# SMT-based Analysis of Biological Computation

**B. Yordanov**, C. M. Wintersteiger, Y. Hamadi, H. Kugler

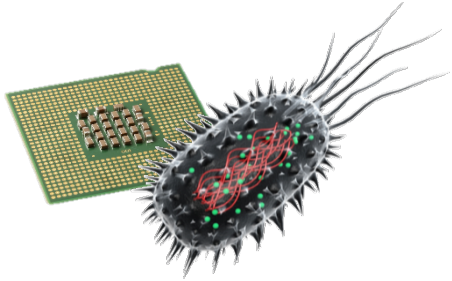
Microsoft Research, Cambridge, UK

NASA Formal Methods Symposium, May 14, 2013

Microsoft Research

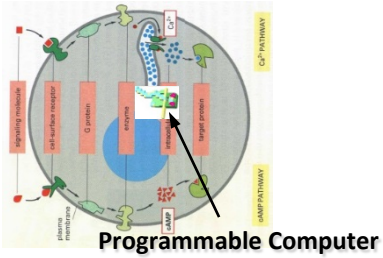
# Biological Computation

- Information processing within living organisms
- Programming biology
  - Synthetic Biology: use DNA "parts" to construct novel systems
  - DNA Computing: engineer chemical reaction networks directly from DNA



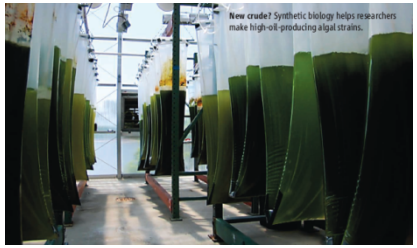
Understanding life

Medicine in 2050: "Doctor in a Cell"




Ehud Shapiro, DNA5, 1999

Medical Treatment




*"Algae's second try", Science, 2011*

Sustainable energy

# Computational Modelling

- Integrate experimental observations and biological insight into formal representations
- Make predictions that can be tested experimentally
- Guide the engineering of biological systems
- Challenges
  - Fragmentation of modelling approaches and formalisms
  - Realistic models are hard to analyse
  - Simulation is often the main analysis strategy

# Formal Methods in Biology

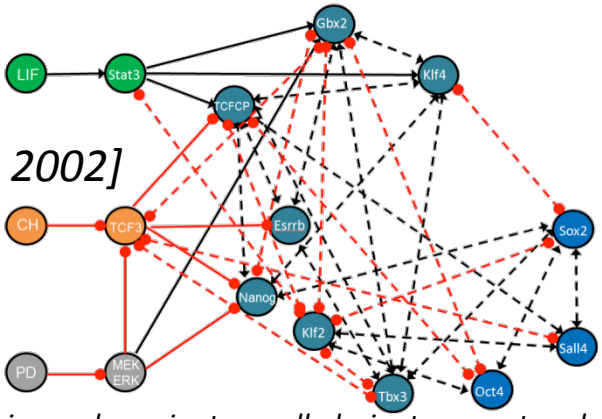
- Model checking *[Chabrier & Fages, 2003; survey in Carrillo et al. 2012]*
- Probabilistic model checking
  [Kwiatowska et al., 2008; Lakin et al., 2012]
- Answer set programming [Gebser 2008]
- Constraint programming [Devloo et al. 2003]
- Formal synthesis [Corblin et al., 2012; Ray et al. 2010]
- BDD-based [Garg et al. 2007]
- SAT-based [Dubrova & Teslenko, 2011; Tiwari et al. 2007, Fagerberger et al. 2012]

# SMT-based Analysis

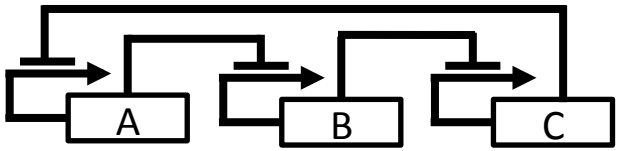
- **Expressive**: captures various modelling formalisms and analysis questions
  - Design constraints
  - Structural properties
  - Functional properties
- **Scalable**: handles models of practical interest
  - Largest available DNA designs, millions of circuits with hundreds of species each, running in parallel
- **Extensible**: additional formalisms and procedures can be integrated
  - Currently: (quantified) bit-vectors, integers
  - Future: reals, probabilistic SMT, etc.

# Synthetic Biology

- Gene Regulation Networks
  - Various modelling approaches *[de Jong, 2002]*
  - Boolean Network Model *[Kauffman, 1969]*
- Synthetic gene networks
  *[Gardner et al., 2000; Elowitz & Leibler, 2000]*
- Genetic Parts *[Knight, 2003]*
- DNA assembly *[Gibson et al., 2009]*
- Computer-aided design
  *[Pedersen & Phillips, 2009; Beal et al. 2012]*
- From modules to systems *[Purnick & Weiss, 2009]*
  - Crosstalk in chemical "wires" *[Tamsir et al., 2011; Moon et al. 2012]*
- Dynamic behaviour requirements [Huynh et al. 2012]

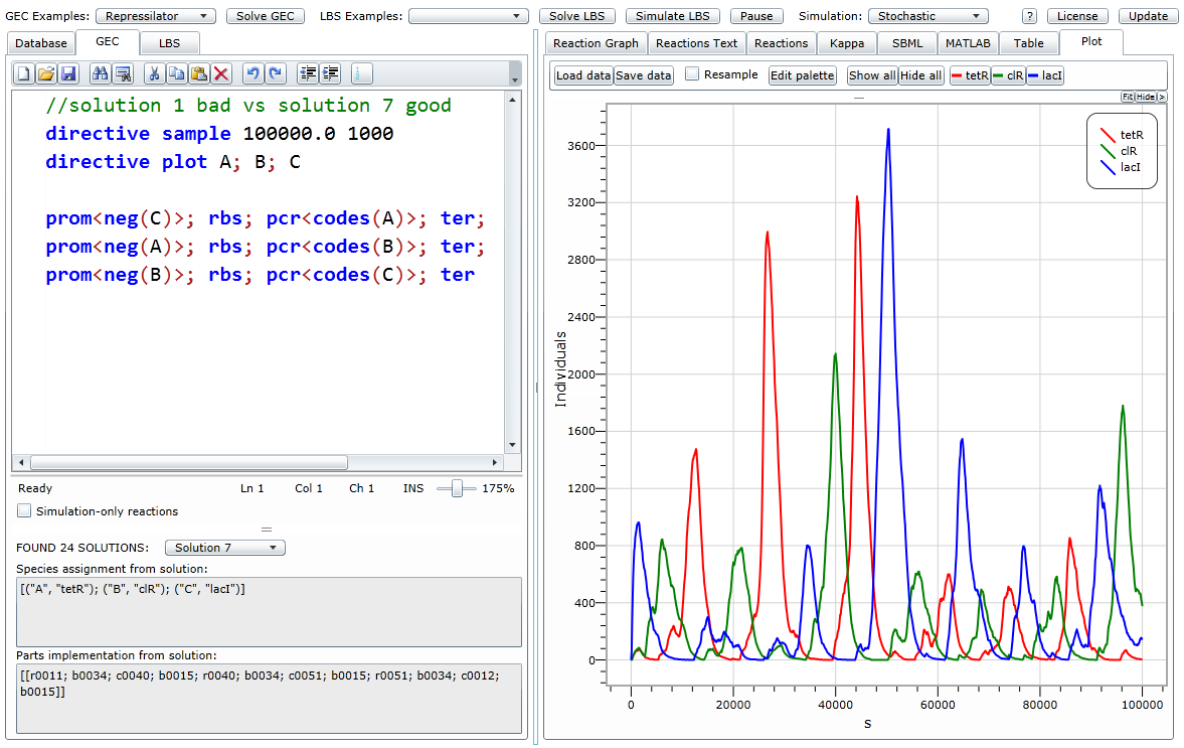



*Murine embryonic stem cell pluripotency network, work with S-J. Dunn and G. Martello*



# Visual GEC

- Given a Visual GEC program, select a set of DNA “parts” to implement the design

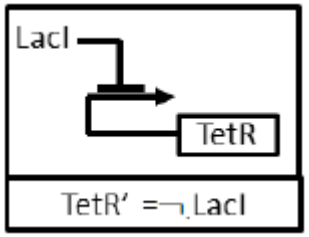



*Pedersen & Phillips , 2009*

# Device Encoding

- Device dynamics are captured using synchronous Boolean update rules
- Bit vector state encoding

Device

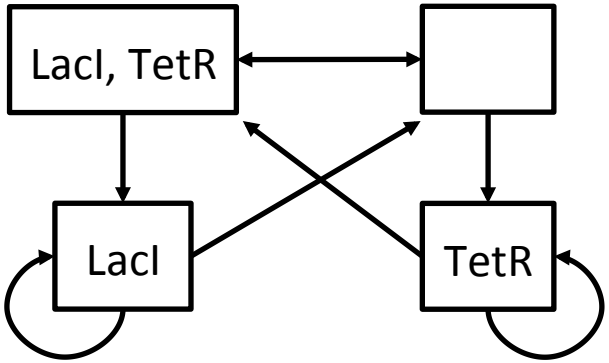



$$d = (I_d, S_d, F_d)$$

$$F_d = \{f_d^s \mid s \in S_d\}$$

$$f_d^s : Q_d \to \mathbb{B}$$

Transition system encoding

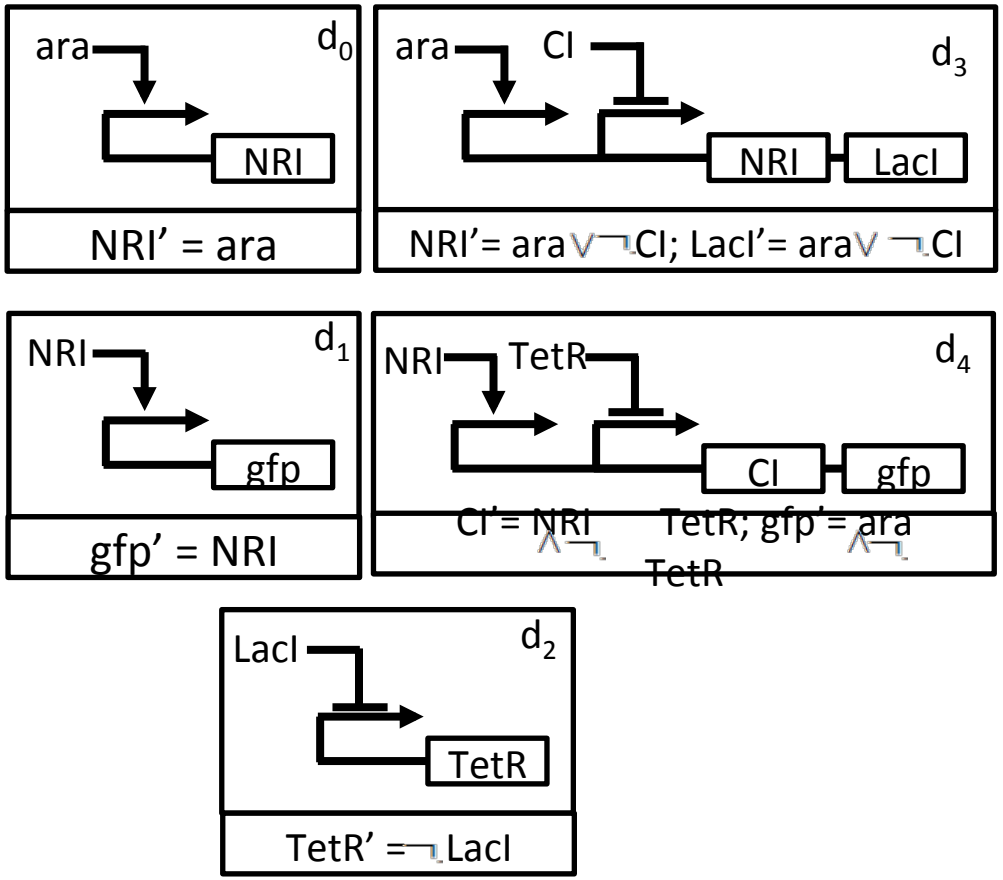



$$\mathcal{T}_d = (Q_d, Q_{d0}, T_d)$$

$$T_d(q, q') \leftrightarrow \left( \bigwedge_{s \in S_d} q'(s) = f_d^s(q) \right)$$

# Device Library Encoding

- Composition of device transition systems




$$\mathcal{D} = \{d_0, \ldots, d_n\}$$

$$\mathcal{S} = \bigcup_{d \in \mathcal{D}} (I_d \cup S_d)$$

$$I \subseteq \mathcal{S} \qquad O \subseteq \mathcal{S}$$

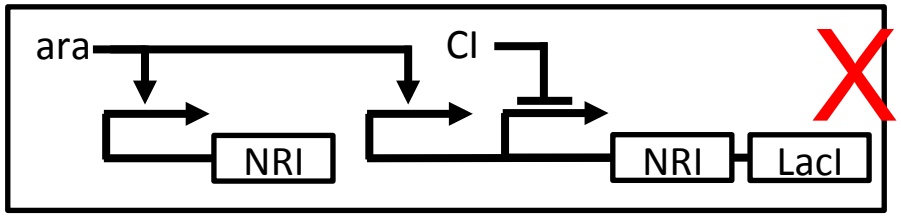
$$\forall q \in Q, \bigwedge_{s \in \mathcal{S} \setminus I} \left( \left( \bigwedge_{d \in D_s} \neg D(d) \right) \rightarrow \neg q(s) \right)$$

$$\mathcal{T} = (Q, Q_0, T)$$

$$T(q, q') \leftrightarrow \left( \bigwedge_{s \in \mathcal{S}} q'(s) = f_d^s(q_d) \right)$$

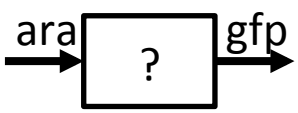
# Additional Constraints

| Constraints | Description |
|---|---|
| $\bigwedge_{s \in \mathcal{S}} \bigwedge_{d,d' \in D_s, d \neq d'} \neg(D(d) \wedge D(d'))$ | To prevent cross-talk, two devices producing the same species are never selected at the same time. |
| $\bigwedge_{s \in I} \bigvee_{d \in D_s} D(d)$ | All species specified as input serve as inputs to a selected device. |
| $\bigwedge_{s \in O} \bigvee_{d \in D^s} D(d)$ | All species specified as output are produced by a selected device. |
| $\bigwedge_{d \in \mathcal{D}} \left( D(d) \rightarrow \bigwedge_{s \in S_d \setminus O} \bigvee_{d' \in D_s} D(d') \right)$ | To prevent the production of species that do not serve any function, all species produced by a selected device are outputs of the circuit or serve as input to another selected device. |
| $\bigwedge_{d \in \mathcal{D}} \left( D(d) \rightarrow \bigwedge_{s \in I_d \setminus I} \bigvee_{d' \in D^s} D(d') \right)$ | All species serving as inputs to a selected device are inputs of the circuit or are produced by another selected device in order to ensure that all device inputs are part of the system. |



# Desired Properties

- The system is influenced by certain chemical inputs and produces specific chemical outputs
- Specific dynamical behaviour is achieved
  - oscillations vs. stabilization




*The output oscillates for one value of the input and stabilizes for the other.*

$$\left.\begin{array}{l} \bigwedge_{i=1}^{K} q_i(\mathrm{ara}) = q_0(\mathrm{ara}) \\ \bigwedge_{i=1}^{K} q'_i(\mathrm{ara}) = q'_0(\mathrm{ara}) \\ q_0(\mathrm{ara}) \neq q'_0(\mathrm{ara}) \end{array}\right\}$$

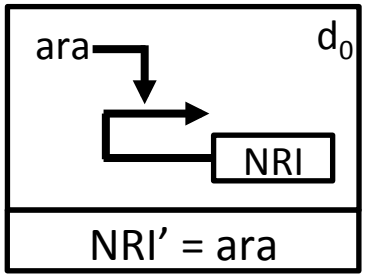
(a different, constant input signal is applied in each case)

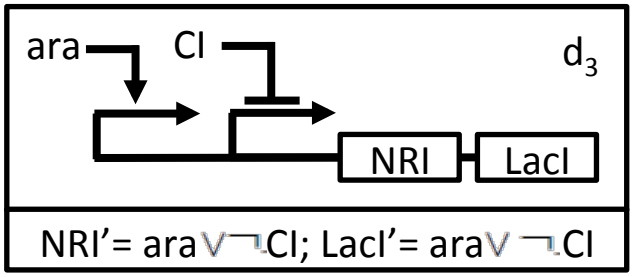
$q_{K-1} = q_K$ (in the first case, the circuit stabilizes)

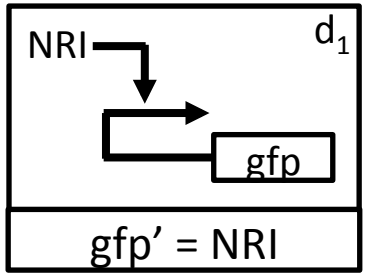
$\bigvee_{i=0}^{K-1}(q'_i = q'_K \wedge \bigvee_{j=i+1}^{K-1} q'_j(\mathrm{gfp}) \neq q'_K(\mathrm{gfp}))$

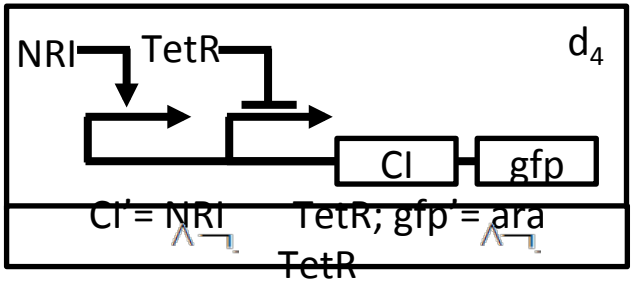
(in the second case, the circuit oscillates)

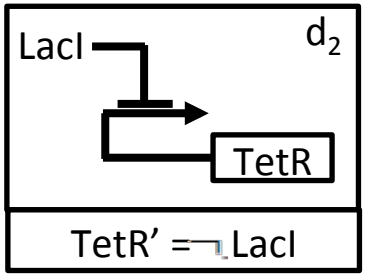
# Identified Designs

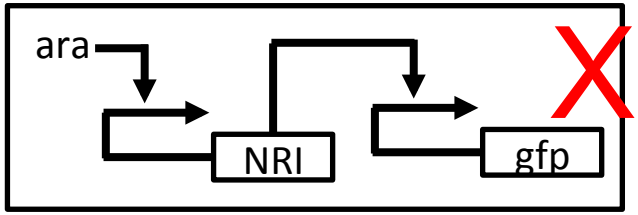


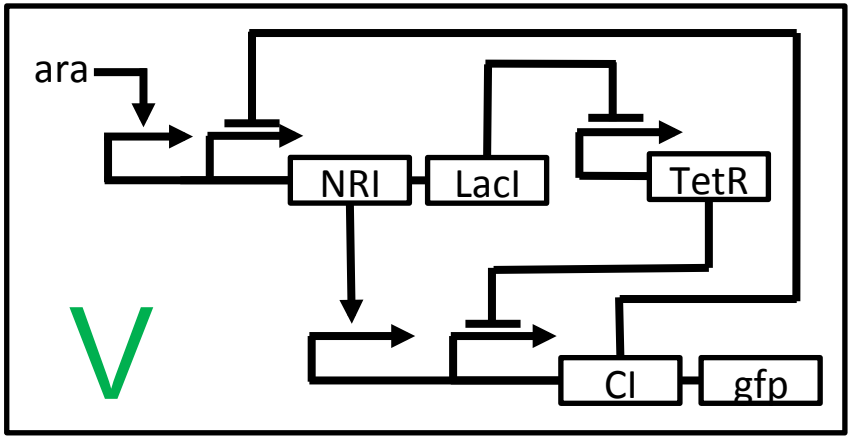



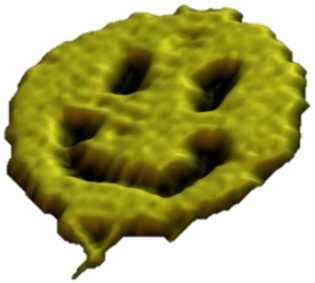



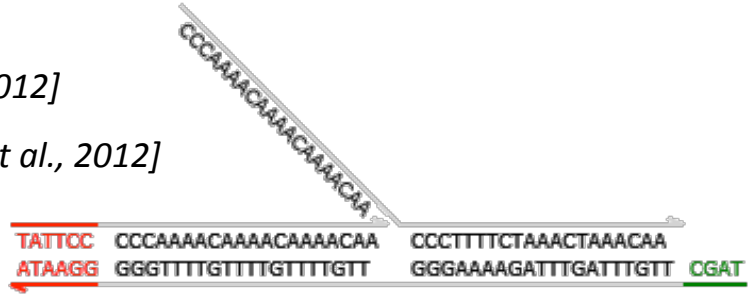



# DNA Computing

- Hamiltonian path problem *[Adleman, 1994]*
- DNA Origami *[Rothemund, 2006; Lin et al., 2006]*
- DNA Strand Displacement (DSD) *[Zhang, 2011]*
  - Arbitrary CRNs *[Soloveichik et al. 2011]*
  - Turing-powerful computation *[Lakin & Phillips, 2011]*
  - Large-scale DSD circuits *[Qian & Winfree, 2011]*
  - DSD Calculus *[Phillips & Cardelli, 2009]*
  - Visual DSD tool *[Lakin et al., 2011]*
  - Square Root Circuit [Qian & Winfree, *2011;Chandran et al., 2011]*

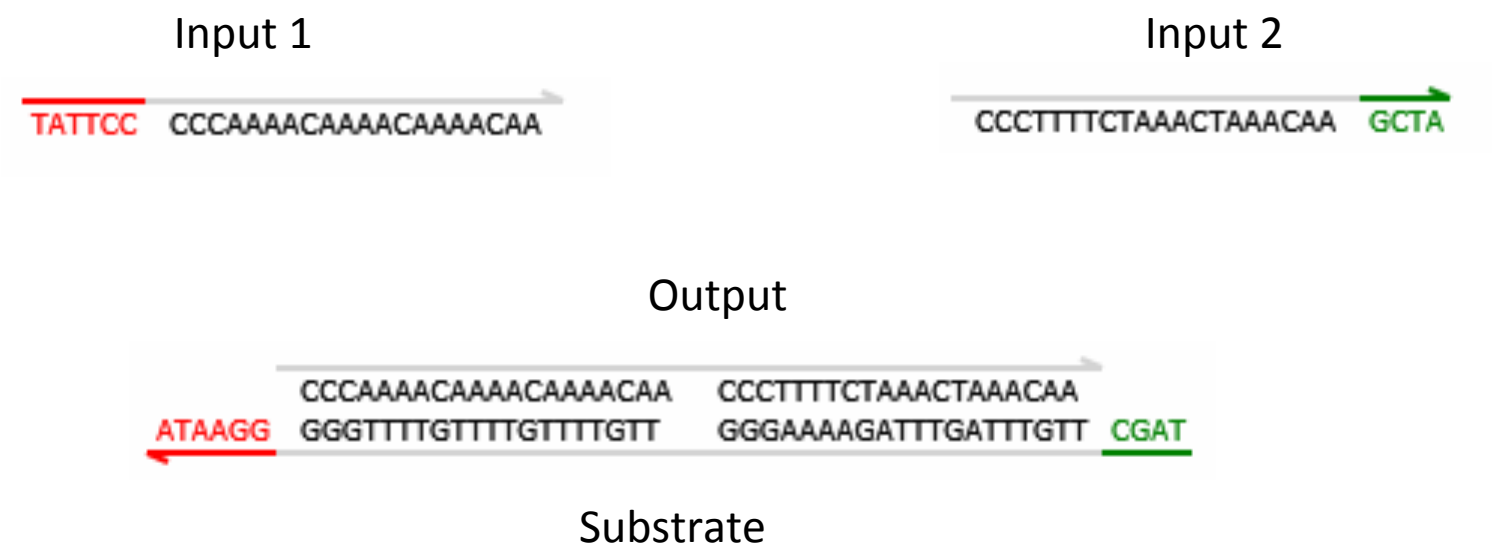


- DSD Analysis
  - Equivalence of CRNs *[Dong, 2012; Shin, 2012]*
  - Probabilistic model checking *[Lakin et al., 2012]*



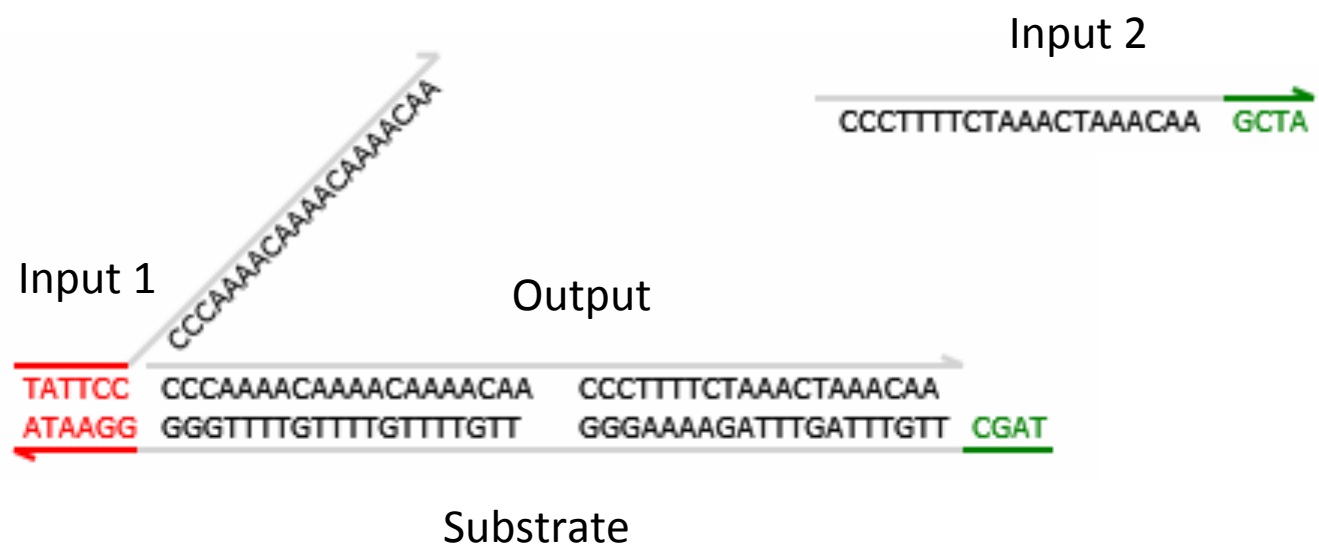
# DNA Strand Displacement

- Chemical reactions between DNA species
- Complementarity of short/long DNA domains
- Example: DSD Logic Gate [Output = Input1 AND Input2]



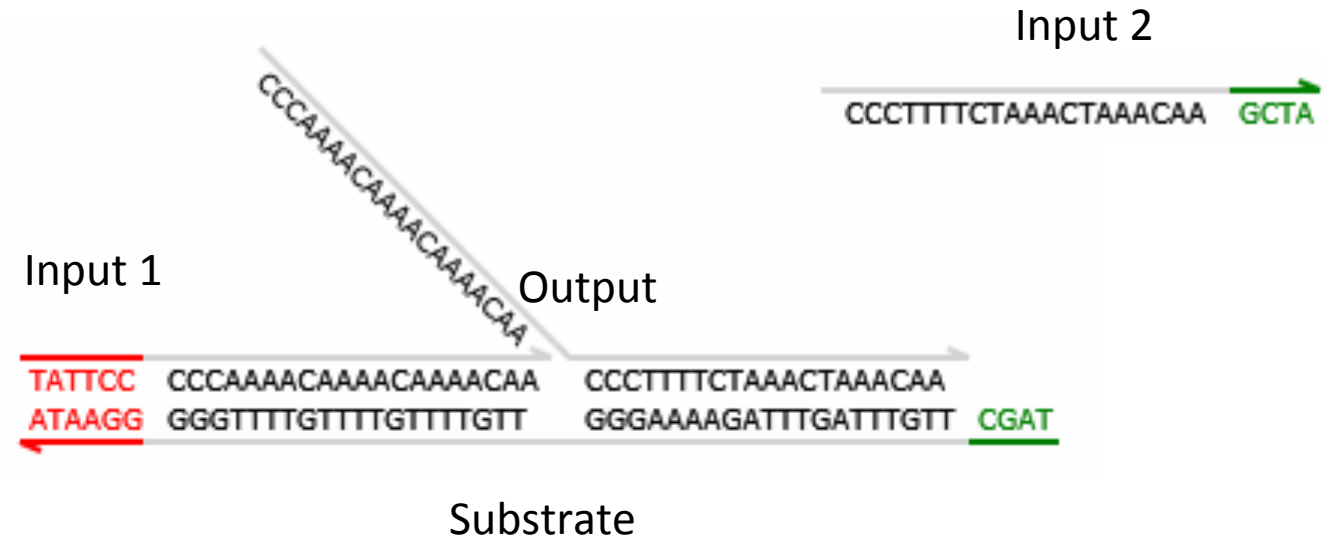
# DNA Strand Displacement

- Chemical reactions between DNA species
- Complementarity of short/long DNA domains
- Example: DSD Logic Gate [Output = Input1 AND Input2]



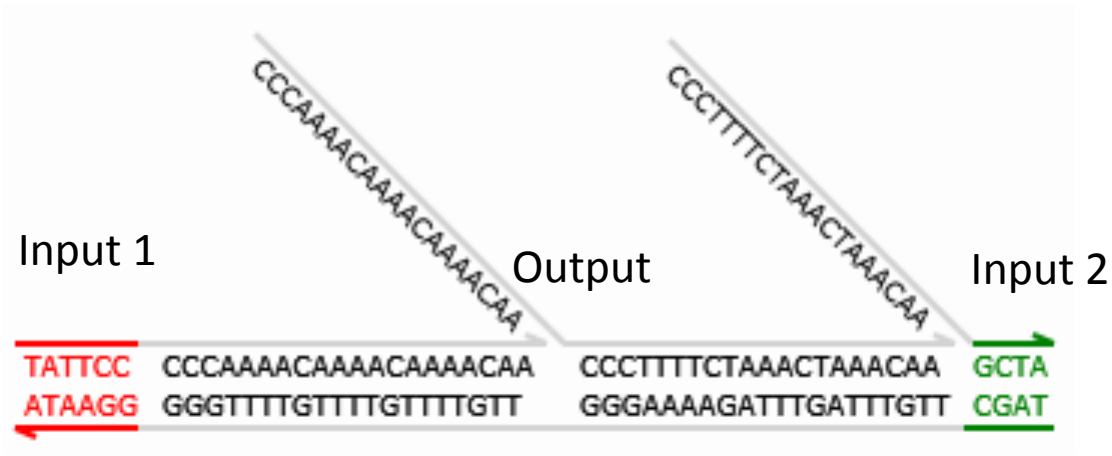
# DNA Strand Displacement

- Chemical reactions between DNA species
- Complementarity of short/long DNA domains
- Example: DSD Logic Gate [Output = Input1 AND Input2]



# DNA Strand Displacement

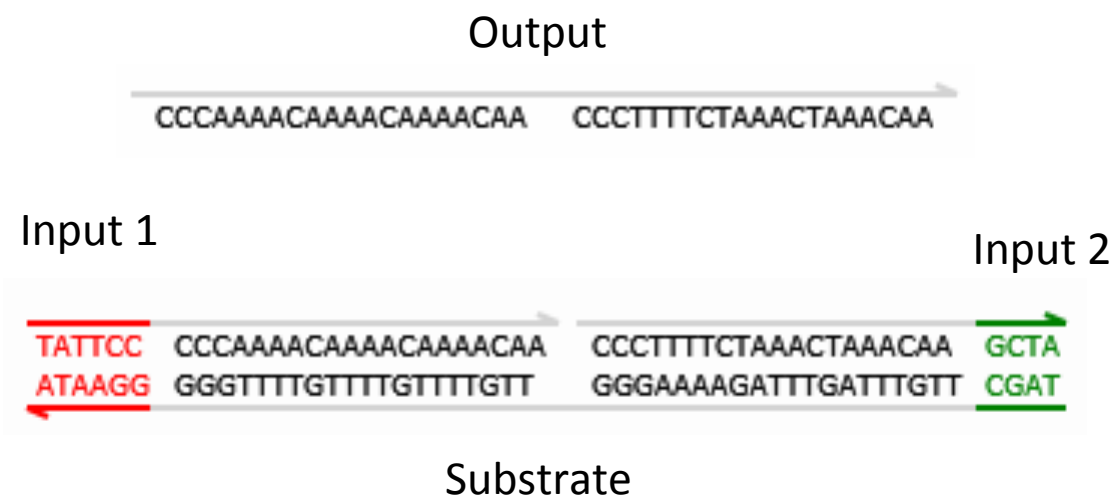
- Chemical reactions between DNA species
- Complementarity of short/long DNA domains
- Example: DSD Logic Gate [Output = Input1 AND Input2]



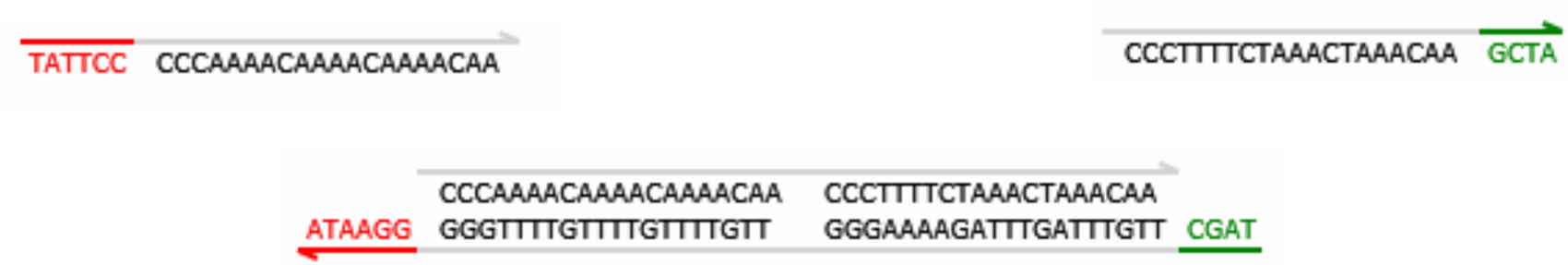

Substrate

# DNA Strand Displacement

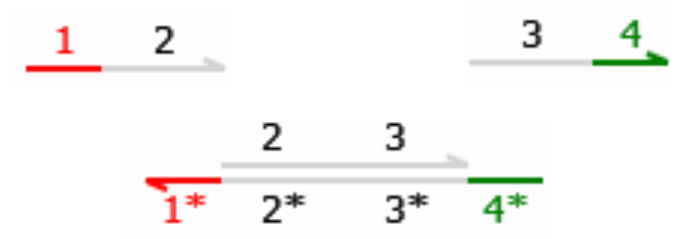
- Chemical reactions between DNA species
- Complementarity of short/long DNA domains
- Example: DSD Logic Gate [Output = Input1 AND Input2]



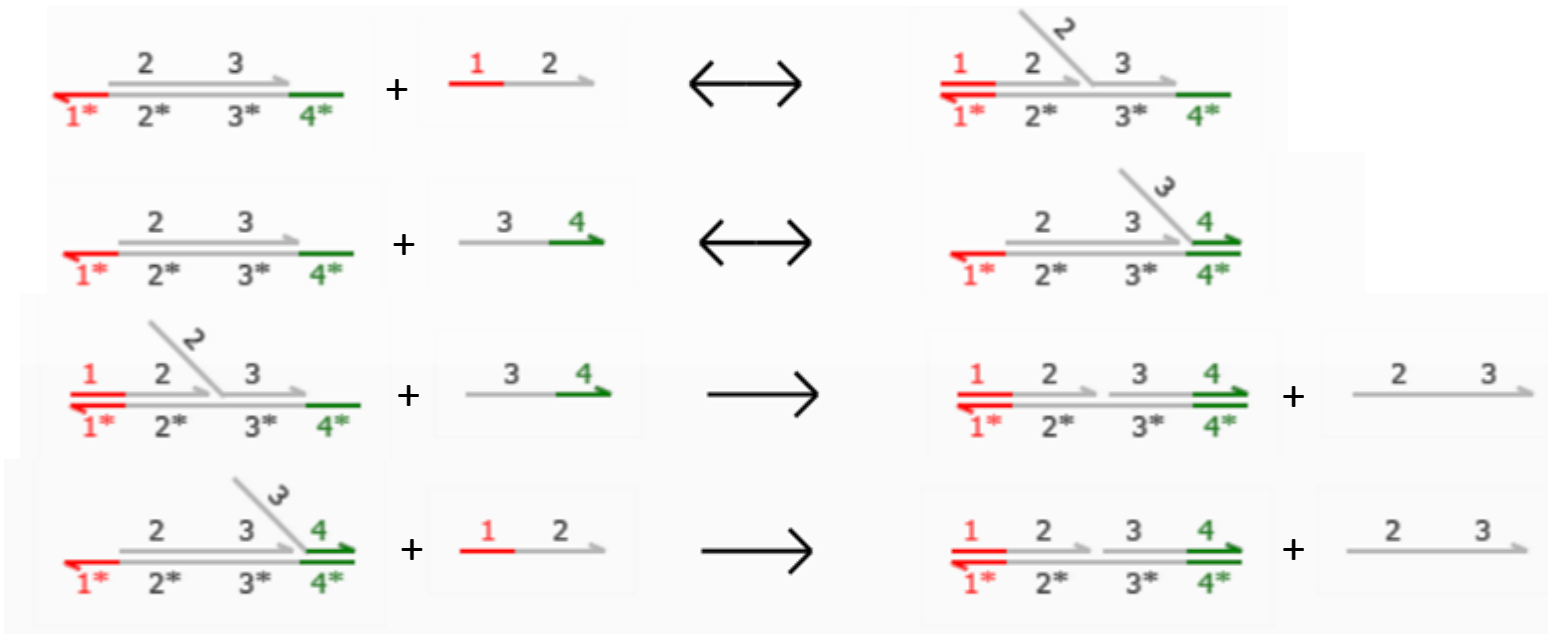
# Domain abstraction




1 --> (5') TATTCC (3')
4 --> (5') GCTA (3')
2 --> (5') CCCAAAACAAAACAAAACAA (3')
3 --> (5') CCCTTTTCTAAACTAAACAA (3')



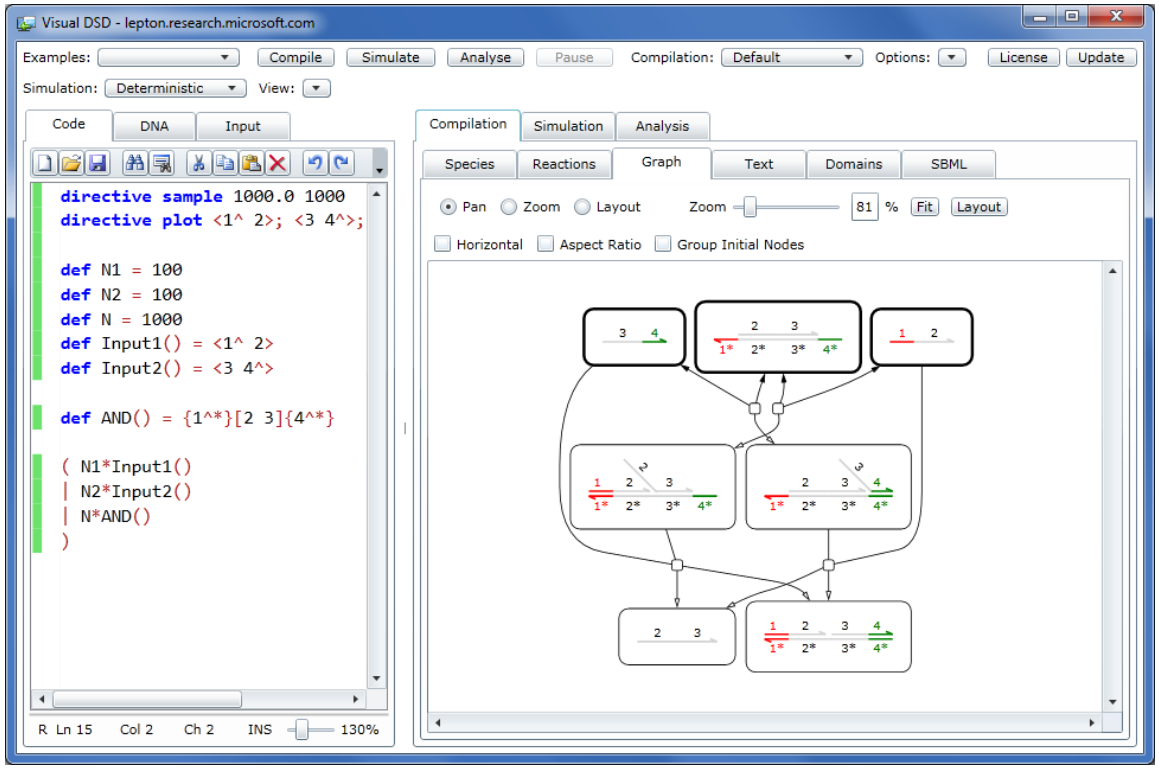
# DSD Reactions




*Currently, reaction rates are abstracted.*

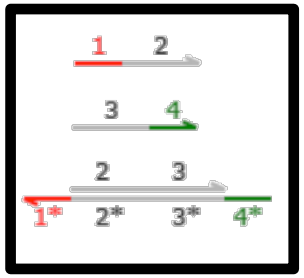
# Visual DSD

- Given a Visual DSD program, compute the set of all possible species and reactions

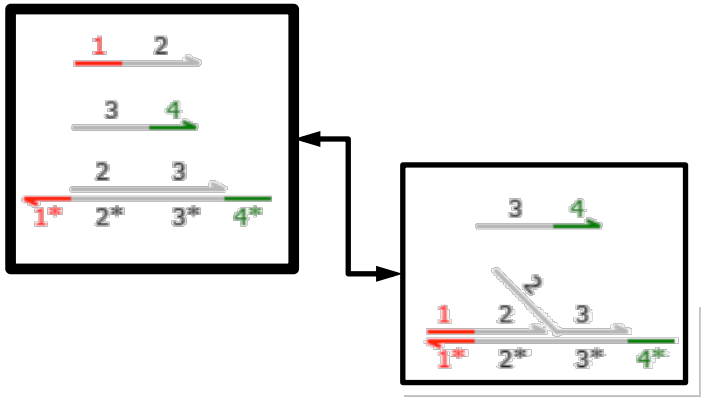



*Phillips & Cardelli, 2009*
*Lakin et al., 2011*

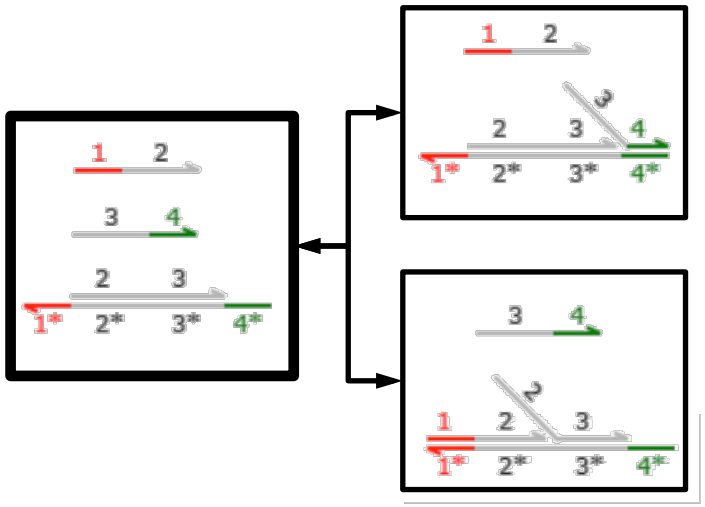
# DSD Transition System



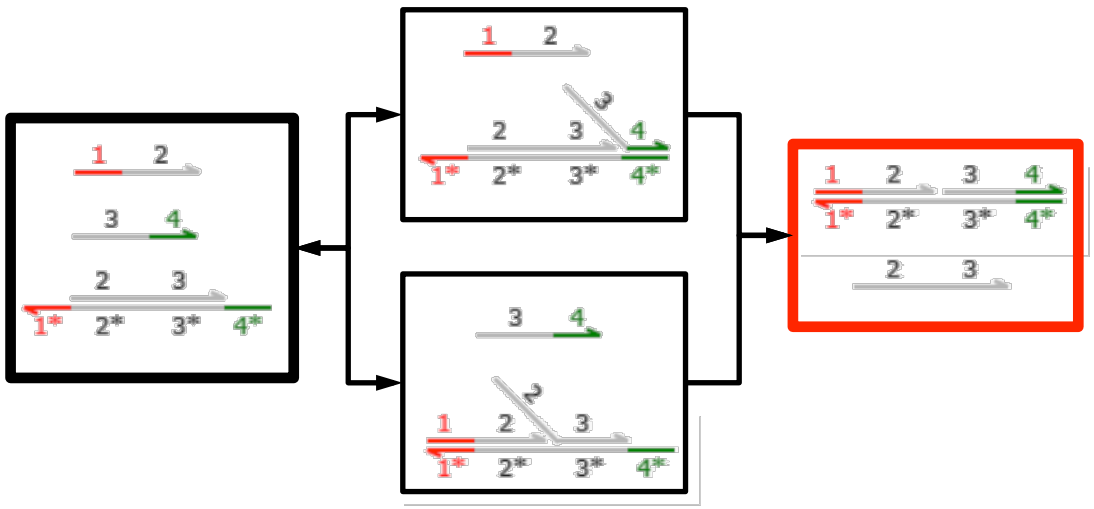
# DSD Transition System



# DSD Transition System



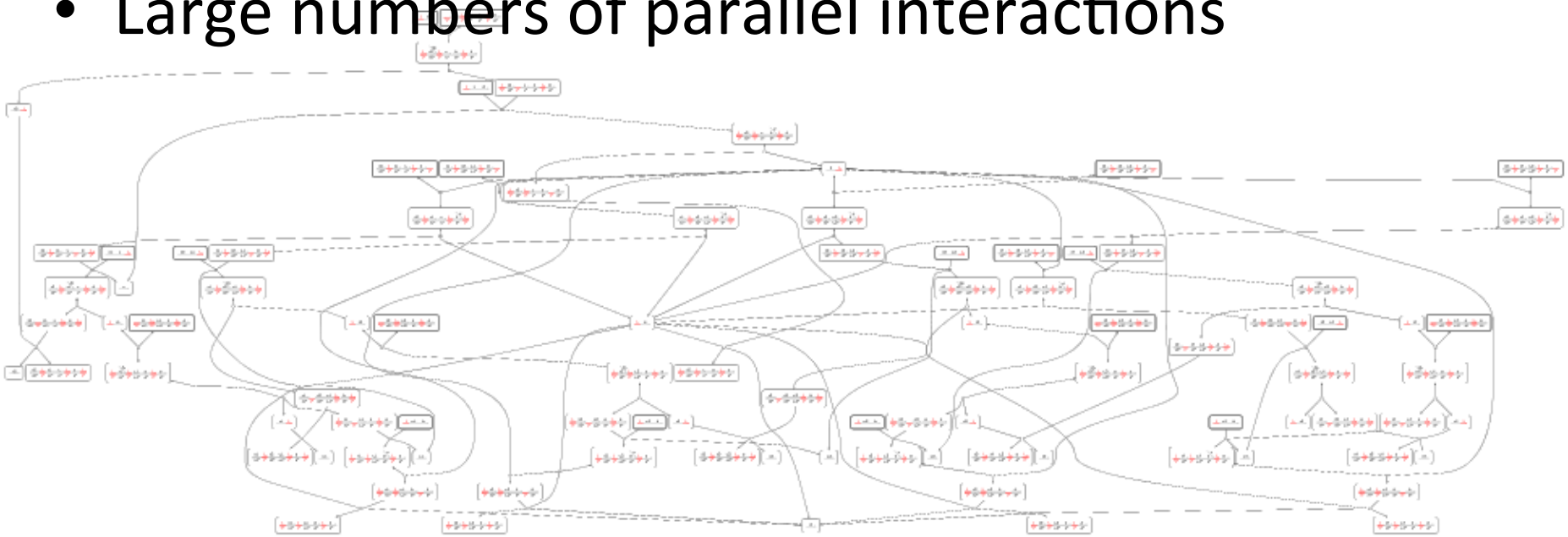
# DSD Transition System




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

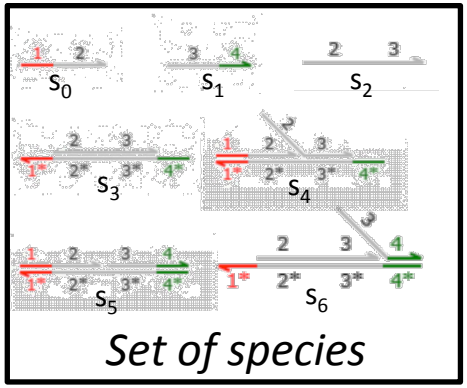
# Realistic Models

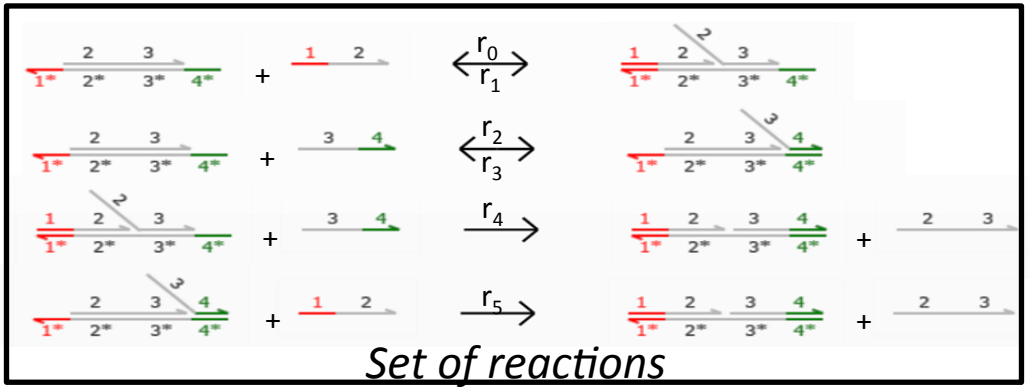
- Large numbers of molecules from each species
- Large numbers of parallel interactions



*Reaction graph for five DSD transducer circuits in series*

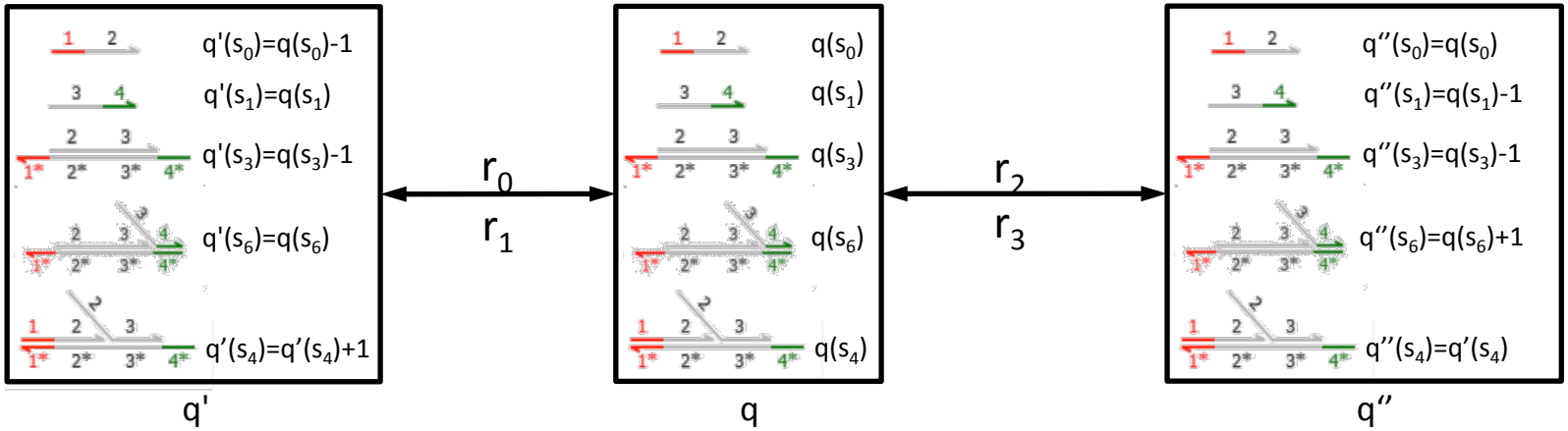
# SMT Encoding



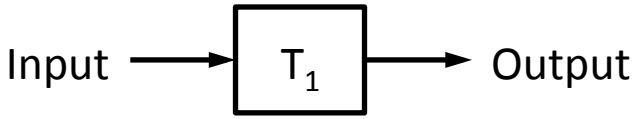



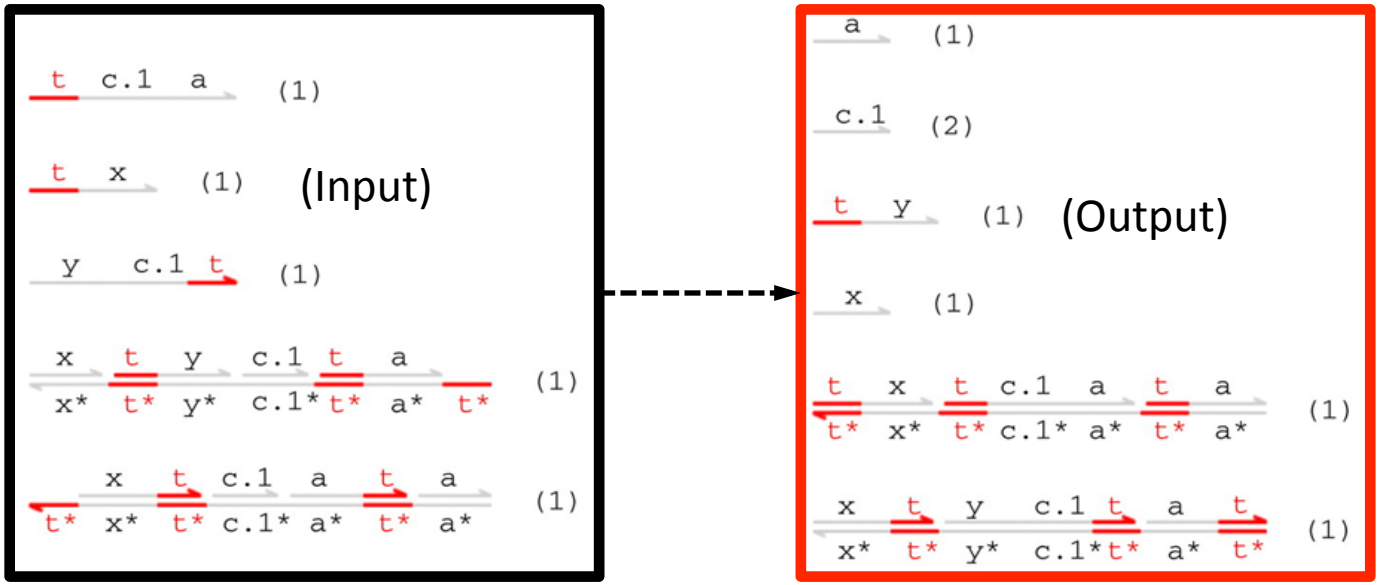

$$Q \subseteq \mathbb{N}^{|\mathcal{S}|} \quad \text{or} \quad Q \subseteq \mathbb{B}^{\hat{N}} \qquad enabled(r,q) \leftrightarrow \bigwedge_{s \in \mathcal{S}} q(s) \geq R_r(s)$$

$$T(q,q') \leftrightarrow \bigvee_{r \in \mathcal{R}} [enabled(r,q) \wedge \bigwedge_{s \in S} q'(s) = q(s) - R_r(s) + P_r(s)].$$



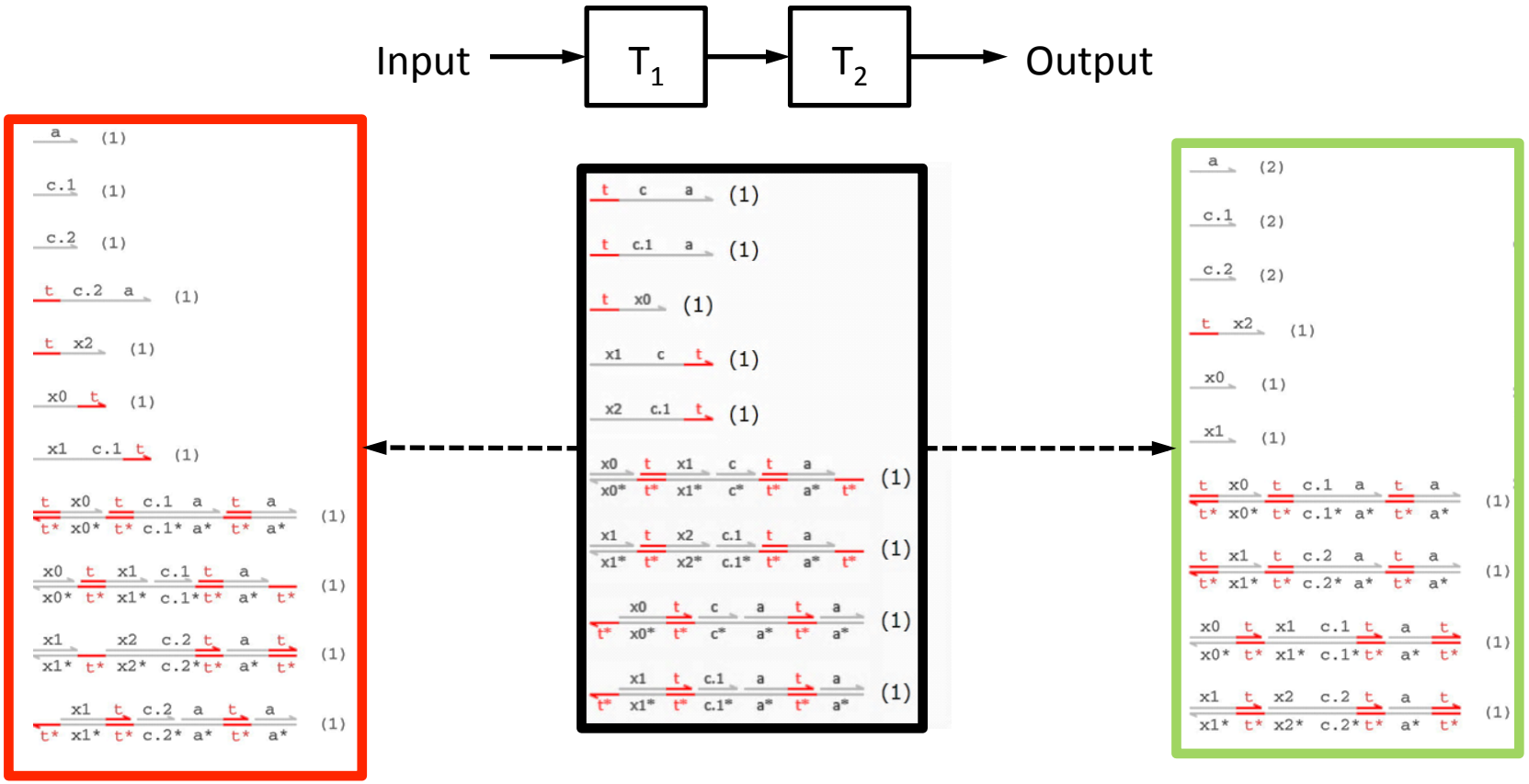
# DSD Transducer Circuits




*Lakin et al., 2011*

# Transducer Error States



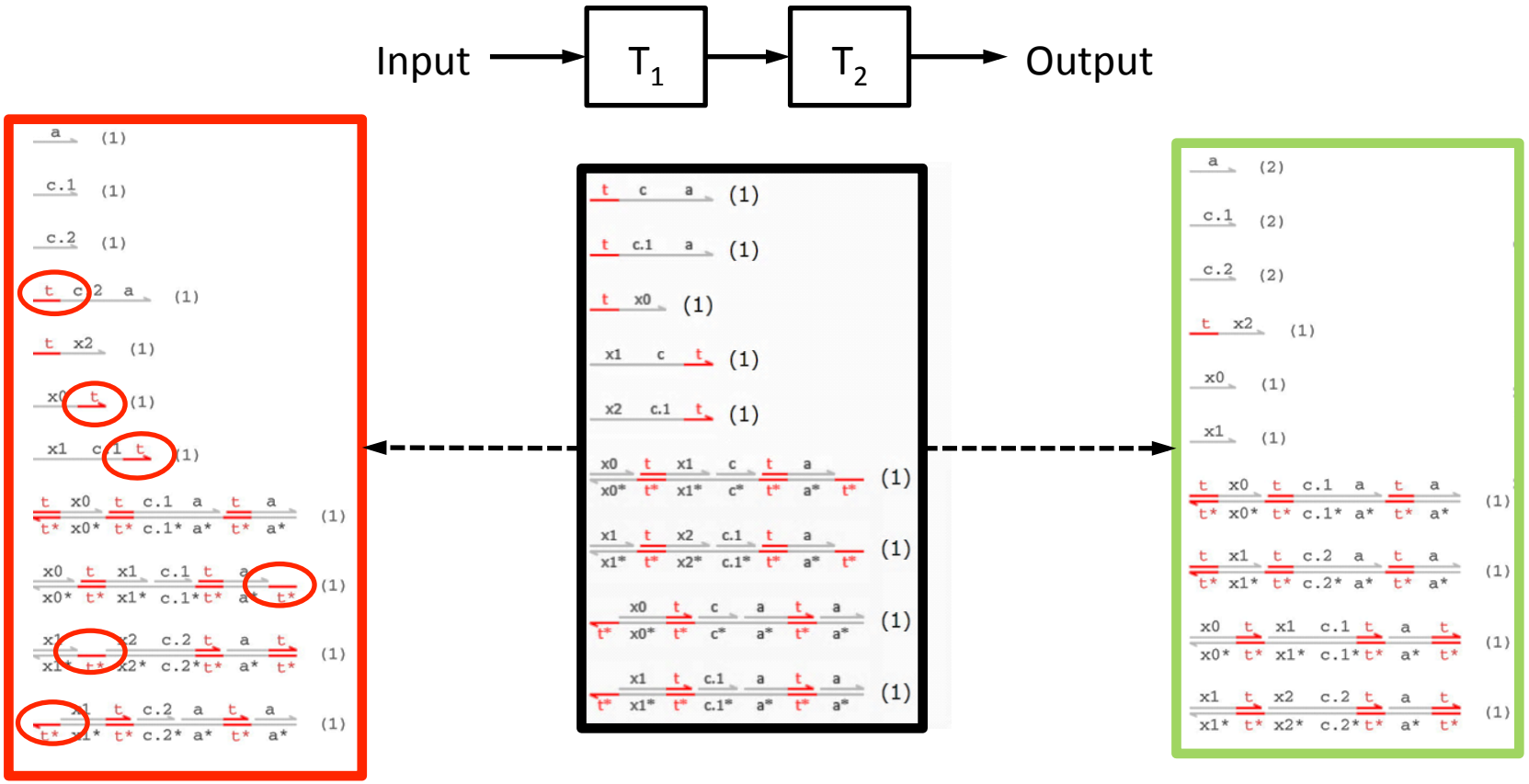

$$good(q) \leftrightarrow \bigwedge_{r \in \mathcal{R}} \neg enabled(r, q) \wedge \bigwedge_{s \in \mathcal{S}_r} s \notin q$$

$$bad(q) \leftrightarrow \bigwedge_{r \in \mathcal{R}} \neg enabled(r, q) \wedge \bigvee_{s \in \mathcal{S}_r} s \in q.$$

$$AG(\neg bad) \wedge EF(good).$$

*Lakin et al., 2011*

# Transducer Error States



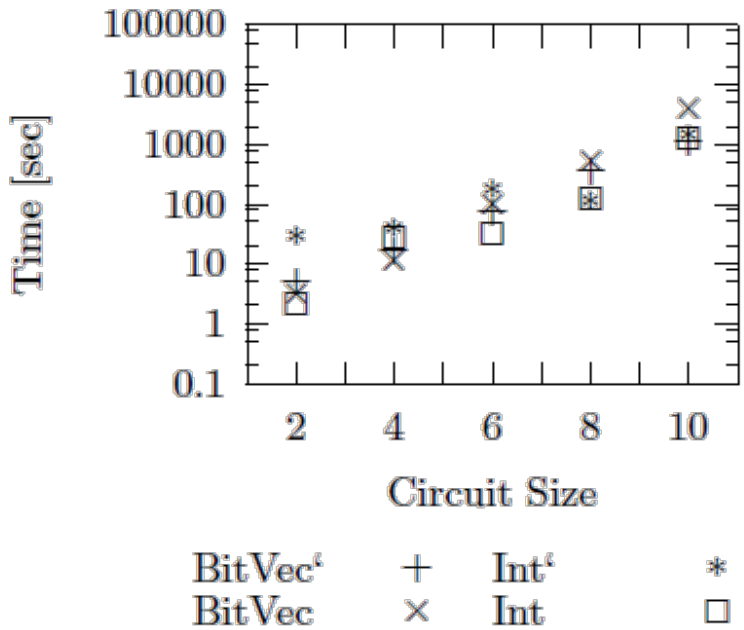

$$good(q) \leftrightarrow \bigwedge_{r \in \mathcal{R}} \neg enabled(r, q) \wedge \bigwedge_{s \in \mathcal{S}_r} s \notin q$$

$$bad(q) \leftrightarrow \bigwedge_{r \in \mathcal{R}} \neg enabled(r, q) \wedge \bigvee_{s \in \mathcal{S}_r} s \in q.$$

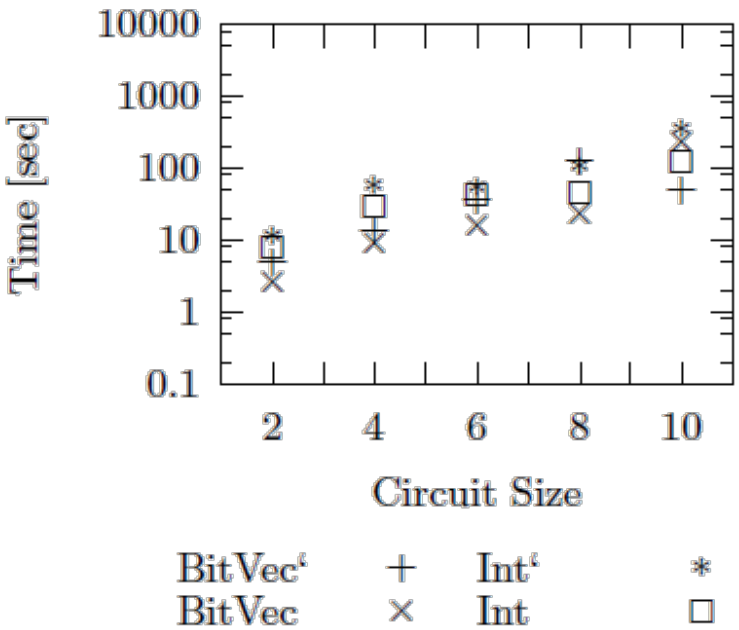
$$AG(\neg bad) \wedge EF(good).$$

*Lakin et al., 2011*

# Flawed Transducer Design

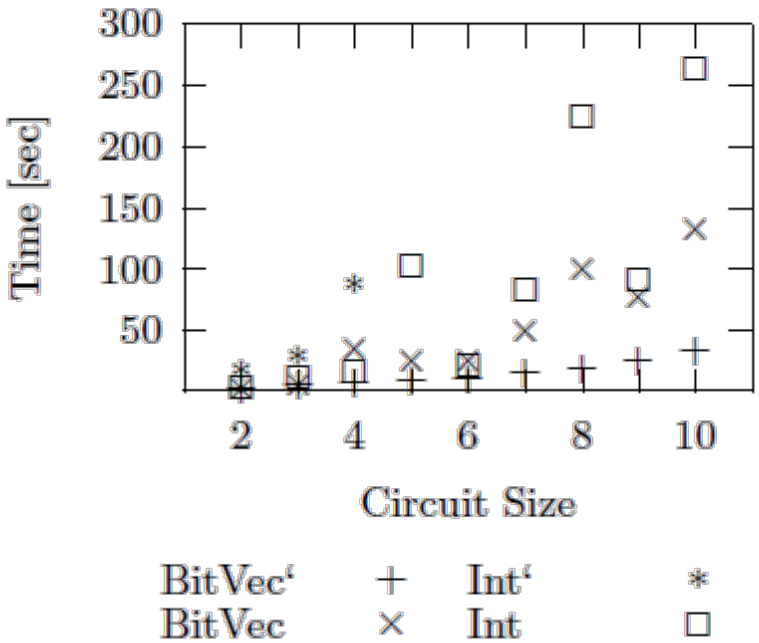



Identification of computation traces reaching a “good” state in up to 100 steps.



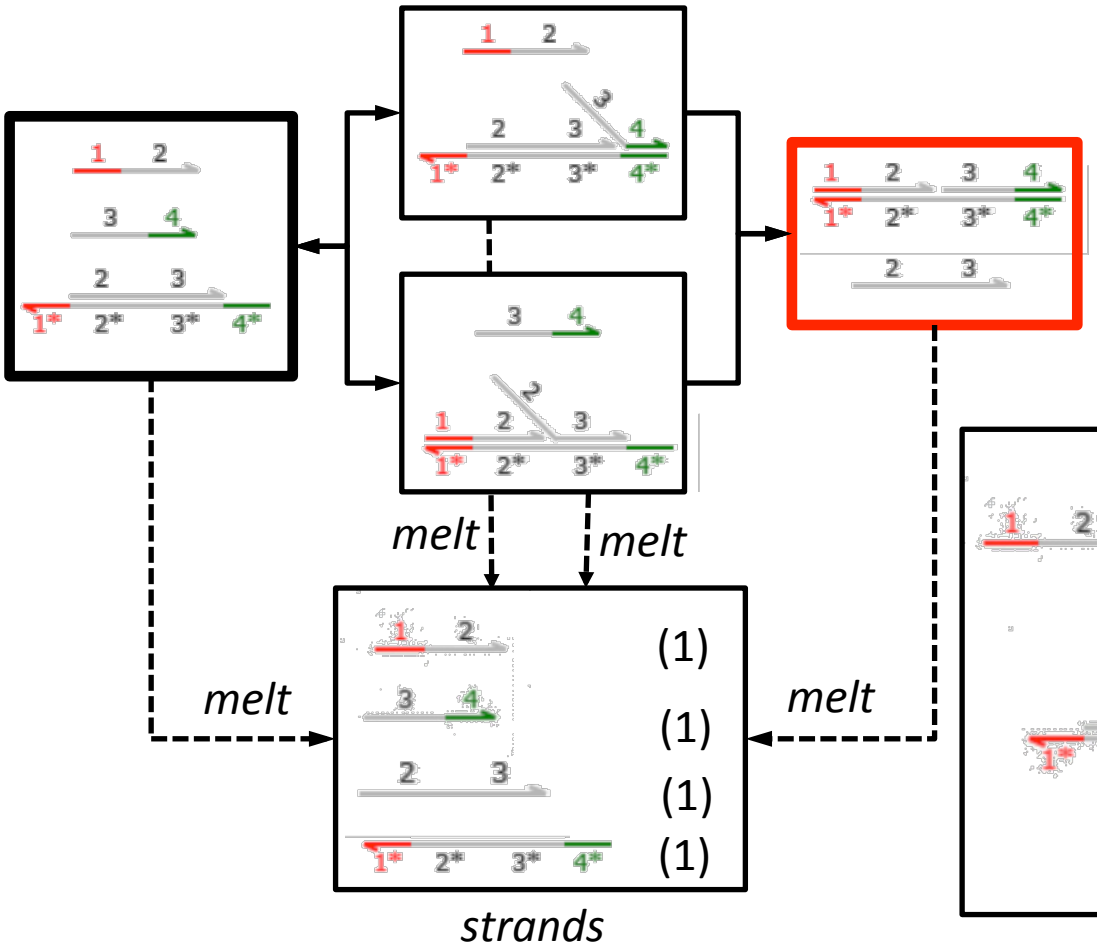

Identification of computation traces reaching a “bad” state in up to 100 steps.

# Corrected Transducer Design

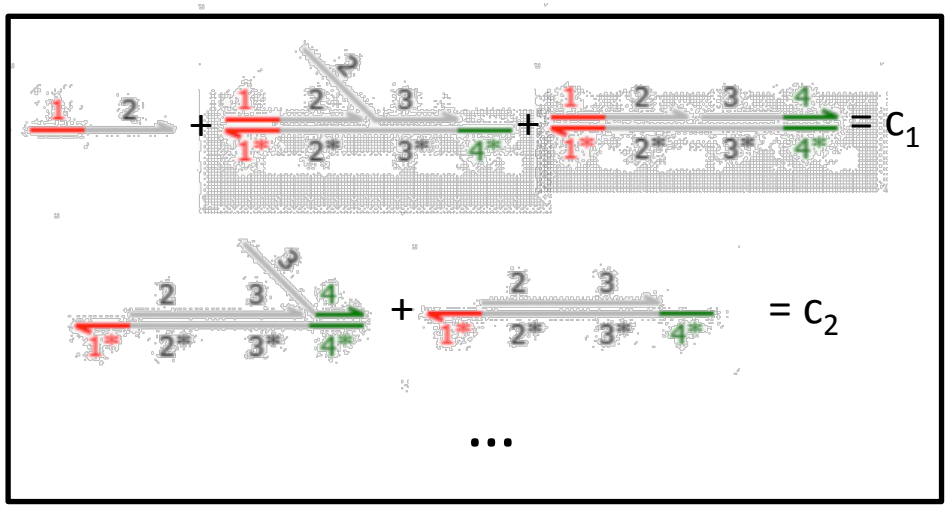



Identification of computation traces reaching a “good” state in up to 100 steps.

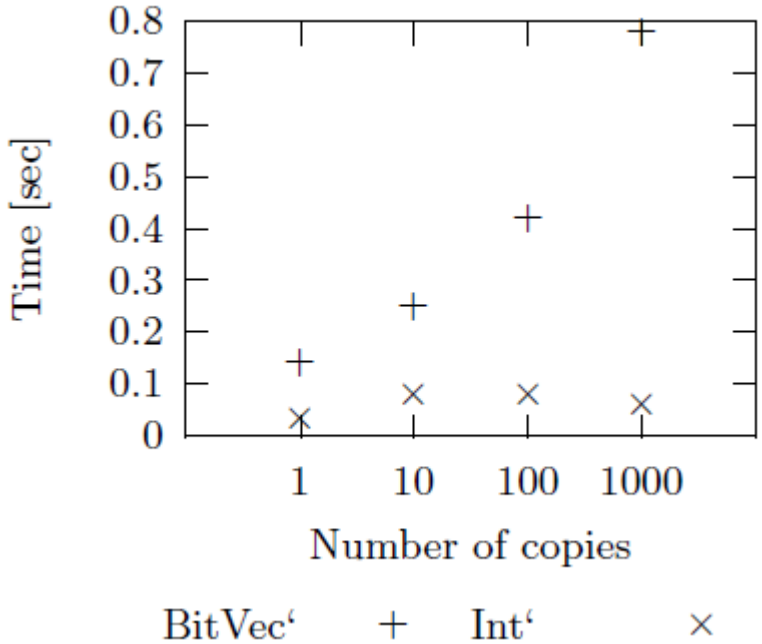
# Conservation of Strands




$$melt(q) \triangleq \biguplus_{s \in \mathcal{S}} q(s)\, melt(s)$$



# Corrected Transducer Design




Verification of multiple copies of a circuit with 10 corrected transducers in series.

# Square Root Circuit Analysis

(Recent progress beyond NFM'13 proceedings)

- Allowed multi-reaction steps
- Strengthened the strand conservation constraints
- Encoded functional properties
- Analysed one of the most complex designs constructed experimentally with millions of copies of the circuit operating in parallel

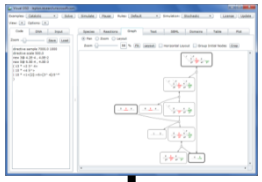
*Yordanov, Wintersteiger, Hamadi, Phillips, Kugler, 2013*

# Summary

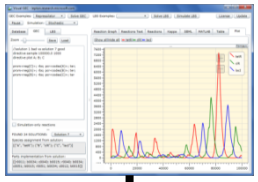
- Analysis methods complementary to simulation can help in understanding and programming biological systems
- SMT-based methods enable an expressive, scalable and extensible analysis framework
- Challenges
  - Biological complexity, parallel interactions, nondeterminism
  - Few realistic benchmarks are available

# Exposing Biology to the Formal Methods Community

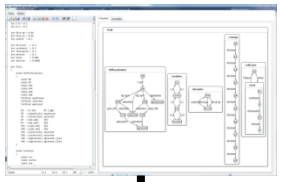
DSD



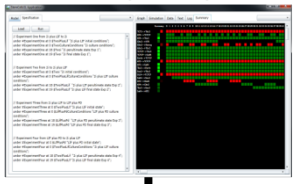
GEC



Biocharts



Varna



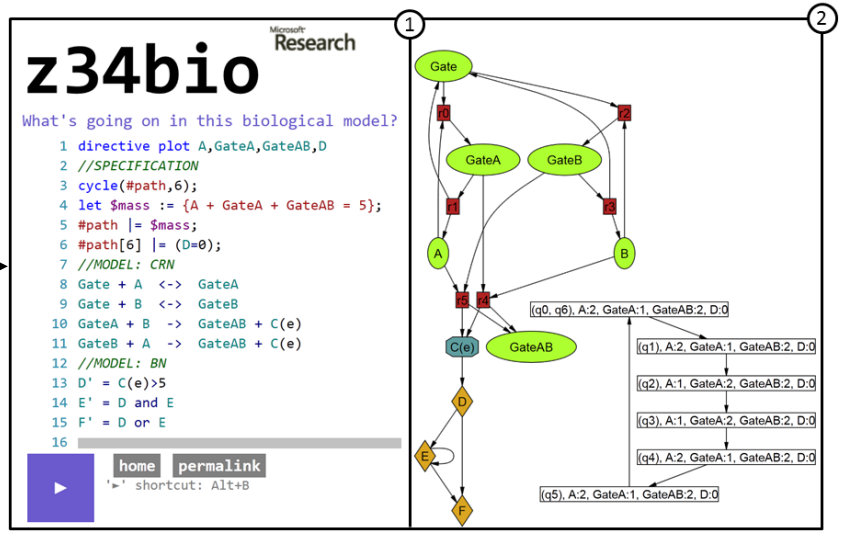
...

Biological Modelling Engine

Z3-4Bio

*SMTLIB*

Z3 ...




http://rise4fun.com/z34biology

# Acknowledgements

**Christoph M. Wintersteiger**
Programming Principles and Tools, Microsoft Research

**Youssef Hamadi**
Constraint Reasoning Group, Microsoft Research

**Hillel Kugler**
Biological Computation Group, Microsoft Research

**Andrew Phillips**
Biological Computation Group, Microsoft Research

**Sara-Jane Dunn**
Biological Computation Group, Microsoft Research

**Graziano Martello**
Wellcome Trust Centre for Stem Cell Research, University of Cambridge