How to Build a Living Cell in Software
or
Can we computerize a bacterium?

Tom Henzinger
IST Austria
Turing Test for E. coli

Fictional ultra-high resolution video showing molecular processes inside the cell

Computer animation of molecular processes inside the cell

Biologist cannot distinguish.
Turing Test for E. coli

Fictional ultra-high resolution video showing molecular processes inside the cell

Computer animation of molecular processes inside the cell

Biologist cannot distinguish.
Fictional ultra-high resolution video showing molecular processes inside the cell

Computer animation of molecular processes inside the cell

Such software must represent all actors and their interaction in real time (think “video game”).
Cell cycle regulation of NOTCH signaling

Nusser-Stein et al. 2012
Which questions about the diagram can be answered with Yes or No?
The Cell as Reactive System [D. Harel]

Reacts to inputs:

- physical (light, mechanical, electrical)
- chemical (molecular receptors, channels)

By producing observable outputs:

- growth, division, death
- movement
- signaling

Generated over time by an internal dynamics:

- metabolism
- cell fate
- cell cycle
\[
\frac{dx}{dt} = f_2(x, y)
\]

until \(f_3(x, y) = 0\)

\[
y = f_1(x, y)
\]

\[
f_4(x, z) < 0
\]

\[
f_4(x, z) \geq 0
\]

Reactive model
\[ \frac{dx}{dt} = f_2(x,y) \]

until \( f_3(x,y) = 0 \)

\[ y = f_1(x,y) \]

\[ f_4(x,z) < 0 \quad f_4(x,z) \geq 0 \]

Distributed state
\[
\frac{dx}{dt} = f_2(x,y) \\
\text{until } f_3(x,y) = 0 \\
y = f_1(x,y) \\
f_4(x,z) < 0, f_4(x,z) \geq 0
\]

\text{Discrete state change (Events)}
\[
dx/dt = f_2(x, y)
\]
until \[ f_3(x, y) = 0 \]

\[
y = f_1(x, y)
\]

\[
f_4(x, z) < 0
\]
\[
f_4(x, z) \geq 0
\]

Continuous state change (Time)
\[ \frac{dx}{dt} = f_2(x,y) \]

until \( f_3(x,y) = 0 \)

\[ y = f_1(x,y) \]

\[ f_4(x,z) < 0 \]

\[ f_4(x,z) \geq 0 \]

\[ \text{Decisions} \]
\[ \frac{dx}{dt} = f_2(x, y) \]

until \( f_3(x, y) = 0 \)

\[ y = f_1(x, y) \]

\[ f_4(x, z) < 0 \quad f_4(x, z) \geq 0 \]

Randomness
\[ \frac{dx}{dt} = f_2(x, y) \]

until \[ f_3(x, y) = 0 \]

\[ y = f_1(x, y) \]

\[ f_4(x, z) < 0 \] \[ f_4(x, z) \geq 0 \]

Interaction
\[ \frac{dx}{dt} = f_2(x,y) \]

until

\[ f_3(x,y) = 0 \]

\[ y = f_1(x,y) \]

\[ f_4(x,z) < 0 \]

\[ f_4(x,z) \geq 0 \]

\[ P_1(x,y) \]

\[ P_2(z) \]

Iteration
Reactive Models are NOT particularly good for

- modeling space (proximity, polarity)
- approximate analysis
Standard Dynamic Models used in Molecular Biology

1. Markovian Population Models for chemical reaction systems
   (continuous time, no decisions, no hierarchy)

2. Boolean/Qualitative Networks for activation/repression systems
   (discrete time, no decisions, no hierarchy)
Chemical Reaction System (Phage $\lambda$)

\[
\begin{align*}
\text{RNA} & \rightarrow \text{RNA} + \text{M} & c_1 &= 0.043 \\
\text{M} & \rightarrow \emptyset & c_2 &= 0.0007 \\
\text{DNA.D} & \rightarrow \text{RNA} + \text{DNA.D} & c_3 &= 0.0715 \\
\text{RNA} & \rightarrow \emptyset & c_4 &= 0.0039 \\
\text{DNA} + \text{D} & \rightarrow \text{DNA.D} & c_5 &= 0.0199 \\
\text{DNA.D} & \rightarrow \text{DNA} + \text{D} & c_6 &= 0.479 \\
\text{DNA.D} + \text{D} & \rightarrow \text{DNA.2D} & c_7 &= 0.0002 \\
\text{DNA.2D} & \rightarrow \text{DNA.D} + \text{D} & c_8 &= 9 \times 10^{-12} \\
\text{M} + \text{M} & \rightarrow \text{D} & c_9 &= 0.083 \\
\text{D} & \rightarrow \text{M} + \text{M} & c_{10} &= 0.5 \\
\end{align*}
\]
Gene Activation/Repression System

RNA $\to$ RNA + M  $c_1 = 0.043$
M $\to$ Φ  $c_2 = 0.0007$
DNA.D $\to$ RNA + DNA.D  $c_3 = 0.0715$
RNA $\to$ Φ  $c_4 = 0.0039$
DNA + D $\to$ DNA.D  $c_5 = 0.0199$
DNA.D $\to$ DNA + D  $c_6 = 0.479$
DNA.D + D $\to$ DNA.2D  $c_7 = 0.0002$
DNA.2D $\to$ DNA.D + D  $c_8 = 9 \times 10^{-12}$
M + M $\to$ D  $c_9 = 0.083$
D $\to$ M + M  $c_{10} = 0.5$

Continuous time, stochastic  
Discrete time, deterministic

No decisions, no hierarchy
Markovian Population Models
Markovian Population Models
Markovian Population Models
Markovian Population Models

Transition:

\((9, 7, 5)\)  \rightarrow  \((8, 6, 6)\)
State Transition Systems

9, 7, 5 → 8, 6, 6
deterministic
State Transition Systems

deterministic
State Transition Systems

deterministic

discrete stochastic

time 0
State Transition Systems

deterministic

9, 7, 5 → 8, 6, 6

discrete stochastic
time 1

9, 7, 5 → 8, 6, 6

0.8

0.2
State Transition Systems

Continuous stochastic time 0
Exit rate 0.5
Expected residence time 2
State Transition Systems

- Continuous stochastic time 1
- Exit rate 0.5
- Expected residence time 2

Diagram of state transitions with states 9, 7, 5 and 8, 6, 6 connected by an edge labeled 0.5.
Markovian Population Models

Syntax: stoichiometric equations (finite object)

Semantics: continuous-time Markov chain (infinite object)
Continuous-Time Markov Chain (CTMC)
Chemist’s syntax:

\[ \delta p_t(x) / \delta t = \sum_{i: x \in H_i} \alpha_i(u_{i^{-1}}(x)) \cdot p_t(u_{i^{-1}}(x)) - \sum_{i: x \in G_i} \alpha_i(x) \cdot p_t(x) \]

Physicist’s syntax (chemical master equation):
Chemist’s syntax:

\[
\frac{\delta p^t(x)}{\delta t} = \sum_{i: x \in H_i} \alpha_i(u_i^{-1}(x)) \cdot p^t(u_i^{-1}(x)) - \sum_{i: x \in G_i} \alpha_i(x) \cdot p^t(x)
\]

Physicist’s syntax (chemical master equation):

Computer scientist’s syntax (stochastic reactive module):

\[
\begin{align*}
\Box x_1 \geq 1 \land x_2 \geq 1 & \quad - 0.2 \cdot x_1 \cdot x_2 \rightarrow x_1 := x_1 - 1; x_2 := x_2 - 1; x_3 := x_3 + 1 \\
\Box x_3 \geq 1 & \quad - 0.1 \cdot x_3 \rightarrow x_3 := x_3 - 1
\end{align*}
\]
Syntax Matters

1. Expressiveness  0
2. Succinctness     IIII IIII IIII
3. Operations       +1

addition
multiplication

XIV x XXXIV   vs.   14 x 34 =

42
56
476
Executable Biology

“Machine model” (rather than equational model):

\[
\begin{align*}
\Box & x_1 \geq 1 \land x_2 \geq 1 \quad - 0.2 \cdot x_1 \cdot x_2 \rightarrow x_1 := x_1 - 1; x_2 := x_2 - 1; x_3 := x_3 + 1 \\
\Box & x_3 \geq 1 \quad - 0.1 \cdot x_3 \rightarrow x_3 := x_3 - 1
\end{align*}
\]

- can be executed (not “simulated”)
- can be composed, encapsulated, and abstracted
- can be dynamically reconfigured
- can be “formally verified” (if you are lucky)

Transient behavior (rather than steady state) of interest.
Four Executions for Phage $\lambda$ Genetic Switch
Probability Mass Propagation for Phage $\lambda$ Genetic Switch

$t = 5000$

$t = 30000$

$t = 15000$

$t = 50000$
Mass Propagation vs. Execution

Program verification vs. Program testing
Phage λ Model

Desired precision: \(3 \times 10^{-6}\)

Probability mass propagation: 55 min runtime
(adaptive uniformization with sliding-window abstraction)
[Didier, H., Mateescu, Wolf 2011]

Execution (\(\beta = 0.95\)): 67 h runtime (\(3 \times 10^8\) runs)
Main Problem with Biochemical Reaction Systems

Well-stirred mixture (gas or fluid)
Crowded cytosol [T. Vickers]
- small numbers of important molecules
- locality, gradients matter
Gene Activation/Repression Systems

Gene A  Gene B  Gene C
DNA

Gene A
Gene B
Gene C

Promoter
Protein code

inactive
inactive
active

GENE REGULATION
DNA

Gene A

Gene B

Gene C

Promoter

Protein code

inactive

inactive

active

Transcription

Protein molecules

Function

GENE REGULATION
DNA

Gene A

inactive

Gene B

Promoter

active

Gene C

Protein code

active

Nucleotides

ATCGCC

Transcription

Protein molecules

Activation

GENE REGULATION
DNA

Gene A  Gene B  Gene C

Promoter  Protein code
active

Transcription
Protein molecules
inactive inactive

Mutation

Nucleotides

Activation

Protein molecules

GENE REGULATION
WAGNER MODEL

A

3,2

activate

3

B

2,1

weight, threshold

activate

C

2,4

repress

1

input
WAGNER MODEL

A

3

3,2

3 - 0 ≥ 2

B

2,1

C

2,4

1 + 0 - 0 < 4

0 < 1

ABC

000

100
WAGNER MODEL

A

3,2

3 - 0 ≥ 2

B

2,1

3 ≥ 1

C

2,4

1 + 3 - 0 ≥ 4

1

ABC

000

100

111
A \rightarrow B: 3 - 2 < 2
B \rightarrow C: 3 \geq 1
C \rightarrow A: 1 + 3 - 2 < 4

WAGNER MODEL

deterministic transition system
MUTATION PROBABILITIES

1. Each nucleotide (A, T, C, or G) mutates to another nucleotide with a given probability $\frac{\pi}{3}$.

2. The weight of a gene $g$ decreases with the number of mutated nucleotides in the promoter region:
   
   if $g$ has a promoter region of length $n$ and $k$ nucleotides are mutated, then $w(g)$ decreases to $w(g) \cdot (1 - k/n)$. 
EVOLVING GENE REGULATORY NETWORK: DISCRETE-TIME MARKOV CHAIN (DTMC)

$p_{00}$

$w_0$  weight function

$p_{01}$

$w_1$

$p_{02}$

$w_2$

$1 - p_{20} - p_{23}$

$p_{03}$

$w_3$

dead
PHENOTYPE = TEMPORAL PROPERTY $\phi$

Oscillation  $\Box((A \Rightarrow \Diamond \neg A) \land (\neg A \Rightarrow \Diamond A))$

Bistability  $\Box(((A \land \neg B) \Rightarrow \Box(A \land \neg B)) \land (\neg A \land B \Rightarrow \Box(\neg A \land B)))$

atomic propositions = genes

Milo et al., Science 2002.
\( \phi \)
Limit probability of truth of $\phi$
1. Repeated Execution

Problem:

The number of weight functions \( w \) grows exponentially with the number of genes.
COMPUTING THE ANSWER

1. Repeated Execution

Problem:

The number of weight functions $w$ grows exponentially with the number of genes.

Solution:

2. Parametric Model Checking

[Giacobbe, Guet, Gupta, H., Paixao, Petrov 2015]
PARAMETRIC MODEL CHECKING

1. Keep inputs, weights, and thresholds as symbolic values ("parameters").
EXAMPLE: TRIPLE REPRESSOR

A
\[w_A, t_A\]

B
\[w_B, t_B\]

C
\[w_c, t_c\]
1. Keep inputs, weights, and thresholds as symbolic values ("parameters").

2. Use SMT solving to compute the constraints on the parameter values such that an LTL formula $\phi$ is satisfied.
EXAMPLE: TRIPLE REPRESSOR

Oscillation:

\[ \phi = \land g \in \{A, B, C\} \Box((g \Rightarrow \lozenge \neg g) \land (\neg g \Rightarrow \lozenge g)) \]

\[ \phi \text{ satisfied iff} \]

\[ i_A > t_A \land i_B > t_B \land i_C > t_C \land \\
 i_B - w_A < t_B \land i_C - w_B < t_C \land i_A - w_C < t_A \]
LIMITATIONS OF THE MODEL

-(over?)simplification of timing

-(over?)simplification of quantitative measures (weights)
The Essence of Computer Science

1. Algorithms
2. Machines/languages: towers of abstraction

- Programming language
- Processor
- Circuit
- Transistor
Tower of Bio-Abstractions

Population
Organism
Organ
Cell
Network
Molecule
Are there useful macros to structure the “hairball”?

We are looking for the organizing principle: the programming language, only then for the program!