Automatic Compilation from High-Level Bio-Languages to Genetic Regulatory Networks

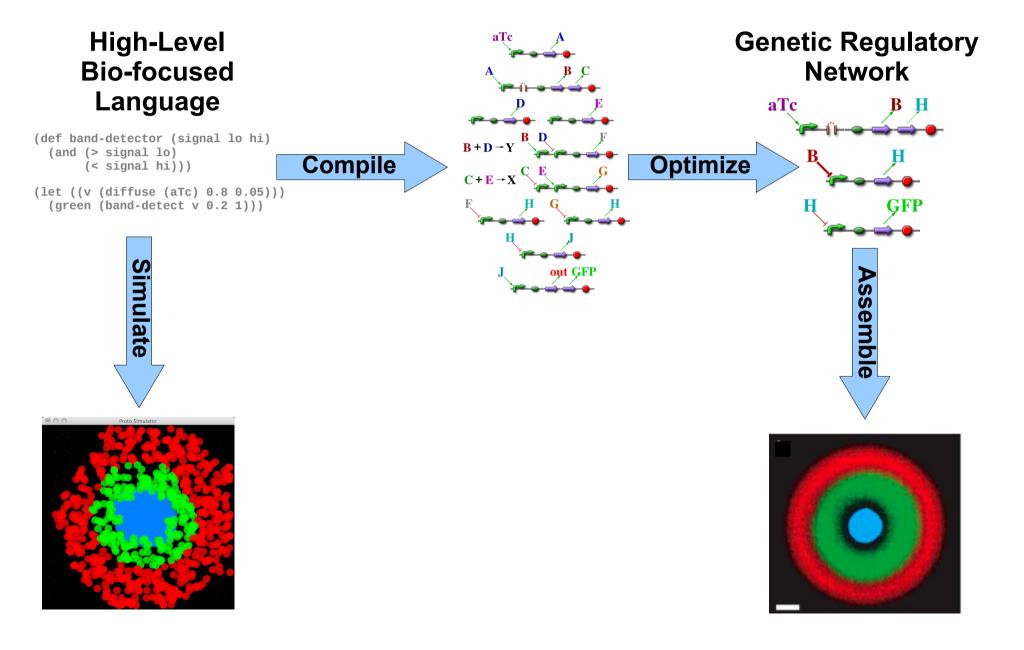
Jacob Beal, Ting Lu, Ron Weiss

IWBDA, June 2010



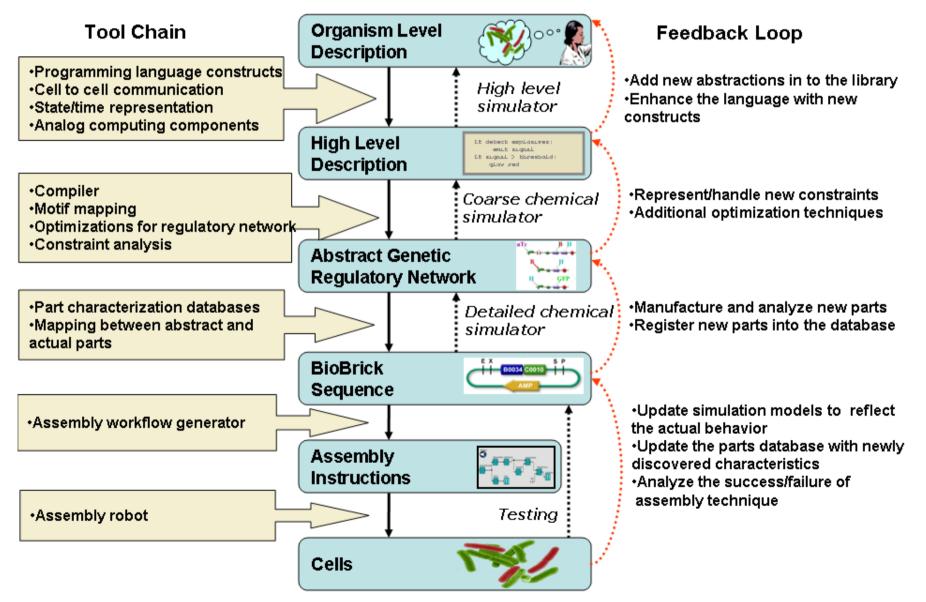


Goal: High-Level Biological Design



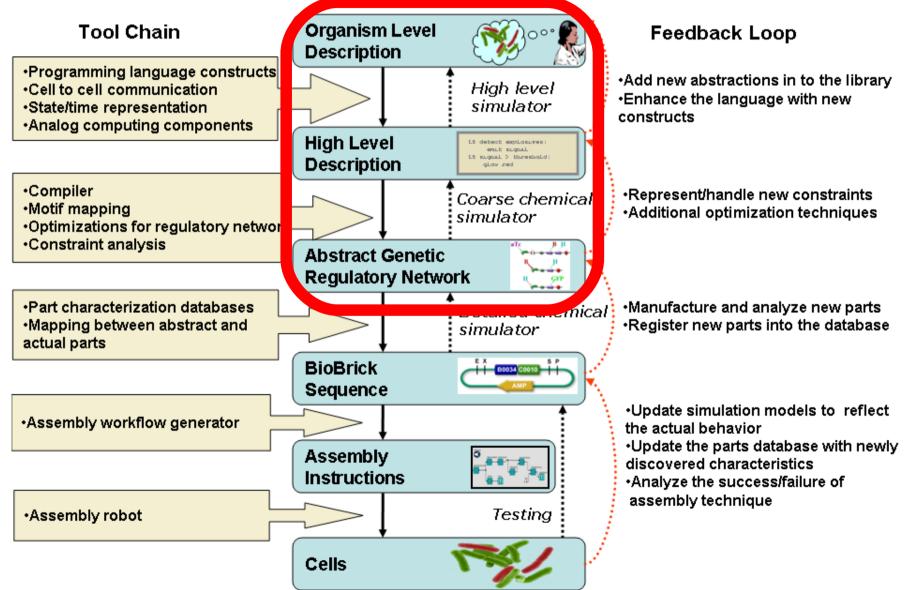
Tool Chain Vision

Stages for Engineering Cells



Tool Chain Vision

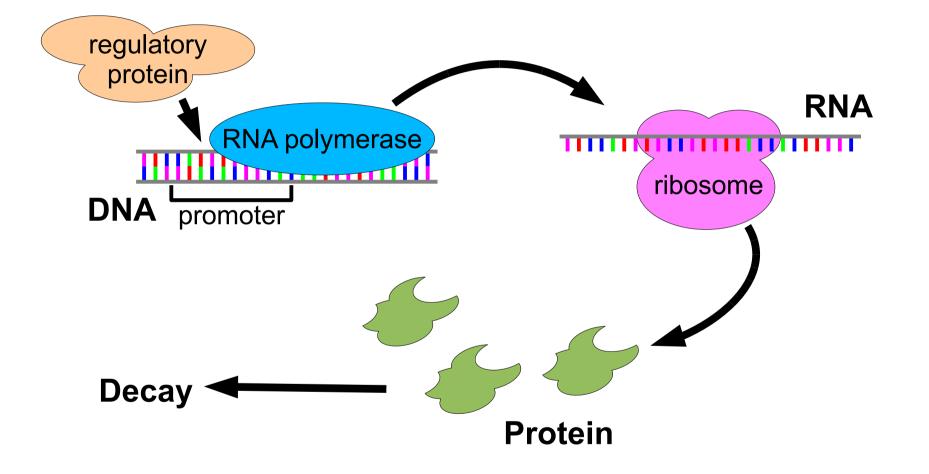
Stages for Engineering Cells



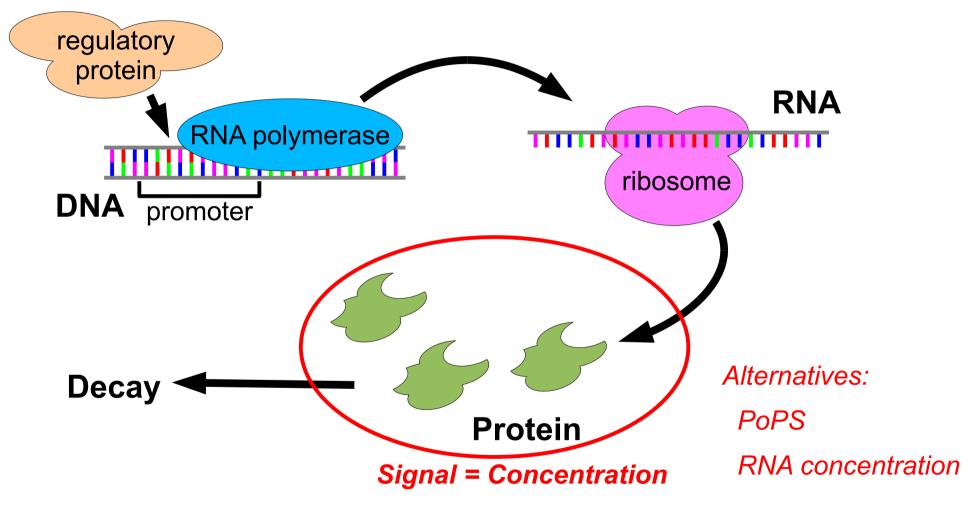
Outline

- Compositional Design
- Motif-Based Compilation
- Simulation Results

Computation via Transcription Network

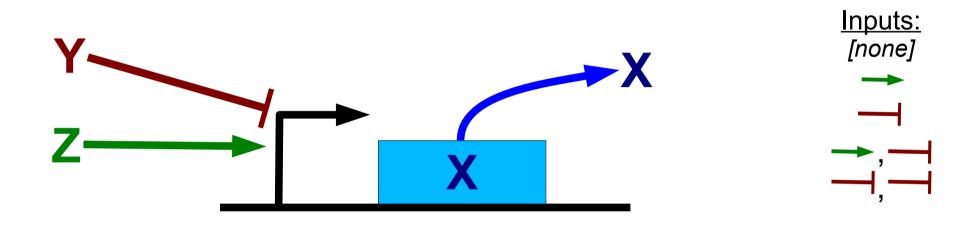


Computation via Transcription Network

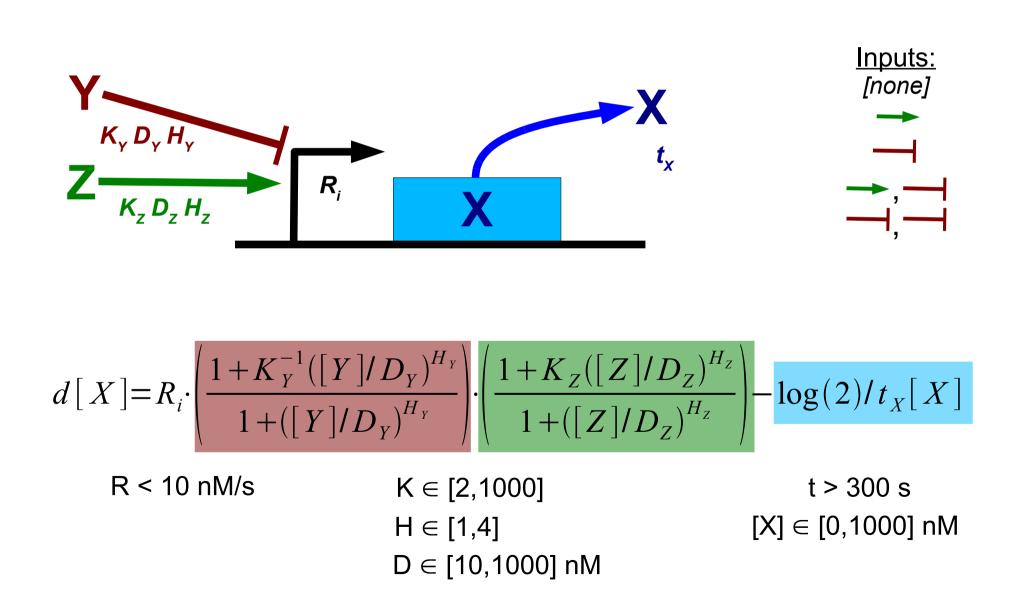


Stablizes at *decay* = *production*

Abstract GRN Design Space

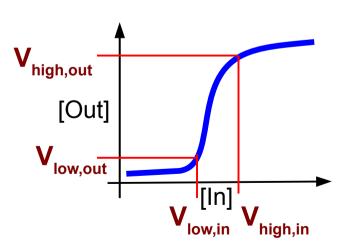


Abstract GRN Design Space



Major Challenge: Interference

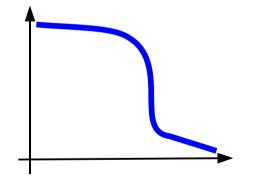
- Effective part characterics changed by:
 - Cellular context (endogenous pathways, synthetic parts)
 - Expression noise
- Our approach: noise-rejection
 - Digital static discipline: V_{low,out} < V_{low,in} < V_{high,in} < V_{high,out}



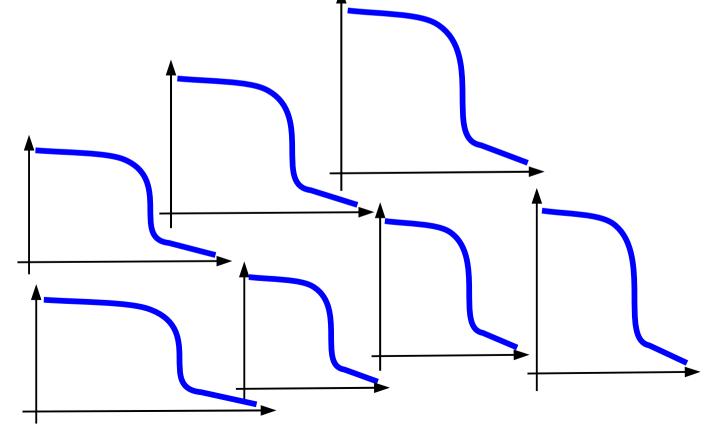
But part variance makes a uniform standard impossible!

- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions

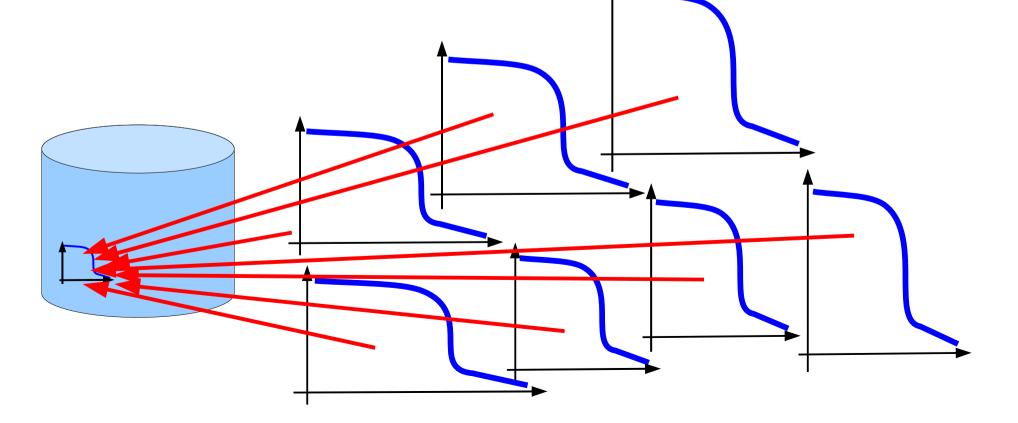
- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



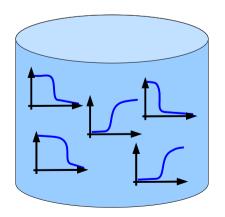
- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



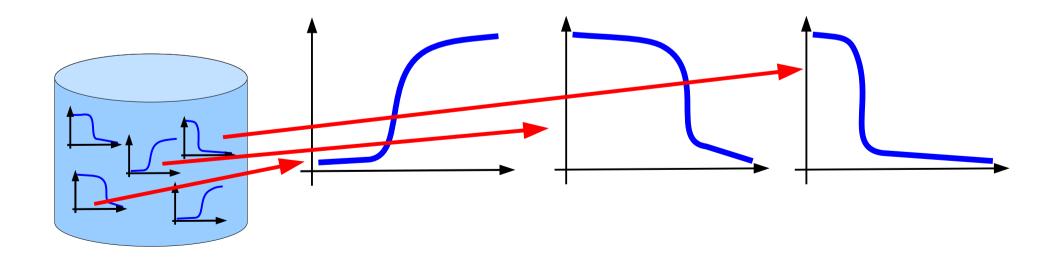
- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



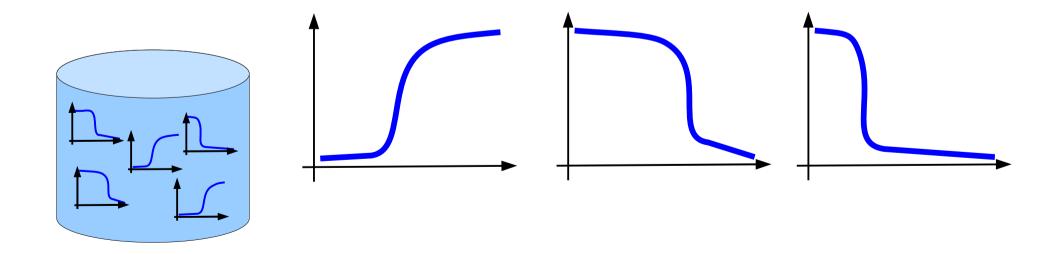
- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



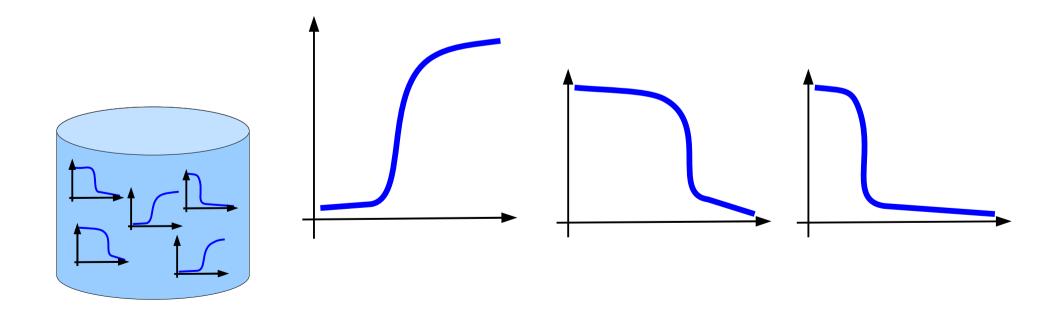
- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



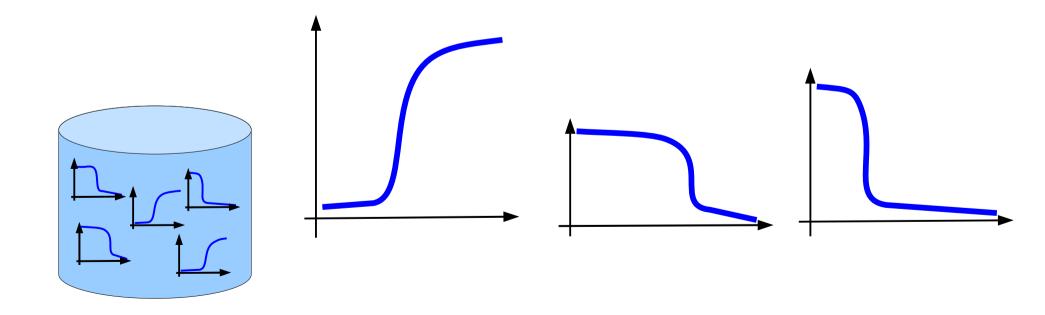
- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



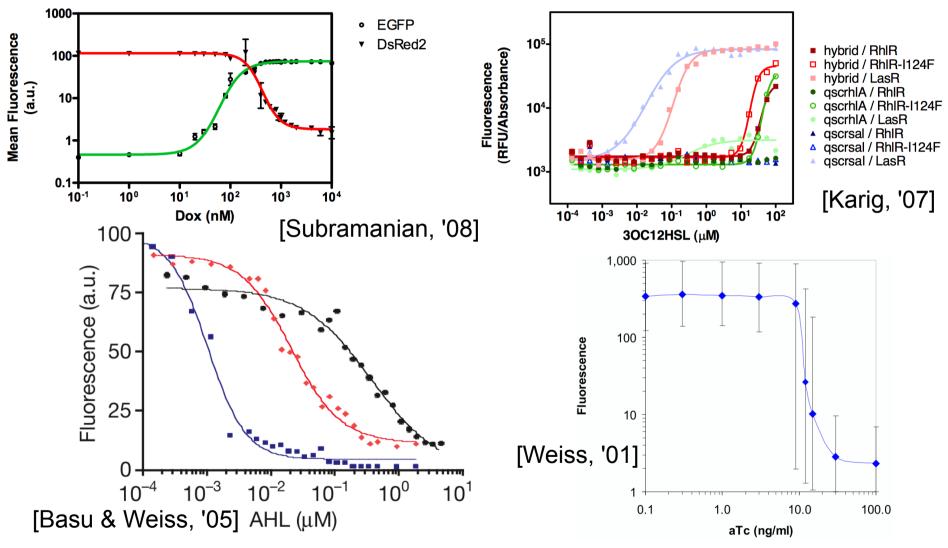
- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



- Identify "standards family" parameter relation
- Create library of characterized part variants
- Adjust part choice to match on junctions



Experimental Input to Family Relation

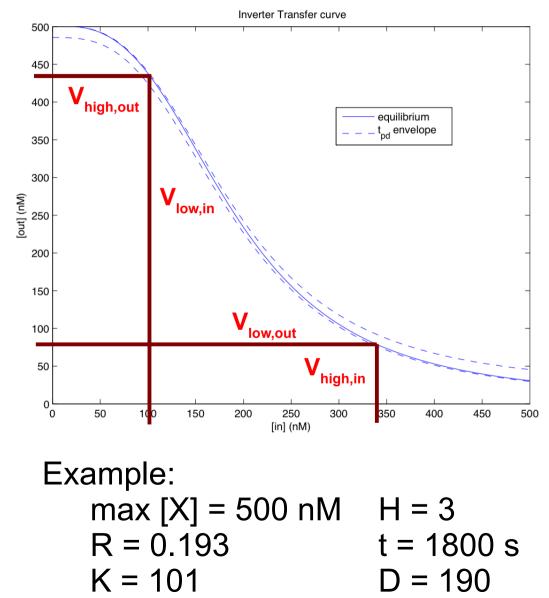


Model constrained by characterization experiments...

Draft Simulation-Based Standards Family Generation

Need: K,H,D,R,t,max [X]

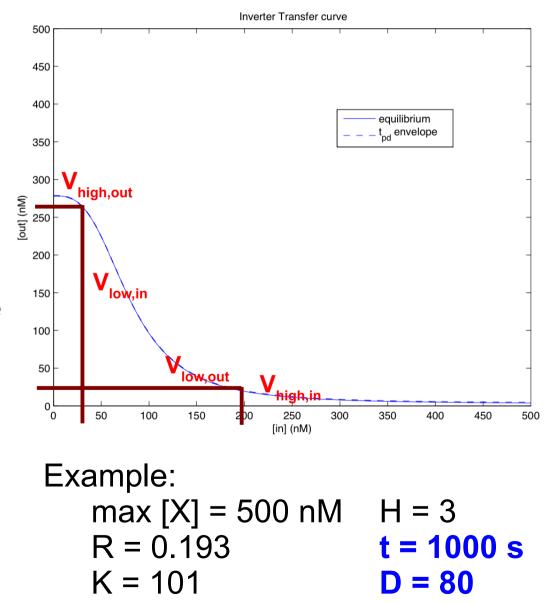
- Choose max [X]
- Assuming large K, amplification of 0.5K when [X]/D ≈ 1
- H → D, static discipline (higher H is better)
- Steady state max: [X]=production*t/log(2)
- production ≈ K*R



Draft Simulation-Based Standards Family Generation

Need: K,H,D,R,t,max [X]

- Choose max [X]
- Assuming large K, amplification of 0.5K when [X]/D ≈ 1
- H → D, static discipline (higher H is better)
- Steady state max: [X]=production*t/log(2)
- production ≈ K*R

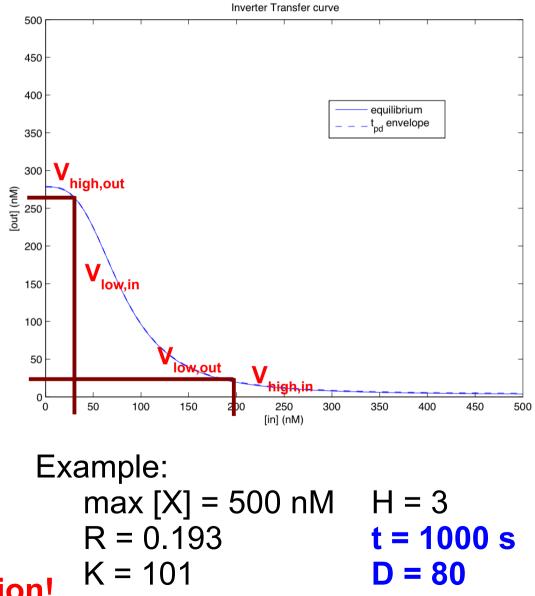


Draft Simulation-Based Standards Family Generation

Need: K,H,D,R,t,max [X]

- Choose max [X]
- Assuming large K, amplification of 0.5K when [X]/D ≈ 1
- H → D, static discipline (higher H is better)
- Steady state max: [X]=production*t/log(2)
- production ≈ K*R

Driver for experimental part creation & characterization!

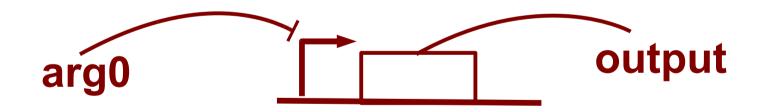


Outline

- Compositional Design
- Motif-Based Compilation
- Simulation Results

- High-level primitives map to GRN design motifs
 - e.g. logical operators:

```
(primitive not (boolean) boolean
  :bb-template ((P 0.193 R- arg0 RBS outputs T)))
```

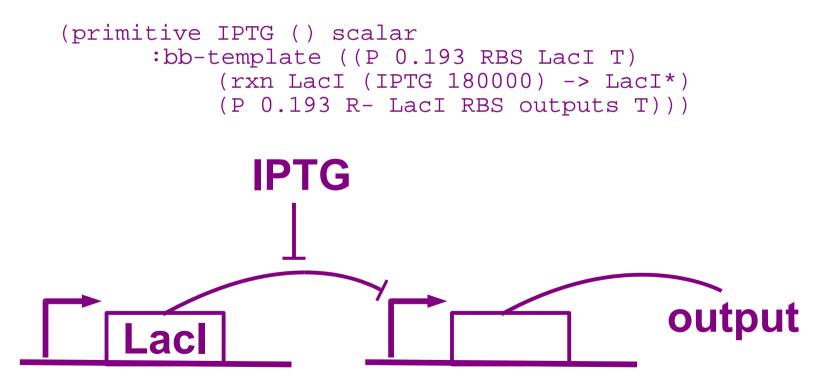


- High-level primitives map to GRN design motifs
 - e.g. logical operators, actuators:

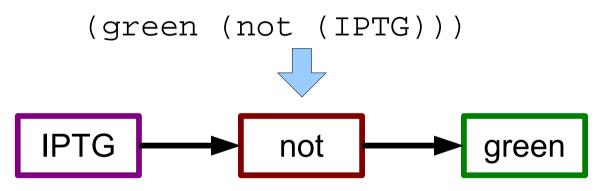
(primitive green (scalar) scalar :side-effect
 :bb-template ((P R+ arg0 RBS GFP outputs T)))



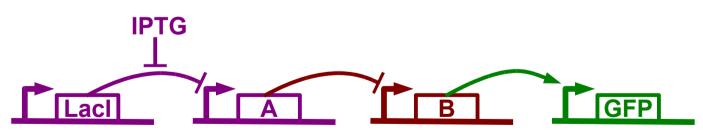
- High-level primitives map to GRN design motifs
 - e.g. logical operators, actuators, sensors:



• Functional program gives dataflow computation:



• Operators translated to motifs:

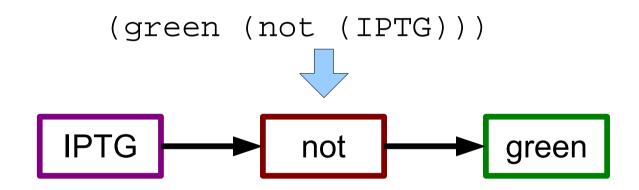


- Standards family sets chemical constants
- Optimizers simplify network

• Functional program gives dataflow computation:

(green (not (IPTG)))

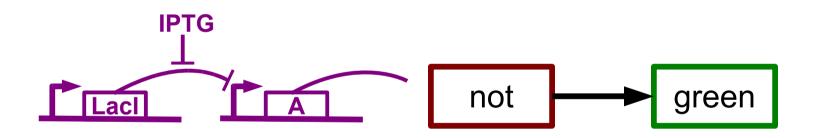
• Functional program gives dataflow computation:



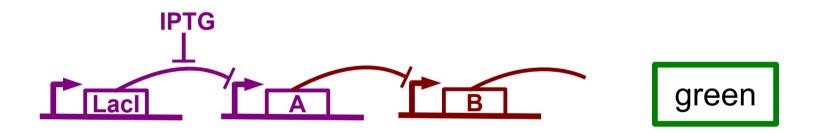
- Operators translated to motifs:
 - Standards family sets chemical constants



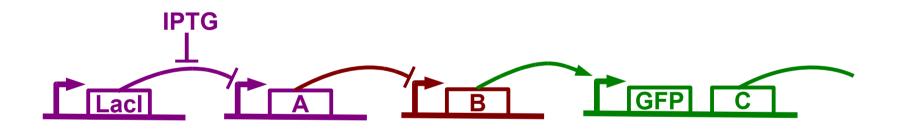
- Operators translated to motifs:
 - Standards family sets chemical constants



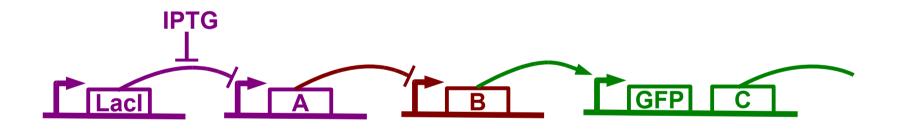
- Operators translated to motifs:
 - Standards family sets chemical constants



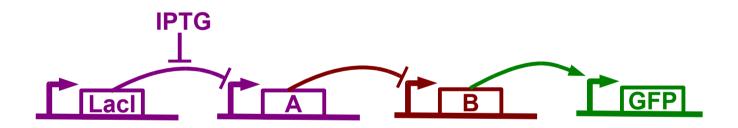
- Operators translated to motifs:
 - Standards family sets chemical constants



• Optimizers simplify network



• Optimizers simplify network



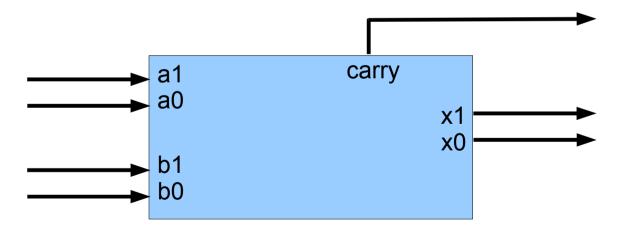
• Optimizers simplify network



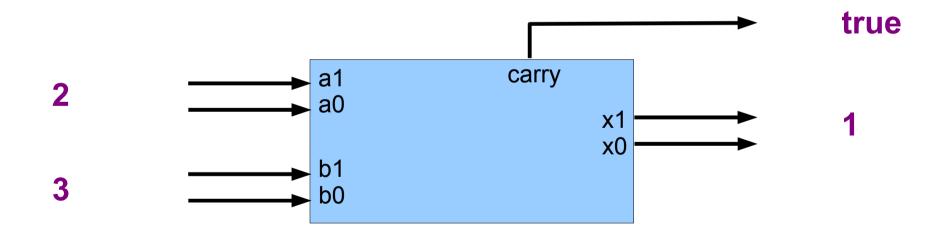
Outline

- Compositional Design
- Motif-Based Compilation
- Simulation Results

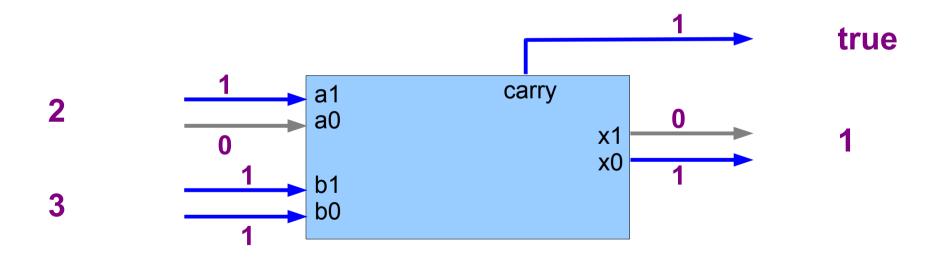
- Prototype compiler generates GRNs that simulate correctly for a limited language subset
- Example: 2-bit adder



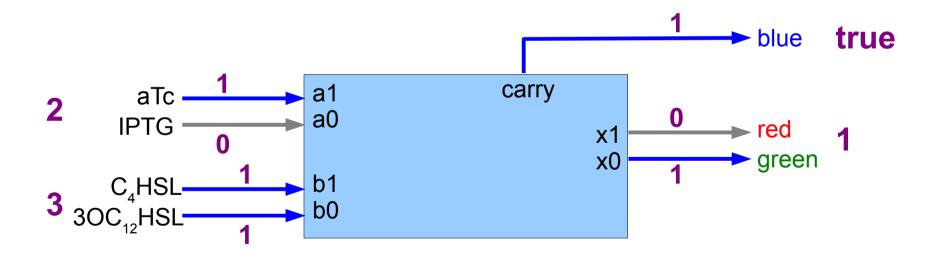
- Prototype compiler generates GRNs that simulate correctly for a limited language subset
- Example: 2-bit adder



- Prototype compiler generates GRNs that simulate correctly for a limited language subset
- Example: 2-bit adder



- Prototype compiler generates GRNs that simulate correctly for a limited language subset
- Example: 2-bit adder



- Prototype compiler generates GRNs that simulate correctly for a limited language subset
- Example: 2-bit adder

```
(macro xor (a b)
 (muxor (muxand ,a (not ,b))
 (muxand ,b (not ,a))))
(macro 2bit-adder (al a0 bl b0)
 (all
 (green (xor ,a0 ,b0)) ; x_0 low bit
 (let ((c0 (muxand ,a0 ,b0))
 (x1 (xor ,a1 ,b1)))
 (red (xor x1 c0)) ; x_1 high bit
 (blue (muxor (muxand x1 c0) ; carry bit
 (muxand ,a1 ,b1))))))
```

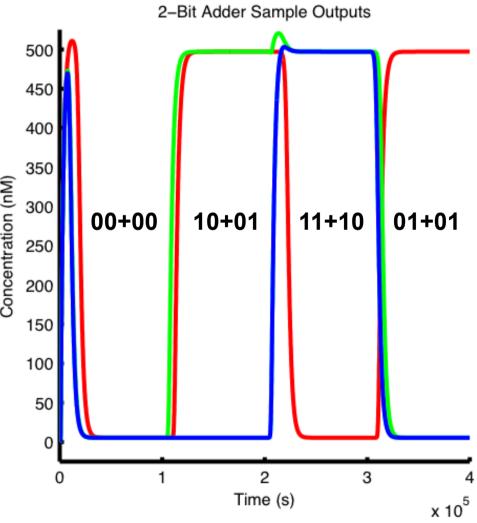
(**2bit-adder** (**aTc**) (**IPTG**) (C4HSL) (30C12HSL))

Bit 0

Bit 1

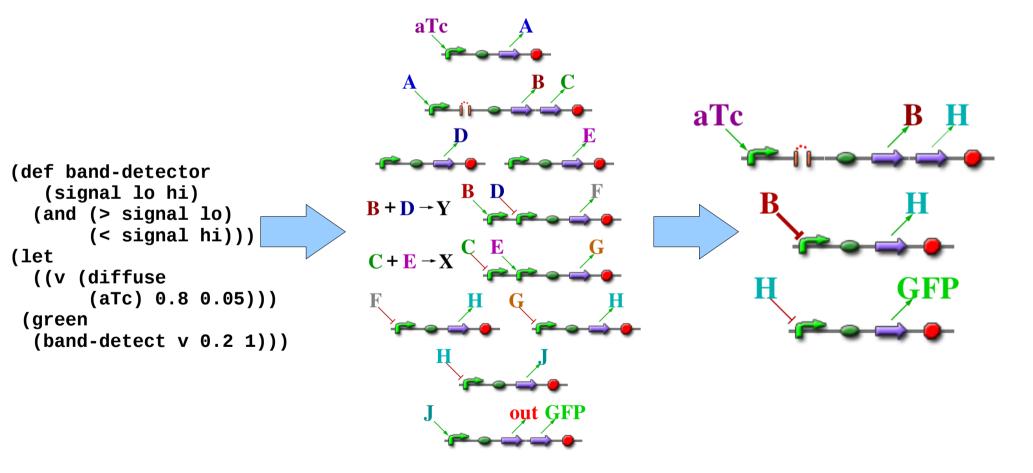
Carry

- Compiled 2-bit adder (unoptimized)
 - 60 signal chemicals
- 52 regulatory regions Generated ODE simulation in MATLAB Generated ODE



On to optimization...

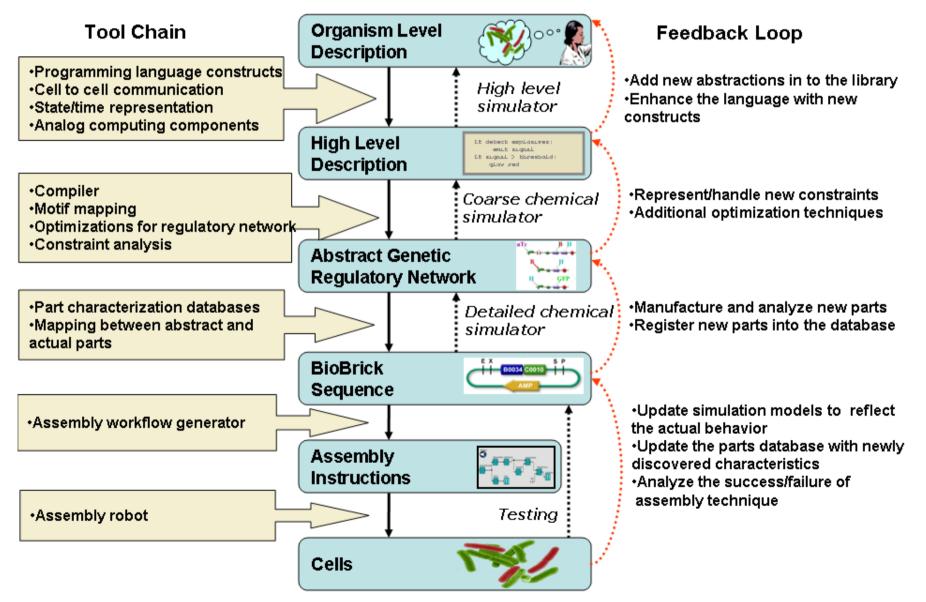
• Adapted classical techniques can be powerful:



Band detector optimization from [Beal & Bachrach, '08]

Tool Chain Vision

Stages for Engineering Cells



Contributions

- Parameterized standards identify all chemical parameters that can produce digital logic in transcriptional networks.
- Prototype motif-based compiler automatically maps high-level programs into GRNs.
- Automatically generated MATLAB ODE simulations verify that GRNs implement program specification.