Formal Verification has proven useful in Reactive Systems Development (Software/Hardware)

What are the main uses / challenges / future research directions in Biology?

Why biology?
What has been achieved so far?
Where the field is going?
Formal verification can be very powerful but we first need:

- Accurate Computational Models
- Relevant Biological Questions

In this tutorial:

- Do not cover lots of important work
- Recommend looking at proceedings of CMSB Computational Methods in Systems Biology annual conference and DNA Computing and Molecular Programming
<table>
<thead>
<tr>
<th>Natural</th>
<th>vs.</th>
<th>Engineered</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biology – understanding life</td>
<td>Building biological</td>
<td></td>
</tr>
<tr>
<td>and predicting system dynamics</td>
<td>devices robustly</td>
<td></td>
</tr>
<tr>
<td>Gene Regulatory Networks</td>
<td>DNA Strand Displacement</td>
<td></td>
</tr>
<tr>
<td>RE:IN</td>
<td>(DSD)</td>
<td></td>
</tr>
<tr>
<td>RE:IN</td>
<td>Network Base</td>
<td></td>
</tr>
<tr>
<td>Logical Models,</td>
<td>Biocomputation (NBC)</td>
<td></td>
</tr>
<tr>
<td>Boolean Networks</td>
<td>Chemical Reactions Networks</td>
<td>(CRN)</td>
</tr>
</tbody>
</table>
Natural Biological Systems

The basic unit is the Cell

Single Cell / Multi-Cellular

Genotype to Phenotype
Modeling Formalisms – Natural Systems

Case Study – C. elegans VPC

How cells decide to differentiate

System is ‘classical’ in Biology and attracted many modeling efforts
**C. elegans**

A Model Organism

Small (1mm long, 959 cells)
Transparent
Short life cycle (~3 days)
Can freeze and use later
Fixed development
Genome is Sequenced
Powerful experimental techniques available
Data on the same worm
Research community has a tradition of sharing resources
Success recognized in several Nobel Prizes

The Nobel Prize in Physiology or Medicine 2002

"for their discoveries concerning 'genetic regulation of organ development and programmed cell death'"

Sydney Brenner, H. Robert Horvitz, John E. Sulston

The Nobel Prize in Physiology or Medicine 2006

"for their discovery of RNA interference - gene silencing by double-stranded RNA"

Andrew Z. Fire, Craig C. Mello

Programmed Cell Death  RNAi  GFP

The Nobel Prize in Chemistry 2008

Osamu Shimomura, Martin Chalfie, Roger Y. Tsien

The Nobel Prize in Chemistry 2008 was awarded jointly to Osamu Shimomura, Martin Chalfie and Roger Y. Tsien "for the discovery and development of the green fluorescent protein, GFP".
... and genetic regulation of aging

Kenyon et al. Nature 93
Cell fate specification

Sulston and Horvitz, 1977
Kimble and Hirsh, 1979
Sulston et al., 1983
A Modeling Proof-of-Principle
Biologists think in terms of models

A Modeling Proof-of-Principle
What's wrong with our models?

Difficult to predict system behavior

- Time
- Concurrency
- Distributed Control
- Interaction with other components

And this will get worse for larger systems!
Vulval Fates

Vulval Precursor Cells

Time

anchor cell

VPCs

1º Fate  2º Fate  3º Fate

vulval fates  non-vulval fate
VPCs form an equivalence group
The normal pattern of fates is specified by cell-cell interactions
Biological understanding based on logical inferences

**Condition/result:** ablation of the gonad abolishes induction

**Inferred ‘mechanism’:** a gonadal signal induces vulval formation

How do we express this so the computer can understand it?
Background for *lin-15(-)* Modeling

The AC induces VPCs to become $1^\circ$

![Diagram](The AC induces VPCs to become $1^\circ$)

In *lin-15(-)*, all VPCs become $1^\circ$ unless prevented by adjacent VPCs

![Diagram](In *lin-15(-)*, all VPCs become $1^\circ$ unless prevented by adjacent VPCs)

$1^\circ$ VPCs prevent adjacent VPCs from becoming $1^\circ$ (via LIN-12/Notch)

![Diagram](1^\circ$ VPCs prevent adjacent VPCs from becoming $1^\circ$ (via LIN-12/Notch))

Thus, in *lin-15(-)* mutants, the VPCs all race to become $1^\circ$

![Diagram](Thus, in *lin-15(-)* mutants, the VPCs all race to become $1^\circ$)
Postulated Mechanism:
Early Activation of the Inductive Pathway
Biases P6.p to Become $1^\circ$
Modeling Formalisms for VPC Models

- Temporal Logic
- Live Sequence Charts
- Statecharts, Reactive Modules
- Petri Nets
- Boolean Networks
- Ordinary Differential Equations
- Dynamic Bayesian Networks
Basic form of a universal LSC

Structure is similar to an experiment or inference

Kam et al 2004 CMSB, Kam et al 2008 Dev Bio
Existential LSC
Statecharts (Harel 87)

Fisher et al 2005 PNAS
Petri Nets (Petri 63)

Weinstein and Mendoza 2013 Front in Genetics
Boolean Networks + Extensions (Kaufman 69)

Weinstein and Mendoza 2013 Front in Genetics
Ordinary Differential Equations

\[ \frac{d(lat_i)}{dt} = k_n^+ - k_n^- (lat_i) - k_{x_2} (mpk_i^*) (lat_i) \]

\[ \frac{d(mpk_i^*)}{dt} = k_m^+ Ind_i (mpk_i) - k_m^- (Ph_T) (mpk_i^*) \]

\[ - k_{x_1} \frac{(lat_i)^2}{K_{M_{lat}} + (lat_i)^2} (mpk_i^*), \]

\[ + k_{x_3} \frac{\left( \frac{mpk_{i+1}^*}{\nu_{i+1}} + \frac{mpk_{i-1}^*}{\nu_{i-1}} \right)^2}{K_{M_{mpk}}^2 + \left( \frac{mpk_{i+1}^*}{\nu_{i+1}} + \frac{mpk_{i-1}^*}{\nu_{i-1}} \right)^2}, \]

Giurumescu Sternberg, and Asthagiri 2005 PNAS
Dynamic Bayesian Networks

Sun and Hung 2007 Bioinformatics
Verification of VPC models

Temporal Logic

Sequence Charts

Statecharts

Boolean Networks

Petri Nets
Using Temporal Logic in Biology

Using LTL:

“If p2 is not present to stimulate its pathway, but p1 is, is the p3 signal silent?”

\( \Box(\Box(p_1 \land \neg p_2) \rightarrow \neg \Diamond p_3) \)  
(alternatively, using truncated semantics in neutral view)

\( \Box (\Box (p_1 \land \neg p_2) \rightarrow \neg F p_3) \)

Eker et al 01

Necessity of eventually reaching a state in which two signals p1 and p2 are activated from some initial state q1

\( q_1 \rightarrow F(p_1 \land p_2) \)

Eker et al 04
Using Temporal Logic in Biology

Using CTL: Branching logic reasons about the tree of computations
E, A path quantifiers
E – there exists a path A – for all paths

[Montiero et al. 08] classify biological specification into patterns:

1) Occurrence/Exclusion pattern
“It is possible for a state \( p \) to occur” \( EF (p) \)
“It is not possible for a state \( p \) to occur” \( \neg EF (p) \)

Could use LTL and then truncated semantics is potentially relevant:
\( \mathcal{G} (\neg p) \) does not hold for occurrence \( EF (p) \)
\( \mathcal{G} (\neg p) \) holds for exclusion \( \neg EF (p) \)
Temporal Logics Patterns

2) Consequence pattern

“If a state $p$ occurs then it is **possibly** followed by a state $q$”

\[ \mathsf{AG}(p \rightarrow \mathsf{EF} \; q) \]

“If a state $p$ occurs then it is **necessarily** followed by a state $q$”

\[ \mathsf{AG}(p \rightarrow \mathsf{AF} \; q) \]

\[ \mathsf{AG}(p \rightarrow \mathsf{EF} \; q) \] possible occurrence is not in LTL

\[ \Box (p \rightarrow \mathsf{F} \; q) \] holds for necessary consecution $\mathsf{AG}(p \rightarrow \mathsf{AF} \; q)$

Monteiro et al 08
Temporal Logics Patterns

3) Sequence pattern

“A state q is reached and is possibly preceded at some time by a state p”
EF(p \land EF(q))

“A state q is reached and is possibly preceded at all times by a state p”
E(p \lor q)

“A state q is reached and is necessarily preceded at some time by a state p”
EF(q) \land \neg E(\neg p \lor q)

“A state q is reached and is necessarily preceded at all times by a state p”
EF(q) \land \neg E(true) \lor (\neg p \land E((true) \lor q)

Monteiro et al 08
Temporal Logics Patterns

4) Invariance pattern

“A state p can persist indefinitely”
   EG (p)

“A state p must persist indefinitely”
   AG (p)

Additional related patterns:

“Can the system reach a given stable state s?”
   EF (AG (s))

“Must the system reach a given stable state s?”
   AF (AG (s))

AF (AG (s)) cannot be expressed in LTL (different than F G p)

Monteiro et al 08

Chabrier-Rivier et al 04
Invariance and Stabilization

Stabilization:

$$\exists k_1, k_2, \ldots, k_n \text{s.t. } F G (\forall v_i, v_i = k_i)$$

Stabilization in BMA (Fisher) “Exists a unique state that is eventually reached in all executions”

Formula requires quantification on values and variables so cannot directly be expressed in propositional temporal logic

$$F G (s)$$ cannot be expresses in CTL (is different than $AF (AG (s))$ discussed before)

BMA supports GUI for patterns

\[
\begin{align*}
\text{NEXT } \varphi & \overset{\text{def}}{=} X \varphi \\
\varphi_1 \text{ UNTIL } \varphi_2 & \overset{\text{def}}{=} [\varphi_1 U \varphi_2] \\
\varphi_1 \text{ ALWAYS } \varphi_2 & \overset{\text{def}}{=} G \varphi \\
\varphi_1 \text{ EVENTUALLY } \varphi_2 & \overset{\text{def}}{=} [\varphi_1 W \varphi_2] \\
\varphi_1 \text{ RELEASE } \varphi_2 & \overset{\text{def}}{=} [\varphi_2 W (\varphi_1 \wedge \varphi_2)]
\end{align*}
\]
Formal Verification for LSCs

Inherent nondeterminism in executing scenarios

Can be resolved using formal verification (Smart Play-Out)

\[ G( \bigvee_{m_i \in M^U} (act_{m_i} = 1)) \]

Existential charts can be considered as properties that system needs to satisfy

HKMP 2002, FHPSS 2005
Formal Verification for LSCs

LSCs can also be directly translated to temporal logic

**Definition 3.** Let $w = m_1 m_2 m_3 \ldots m_k$ be a finite trace. Let $R = \{ e_1, e_2, e_3 \ldots e_l \}$ be a set of events. The temporal logic formula $\phi_w^R$ is defined as:

$$\phi_w^R = NU(m_1 \land (X(NU(m_2 \land (X(NU(m_3 \ldots)))))))$$

where the formula $N$ is given by $N = \neg e_1 \land \neg e_2 \ldots \land \neg e_l$.

**Definition 4.** Let $LS = \langle M, amsg, mod \rangle$ be an LSC specification. For a chart $m \in M$, we define the formula $\psi_m$ as follows:

- If $mod(m) = \text{universal}$, then $\psi_m = AG(\text{amsg}(m) \rightarrow X(\bigvee_{w \in L_{m \text{trc}}} \phi_w^R))$.
- If $mod(m) = \text{existential}$, then $\psi_m = EF(\bigvee_{w \in L_{m \text{trc}}} \phi_w^R)$.

KHPLB05, KPP11
Statecharts (and other state-based languages)

Exhaustive testing of statechart based models  [Sadot]

Challenges for verification
  Extensions of statecharts
    C++ code
    Variables
    Dynamic object construction

Reactive Modules and Mocha tool [Fisher, and Henzinger]

Petri Nets (Petri 63)

Computation of Attractors [Chatain et al]

Monte Carlo Simulations [Krepska et al]

Simulation Based Model Checking [Li and Miyano]

Colored Petri Nets Verification Tools [Liu and Heiner]

Boolean Networks + Extensions  (Kaufman 69)

Temporal Logic and Model Checking of Boolean Networks, Synchronous and Asynchronous

Finding Fixed Points

Computing Attractors and Basins of Attraction

Stability Analysis (Modular Proof Techniques)

Identifying new Interactions

Weinstein and Mendoza 2013 Front in Genetics, Weinstein et al. BMC Bioinformatics, Cook et al. VMCAI 2005
Dynamic Bayesian Networks

Learns network models from examples and assumptions on influence between components

Can learn different networks with confidence scores

Learning approaches are dominant in Gene Network Inferences

Pros - Deal with noise and stochastic behavior
         Scalability

Cons - Limited in identifying inconsistencies
        Not always mechanistic and hard to explain

Sun and Hung 2007 Bioinformatics
Modeling Gene Regulatory Networks (GRNs)

Every cell’s identity and function is defined by the different genes that it “expresses”.

Genes can activate and inhibit each other’s expression. Gene regulatory networks thus determine which genes are switched on, and which are switched off.

Computational Models can represent dynamics of GRN

- Mechanistic Models based on experimental data
- Allows to simulate new experiments in-silico
  - Starting from new conditions
  - Knockouts or Over Expressions
Which of the optional interactions (1,2,3,4) are necessary to meet these two experimental conditions?

Example: A simple network of 5 genes

<table>
<thead>
<tr>
<th>Gene</th>
<th>Experiment 1</th>
<th>Experiment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>CH</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>Klf4</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>Esrrb</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>Oct4</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>
There are 16 possible networks, but not all of these will satisfy the experimental observations.

Do we have to check all of them?
6 of the networks can explain the experimental data
What happens when things get a little more complicated?
Do we have to check all of them?!
What are Embryonic Stem Cells?

Self-renewing: Divide indefinitely

Pluripotent: Generate all adult cell types
Constraining The Set of Possible Models

Dunn et al., Science 2014
Predictions of ES Cell Behaviour

Self-renewal? Yes / no
Abstract Boolean Network (ABN)

Tuple $N = (G, E, E^?, R)$

$G$ - Finite set of genes

$E$ - Definite interactions (positive or negative)

$E^?$ - Optional interactions (positive or negative)

$R$ – Set of Regulation Conditions
Regulation Conditions

- Defines a set of logical function given positive and negative interactions
- Takes into account if none / some / all activators are present
- Restrict to monotonic functions
- Aims to capture biologically plausible regulation functions
- Recent unpublished work extends regulation conditions
Synthesize Concrete Boolean Network

Tuple $N = (G, E, E^?, R)$

$G$ - Finite set of genes

$E$ - Definite interactions (positive or negative)

$E^?$ - Identify Optional interactions

$R$ – Identify Regulation Conditions
// Settings
directive regulation noThresholds;
directive updates sync;

// Components
S1(0); S2(0); // Signals
A(0..8); B(0..8); C(1,3,5); // TFs

// Definite interactions
S1 S1 positive;
S2 S2 positive;
S1 A positive;
S2 B positive;

// Possible interactions
A C positive optional;
A B positive optional;
B A positive optional;
B C positive optional;

// Observation predicates
$Conditions1 := { S1 = 0 and S2 = 1 };
$Conditions2 := { S1 = 1 and S2 = 1 };
$Expression1 := { A = 1 and B = 1 and C = 1 };
$Expression2 := { A = 0 and B = 1 and C = 1 };

// Observations
#Experiment1[0] | = $Conditions1 and
#Experiment1[0] | = $Expression1 and
#Experiment1[18] | = $Expression2 and
fixpoint(#Experiment1[18]);

#Experiment2[0] | = $Conditions2 and
#Experiment2[0] | = $Expression2 and
#Experiment2[18] | = $Expression1 and
fixpoint(#Experiment2[18]);

Synthesis Algorithm: Find Solutions that satisfy all constraints if possible (Z3-4Bio Framework)

Inconsistent: no concrete programs exist
RE:IN Tool - A Method to Identify and Analyze Gene Regulatory Networks through Automated Reasoning

Network Motifs

Motif finding

+ Scalable motif finding algorithms
- Often, static networks are considered
- Networks are rarely precisely known

Motif dynamics

+ Detailed quantitative predictions
- Motifs are studied in isolation
- Parameters are often not known
### Stem Cell Motifs

<table>
<thead>
<tr>
<th>Motif</th>
<th>$M$</th>
<th>$\overline{M}$</th>
<th>$t_M$ (t'$_M$)</th>
<th>$t_{\overline{M}}$ (t'$_{\overline{M}}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPFFa</td>
<td>true</td>
<td>true</td>
<td>630.62(19.19)</td>
<td>460.10(18.65)</td>
</tr>
<tr>
<td>CPFFb</td>
<td>true</td>
<td>true</td>
<td>277.47(18.40)</td>
<td>586.21(18.17)</td>
</tr>
<tr>
<td>IFFd</td>
<td>false</td>
<td>true</td>
<td>26.27(19.23)</td>
<td>636.10(19.16)</td>
</tr>
<tr>
<td>CNFFa</td>
<td>true</td>
<td>true</td>
<td>489.93(18.43)</td>
<td>611.28(18.90)</td>
</tr>
<tr>
<td>CNFFb</td>
<td>true</td>
<td>true</td>
<td>537.20(18.28)</td>
<td>493.21(17.90)</td>
</tr>
<tr>
<td>IFFa</td>
<td>true</td>
<td>false</td>
<td>609.35(19.95)</td>
<td>540.18(18.39)</td>
</tr>
<tr>
<td>IFFb</td>
<td>true</td>
<td>true</td>
<td>568.20(19.15)</td>
<td>555.36(17.77)</td>
</tr>
<tr>
<td>IFFc</td>
<td>true</td>
<td>true</td>
<td>515.28(20.74)</td>
<td>619.15(19.11)</td>
</tr>
<tr>
<td>PFB2a</td>
<td>true</td>
<td>false</td>
<td>467.51(18.01)</td>
<td>112.02(18.40)</td>
</tr>
<tr>
<td>PFB2b</td>
<td>true</td>
<td>true</td>
<td>505.46(18.21)</td>
<td>505.40(18.74)</td>
</tr>
<tr>
<td>NFB2</td>
<td>false</td>
<td>true</td>
<td>25.05(17.91)</td>
<td>640.31(17.72)</td>
</tr>
<tr>
<td>PFB3a</td>
<td>true</td>
<td>false</td>
<td>575.88(17.62)</td>
<td>525.42(19.04)</td>
</tr>
<tr>
<td>PFB3b</td>
<td>true</td>
<td>true</td>
<td>410.93(17.84)</td>
<td>531.24(17.73)</td>
</tr>
<tr>
<td>NFB3a</td>
<td>false</td>
<td>true</td>
<td>25.04(17.73)</td>
<td>572.96(17.65)</td>
</tr>
<tr>
<td>NFB3b</td>
<td>true</td>
<td>true</td>
<td>522.14(18.03)</td>
<td>531.86(17.96)</td>
</tr>
</tbody>
</table>

**Diagram:**
- **Red lines:** Definite repression.
- **Blue lines:** Definite activation.
- **Dotted lines:** Possible activation.
- **Dashed lines:** Possible repression.
Systematic Motif Exploration
Temporal gene expression data + spatial domains

FULLY EXPLAINS EXPERIMENTAL DATA

- Solved discrepancies
- No need for hard-coded terms

Paoletti, Yordanov, Wintersteiger, Hamadi, Kugler. *CAV* 2014
Neuron Specification in mammalian Cortex  Shavit et al. (with Livesey Lab)
Engineered Biological Systems

Build new Computational Devices
  Fast
  Energy efficient

To better understand Biology

Interact with living systems
  Diagnostic
  Medicine
DNA Computing

Use biological material to design computational circuits (Adleman, 1994)

One promising paradigm is DNA Strand Displacement

Based on complementarity of DNA strands

Programming Language and simulator translates to CRN representations

Programmable DNA binding

• Short complementary domains bind \textit{reversibly}

• Long complementary domains bind \textit{irreversibly}
DNA Strand Displacement

DSD Logic Gate [Output = Input1 AND Input2]
DNA Strand Displacement

DSD Logic Gate [Output = Input1 AND Input2]
DNA Strand Displacement

DSD Logic Gate [Output = Input1 AND Input2]
DNA Strand Displacement

DSD Logic Gate [Output = Input1 AND Input2]
DNA Strand Displacement

DSD Logic Gate [Output = Input1 AND Input2]
Chemical Reaction Networks (CRNs)

\[
\begin{align*}
X + G & \leftrightarrow XG \\
Y + G & \leftrightarrow GY \\
XG + Y & \rightarrow XGY + O \\
GY + X & \rightarrow XGY + O
\end{align*}
\]
DNA Strand Displacement
### Programming Examples

<table>
<thead>
<tr>
<th>Specification</th>
<th>Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y := 2 X</td>
<td>X -&gt; Y + Y</td>
</tr>
<tr>
<td>Y := ⌊X/2⌋</td>
<td>X + X -&gt; Y</td>
</tr>
<tr>
<td>Y := X1 + X2</td>
<td>X1 -&gt; Y</td>
</tr>
<tr>
<td>Y := min (X1, X2)</td>
<td>X2 -&gt; Y</td>
</tr>
<tr>
<td></td>
<td>X1 + X2 -&gt; Y</td>
</tr>
</tbody>
</table>

Luca Cardelli, 2019
<table>
<thead>
<tr>
<th>Specification</th>
<th>Program</th>
</tr>
</thead>
</table>
| $Y := \max (X_1, X_2)$ | $X_1 \rightarrow L_1 + Y$
| | $X_2 \rightarrow L_2 + Y$
| | $L_1 + L_2 \rightarrow K$
| | $Y + K \rightarrow$ |
| | $\max (X_1, X_2) := X_1 + X_2 - \min(X_1, X_2)$ |

Luca Cardelli, 2019
Computing with CRNs

What does the following CRN compute?

\[
\begin{align*}
X + Y &\rightarrow X + B \\
Y + X &\rightarrow Y + B \\
B + X &\rightarrow X + X \\
B + Y &\rightarrow Y + Y
\end{align*}
\]
Approximate Majority

Microsoft Research
Approximate Majority – Visual DSD

directive simulation { final=0.15; points=1000 }
directive parameters [r = 1.0]

| X + Y ->{r} X + B |
| Y + X ->{r} Y + B |
| X + B ->{r} X + X |
| Y + B ->{r} Y + Y |

Approximate Majority – Visual DSD

directive simulation { final=0.15; points=1000 }
directive parameters [r = 1.0]

| X + Y ->{r} X + B
| Y + X ->{r} Y + B
| X + B ->{r} X + X
| Y + B ->{r} Y + Y

| 30 X
| 20 Y
Approximate Majority – Continuous Semantics
Formal Verification of Strand Displacement Systems – Discrete Semantics

(* Signal strand *)
def S(N, x) = N * <t^ x>

(* Transducer gate *)
def T(N, x, y) = new c
    (N * {t^*}: [x t^]: [c]: [a t^]: [:]
     | N * [x]: [t^ y]: [c]: [t^ a]: {t^},
     | N * <t^ c a>
     | N * <y c t^> )

DSD Code - Transducer

label "all_done"
    = strands_reactive=output &
      output=N & gates_reactive=0;

A [ G "deadlock" => "all_done" ]
E [ F "all_done" ]

CTL property checked by PRISM

Lakin, Parker, Cardelli, Kwiatkowska, Phillips RSIF 2009
Probabilistic Verification – CTMC Semantics

(*) Signal strand *)
def S(N, x) = N * <t^< x>

(*) Transducer gate *)
def T(N, x, y) = new c
  ( N * {t^*}:[x t^]:[c]:[a t^]:[a]
  | N * [x]:[t^ y]:[c]:[t^ a]:{t^*}
  | N * <t^ c a>
  | N * <y c t^> )

DSD Code - Transducer

P=? [ F[T,T] "deadlock" ]
P=? [ F[T,T] "deadlock" & "all_done" ]
P=? [ F[T,T] "deadlock" & !"all_done" ]

PCTL property checked by PRISM

Lakin, Parker, Cardelli, Kwiatkowska, Phillips RSIF 2009
DNA device verification

Among DNA circuit constructed experimentally

[Qian, Winfree, Science, 2011; Chandran, Gopalkrishnan, Phillips, Reif, DNA17, 2011]

\[ \text{Output} = \left\lfloor \sqrt{\text{Input}} \right\rfloor \]

Yordanov, Wintersteiger, Hamadi, Phillips, Kugler. DNA19, 2013
Model Generation

def Input1() = <1^ 2>
def Input2() = <3 4^>
def Output() = <2 3>
def AND() = {1^*}{2 3}{4^*}
(N1*Input1()
| N2*Input2()
| N*AND()
| 0*Output())

Visual DSD

SMT encoding

$\mathcal{T} = (Q, q_0, T)$
DNA Verification Strategies

• Inductive invariants - conservation of strands

• Acceleration - multiple reactions firing

Yordanov, Wintersteiger, Hamadi, Phillips, Kugler. DNA’19
Yordanov, Wintersteiger, Hamadi, Kugler. NFM’13
Network Based Biocomputation

Bio4Comp Horizon2020 European Project:

Lund University, Technische Universität Dresden, Linnaeus University, Molecular Sense Ltd, Bar-Ilan University, Fraunhofer Gesellschaft

Develop Network Based Biocomputation:
   Speed up solution of complex problems
   Low energy
   Interface with Biological material

Formal Verification of Biocomputation Circuits
Network Based Biocomputation (NBC)

Build Nanofabricated device (using Electron Beam Lithography)

Molecular agents can travel through network exploring it in parallel

Use actin-myosin filaments or microtubules-kinesin (speeds 5-10 μms⁻¹, 0.5-1 μms⁻¹)

Two Types of Junctions: Pass, Split

Measure agents exiting device

Deduce the answer using exit information

Nicolau et al. PNAS 2016
The Subset Sum Problem (SSP)

SSP: Given $S = \{a_1, a_2, \ldots, a_n\}$ each $a_i$ an integer and integer $T$ decide if there is a subset $S^* \subseteq S$ whose sum is $T$, i.e. $\sum a_{i_k} = T$ where $a_{i_k} \in S^*$

Decision Problem – only need to answer:

Yes if there is such a subset
No if there isn’t

Example $S = \{2, 5, 9\}$ $T = 11$ Yes $T = 13$ No

SSP is known to be NP Complete (NPC)
⇒ No polynomial algorithm unless $P = NP$
⇒ Can reduce many other interesting problems to SSP
Network Based Biocomputation (NBC)

Solving SSP using NBC:

Agents start at top left entry and exit at bottom row

At split junctions ● agents can decide if to move down or diagonally

At pass junctions ○ agents continue in the direction it was travelling

A choice to move diagonally in split junction -> adding a number to the sum
Agent traversing yellow path $11 = 2 + 9$
Network Based Biocomputation

Some experimental challenges:

Distribution in split junction not symmetric

Errors in pass junctions

Agents get stuck in junction

Agents “climb out” of tracks and land in wrong positions
Network Encoding of ExCov (Exact Cover)

- EXCOV sets represented as binary numbers
- Each EXCOV Set encoded into the network as one decimal number
- RESET junctions prevent addition of colliding sets

<table>
<thead>
<tr>
<th>Set 1 {2;4}</th>
<th>Set 2 {2;3}</th>
<th>Set 3 {1;3}</th>
<th>Set 4 {1;2}</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^0$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$2^1$</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$2^2$</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$2^3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Decimal Numbers</td>
<td>10</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>Target Sum</td>
<td>15</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
10-set EXCOV: One Solution
Formal Verification of NBC Circuits

Eliminate logical errors before manufacturing circuits

Prototype new NBC ideas, complementing simulation tools

Identify faulty junctions using experimental measurements of exits
Formal Verification of SSP Network

Define Transition System:

**Variables**  \( x, y, dir \)
\( x, y : 1 .. (\sum a_i) \)
\( dir : \{0,1\} \)  \((0 – down, 1 – diagonally)\)

**Transition Relation**
\( y' = y + 1 \)
\( (x' = x \land dir = 0) \lor (x' = x + 1 \land dir = 1) \)

\( dir \) allowed to non-deterministically choose 0 or 1 at split junction, no change in pass junction.

**Initial Condition**
\( (x' = 1 \land y = 1 \land (dir = 0 \lor dir = 1)) \)
Formal Verification of SSP Network

CTLSPEC NAME csum :=!(EX(AG((flag = FALSE)|(column = sum))));

CTLSPEC NAME nsum :=!(EF((flag = TRUE)&(column = xsum)));

<table>
<thead>
<tr>
<th>ID</th>
<th>Set Size</th>
<th>Set</th>
<th>Spec</th>
<th>Spec Validity</th>
<th>Tag Runtime</th>
<th>No Tag Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3</td>
<td>[2, 3, 5]</td>
<td>csum</td>
<td>VALID</td>
<td>0.0011</td>
<td>0.0011</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>[2, 3, 5]</td>
<td>nsum</td>
<td>VALID</td>
<td>0.0009</td>
<td>0.0009</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>[2, 3, 5, 7]</td>
<td>csum</td>
<td>VALID</td>
<td>0.0013</td>
<td>0.0012</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>[2, 3, 5, 7]</td>
<td>nsum</td>
<td>VALID</td>
<td>0.0009</td>
<td>0.0009</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>[2, 3, 5, 7, 11]</td>
<td>csum</td>
<td>VALID</td>
<td>0.0018</td>
<td>0.0013</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>[2, 3, 5, 7, 11]</td>
<td>nsum</td>
<td>VALID</td>
<td>0.0009</td>
<td>0.0009</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>[2, 3, 5, 7, 11, 13]</td>
<td>csum</td>
<td>VALID</td>
<td>0.0037</td>
<td>0.0018</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>[2, 3, 5, 7, 11, 13]</td>
<td>nsum</td>
<td>VALID</td>
<td>0.0009</td>
<td>0.0009</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>[2, 3, 5, 7, 11, 13, 17]</td>
<td>csum</td>
<td>VALID</td>
<td>0.0092</td>
<td>0.0025</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>[2, 3, 5, 7, 11, 13, 17]</td>
<td>nsum</td>
<td>VALID</td>
<td>0.0009</td>
<td>0.0009</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>[2, 3, 5, 7, 11, 13, 17, 19]</td>
<td>csum</td>
<td>VALID</td>
<td>0.0260</td>
<td>0.0042</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>[2, 3, 5, 7, 11, 13, 17, 19]</td>
<td>nsum</td>
<td>VALID</td>
<td>0.0009</td>
<td>0.0009</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>[2, 3, 5, 7, 11, 13, 17, 19, 23]</td>
<td>csum</td>
<td>VALID</td>
<td>0.0821</td>
<td>0.0074</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>[2, 3, 5, 7, 11, 13, 17, 19, 23]</td>
<td>nsum</td>
<td>VALID</td>
<td>0.0010</td>
<td>0.0009</td>
</tr>
</tbody>
</table>

Table 8 SSP general sum verification runtimes in minutes.
Future Outlook

Formal Verification tools used as mainstream approach in Genetic Network Inference and Analysis

Whole Tissue models – Verification and Reasoning

Industrial applications for biodevices will require certification opening key role for FV tools
Thanks for Listening!

Til Korten, Stefan Diez - Technische Universität Dresden
Dan Nicolau Jr. - Molecular Sense Ltd.

Sara Jane Dunn, Boyan Yordanov, Andrew Phillips - Microsoft Research Cambridge

Michelle Aluf-Medina, Tamar Viclizky, Ani Amar, Amit Schussheim, Avraham Raviv - Bar Ilan University

Jane Hubbard NYU

David Harel Weizmann

All Bio4Comp members

Funding: European Commission Horizon 2020
Israeli Science Foundation (ISF)