Biochemical Algorithms

Luca Cardelli, Microsoft Research & Oxford University

related work: Attila Csikász-Nagy, Neil Dalchau, Mirco Tribastone, Max Tschaikowski, Andrea Vandin

IMT Lucca, 2017-03-30
Introduction
Cellular Computation

- No survival without computation!
  - Finding food
  - Avoiding predators

- How do cells compute?
  - Clearly doing “information processing”
  - What are their computational principles?
  - What are their algorithms?

More concretely

- Give substance to the claim that “cells compute”
  - Yes, but *what* do they compute?

- Catch nature red-handed in the act of running a computational task
  - Something that a computer scientist would recognize as an *algorithm*
Reality is Complicated

- *Every* biochemical species that we may just call “X” is actually a sophisticated machine that has evolved for billions of years.
Biochemical Networks
Abstract Machines of Biochemistry

- **Gene Machine**
  - Nucleotides
  - Regulation
  - Direct construction
  - Hold receptors, host reactions

- **Protein Machine**
  - Aminoacids
  - Make proteins
  - Send signals
  - Enact fusion, fission

- **Membrane Machine**
  - Phospholipids
  - Confinement, Storage
  - Bulk Transport

Metabolism, Propulsion, Signaling, Transport

Glycan Machine
- Surface and Extracellular Features
Bioinformatics View (Data Structures)

- Gene Machine
- Protein Machine
- Membrane Machine
  - Records (sets of features)
  - Trees
  - Glycan Machine
  - Hierarchical Multisets
  - Strings
Systems Biology View (Networks)
Algorithmic View (Languages)

Gene Machine

Protein Machine

Membrane Machine

Molecular Interaction Maps

Gene Networks

Transport Networks

These 3 machines are Turing powerful!
Network Evolution

Across species: Ortholog genes

Within species: Paralog genes

“same function”

“new function”
Influence Networks
How to model “Influence”

“Influence” molecular interactions.

“Equivalent” influence interactions.

Chemical Reaction Network

Evolving a Primitive Eukaryotic Cell Cycle Model

Malte Lücken, Jotun Hein, Bela Novak

Instead of modeling basic interactions, such as binding, synthesis, and degradation of molecular components, this framework models interactions simply as activation or inhibition. This approach also reduces the number of nodes necessary in the network, as e.g. the inhibitor binding tightly to the activator to form a complex, which produces phosphorylated inhibitor to be degraded under catalysis by the activator, is now simply a double negative feedback loop shown in Figure 1. This type of interaction is the basis of both aforementioned molecular model, therefore they can both be summarized in a single Reinitz model.
The Reinitz Model of Influence

- Based on early connectionist (neural network) modeling
- Each activation/inhibition interaction is modeled as a flexible sigmoid function with 4+ parameters per node

- We prefer to stick to mass action kinetics
- It will later become clear why

- We model activation/inhibition nodes by a mass action motif:
  - Using 4 rate parameters per node
  - Akin to multisite modification
The Triplet Model of Influence

Usually modeled by sigmoid (e.g. Hill or Reinitz) functions

They actually implement a Hill function of coefficient 2:

For example:

biological mechanism: (e.g.:) multisite phosphorylation

Approximate Majority
The Triplet Model of Influence

- Solving this mass action model at steady state with \( tot = x_0 + x_1 + x_2 \), obtain \( x_0 \) as a function of \( a \) and \( i \):
  \[
x_0 = \frac{k_10 k_{21} tot \ a^2}{k_{10}k_{21}a^2 + k_{01}k_{21}ai + k_{01}k_{12}i^2}
\]

- Assuming \( i = tot - a \) (inhibition decreases as activation increases) obtain \( x_0 \) as a function of \( a \in [0..tot] \) (max stimulus = max response)
  \[
x_0 = \frac{k_{10}k_{21}tot \ a^2}{(k_{10}k_{21} - k_{01}k_{21} + k_{01}k_{12})a^2 + (k_{01}k_{21} - 2k_{01}k_{12})tot \ a + k_{01}k_{12}tot^2}
\]

- By regulating the rates of flow through \( x_1 \) within 2 orders of magnitude we can obtain a range of linear, hyperbolic and sigmoid responses in the range \([0..1]\) to linear activation \( a \in [0..1] \).

steady state transitions from inhibited to activated with \( tot = 1 \) and \( a \in [0..1] \)
Influence Network Notation

- Catalytic reaction
  \[ x \rightarrow y = x \rightarrow y \]
  \[ x + z \rightarrow z + y \]
  \( z \) is the catalyst

- Triplet motif

![Diagram](image)

influence node
catalytic node

Duality

\( \sim x_2 \) output \( \sim x_0 \) output

\( (\sim x)_0 = x_2, \)
\( (\sim x)_1 = x_1, \)
\( (\sim x)_2 = x_0 \)
Influence Network Duality

- Let $\sim x$ be the species such that

  \[(\sim x)_0 = x_2, \quad (\sim x)_1 = x_1, \quad (\sim x)_2 = x_0\]

  so that promoting $x$ is the same as inhibiting $\sim x$ etc. Then:
**Network model**

- **Influence networks**
  - Influence species: two main molecular states (high/low or modified/unmodified)
  - High-low transitions are nonlinear (e.g. sigmoidal)
  - Transition kinetics may vary (but we fix one uniformly)

  ![Diagram of network model](image)

  **Nodes**

  - Very much like gene regulatory networks, but with the extra option of the “unmodified” state being active too
Consensus Networks
A Consensus Problem

• Population Consensus
  • Given two populations of x and y “agents”
  • We want them to “reach consensus”
  • By converting all agents to x or to y depending on which population was in majority initially
  
  \[
  \begin{align*}
  X, Y &:= X + Y, 0 \quad \text{if} \quad X_0 \geq Y_0 \\
  X, Y &:= 0, X + Y \quad \text{if} \quad Y_0 \geq X_0
  \end{align*}
  \]

• Population Protocols Model
  • Finite-state identity-free agents (molecules) interact in randomly chosen pairs \(\Rightarrow\) stochastic symmetry breaking
  • Each interaction (collision) can result in state changes
  • Complete connectivity, no centralized control (well-mixed solution)
A Consensus Algorithm

- Approximate Majority (AM) Algorithm
  - Uses a third “undecided” population $b$
  - Disagreements cause agents to become undecided
  - Undecided agents agree with any non-undecided agent

Dana Angluin · James Aspnes · David Eisenstat

A Simple Population Protocol for Fast Robust Approximate Majority
Consensus Algorithms

Direct Competition

\[ x + y \rightarrow y + y \]
\[ y + x \rightarrow x + x \]

Approximate Majority

\[ x_2 + x_0 \rightarrow x_0 + x_1 \]
\[ x_1 + x_0 \rightarrow x_0 + x_0 \]
\[ x_0 + x_2 \rightarrow x_2 + x_1 \]
\[ x_1 + x_2 \rightarrow x_2 + x_2 \]

Bad: \( O(n) \)

Good: \( O(\log n) \)

Worse-case scenario example, starting with \( x_0=x_2, x_1=0 \):
A Biological Implementation

Approximate Majority (AM)

1) Bistable
   Even when initially x=y (stochastically)

2) Fast (asymptotically optimal)
   $O(\log n)$ convergence time

3) Robust to perturbation
   above a threshold, initial majority wins \textit{whp}

Epigenetic Switch

Figure 1. Basic Ingredients of the Model

Dana Angluin, James Aspnes, and David Eisenstat
A Simple Population Protocol for Fast Robust Approximate Majority 2007

Cell 2007
Not always that simple

- The epigenetic switch seems a *direct* biological implementation of an algorithm
  - Although we may have to qualify that with some notion of approximation of the (enzymatic) kinetics

- In most cases the biological implementation seems more *indirect* or *obfuscated*
  - "*Nature is subtle but not malicious* - *Einstein*" Ha! think again!
  - Other implementations of Approximate Majority seem more convoluted and approximate
Antagonistic Networks
Antagonistic Networks

• Let’s generalize:
  • AM is based on antagonism between two species (inside the triplet)
  • So (essentially) are many standard biological networks

• Are they somehow related?
  • We could try the same empirical analysis as for CC/AM
  • But we can do better
Mutual Inhibition (1 vs. 1)

- “All” cellular switches in all phases of the cell cycle follow (abstractly) a mutual inhibition pattern:

- Also found in other areas (cell polarity establishment):
Septation Initiation (1 vs. 1)

- Other (inherently different) biological networks are based on mutual inhibition, and share characteristics with AM.

Dynamics of SIN Asymmetry Establishment

SIN inhibiting Byr4, absence of SIN promoting Byr4
Byr4 inhibiting SIN, absence of Byr4 promoting SIN
Delta-Notch (2 vs. 2)

- A mutual inhibition pattern
  - Involving *two* species in each cell

- In two cells a,b
  - \(D_a, N_b\) antagonize \(D_b, N_a\)

---

*Lateral Inhibition through Delta-Notch Signaling: A Piecewise Affine Hybrid Model*

Romejoy Ghosh and Claire J. Tomlin

© Springer-Verlag Berlin Heidelberg 2001
Antagonistic Networks

1 vs. 1
Mutual Inhibition & Self Activation

1 vs. 1
Mutual Inhibition & Mutual Anti-activation

Cell cycle transitions

Polarity establishment

Gene networks

Septation Initiation
Antagonistic Networks

1 vs. 2

2 vs. 2

3 vs. 3

The $G_{2}/M$ cell cycle switch

Delta-Notch

The “new” cell cycle switch
The Cell Cycle Switch
Decisions, decisions...

- The AM algorithm has ideal properties for settling a population into one of two states

- Seems like this would be useful in Biology
  - Can we find biological implementations of this algorithm?
  - Could it be related to the cell cycle switch?
The Cell Cycle Switch

- This basic network is universal in Eukaryotes [P. Nurse]
  - The switching function and the basic network is the same from yeast to us.
  - In particular detail, in frog eggs, G₂/M transition:
    - Double positive feedback on x
    - Double negative feedback on x
    - No feedback on y. Why ???

- The function is very well-studied. But why this network structure?
- That is, why this peculiar algorithm?
How to Build a Good Switch

• What is a “good” switch?
  • We need first a bistable system: one that has two distinct and stable states.
    I.e., given any initial state the system must settle into one of two states
  • The settling must be fast (not get stuck in the middle for too long)
    and robust (must not spontaneously switch back)
  • Finally, we need to be able to flip the switch by external inputs

• “Population protocol” switches
  • Identical agents (‘molecules’) in a population start in some state, say x or y
  • A pair of agents is chosen randomly at each step,
    they interact (‘collide’) and change state
  • The whole population must eventually agree on a
    majority value (all-x or all-y) with probability 1
A Bad Algorithm

- Direct Competition
  - x catalyzes the transformation of y into x
  - y catalyzes the transformation of x into y
  - when all-x or all-y, it stops

- This system has two end states, but
  - Convergence to an end state is slow (a random walk)
  - Any perturbation of an end state can start a random walk to the other end state (hence not really bistable)
A Good Algorithm

• Approximate Majority (AM)
  • Third, undecided, state \( b \)
  • Disagreements cause agents to become undecided
  • Undecided agents believe any non-undecided agent

• With high probability, for \( n \) agents
  • The total number of interactions before converging is \( O(n \log n) \)
    \( \Rightarrow \) fast (optimal)
  • The final outcome is correct if the initial disparity is \( \omega(\sqrt{n} \log n) \)
    \( \Rightarrow \) solution states are robust to perturbations

• Logarithmic time bound in parallel time
  • Parallel time is the number of steps divided by the number of agents
  • In parallel time the algorithm converges with high probability in \( O(\log n) \)

Worst-case scenario, starting with \( x=y, b=0 \):

\[
\begin{align*}
  x + y & \rightarrow y + b \\
  y + x & \rightarrow x + b \\
  b + x & \rightarrow x + x \\
  b + y & \rightarrow y + y
\end{align*}
\]
An “Ugly” Algorithm: Cell Cycle Switch

- Is it a good algorithm? Is it bad?
- Is it optimal or suboptimal?
Convergence Analysis - CONSENSUS

- Switches as computational systems

CC converges in O(log n) time (like AM) (but 2x slower than AM, and does not fully switch)

Start symmetrical ($x_0=x_1=x_2$ etc.)

Black lines: several stochastic simulation traces
Color: full probability distribution of small-size system
Steady State Analysis - SWITCH

• Switches as dynamical systems

Black lines: deterministic ODE bifurcation diagrams
Red lines: noisy stochastic simulations
Color: full probability distribution of small-size system
A Bug in the Algorithm
In Summary

CC converges in $O(\log n)$ time (like AM) (but 2x slower than AM, and does not fully switch)

Symmetrical initial conditions ($x_0 = x_1 = x_2$)

Black lines: high-count stochastic simulation traces
Color: full probability distribution of low-count system

Hor axis is time.

AM shows hysteresis (like CC)

Black lines: deterministic ODE bifurcation diagrams
Red lines: medium-count stochastic simulations
Color: full probability distribution of low-count system

Hor axis is stimulus pushing towards $x_0$ against fixed bias.

But there is a deficiency in CC performance!

The “classical” Cell Cycle Switch $CC$ approximates AM performance
Why is CC worse than AM?

- The classical CC has an algorithmic "bug"
  - It works ok but never as well as AM
  - Because s continuously inhibits x through z, so that x cannot fully express

- So let’s fix the bug!
  - Easy: let x inhibit s and t “in retaliation”
  - Q: Why didn’t nature fix it?

The corresponding cell cycle oscillator is also depressed
Nature fixed it!

- There is another known feedback loop
  - By which $x$ suppresses $s$ “in retaliation” via the so-called Greatwall loop
  - Also, $s$ and $t$ happen to be the same molecule ($=s$)

- $s$ and $x$ now are antagonists: they are the two halves of the switch, mutually inhibiting each other (through intermediaries).
More surprisingly

- The fix makes it faster too!
- The extra feedback also speeds up the decision time of the switch, making it about as good as the ‘optimal’ AM switch:

Conclusion:
Nature is trying as hard as it can to implement an AM-class algorithm!

The “classical” cell cycle switch is only half of the picture: the extra feedback completes it *algorithmically.*
The Greatwall Loop

- Our paper appeared, suggesting GW is a better cell cycle switch than CC:

GW = AM “obfuscated”

- Another paper appeared that same week:

Showing experimentally that the Greatwall loop is a necessary component of the switch. The not-as-good-as-AM network has been ‘refuted’
More Recent Developments
The basic “revised” Cell Cycle Switch

This is an AM-class algorithm (identical performance)

Vs.
New Cell Cycle Switch Network

- A recent paper presents a more complete view of the cell cycle switch
- N.B. “phosphorylation network dynamics” here is the same as our $x_0-x_1-x_2$ motif

Mutual inhibition between *three* species each
Molecular Implementation of AM

- We produced a chemical implementation of AM using DNA gates
- I.e., a ‘synthetic reimplementation’ of the central cell-cycle switch.
Network Equivalences
What we learned

- The network structure of AM implements an input-driven switching function (in addition to the known majority function).
- The network structure of CC/GW implements a input-less majority function (in addition to the known switching function).

- The behavior of AM and CC/GW in isolation are related.
- The behavior of AM and CC/GW in oscillator contexts are related.

- A refinement (GW) of the core CC network, known to occur in nature, improves its switching performance and brings it in line with AM performance.
Can we make this precise?

- Our evidence for computational content of biochemical networks is so far
  - Quantitative, covering both kinetic and steady state behavior of what networks do
  - But empirical (based on simulations/numerical solutions)
  - And it does not yet explain how the CC/GW network relates to the AM network, that is, how each piece of CC/GW corresponds to each piece of AM

- Analytical evidence is harder to obtain
  - The proofs of the computational properties (optimality etc.) for the AM algorithm are hard and do not generalize easily to more complex networks
  - Quantitative theories of behavioral equivalence and behavioral approximation, e.g. in process algebra, are still lacking (although rich qualitative theories exist)

- How exactly is CC (or CCr, GW, etc.) the “same” as AM?
Network Emulation  CCr emulates AM

- For any rates and initial conditions of AM, we can find some rates and initial conditions of CCr such that the (9) trajectories of CCr retrace those (3) of AM:

\[ \sim S, r \rightarrow x \]

- How do we find these matching parameters? By a network morphism!

initialize:
\[ z = x \]
\[ \sim y = x \]
\[ y_2 = x_0 \]
\[ y_1 = x_1 \]
\[ y_0 = x_0 \]

(9 species on 3 trajectories)  (3 species on 3 trajectories)
Network Emulation: MI emulates AM

A mapping of species and reactions

**homomorphic mapping**

\[ z \rightarrow x \]
\[ \sim y \rightarrow x \]

any initial conditions

initial conditions:

\[ z_0 = y_2 = x_0 \]
\[ z_1 = y_1 = x_1 \]
\[ z_2 = y_0 = x_2 \]

less trivial than you might think:
it need not preserve the out-degree of a node!
Network Emulation: SI emulates AM

A mapping of species and reactions

**homomorphic mapping**

\[ z \rightarrow x \]

\[ y \rightarrow x \]

**Initial conditions:**

\[ z_0 = y_2 = x_0 \]

\[ z_1 = y_1 = x_1 \]

\[ z_2 = y_0 = x_2 \]
How to find emulations

- How do we check a potential mapping for all possible initial conditions of the target?
  - Statically! Check conditions on the joint stoichiometric matrices of the two networks under the mapping.

- How do we check a potential emulation morphism for all possible rates of the target?
  - Can’t; but if one emulation is found, then the rates of the target network can be changed arbitrarily and a related emulation will again exist.
Applications of Emulation

- Model Reduction
  - Find reduced networks
  - Compute quotient CRNs
  - Find network symmetries that may be of biological interest

- Morphism Generation
  - Find morphisms between networks (e.g. all the ones for a fixed rate assignment)

<table>
<thead>
<tr>
<th>Models from the BioNetGen database</th>
<th>Aggregation reduction</th>
<th>Emulation reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>[R]</td>
<td>[S]</td>
</tr>
<tr>
<td>M1</td>
<td>3539544</td>
<td>230186</td>
</tr>
<tr>
<td>M2</td>
<td>768567</td>
<td>66548</td>
</tr>
<tr>
<td>M3</td>
<td>1202029</td>
<td>18386</td>
</tr>
<tr>
<td>M4</td>
<td>48</td>
<td>18</td>
</tr>
<tr>
<td>M5</td>
<td>190454</td>
<td>14531</td>
</tr>
<tr>
<td>M6</td>
<td>187468</td>
<td>10734</td>
</tr>
<tr>
<td>M7</td>
<td>32776</td>
<td>2506</td>
</tr>
<tr>
<td>M8</td>
<td>41233</td>
<td>2662</td>
</tr>
<tr>
<td>M9</td>
<td>5634</td>
<td>471</td>
</tr>
<tr>
<td>M10</td>
<td>5797</td>
<td>796</td>
</tr>
<tr>
<td>M11</td>
<td>5532</td>
<td>730</td>
</tr>
<tr>
<td>M12</td>
<td>487</td>
<td>85</td>
</tr>
<tr>
<td>M13</td>
<td>24</td>
<td>18</td>
</tr>
</tbody>
</table>
Network Evolution and Network Robustness
Walks in Network Space

(\[ \text{homomorphism and stoichiomorphism (transitive)} \])
Walks in Network Space

( homomorphism and stoichiomorphism (transitive))
Walks in Network Space

Neutral paths in network space

( homomorphism and stoichiomorphism (transitive))

Side jumps
Another Zoo
Network Perturbations

Network       Normal Behavior       Removing each link in turn

A complex but robust implementation of the simple network

never dead “on average”
Conclusion
Networks are Algorithms

• They are *methods* for achieving a function
  • We need to understand how these methods relate to each other
  • In addition to how and how well they implement function
  • Algorithms can be obfuscated, and nature can obfuscate networks (to what end?)

• Network emulation can be checked *statically*
  • By stoichiometric/reaction-rate (*structural*) properties
  • That is, no need to compare ODE (*functional*) properties
  • For *any* initial conditions and rates of (one of) the networks

• We can efficiently discover emulations
  • Automatic model reduction of large networks
Nature likes good algorithms

Approximate
“default” rates and initial conditions

Exact
any rates and initial conditions

These additional feedbacks do exist in real cell cycles (via indirections)

The cell cycle switch can exactly emulate AM
What Contributes to Complexity?

- Indifference? (does not really cost much)
- Robustness? (resist point failures)
- Adaptability? (neutral paths)
- Noise resistance? (improve signal processing)
- Temperature compensation?
- Etc.