Executable Biology

Tom Henzinger

IST Austria
Executable Biology

Tom Henzinger
IST Austria

Modeling of biological objects (e.g. cells) and processes (e.g. pathways) as interacting computer programs.
Biology

- complex natural systems

Computer Science

- complex engineered systems
<table>
<thead>
<tr>
<th>Biology</th>
<th>Computer Science</th>
</tr>
</thead>
<tbody>
<tr>
<td>complex natural systems</td>
<td>complex engineered systems</td>
</tr>
<tr>
<td>primacy of system</td>
<td>primacy of method (&quot;algorithm&quot;)</td>
</tr>
<tr>
<td>Biology</td>
<td>Computer Science</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>------------------------------------------------------</td>
</tr>
<tr>
<td>-complex natural systems</td>
<td>-complex engineered systems</td>
</tr>
<tr>
<td>-primacy of system</td>
<td>-primacy of method (“algorithm”)</td>
</tr>
<tr>
<td>-numeric; stochastic: continuous variables</td>
<td>-symbolic; causalistic: discrete events</td>
</tr>
</tbody>
</table>

Science of *approximation*: bounds errors  
Science of *abstraction*: preserves properties
<table>
<thead>
<tr>
<th>Computational Science</th>
<th>Biology</th>
<th>Computer Science</th>
</tr>
</thead>
<tbody>
<tr>
<td>complex engineered systems</td>
<td>complex natural systems</td>
<td>symbolic; causalistic: discrete events</td>
</tr>
<tr>
<td>primacy of method (&quot;algorithm&quot;)</td>
<td>primacy of system</td>
<td>numeric; stochastic: continuous variables</td>
</tr>
<tr>
<td>Science of <em>approximation</em>: bounds errors</td>
<td></td>
<td>Science of <em>abstraction</em>: preserves properties</td>
</tr>
</tbody>
</table>
The Essence of Computer Science

1. Algorithms
The Essence of Computer Science

1. Algorithms
2. Towers of Abstraction

Property Preservation

High-level Programming Language

Processor

Boolean Gates

Transistors

Compilers CAD etc.
The Essence of Computer Science

1. Algorithms

2. Towers of Abstraction
   - Cloud
   - Virtual Machine

- Property Preservation
- High-level Programming Language
- Processor
- Boolean Gates
- Transistors
- Compilers
- CAD
- etc.
Markovian Population Models
Markovian Population Models
Markovian Population Models
Markovian Population Models
Markovian Population Models

State: (8, 6, 6)
Markovian Population Models

State: $(8, 6, 6)$

State: $(9, 7, 5)$

Transition:

$(9, 7, 5)$

$(8, 6, 6)$
Markovian Population Models

Syntax: stoichiometric equations (finite object)
Markovian Population Models

Syntax: stoichiometric equations (finite object)

Semantics: CTMC (infinite object)
Syntax: Guarded Commands

Dimension $n$: state $\langle x_1, \ldots, x_n \rangle \in S$

state space $S = \mathbb{N}^n$

Finite set of guarded commands:

each consists of
1. guard $G \subseteq S$
2. update function $u : G \rightarrow S$
3. rate function $\alpha : G \rightarrow \mathbb{R}^+$

$G_1 : x_1 \geq 1 \land x_2 \geq 1$
$u_1(x_1, x_2, x_3) = (x_1 - 1, x_2 - 1, x_3 + 1)$
$\alpha_1(x_1, x_2, x_3) = 0.2 \cdot x_1 \land 0.2 \cdot x_2$

$G_3 : x_3 \geq 1$
$u_1(x_1, x_2, x_3) = (x_1, x_2, x_3 - 1)$
$\alpha_1(x_1, x_2, x_3) = 0.1 \cdot x_3$
Syntax matters!

1. Composition and compositionality
2. Expressiveness and succinctness
3. Executability
4. Encapsulation and abstraction

e.g.
- Petri nets: not compositional
- Stoichometric equations: not expressive
- Rate matrices: not succinct
- Differential equations: not executable
Guarded Commands

- compositional
- expressive and succinct
- executable
- counterexample-guided abstraction refinement
- model checking
Genetic Toggle Switch: Four Simulation Runs
Genetic Toggle Switch: Model Checking [DHMW09]

\[ t = 5000 \]

\[ t = 30000 \]

\[ t = 15000 \]

\[ t = 50000 \]
Bacteriophage $\lambda$ Model [DHW09]

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

Model checking: 55 min runtime

Gillespie simulation ($\beta = 0.95$): 67 h runtime ($3 \times 10^8$ runs)
Model Checking CTMCs  [DHMW09]

Transient distributions can be efficiently approximated by combining adaptive uniformization with ideas from Computer Science:

1. on-the-fly state space exploration
2. removal of low-probability states
Transient distributions can be efficiently approximated by combining adaptive uniformization with ideas from Computer Science:

1. on-the-fly state space exploration
2. removal of low-probability states
Opportunity and Challenges

1. Computer Science can contribute to biology far more than *algorithms* and *tools*, e.g. modeling principles such as *reactivity*, *compositionality*, *executability*, *hybrid systems*, *property preserving abstraction*, and *model checking*.

Example: bounded asynchrony [FHMP08]
Opportunity and Challenges

1. Computer Science can contribute to biology far more than *algorithms* and *tools*, e.g. modeling principles such as *reactivity*, *compositionality*, *executability*, *hybrid systems*, *property preserving abstraction*, and *model checking*.

2. Computer Science needs to develop more *quantitative* models.

B versus R
Opportunity and Challenges

1. Computer Science can contribute to biology far more than algorithms and tools, e.g. modeling principles such as reactivity, compositionality, executability, hybrid systems, property preserving abstraction, and model checking.

2. Computer Science needs to develop more quantitative models.

3. Training of scientists!!!
IST Austria

8 biologists, 4 computer scientists, 1 PhD program