

Expressed
Repressed
Expressing
Gene Regulatory Networks

http://strc.herts.ac.uk/bio/maria/NetBuilder/

(The Classical ODE Approach)

\[ \frac{dr}{dt} = f(p) - Vr \]

\[ \frac{dp}{dt} = Lr - Ur \]

\( n \): number of genes

\( r \): mRNA concentrations (n-dim vector)

\( p \): protein concentrations (n-dim vector)

\( f(p) \): transcription functions: (n-dim vector polynomials on \( p \))
A stochastic rate $r$ is always associated with each channel $a_r$ (at channel creation time) and delay $\tau_r$, but is often omitted when unambiguous.
Production and Degradation

Degradation is extremely important and often deliberate; it changes unbounded growth into (roughly) stable signals.

\[ \text{tr}(p) \triangleq (\lambda p_r \wedge \text{tr}(p)) + \tau_\delta \]

- transcription factor
- interaction site of transcription factor
- stochastic choice (race between \( r \) and \( \delta \))
- degradation rate \( \delta \)
- (output, \( l \)) interaction with rate \( r \)

A transcription factor is a process (not a message or a channel): it has behavior such as interaction on \( p \) and degradation.

\[ \text{null}(b) \triangleq \tau_\epsilon \wedge (\text{tr}(b) \mid \text{null}(b)) \]

- product
- combined effect of production and degradation (without any interaction on \( b \))

interaction offers on \( b \)
(\( \approx \) number of \( \text{tr} \) processes)

null(b)

\( \epsilon=0.1, \delta=0.001 \)
Unary Pos Gate

\[ \text{pos}(a, b) \triangleq q_r; \tau_\eta (\text{tr}(b) \mid \text{pos}(a, b)) + \tau_\epsilon (\text{tr}(b) \mid \text{pos}(a, b)) \]

(input, ?) interaction with rate \( r \)
or constitutive transcription to always get things started
parallel, not sequence, to handle self-loops without deadlock

transcription delay with rate \( \eta \)
race between \( r \) and \( \epsilon \)
output protein

unlimited amount of
*\text{tr}(q_r) \mid \text{pos}(q_r, b)*

pos(a, b)
Unary Neg Gate

(input, \(?\)) interaction with rate \(r\)

or constitutive transcription to always get things started

\[ \text{neg}(a,b) \triangleq \begin{cases} \text{?}a_r; \tau_r; \text{neg}(a,b) + \\ \tau_{\varepsilon}; (\text{tr}(b) | \text{neg}(a,b)) \end{cases} \]

inhibition delay with rate \(\eta\)

race between \(r\) and \(\varepsilon\)

\(r=1.0, \varepsilon=0.1, \eta=0.01, \delta=0.001\)

\[ \text{neg}(a_r,b) \]

\[ \text{*tr}(a_r) | \text{neg}(a_r,b) \]

Constitutive

Inhibited
Signal Amplification

\[ \text{pos}(a, b) \mid \text{pos}(b, c) \]

\[ \text{pos}(a, b) \triangleq \ ?a_r; \ \tau_{\eta}; \ (\text{tr}(b) \mid \text{pos}(a, b)) + \ \tau_{c}; \ (\text{tr}(b) \mid \text{pos}(a, b)) \]

\[ \text{tr}(p) \triangleq (\text{!p} \mid \text{tr}(p)) + \tau_{\delta} \]

E.g. 1 a that interacts twice before decay can produce 2 b that each interact twice before decay, which produce 4 c.

Even with no a input, constitutive production of b gets amplified to a high c signal.
Signal Normalization

\[ \text{neg}(a,b) \mid \text{neg}(b,c) \]

\[ \text{neg}(a,b) \triangleq \]
\[ ?_{a_r}; \tau_h; \text{neg}(a,b) + \tau_e; (\text{tr}(b) \mid \text{neg}(a,b)) \]

\[ \text{tr}(p) \triangleq (!_{p_r}; \text{tr}(p)) + \tau_\delta \]

A non-zero input level, \( a \), whether weak or strong, is renormalized to a standard level, \( c \).

\[ r=1.0, \ v=0.1, \ \eta=0.01, \ \delta=0.001 \]
**Self Feedback Circuits**

\[
\text{pos}(a,a) \quad \Downarrow \quad a \\
\text{pos}
\]

\[
\text{pos}(a,b) \triangleq \\
\begin{align*}
?a_r; & \quad (\text{tr}(b) \mid \text{pos}(a,b)) + \\
\tau_{\epsilon}; & \quad (\text{tr}(b) \mid \text{pos}(a,b))
\end{align*}
\]

\[
\text{tr}(p) \triangleq (\lnot p_r; \text{tr}(p)) + \tau_{\delta}
\]

*(Can overwhelm degradation, depending on parameters)*

\[
\text{neg}(a,a) \quad \Downarrow \quad a \\
\text{neg}
\]

\[
\text{neg}(a,b) \triangleq \\
\begin{align*}
?a_r; & \quad \tau_{h}; \text{neg}(a,b) + \\
\tau_{\epsilon}; & \quad (\text{tr}(b) \mid \text{neg}(a,b))
\end{align*}
\]

\[
\text{tr}(p) \triangleq (\lnot p_r; \text{tr}(p)) + \tau_{\delta}
\]

*high, to raise the signal*
Two-gate Feedback Circuits

Monostable:
For some degradation rates is quite stable:

pos(b,a) | neg(a,b)

But with a small change in degradation, it goes wild:

pos(b,a) | neg(a,b)

Bistable:

r=1.0, ε=0.1, h=0.01, δ=0.0005

5 runs with r(a)=0.1, r(b)=1.0 shows that circuit is now biased towards expressing b
Repressilator

\[ \text{neg}(a,b) \quad \text{neg}(b,c) \quad \text{neg}(c,a) \]

Same circuit, three different degradation models by chaning the tr component:

- \( \text{tr}(p) \triangleq \neg p_r \) interact once and die otherwise stick around
  
- \( \text{tr}(p) \triangleq \neg p_r + \tau_\delta \) interact once and die otherwise decay
  
- \( \text{tr}(p) \triangleq (\neg p_r; \text{tr}(p)) + \tau_\delta \) interact many times and decay

Subtle... at any point one gate is inhibited and the other two can fire constitutively. If one of them fires first, nothing really changes, but if the other one fires first, then the cycle progresses.
Repressilator in SPiM

val dk = 0.001 (* Decay rate *)
val eta = 0.001 (* Inhibition rate *)
val cst = 0.1 (* Constitutive rate *)

let tr(p:chan()) =
do !p; tr(p)
or delay@dk

let neg(a:chan(), b:chan()) =
do ?a; delay@eta; neg(a,b)
or delay@cst; (tr(b) | neg(a,b))

(* The circuit *)
val bnd = 1.0 (* Protein binding rate *)
new a@bnd: chan()
new b@bnd: chan()
new c@bnd: chan()

run (neg(c,a) | neg(a,b) | neg(b,c))
Repressilator ODE Model and Simulation

\[ \begin{align*}
\frac{d[X]}{dt} &= \alpha_0 + \frac{\alpha + \alpha_1 [PY]^n}{K^n + [PY]^n} - k[X], \\
\frac{d[PY]}{dt} &= \beta [[X] - [PX]], \\
\frac{d[Y]}{dt} &= \alpha_0 + \frac{\alpha + \alpha_1 [PZ]^n}{K^n + [PZ]^n} - k[Y], \\
\frac{d[PY]}{dt} &= \beta [[Y] - [PY]], \\
\frac{d[Z]}{dt} &= \alpha_0 + \frac{\alpha + \alpha_1 [PX]^n}{K^n + [PX]^n} - k[Z], \\
\frac{d[PZ]}{dt} &= \beta [[Z] - [PZ]].
\end{align*} \]

Bruce E Shapiro
Cellerator

The figure illustrates the genetic network with operators and genes involved:

- **aTc** activates the TetR gene
- **IPTG** activates the LacI gene
- **λcI** activates the GFP gene
- **tet** gene
- **lac** gene
- **cI** gene
- **gfp** gene

The experiment conditions are:
- **aTc**: - + - +
- **IPTG**: - - + +
- **GFP**: - + - -

We can model an inducer like aTc as something that competes for the transcription factor.

IPTG de-represses the lac operon, by binding to the lac repressor (the lac I gene product), preventing it from binding to the operator.
Gene-Protein Networks
Indirect Gene Effects

No combination of standard high-throughput experiments can reconstruct an a-priori known gene/protein network [Wagner].

One of many bistable switches that cannot be described by pure gene regulatory networks [Francois & Hakim].
François & Hakim Fig3A

**Reactions**

<table>
<thead>
<tr>
<th>Reaction</th>
<th>Constants</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a \rightarrow a + A$</td>
<td>0.20</td>
<td>0.9 - 1.1</td>
</tr>
<tr>
<td>$A \rightarrow$ Nothing</td>
<td>0.0085</td>
<td>0.0 - 1.5</td>
</tr>
<tr>
<td>$b \rightarrow b + B$</td>
<td>0.37</td>
<td>0.7 - 1.3</td>
</tr>
<tr>
<td>$B \rightarrow$ Nothing</td>
<td>0.034</td>
<td>0.0 - 8.9</td>
</tr>
<tr>
<td>$A + B \rightarrow A : B$</td>
<td>0.72</td>
<td>0.1 - &gt; 10</td>
</tr>
<tr>
<td>$A : B \rightarrow$ Nothing</td>
<td>0.53</td>
<td>Irrelevant</td>
</tr>
<tr>
<td>$b + A \rightarrow b : A$</td>
<td>0.19</td>
<td>0.7 - 7.6</td>
</tr>
<tr>
<td>$b : A \rightarrow b + A$</td>
<td>0.42</td>
<td>0.2 - 1.5</td>
</tr>
<tr>
<td>$b : A \rightarrow b : A + B$</td>
<td>0.027</td>
<td>0.0 - 2.3</td>
</tr>
</tbody>
</table>

**Fig 3A**

**Reaction oriented**

**Fig 14A**

**Protein A**

**Number of proteins**

**Free evolution**

**Fig 14A**

**Protein A**

**Number of proteins**

**Fig 14A**
François & Hakim Fig3A, SPiM simulation

Parameters as in paper

3 copies of each gene.

Modified for stability: \( \Delta k_A = 0.02, \Delta k_B = 0.02 \)
François & Hakim Fig3Ast8

Circuit of Fig 3A with parameters from Supporting Text Fig 8, plotted in Fig 13A

<table>
<thead>
<tr>
<th>Reactions</th>
<th>Constants</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a \rightarrow a+A$</td>
<td>0.52</td>
</tr>
<tr>
<td>A $\rightarrow$ Nothing</td>
<td>0.00019</td>
</tr>
<tr>
<td>$b \rightarrow b+B$</td>
<td>0.79</td>
</tr>
<tr>
<td>B $\rightarrow$ Nothing</td>
<td>0.0030</td>
</tr>
<tr>
<td>A+B $\rightarrow$ A:B</td>
<td>0.053</td>
</tr>
<tr>
<td>A:B $\rightarrow$ Nothing</td>
<td>0.15</td>
</tr>
<tr>
<td>$b+A \rightarrow b:A$</td>
<td>0.22</td>
</tr>
<tr>
<td>b:A $\rightarrow$ b+A</td>
<td>0.31</td>
</tr>
<tr>
<td>b:A $\rightarrow$ b:A+B</td>
<td>0.43</td>
</tr>
</tbody>
</table>

Fig 8

Fig 13A

SPIM simulation

Free evolution

200xA injected at ~2500
500xA injected at ~5000
200xA injected at ~7500

200xB injected at 0
600xA injected at ~2500
600xB injected at ~7500

Pulse of A
Pulse of B

Number of proteins

0 2000 4000 6000 8000
0 200 400 600 800

0 500 1000 1500
0 100 200 300 400 500
0 10 20 30 40 50
0 10 20 30 40 50

dkA 0.00019()
dkB 0.003()
let ptnA() =
    (new unb@pntAunb
       do delay@dkA or !AB or ![A(unb);(?unb; ptnA()))

let ptnB() =
    do delay@dkB or ![AB;cpxAB()]

let cpxAB() = delay@dkAB

let geneA() =
    delay@geneACSt; (ptnA() | geneA())

let geneBfree() =
    do delay@geneBCSt; (ptnB() | geneBfree())
    or ![A(unb); geneBbound(unb)

and geneBbound(unb:ch()) =
    do delay@geneBInh; (ptnB() | geneBbound(unb))
    or ![unb; geneBfree()
Scaling up:
ODE vs Process Descriptions
From Chemical Reactions to ODE's

\[ \begin{align*}
  r_1: & \quad A+B \rightarrow_{k_1} C+C \\
  r_2: & \quad A+C \rightarrow_{k_2} D \\
  r_3: & \quad C \rightarrow_{k_3} E+F \\
  r_4: & \quad F \rightarrow_{k_4} B
\end{align*} \]

Write the coefficients by columns

\[
\begin{array}{c|cccc}
\text{N} & r_1 & r_2 & r_3 & r_4 \\
\hline
A & -1 & -1 & & \\
B & -1 & & & 1 \\
C & 2 & -1 & -1 & \\
D & 1 & & & \\
E & & & & 1 \\
F & & & 1 & -1 \\
\end{array}
\]

Read the rate laws from the columns

\[ v_i(x,e_i,k_i) \]

\( x \): chemical species
\( [\cdot] \): concentrations
\( v \): rate laws
\( k \): kinetic parameters
\( N \): stoichiometric matrix
\( e \): catalysts (if any)

Rate laws:
\[ \begin{align*}
  d[A]/dt & = -v_1 - v_2 \\
  d[B]/dt & = -v_1 + v_4 \\
  d[C]/dt & = 2 \cdot v_1 - v_2 - v_3 \\
  d[D]/dt & = v_2 \\
  d[E]/dt & = v_3 \\
  d[F]/dt & = v_3 - v_4
\end{align*} \]

E.g.
\[ d[A]/dt = -k_1 [A][B] - k_2 [A][C] \]
From Chemical Reactions to Processes

\[ r_1: \text{A} + \text{B} \rightarrow k_1 \text{C} + \text{C} \]
\[ r_2: \text{A} + \text{C} \rightarrow k_2 \text{D} \]
\[ r_3: \text{C} \rightarrow k_3 \text{E} + \text{F} \]
\[ r_4: \text{F} \rightarrow k_4 \text{B} \]

Write the coefficients by columns

For binary reactions, first species in the column does an input and produces result, second species does an output. For unary reactions, species does a tau action and produces result. No ternary reactions.

\[
\begin{array}{cccc}
\text{N} & r_1 & r_2 & r_3 & r_4 \\
\hline
\text{A} & -1 & -1 & & \\
\text{B} & -1 & 1 & & \\
\text{C} & 2 & -1 & -1 & \\
\text{D} & 1 & & & \\
\text{E} & 1 & & & \\
\text{F} & 1 & -1 & & \\
\end{array}
\]

\[ \text{A} = ?v_1k_1(C|C) + ?v_2k_2.D + ?a \]
\[ \text{B} = !v_1k_1 + ?b \]
\[ \text{C} = !v_2k_2 + \tau k_3(E|F) + ?c \]
\[ \text{D} = 0 + ?d \]
\[ \text{E} = 0 + ?e \]
\[ \text{F} = \tau k_3.B + ?f \]

Add a bar for counting and plotting

Read the process interactions from the rows

(Rate laws are implicit in stochastic semantics)
Stoichiometric Matrices Blow Up

• **We can translate Chemistry to ODE’s or Processes**
  - It is standard to go from chemical equations to ODE’s via a stoichiometric matrix.
  - It is similarly possible to go from chemical equations to processes via a stoichiometric matrix.

• **But there is a better way:**
  - Stoichiometric matrices blow-up exponentially for biochemical systems (unlike for ordinary chemical systems) because proteins have combinatorial state and complexed states are common.
  - To avoid this explosion, we should describe biochemical systems compositionally without going through a stochiometric matrix (and hence without ODE’s).
Complexes: The ODE Way

- **n** domains: A, B, C
- **2n** domain reactions:
  - A ≈ A_p
  - B ≈ B_p
  - C ≈ C_p
- **1** complex: ABC
- **2n** species:
  - A_pBC
  - A_pBC
  - A_pC
  - A_pBC
  - A_pB_pC
  - A_pB_pC
  - A_pB_pC
  - A_pB_pC

The matrix is very sparse, so the corresponding ODE system is not dense. But it still has $2^n$ equations, one per species, plus conservation equations ($[ABC]+[A_pBC]=\text{constant}$, etc.).

System description is exponential in the number of basic components.

Stoichiometric Matrix

| N     | V_1 | V_2 | V_3 | V_4 | V_5 | V_6 | V_7 | V_8 | V_9 | V_10 | V_11 | V_12 | V_13 | V_14 | V_15 | V_16 | V_17 | V_18 | V_19 | V_20 | V_21 | V_22 | V_23 | V_24 |
|-------|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| ABC   |     |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| A_pBC |     |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| A_BpC |     |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| A_pBC |     |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| A_pB_pC | |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| A_pB_pC | |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| A_pB_pC | |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
| A_pB_pC | |     |     |     |     |     |     |     |     |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |

$2^n \times 2n(2^{n-1})$
Complexes: The Reactive System Way

When the local domain reactions are not independent, we can use lateral communication so that each component is aware of the relevant others.

System description is linear in the number of basic components.

(Its “run-time” behavior or analysis potentially blows-up just as in the previous case, but its description does not.)
Model Validation
Model Validation: Simulation

- **Basic stochastic algorithm: Gillespie**
  - Exact (i.e. based on physics) stochastic simulation of chemical kinetics.
  - Can compute concentrations and reaction times for biochemical networks.

- **Stochastic Process Calculi**
  - **BioSPI** [Shapiro, Regev, Priami, et. al.]
    - Stochastic process calculus based on Gillespie.
  - **BioAmbients** [Regev, Panina, Silverma, Cardelli, Shapiro]
    - Extension of BioSPI for membranes.
  - **Case study: Lymphocytes in Inflamed Blood Vessels** [Lecca, Priami, Quaglia]
    - Original analysis of lymphocyte rolling in blood vessels of different diameters.
  - **Case study: Lambda Switch** [Celine Kuttler, IRI Lille]
    - Model of phage lambda genome (well-studied system).
  - **Case study: VICE** [U. Pisa]
    - Minimal prokaryote genome (180 genes) and metabolism of whole Virtual Cell, in stochastic π-calculus, simulated under stable conditions for 40K transitions.

- **Hybrid approaches**
  - **Charon language** [UPenn]
    - Hybrid systems: continuous differential equations + discrete/stochastic mode switching.
  - Etc.
Model Validation: “Program” Analysis

- **Causality Analysis**
  - *Biochemical pathways*, (“concurrent traces” such as the one here), are found in biology publications, summarizing known facts.
  - This one, however, was automatically generated from a program written in BioSpi by comparing traces of all possible interactions. [Curti, Priami, Degano, Baldari]
  - One can play with the program to investigate various hypotheses about the pathways.

- **Control Flow Analysis**
  - Flow analysis techniques applied to process calculi.
  - Overapproximation of behavior used to answer questions about what “cannot happen”.
  - Analysis of positive feedback transcription regulation in BioAmbients [Flemming Nielson].

- **Probabilistic Abstract Interpretation**
  - [DiPierro Wickicky].

![Diagram](image1)

![Diagram](image2)
Model Validation: Modelchecking

- **Temporal**
  - Software verification of biomolecular systems (NA pump) [Ciobanu]
  - Analysis of mammalian cell cycle (after Kohn) in CTL. [Chabrier-Rivier Chiaverini Danos Fages Schachter]
    - E.g. is state S₁ a necessary checkpoint for reaching state S₂?

- **Quantitative: Simpathica/xssys** [Antioniotti Park Policriti Ugel Mishra]
  - Quantitative temporal logic queries of human Purine metabolism model.
    - Eventually(Always(PRPP = 1.7 * PRPP₁) implies steady_state() and Eventually(Always(IMP < 2 * IMP₁)) and Eventually(Always(hx_pool < 10*hx_pool₁)))

- **Stochastic: Spring** [Parker Normal Kwiatkowska]
  - Designed for stochastic (computer) network analysis
    - Discrete and Continuous Markov Processes.
    - Process input language.
    - Modelchecking of probabilistic queries.
What Reactive Systems Do For Us

We can write things down precisely
- We can modularly describe high structural and combinatorial complexity (“do programming”).

We can calculate and analyze
- Directly support simulation.
- Support analysis (e.g. control flow, causality, nondeterminism).
- Support state exploration (model checking).

We can reason
- Suitable equivalences on processes induce algebraic laws.
- We can relate different systems (e.g. equivalent behaviors).
- We can relate different abstraction levels.
- We can use equivalences for state minimization (symmetries).

We can visualize
- Automata-like presentations.
- Petri-Net-like presentations.
- State Charts, Live Sequence Charts [Harel]
  • Hierarchical automata.
  • Scenario composition.

Disclaimers
- Some of these technologies are basically ready (medium-scale stochastic simulation and analysis, medium-scale nondeterministic and stochastic model checking).
- Others need to scale up significantly to be really useful. This is (has been) the challenge for computer scientists.

Many approaches, same basic philosophy, tools being built:
Q: “The data are accumulating and the computers are humming, what we are lacking are the words, the grammar and the syntax of a new language…”
D. Bray (TIBS 22(9):325-326, 1997)

A: “The most advanced tools for computer process description seem to be also the best tools for the description of biomolecular systems.”
E. Shapiro (Lecture Notes)
References


Papers

BioAmbients
a stochastic calculus with compartments.

Brane Calculi
process calculi with computation “on” the membranes, not inside them.

Bitonal Systems
membrane reactions and their connections to “local” patch reactions.

Abstract Machines of Systems Biology
the abstract machines implemented by biochemical toolkits.

www.luca.demon.co.uk/BioComputing.htm