

Spatial Computing: From Manifold Geometry to Networking and Biology

Jacob Beal

February, 2012

Work partially sponsored by DARPA; the views and conclusions contained in this document are those of the authors and not DARPA or the U.S. Government.

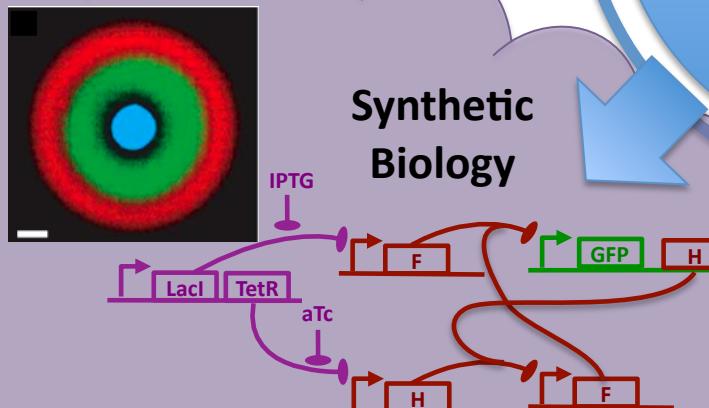
**Raytheon
BBN Technologies**



Distributed Power Demand Response

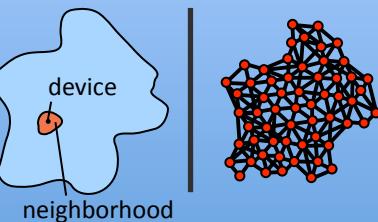
How can millions of appliances coordinate to change how we use energy?

How can the parts of a design work together to adapt it to new uses?



Synthetic Biology

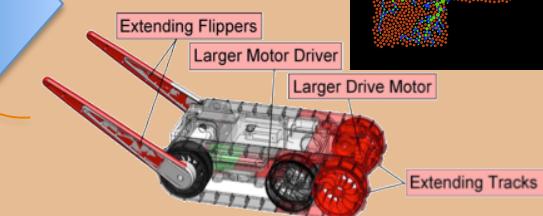
How do you program space?



Spatial Computing

How do you program the behavior of 10^{12} cells?

Morphogenetic Engineering

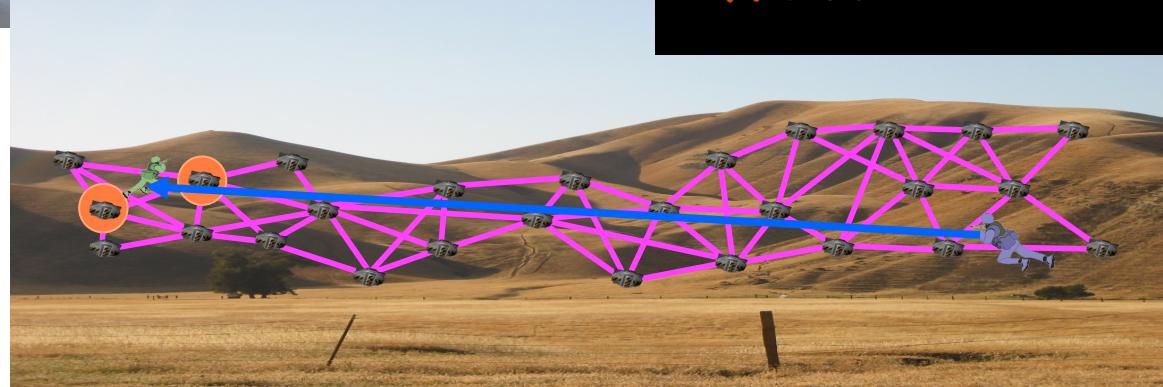
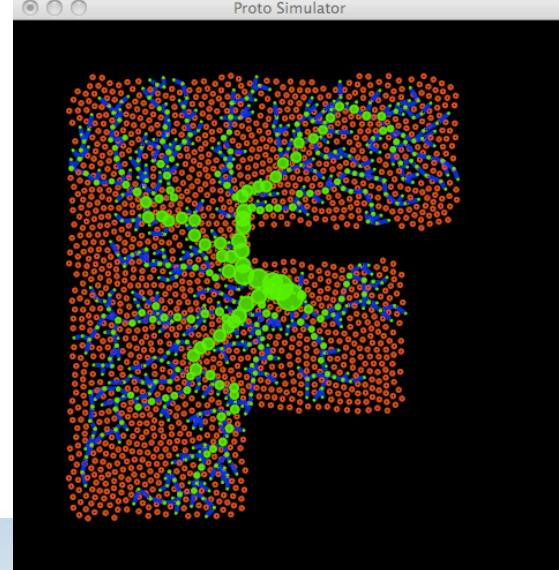


When the world is geometric...



Pervasive Computing

Morphogenetic Engineering

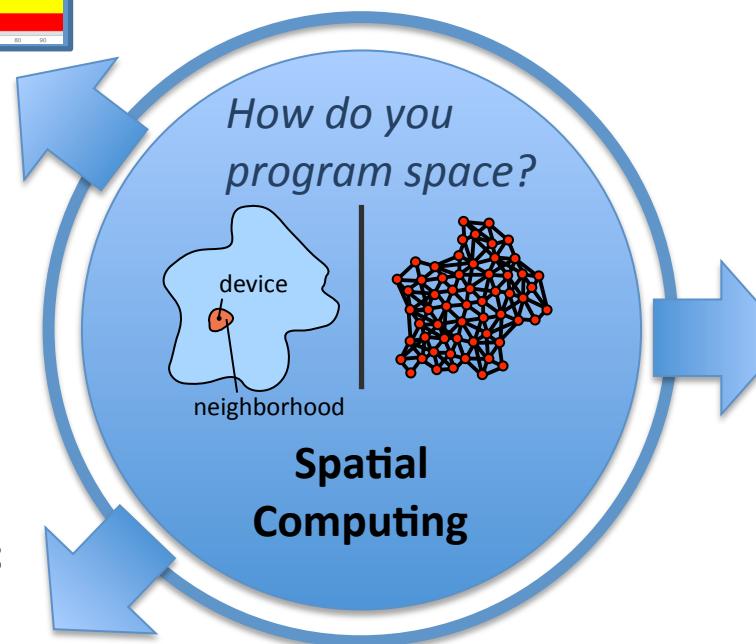


Sensor Networks

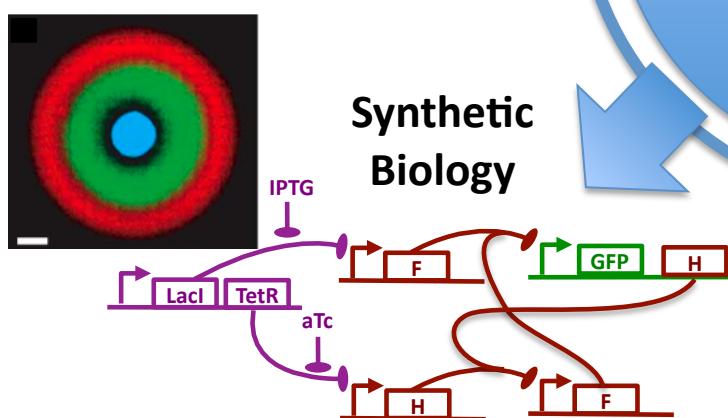
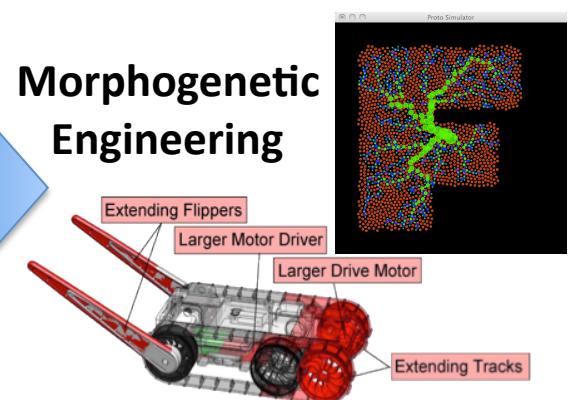
... take advantage of it!



Distributed Power Demand Response



Morphogenetic Engineering



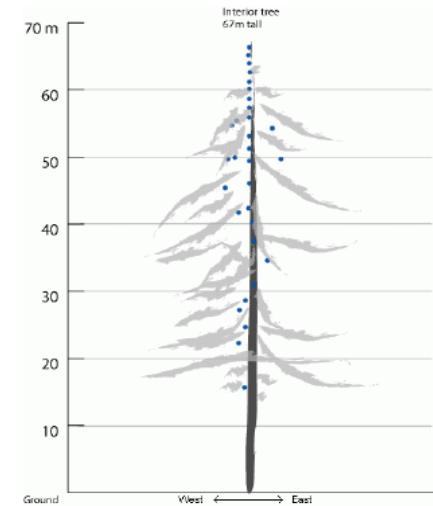
Spatial Computers



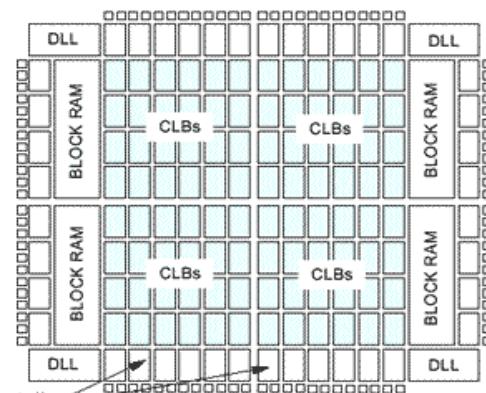
Robot Swarms



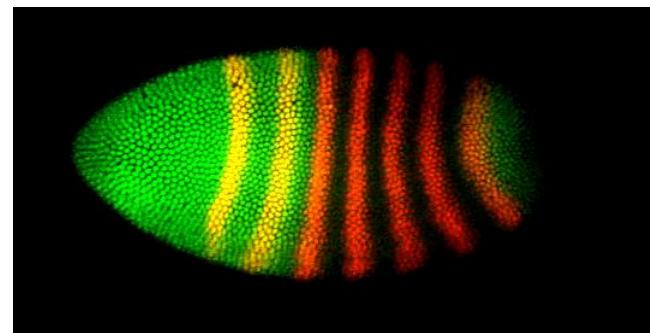
Biological Computing



Sensor Networks



Reconfigurable Computing



Cells during Morphogenesis

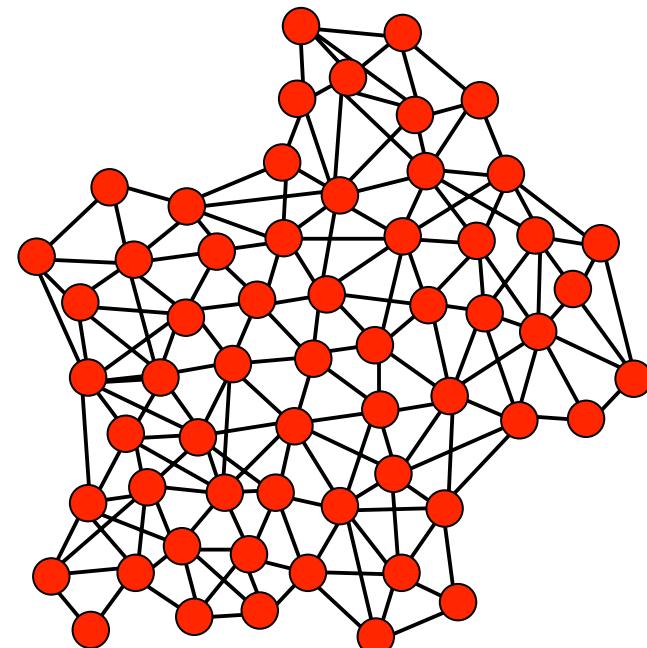
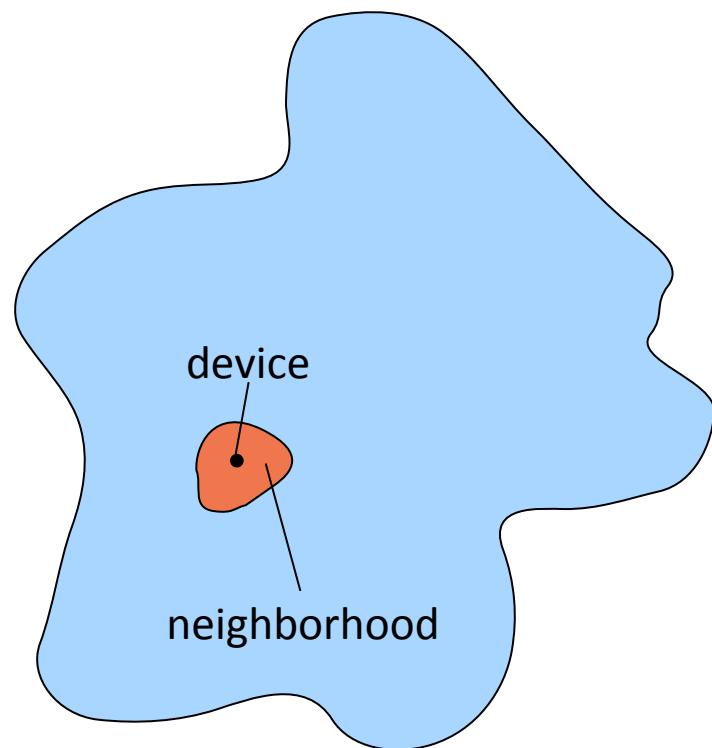


Modular Robotics

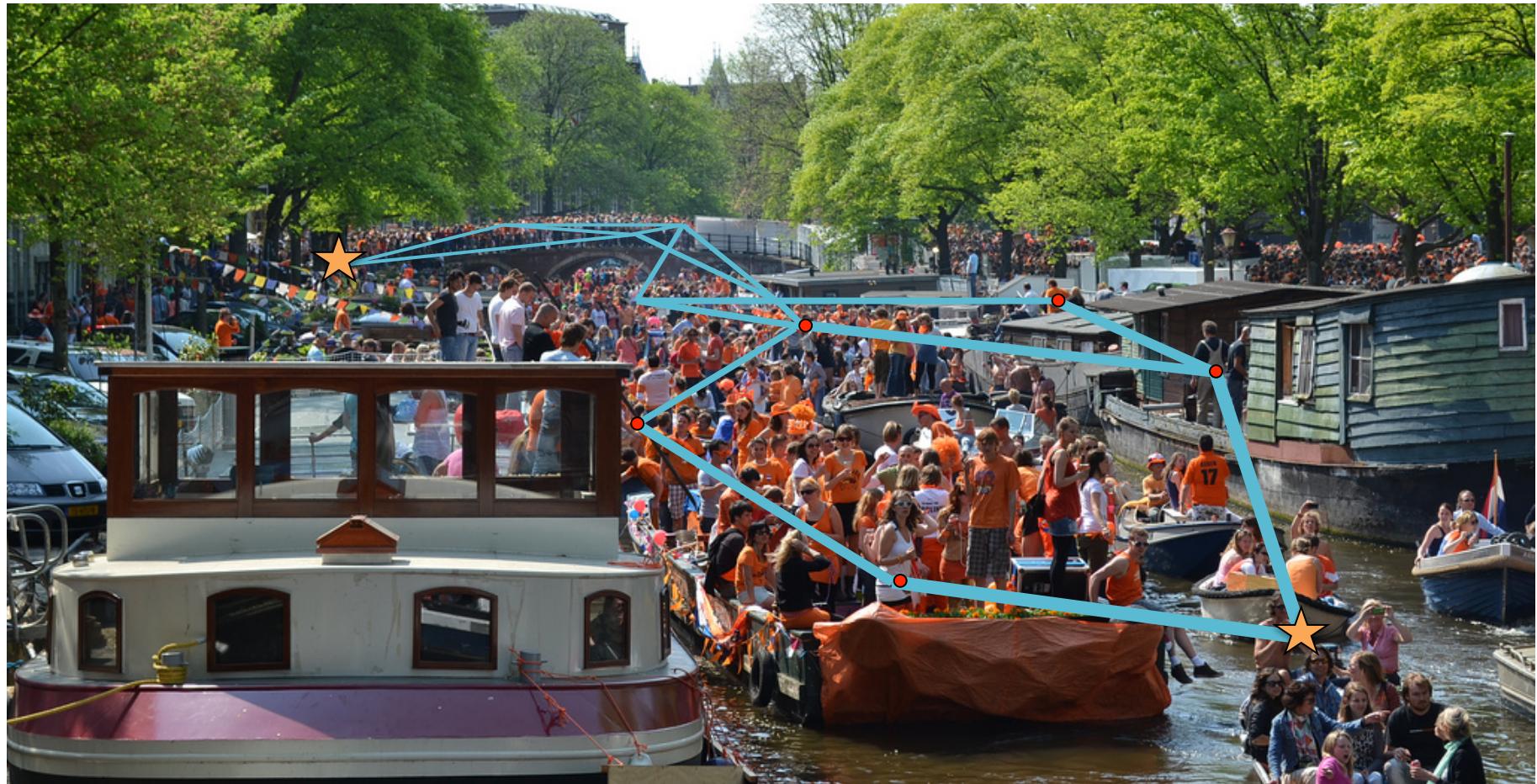
More formally...

- A spatial computer is a collection of computational devices distributed through a physical space in which:
 - the difficulty of moving information between any two devices is strongly dependent on the distance between them, and
 - the “functional goals” of the system are generally defined in terms of the system's spatial structure

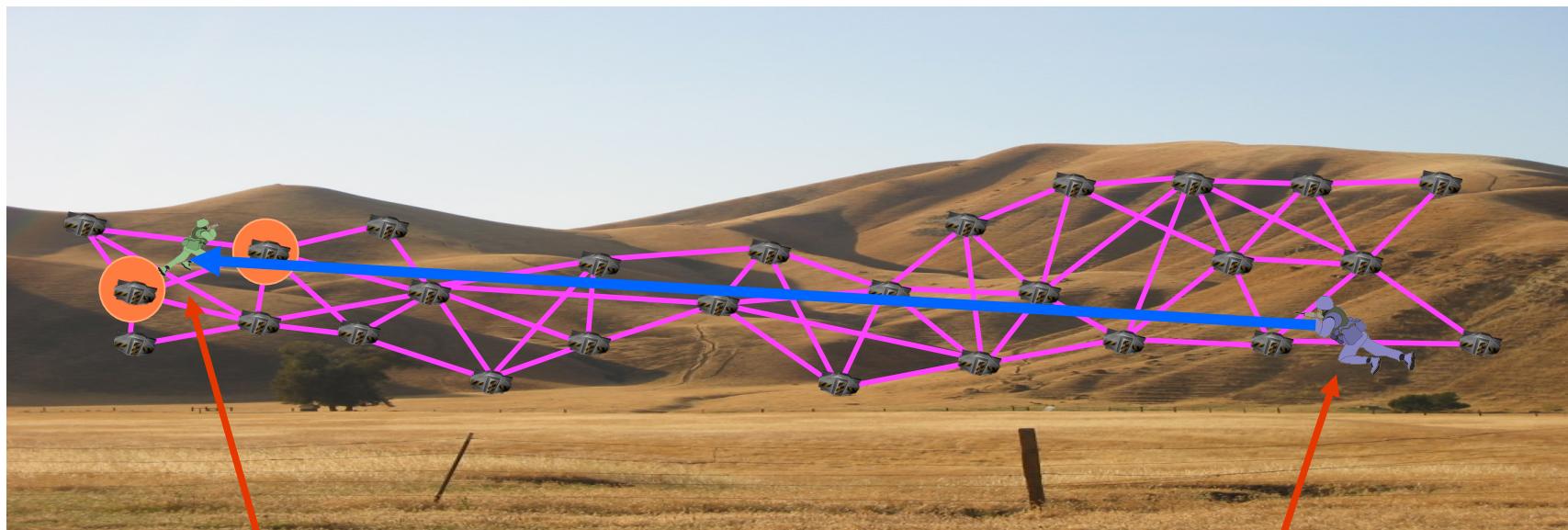
Space/Network Duality



Example: Mesh-Network Cell Phones



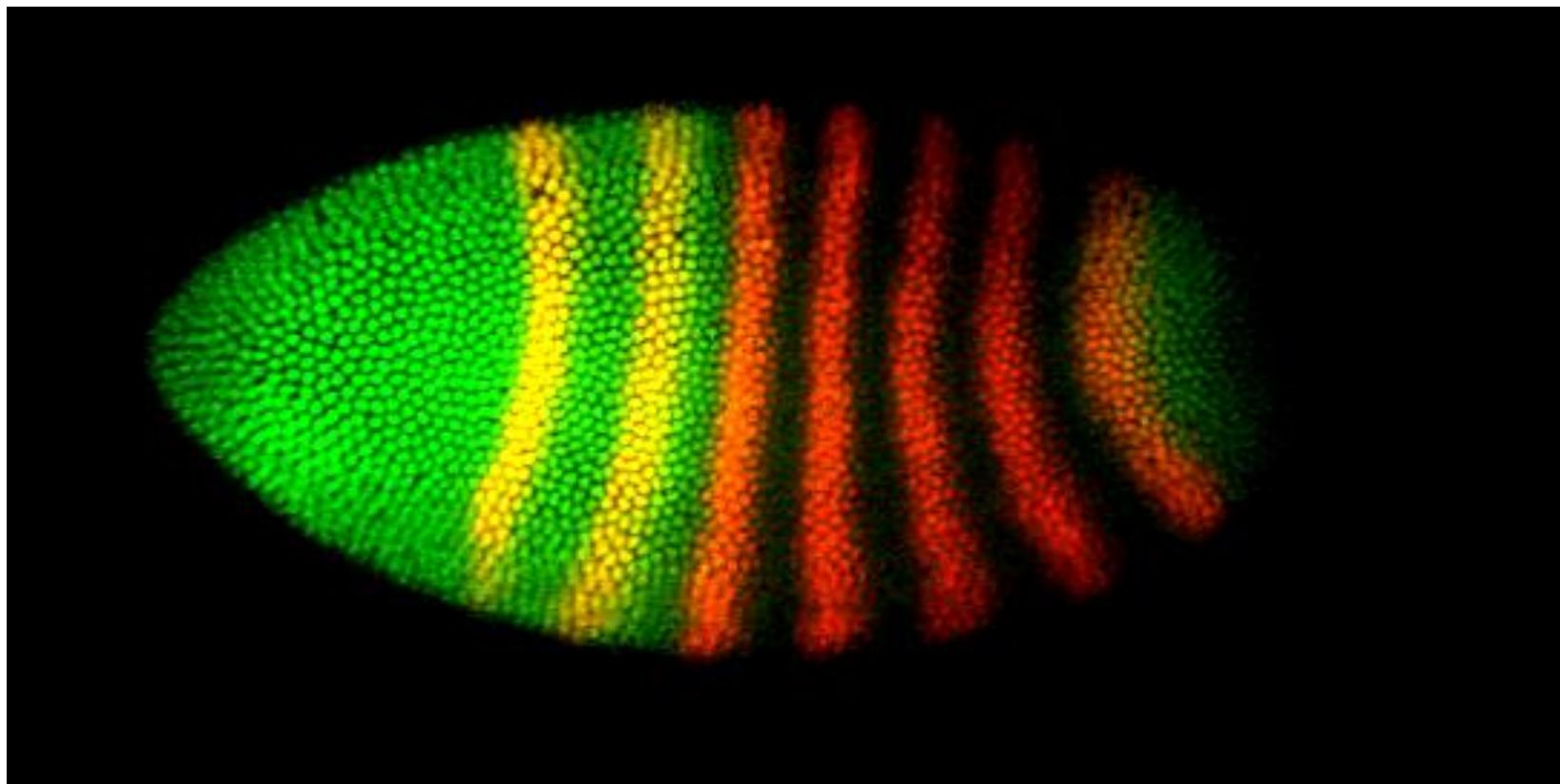
Example: Tracking in Sensor Networks



Intruder

Guard

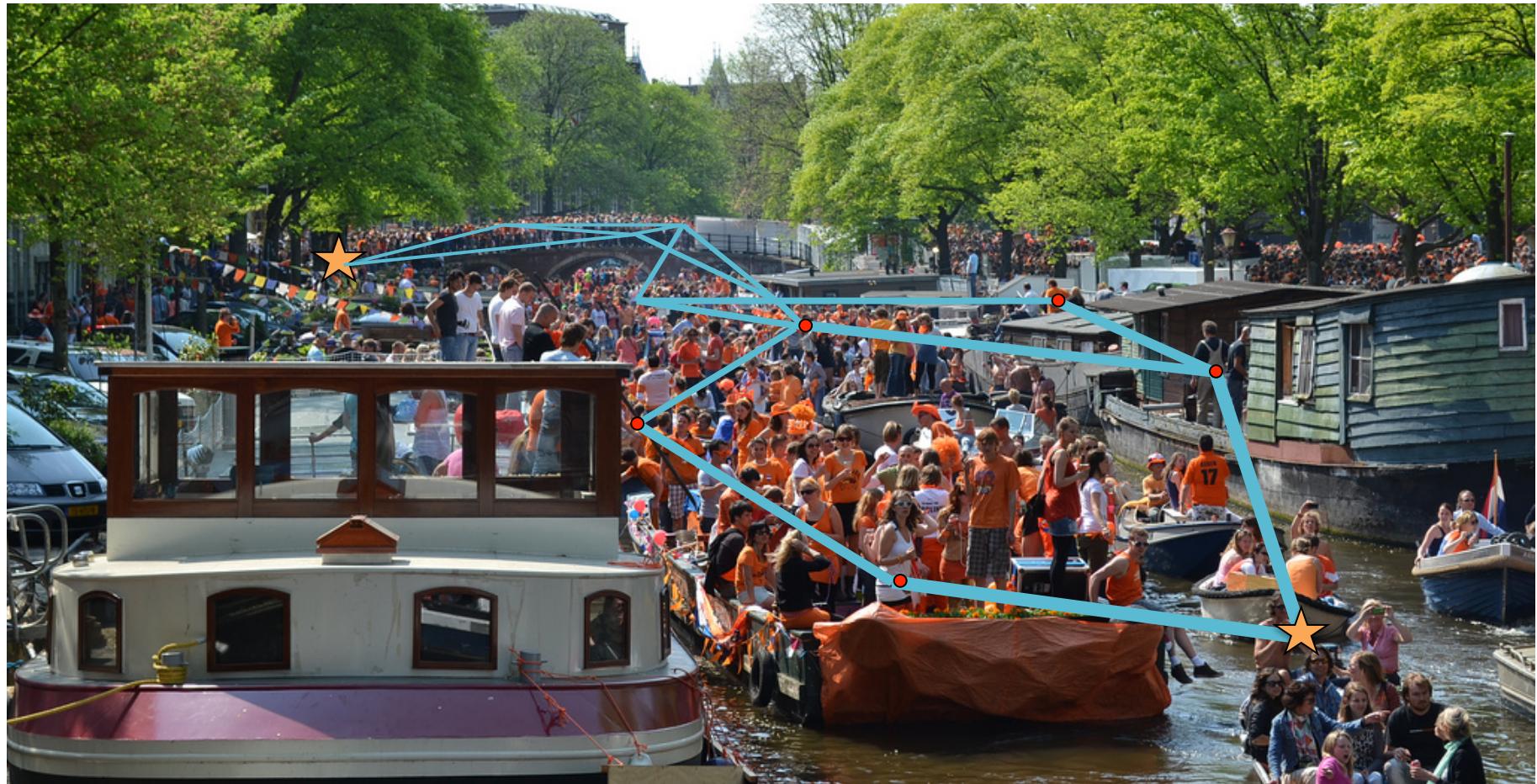
Example: Morphogenesis



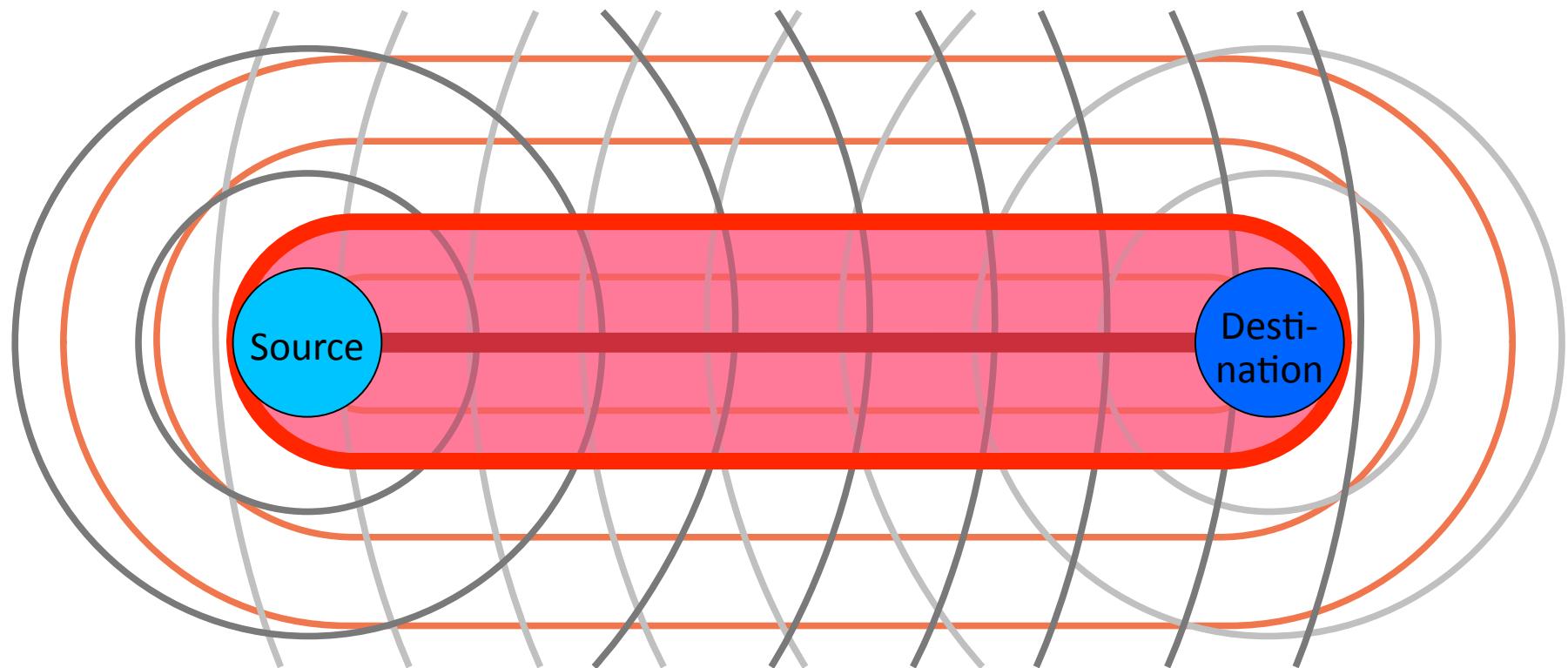
How can we program these?

- Desiderata for approaches:
 - Simple, easy to understand code
 - Robust to errors, adapt to changing environment
 - Scalable to potentially vast numbers of devices
 - Take advantage of spatial nature of problems

Example: Mesh-Network Cell Phones

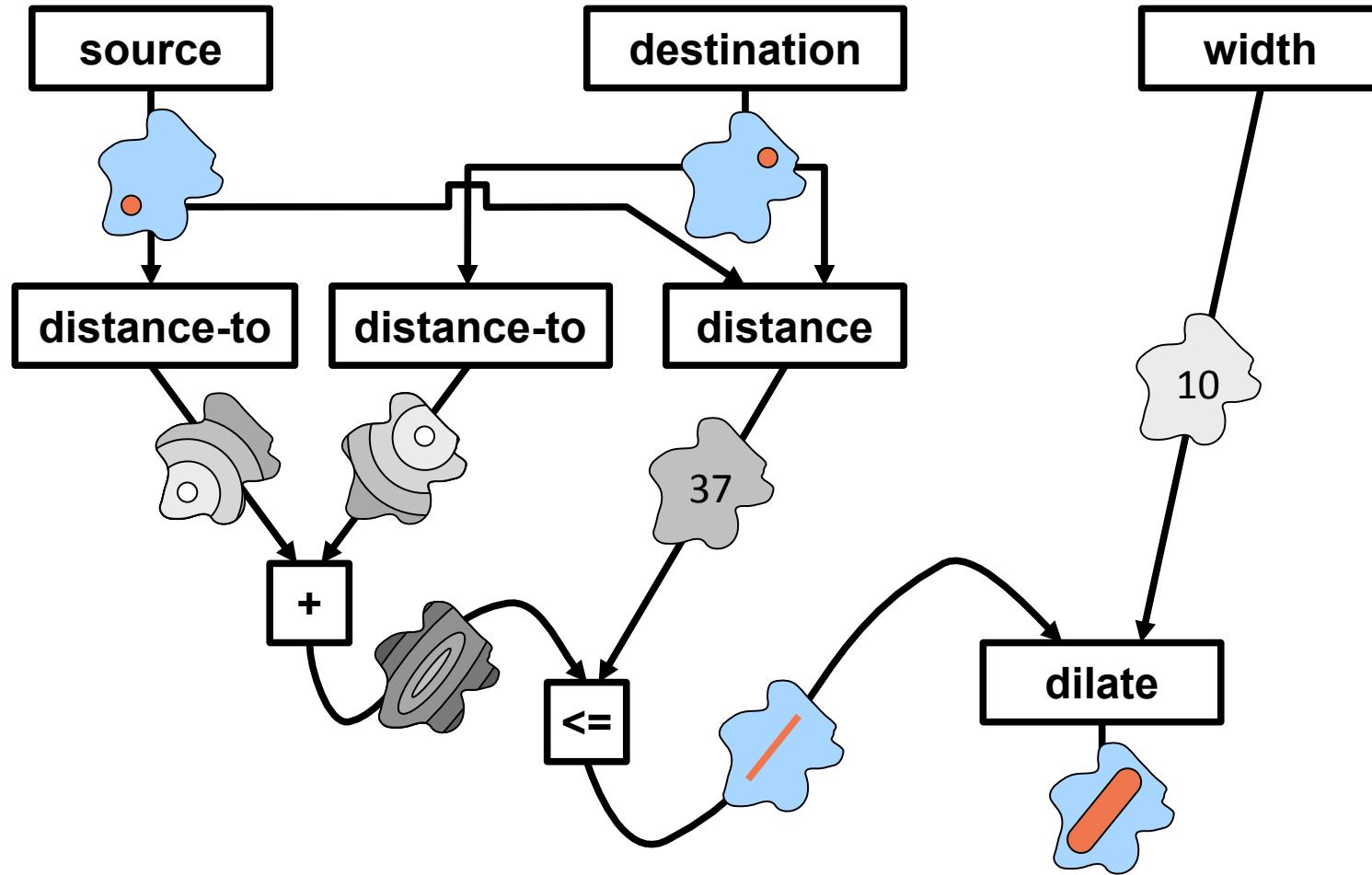


Geometric Program: Channel

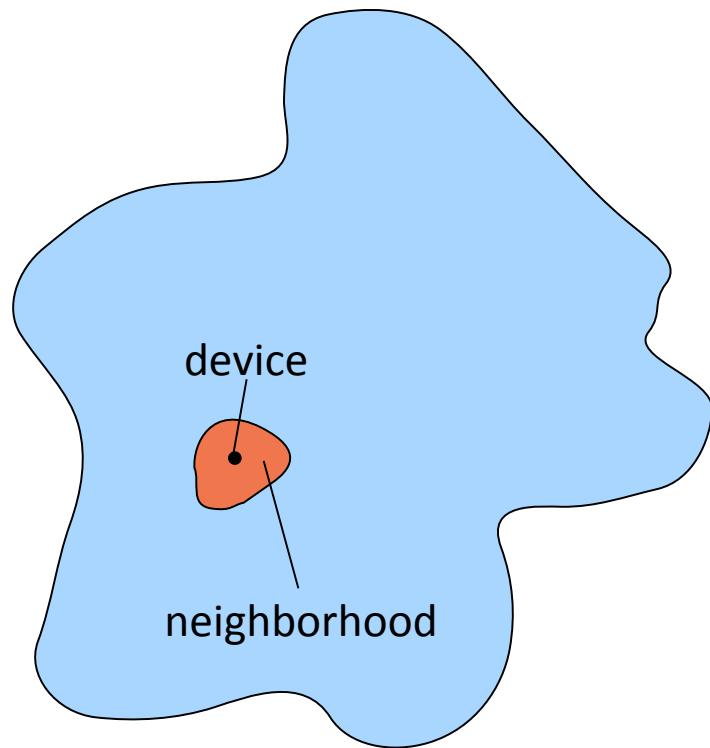


(cf. Butera)

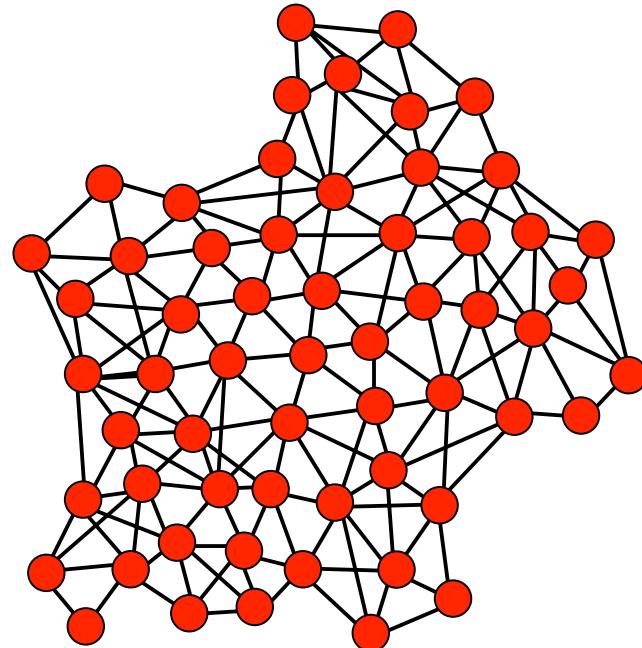
Computing with fields



Amorphous Medium



- Continuous space & time
- Infinite number of devices
- See neighbors' past state



- Approximate with:
- Discrete network of devices
- Signals transmit state

```
(def gradient (src) ...)
(def distance (src dst) ...)
(def dilate (src n)
  (<= (gradient src) n))
(def channel (src dst width)
  (let* ((d (distance src dst))
         (trail (<= (+ (gradient src)
                        (gradient dst))
                    d)))
    (dilate trail width)))
```

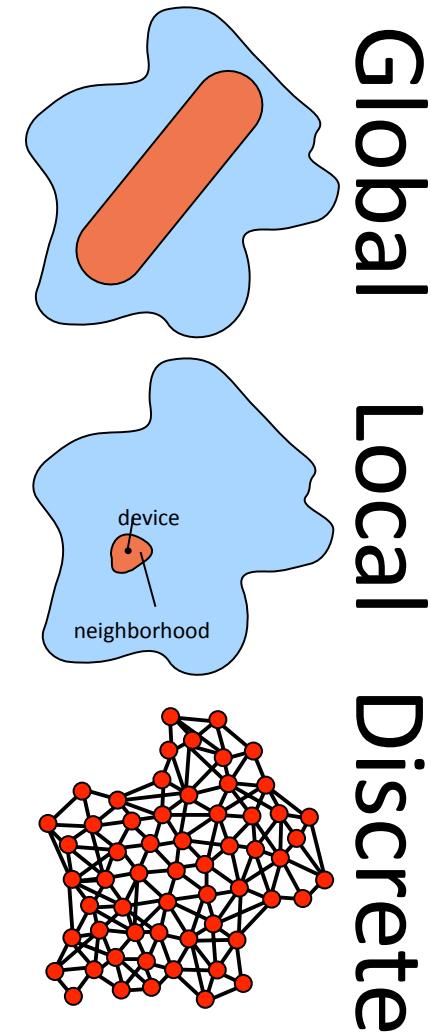
platform
specificity &
optimization

evaluation

global to local
compilation

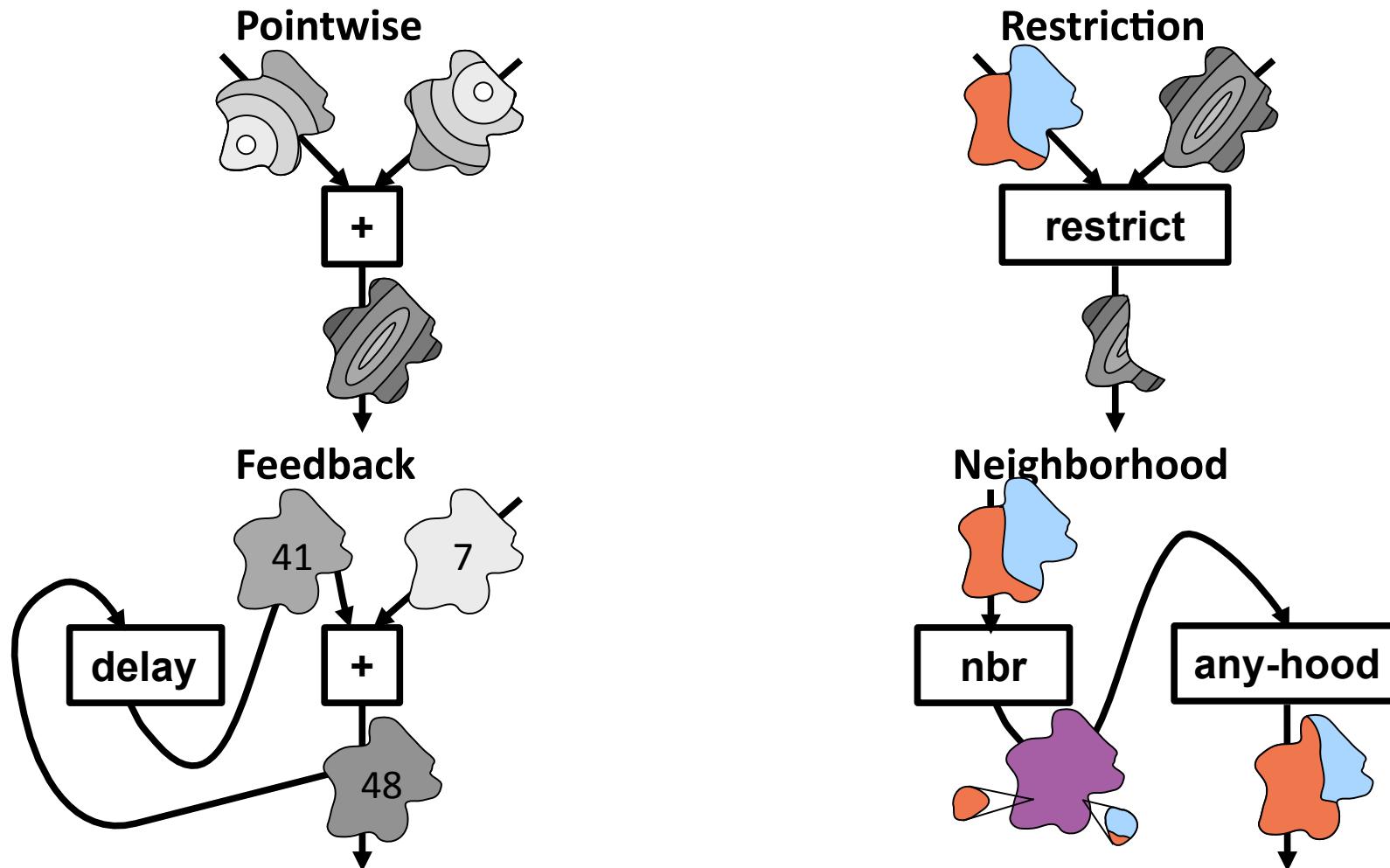
discrete
approximation

Device
Kernel

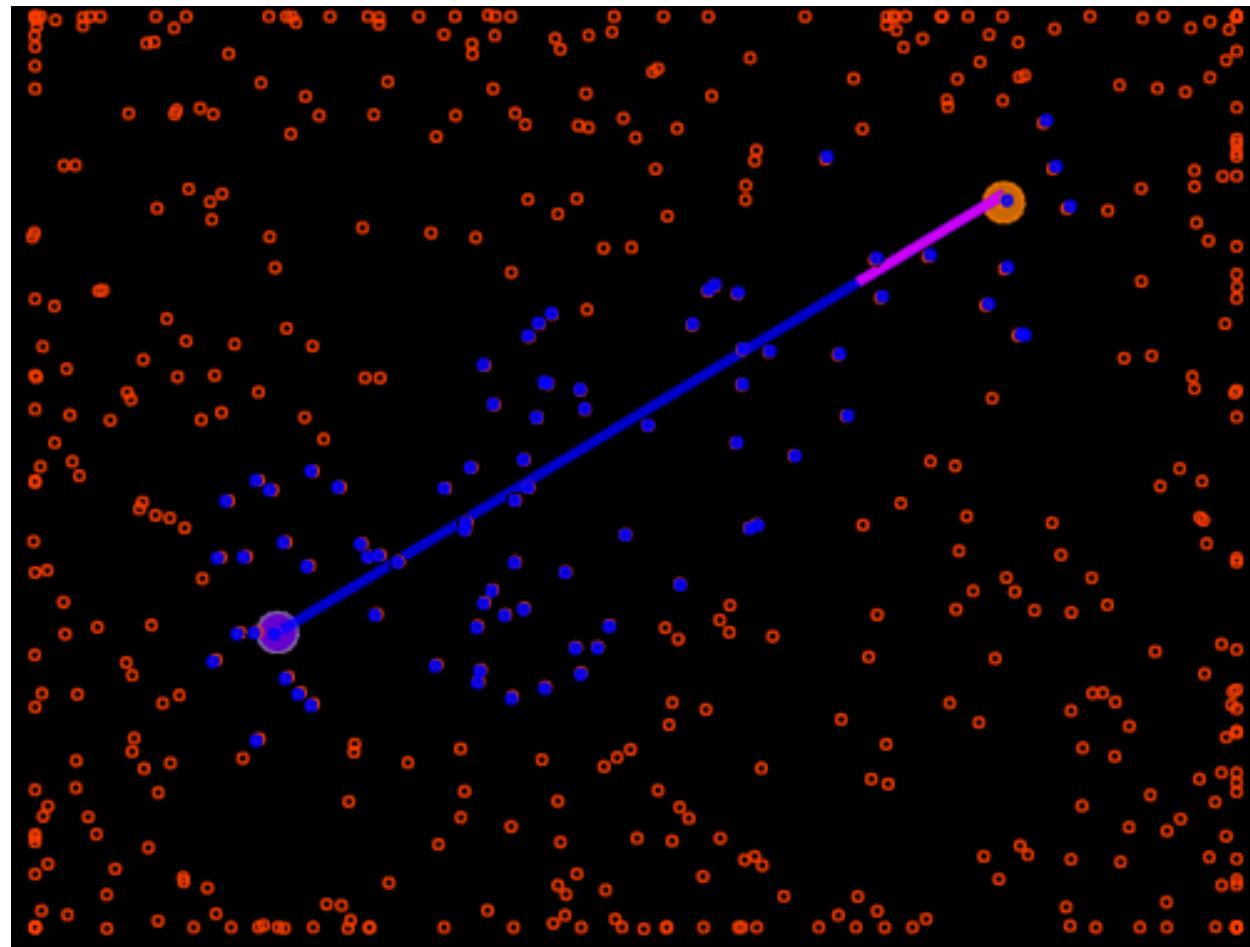


[Beal & Bachrach, '06]

Proto's Families of Primitives



In simulation...



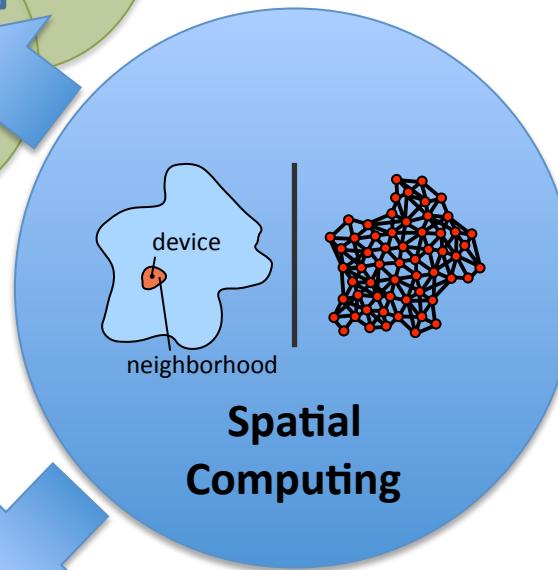
Weaknesses

- Programmers can break the abstraction
 - Functional programming scares people
 - No dynamic allocation of processes
 - No formal proofs available for quality of approximation in a composed program
-
- The diagram consists of four vertical brackets on the right side of the slide. From top to bottom:
 - A green bracket groups the first two items: "Programmers can break the abstraction" and "Functional programming scares people".
 - A blue bracket groups the next two items: "No dynamic allocation of processes" and "No formal proofs available for quality of approximation in a composed program".
 - A red bracket groups the last two items: "By design Upgrades Coming Soon!" and "Active research area".

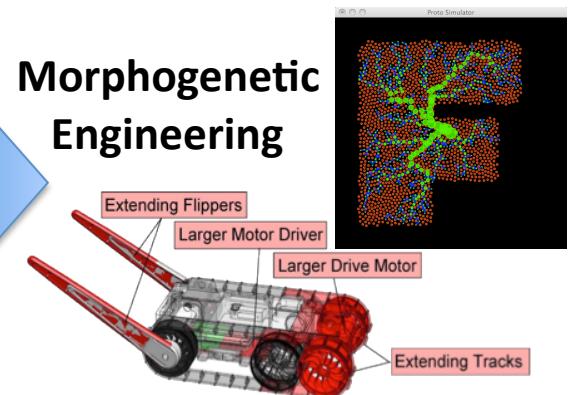


Distributed Power Demand Response

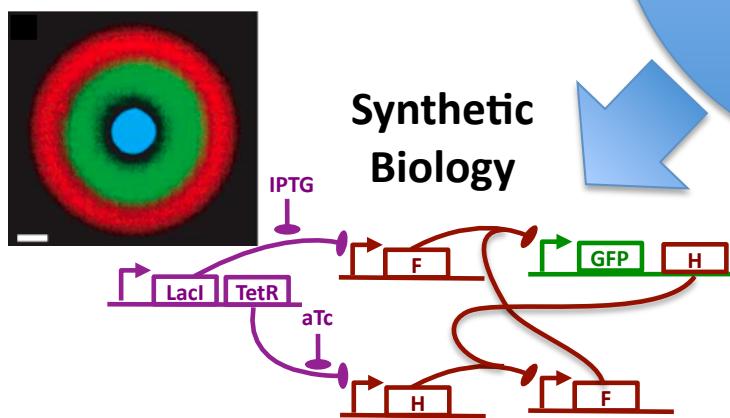
How can millions of appliances coordinate to change how we use energy?



Morphogenetic Engineering



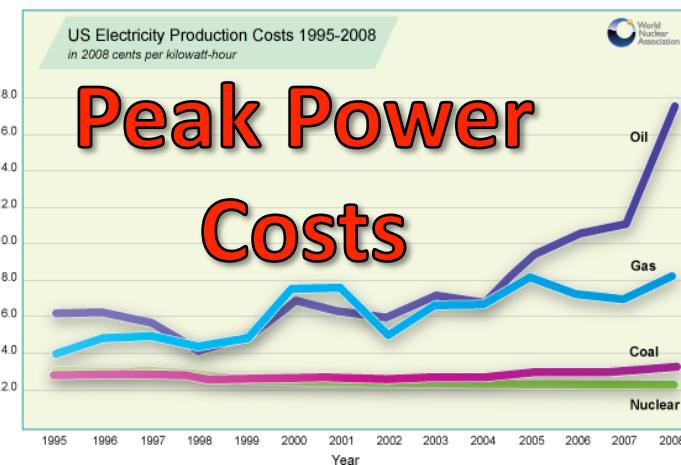
Synthetic Biology



Distributed Power Demand Response



Why is DR important?



Production Costs = Operations & Maintenance + Fuel. Production costs do not include indirect costs or capital.
Source: Vertix Velocity Suite, via NEI

Inefficiency of Demand vs. Intention

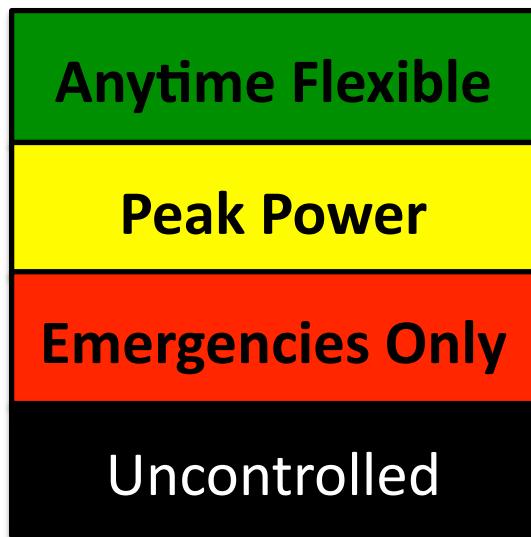
- Demand/supply mismatch is extremely costly
 - \$ billions to utilities, local governments
- Consumers dramatically reduce demand when:
 - ... aware of actual appliance energy use
 - ... informed about neighbors' energy use
 - ... aware of stress on power grid

*Coordination opportunity: peak-shaving & demand management by **automating volunteerism***

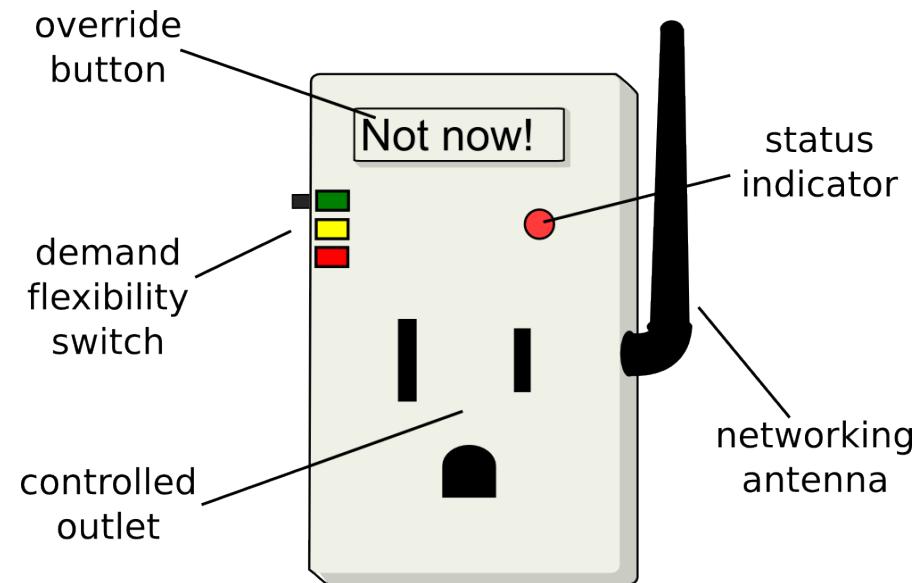
Key Challenges

- Scalability:
 - Safe, reliable coordinated response from millions of devices in <5 minutes
- Consumer interface:
 - High benefit, low “annoyance factor”
 - Eliciting useful information
 - Privacy concerns
- Deployability:
 - Technology alignment with market structure
 - Low cost devices
 - Market fragmentation across grid & in home

Capturing User Requirements

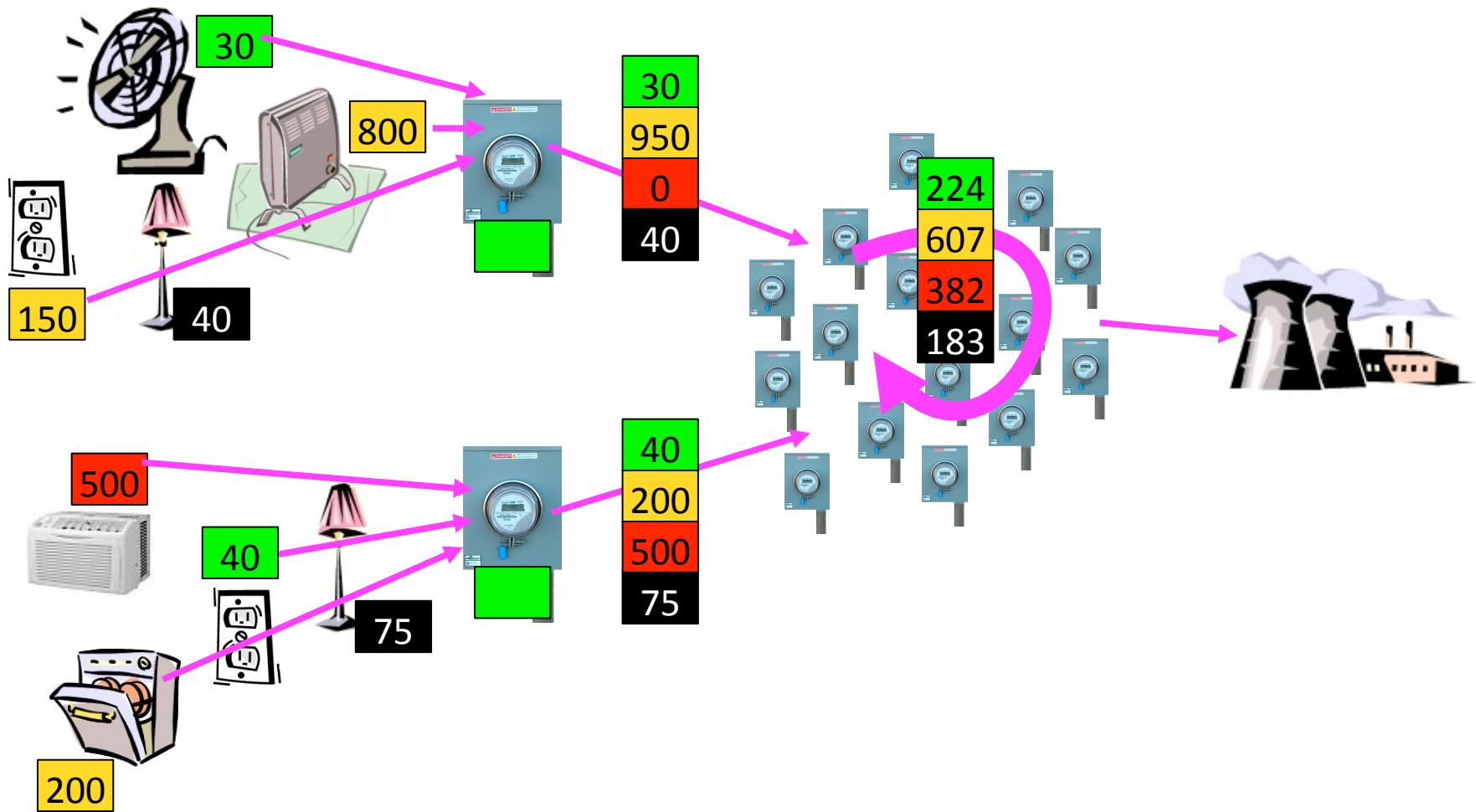


Qualitative
Energy Flexibility



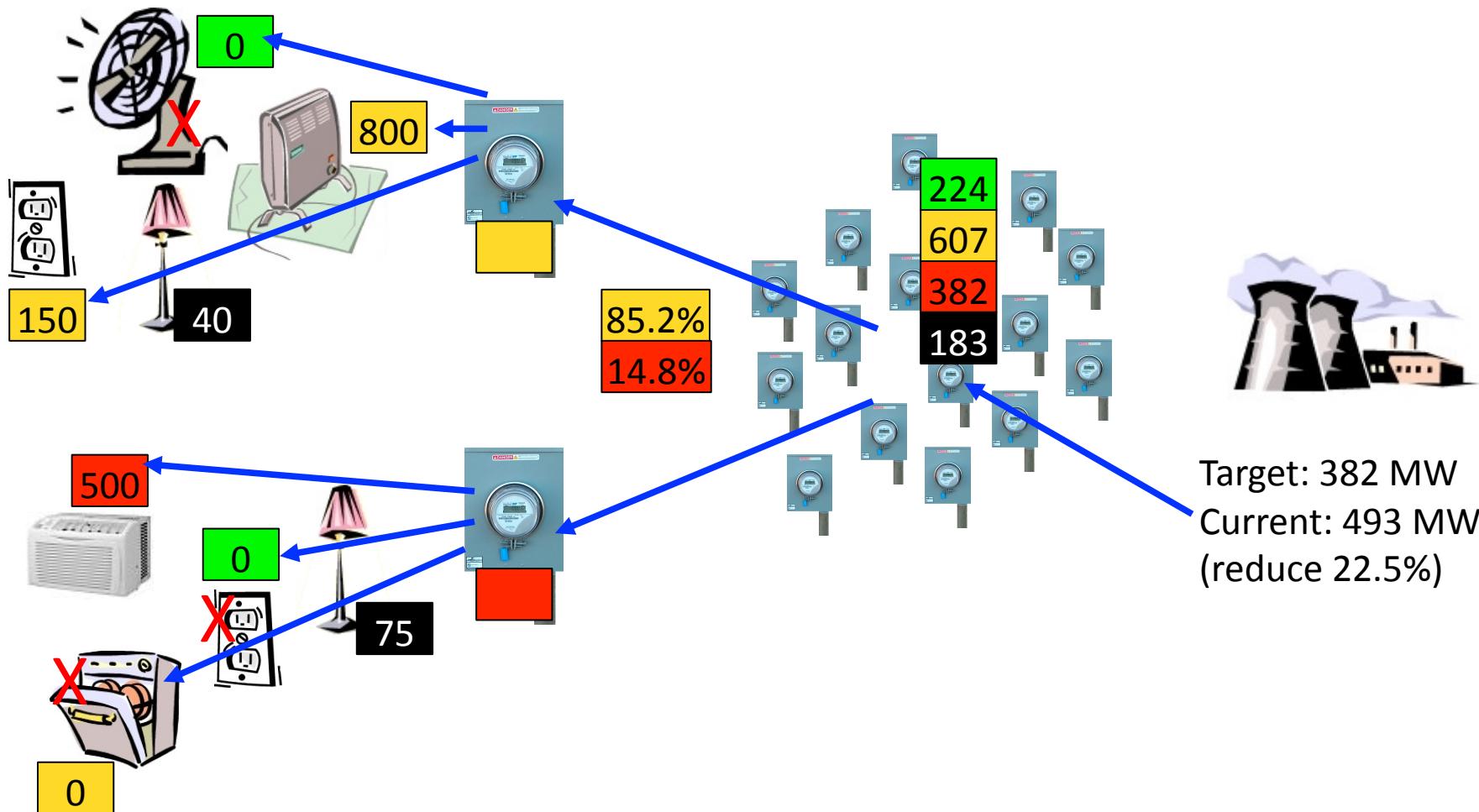
Smart plugs, new appliances,
home automation, ...

Distributed Creation of Aggregate Model



Model Coordinates Local Control Actions

Raytheon
BBN Technologies



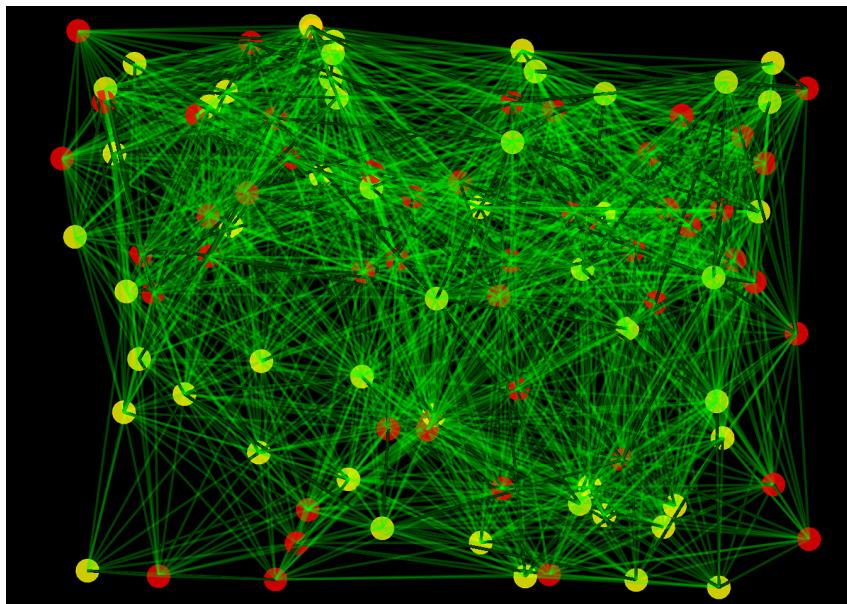
ColorPower Algorithm

- Challenge: fast, private, robust, non-intrusive
- Approach: randomized distributed control
 - Aggregate flexibility information to shared model
 - Disseminate control signals via gossip
 - Local decision; coin-flip for fractional color
 - Weight for availability, overdamped control

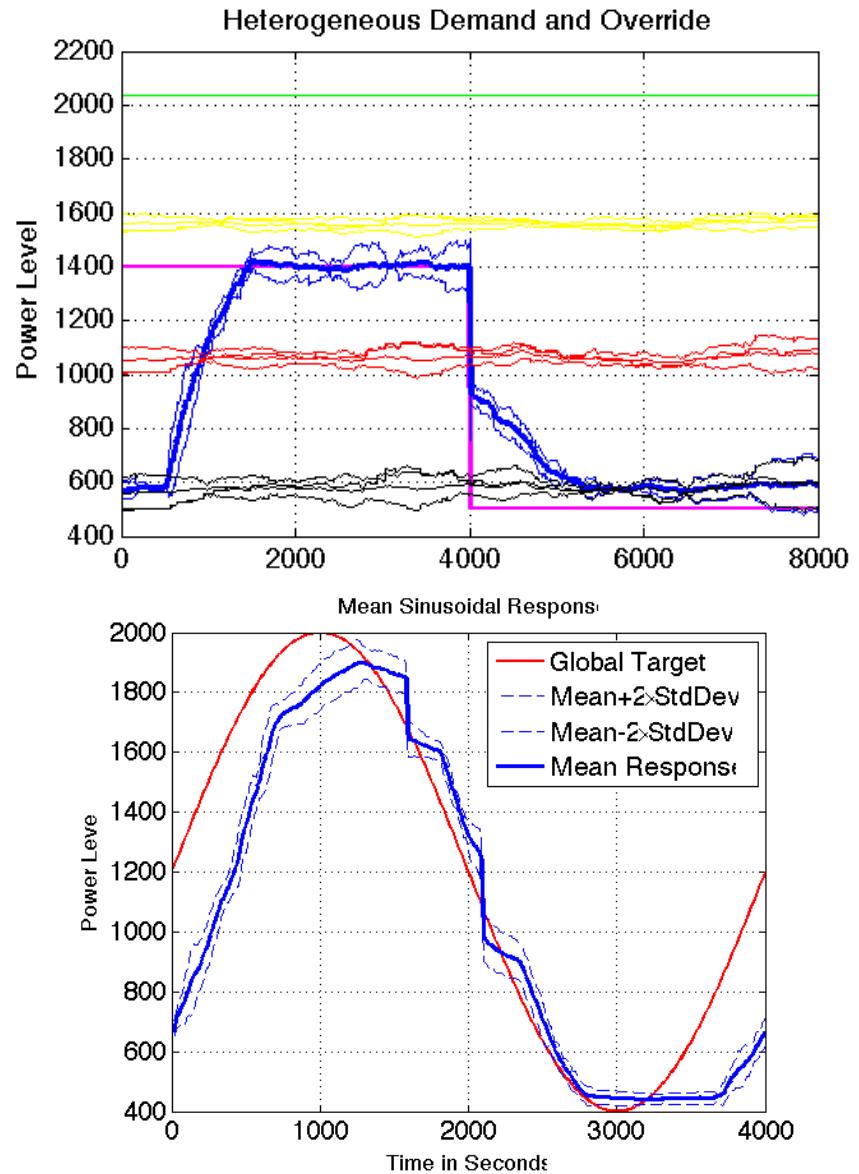
Control problem: long timeouts on state changes

ColorPower Initial Algorithm

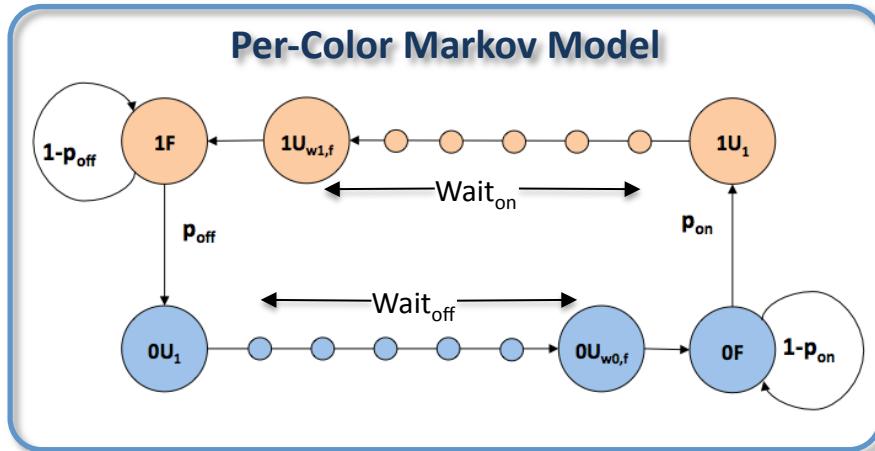
- Simulation on 100 device network using PID local controller



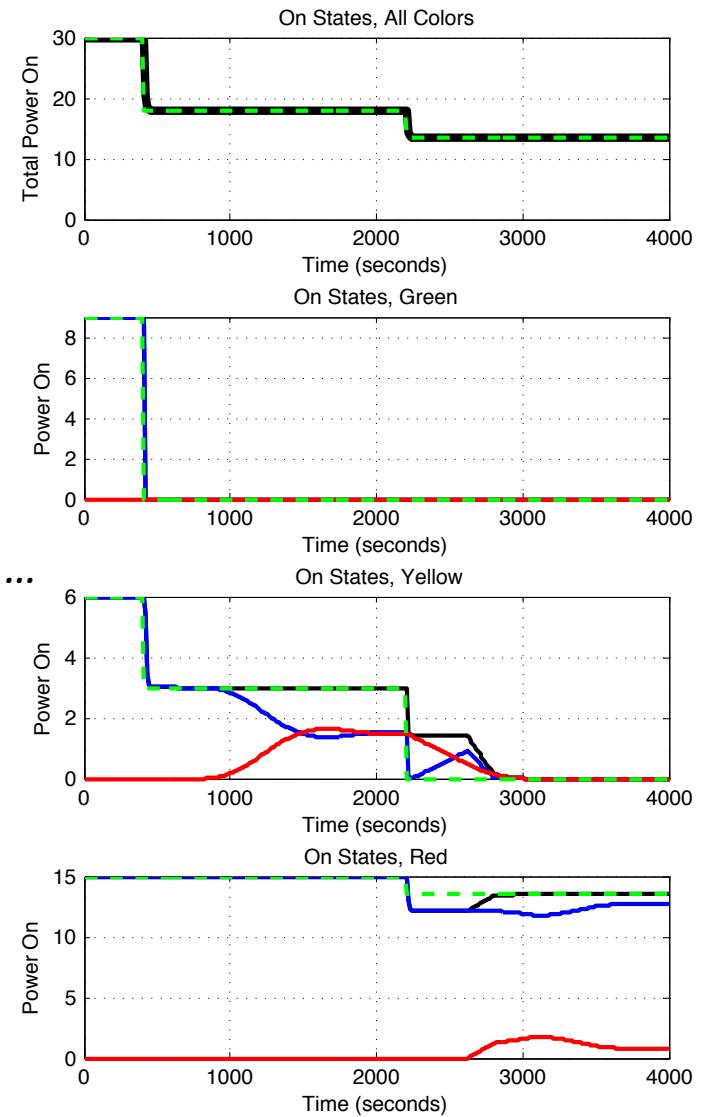
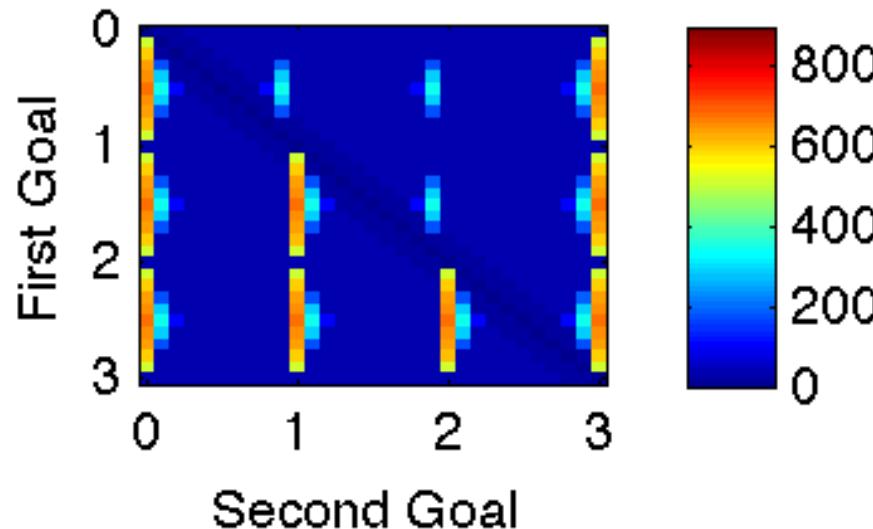
[Ranade & Beal, IEEE SASO 2010]



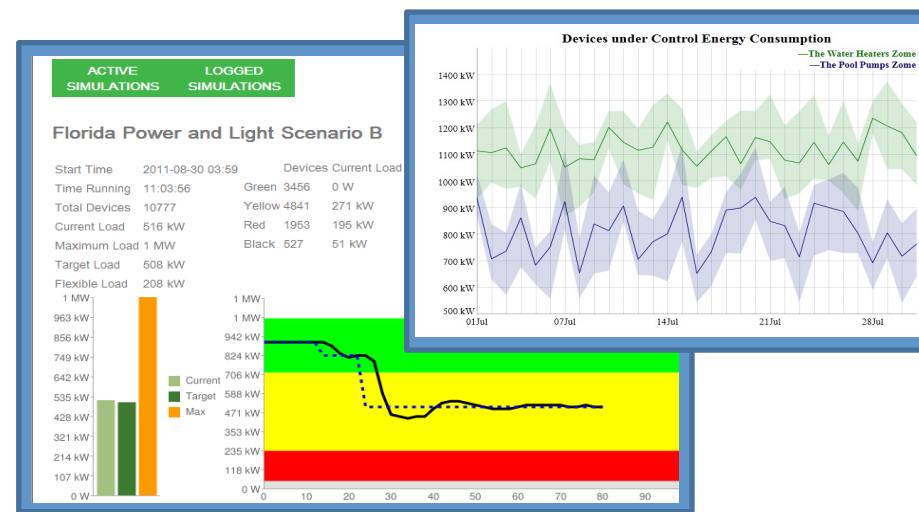
Improved constraint-based controller



*Based on a Markov model of state flux...
... devices use constraint-based controls ...
... allowing <1 minute aggregate convergence time.*



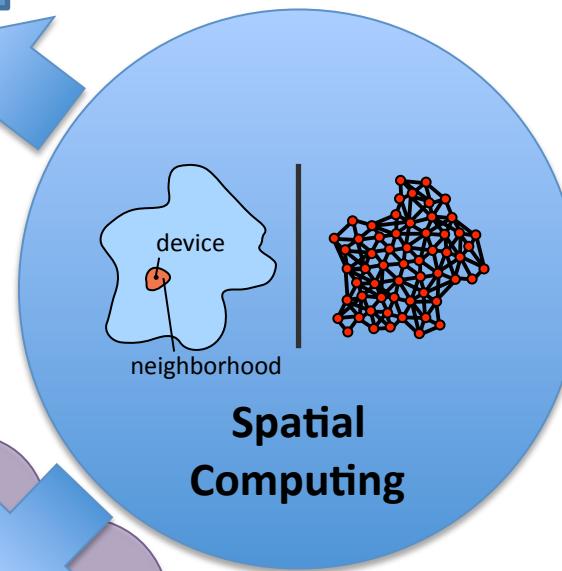
Bringing it to the real world...



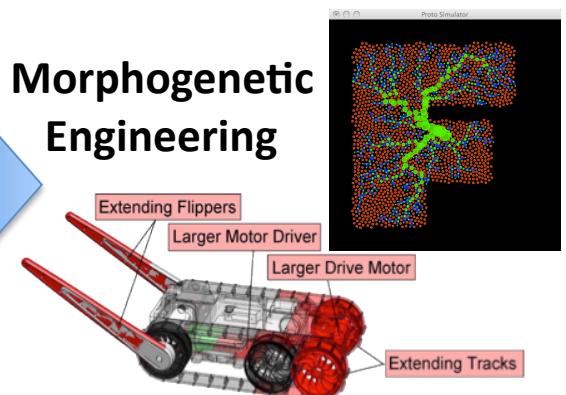
*Founded 2010 with team of serial entrepreneurs,
well-funded, offices in San Francisco and
Cambridge, first products in launch process...*



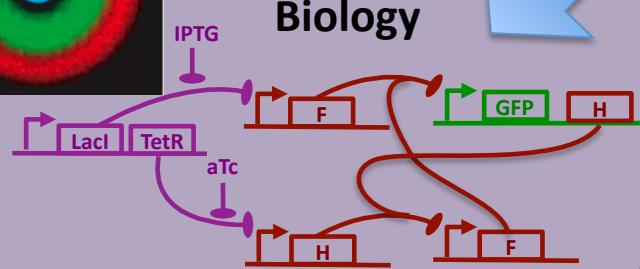
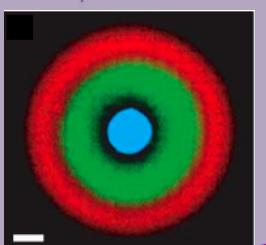
Distributed Power Demand Response



Morphogenetic Engineering



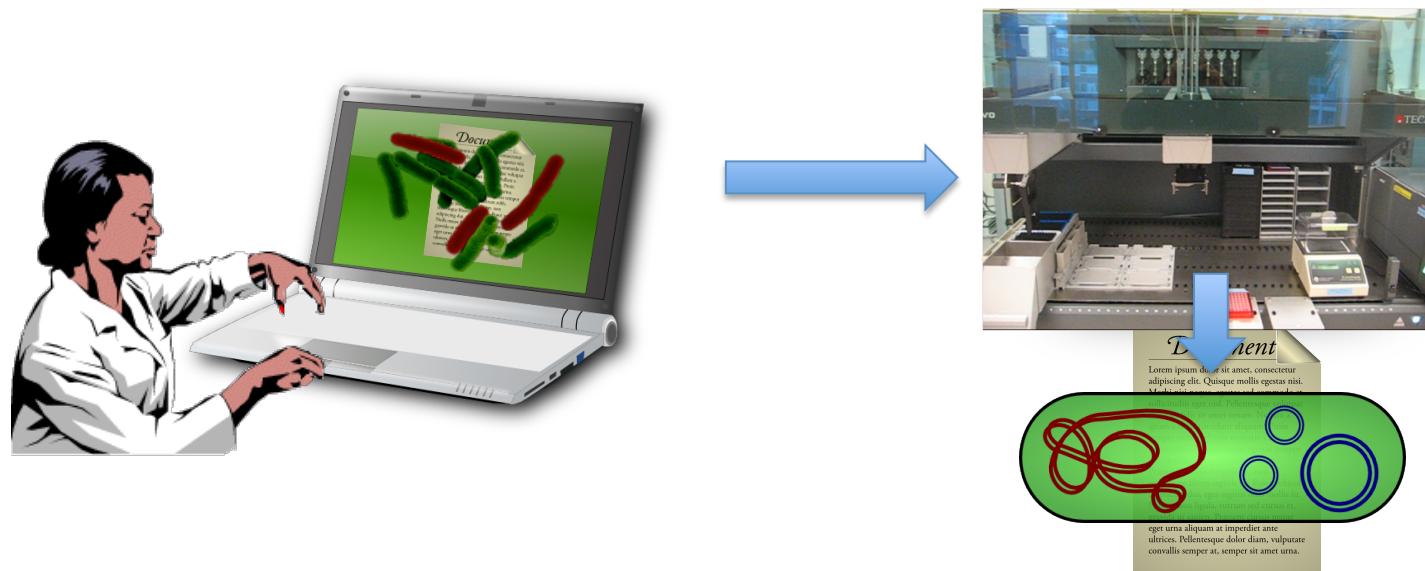
Synthetic Biology



How do you program the behavior of 10^{12} cells?

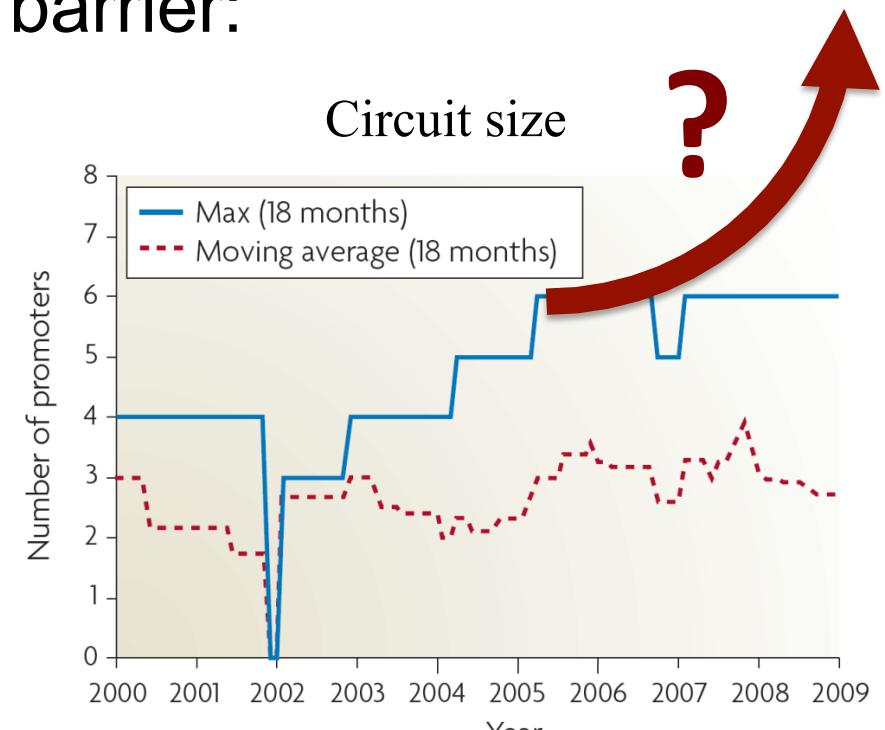
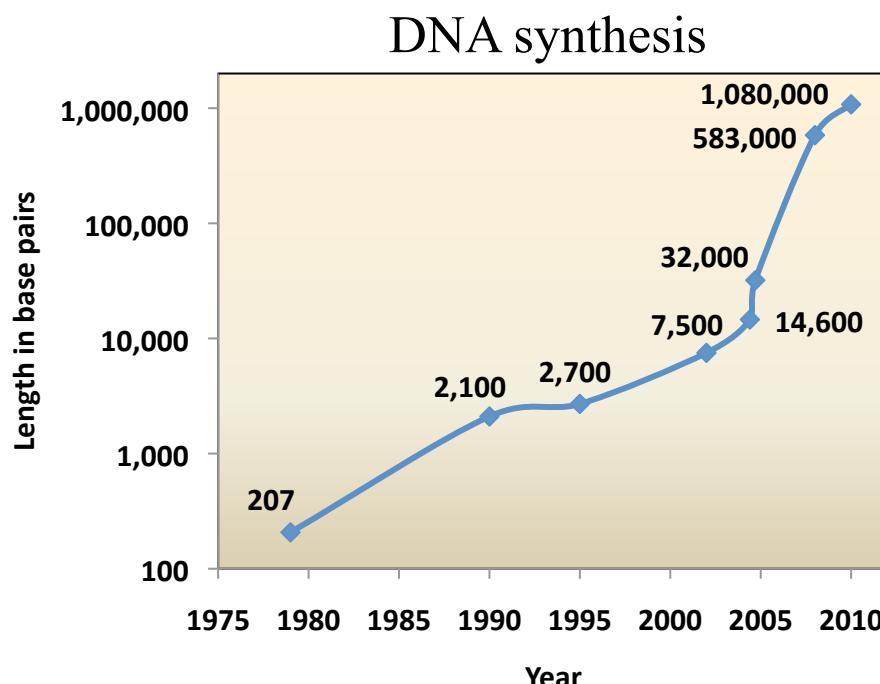
Vision: WYSIWYG Synthetic Biology

Bioengineering should be like document preparation:



Why is this important?

- Breaking the complexity barrier:



[Purnick & Weiss, '09]

- Multiplication of research impact
- Reduction of barriers to entry

*Sampling of systems in publications with experimental circuits

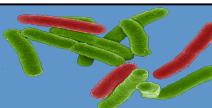
Why a tool-chain?

Organism Level Description

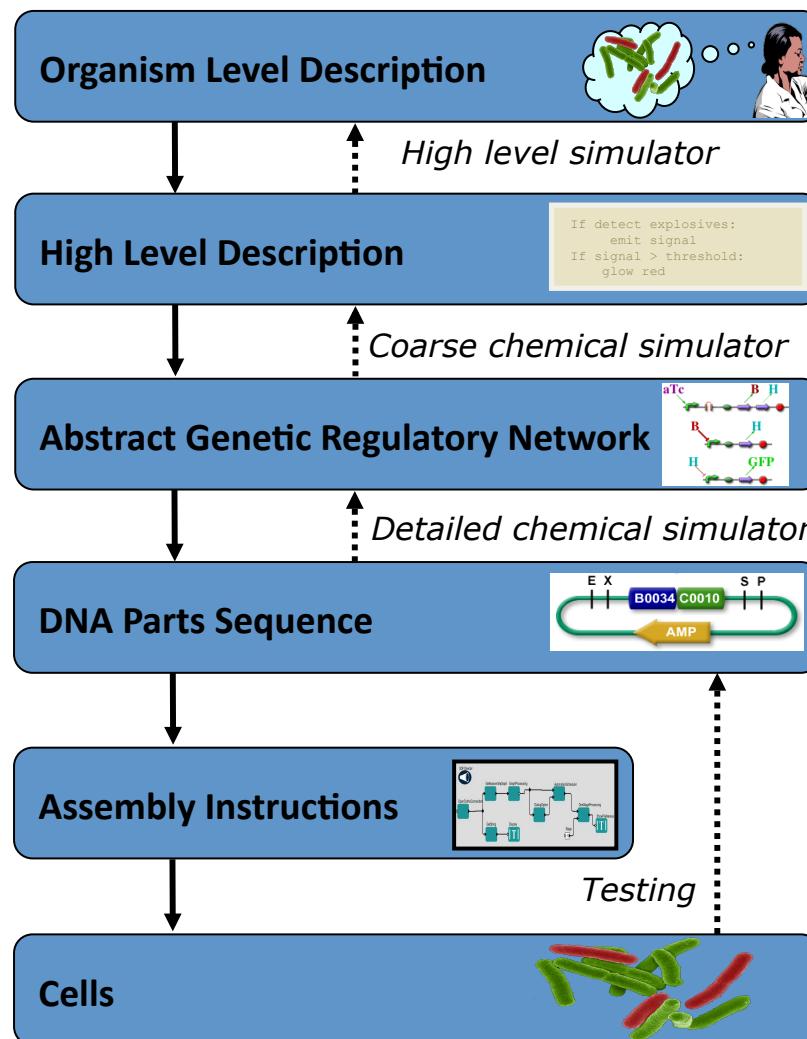


*This gap is too big
to cross with a
single method!*

Cells



The TASBE architecture:



Collaborators:



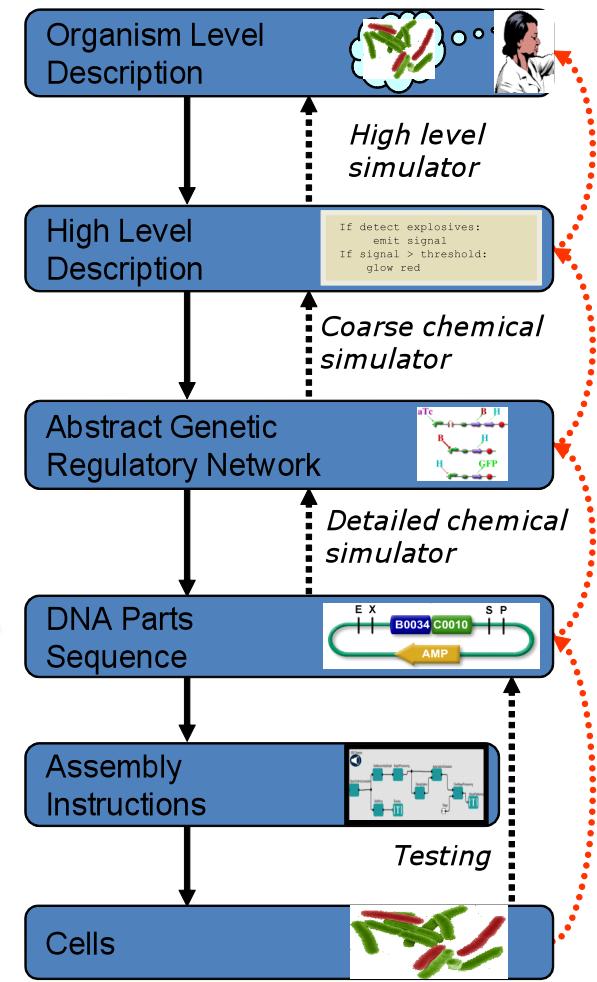
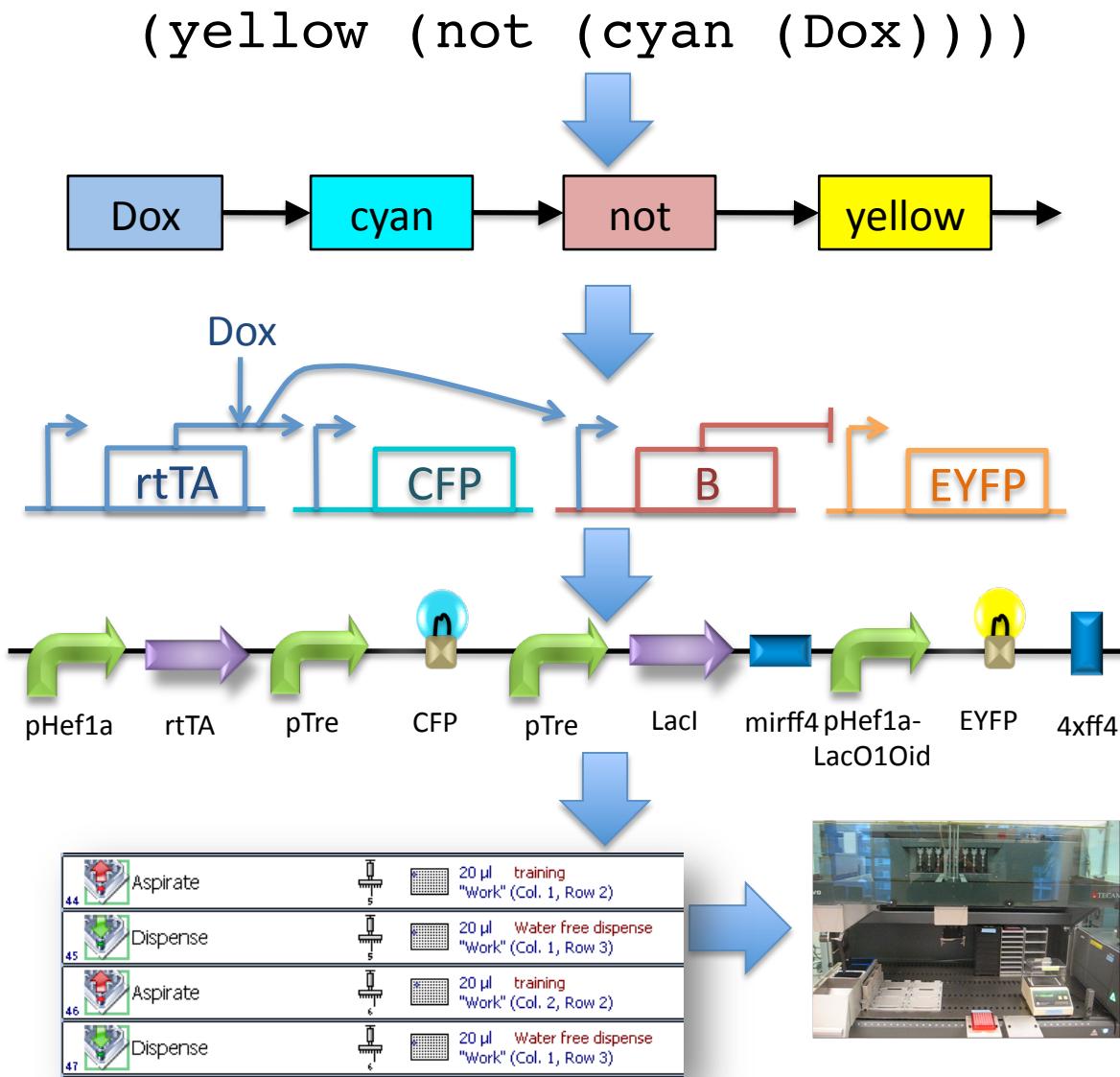
Ron
Weiss



Douglas
Densmore

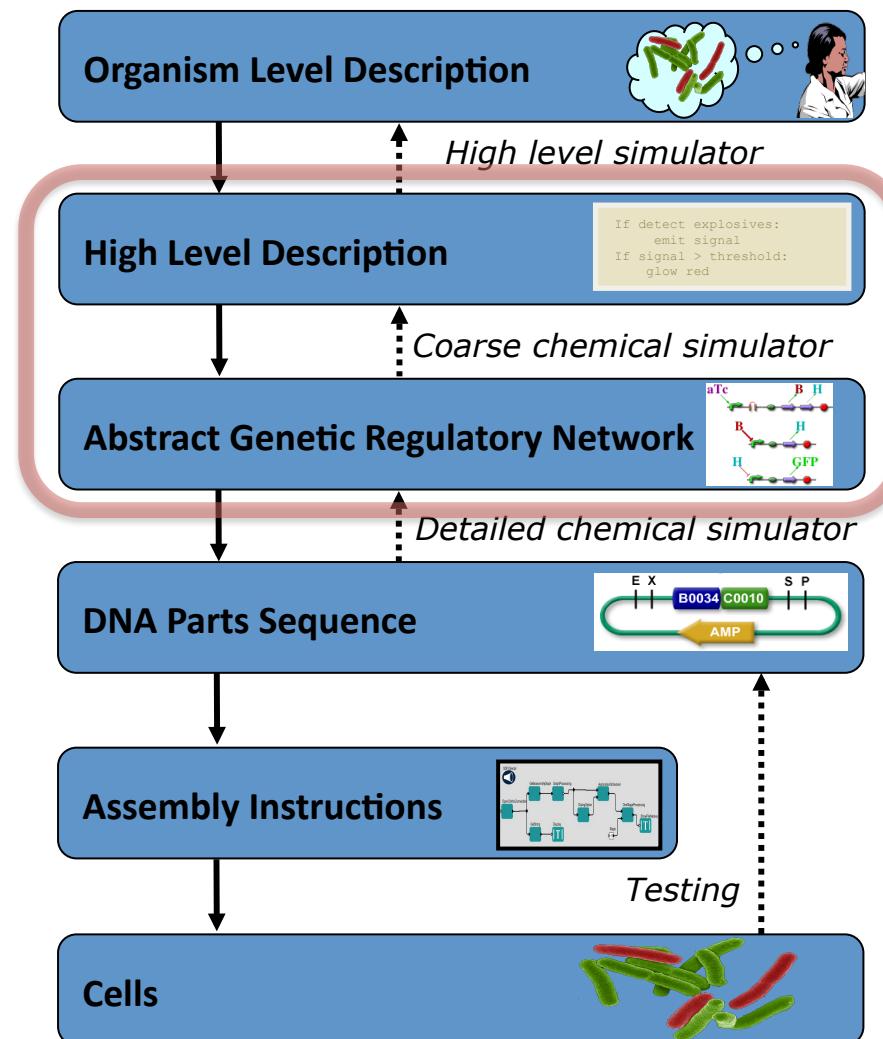
*Modular architecture
also open for flexible
choice of organisms,
protocols, methods, ...*

A Tool-Chain Example

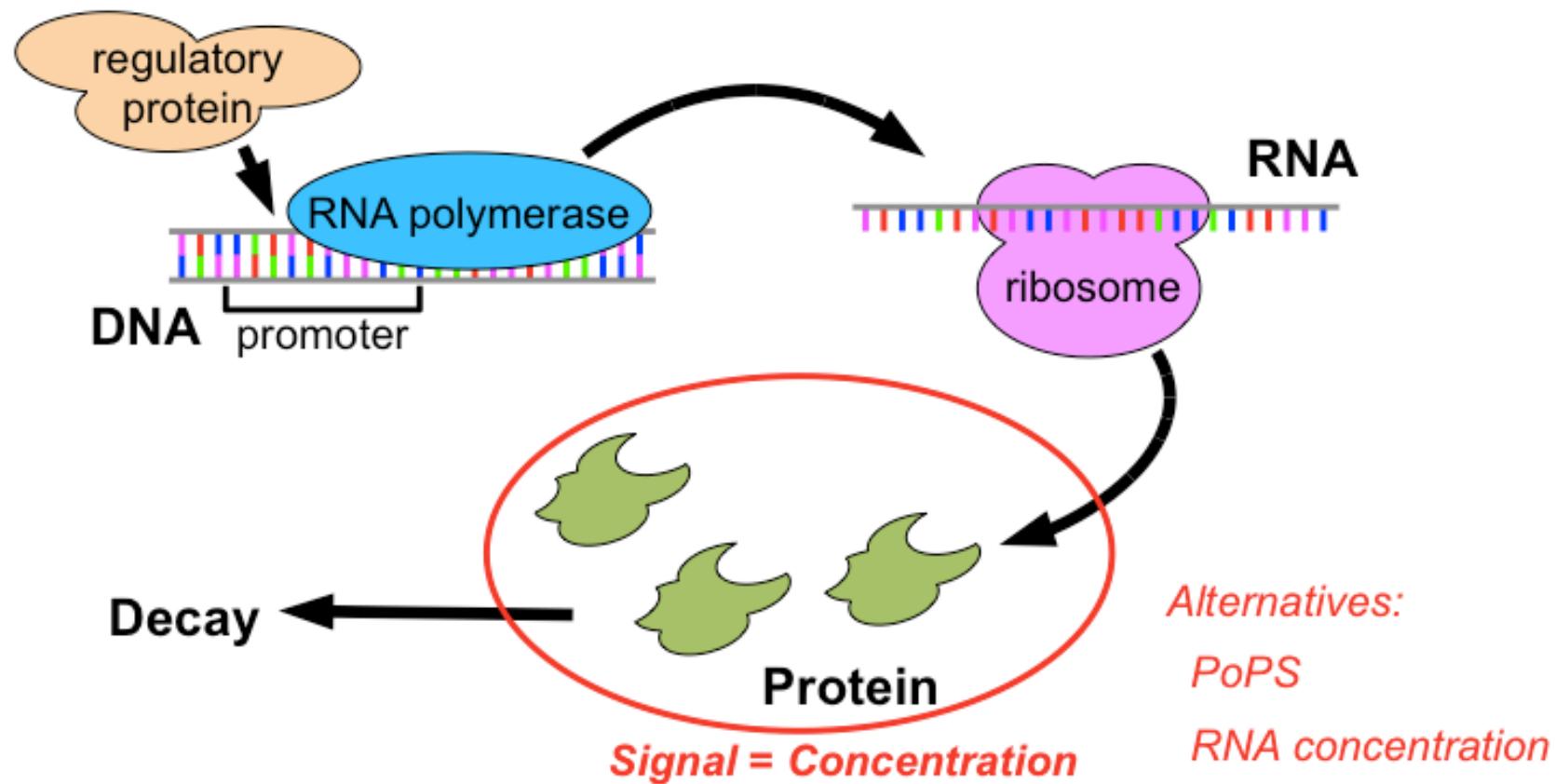


Today's focus: BioCompiler

Compilation & Optimization



Transcriptional Logic

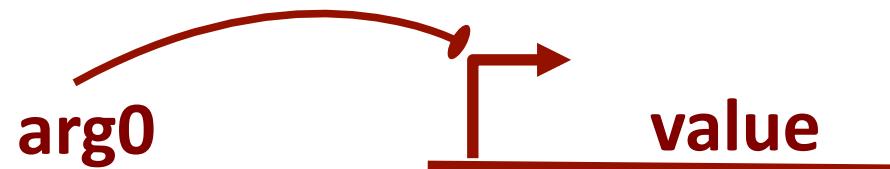


Stabilizes at $decay = production$

Motif-Based Compilation

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

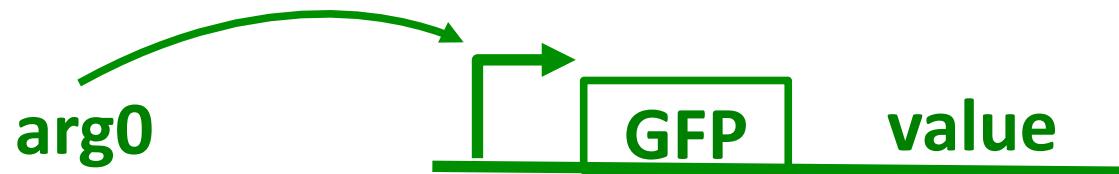
```
(primitive not (boolean) boolean  
:grn-motif ((P high R- arg0 value T)))
```



Motif-Based Compilation

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

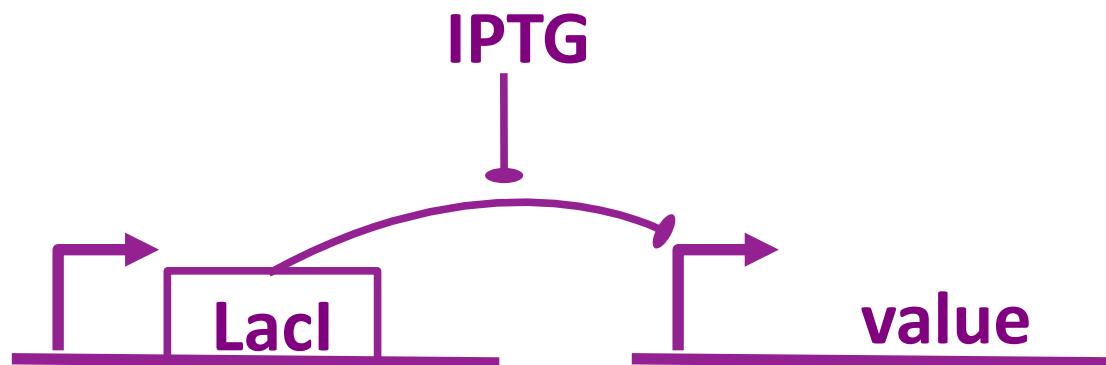
```
(primitive green (boolean) boolean :side-effect  
:type-constraints ((= value arg0))  
:grn-motif ((P R+ arg0 GFP|arg0 value T)))
```



Motif-Based Compilation

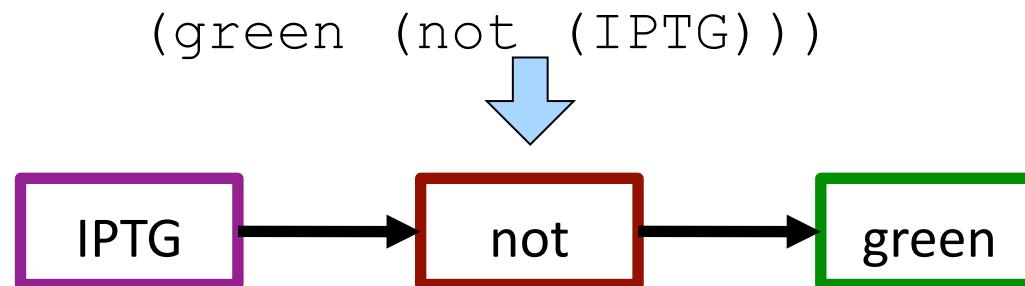
- High-level primitives map to GRN design motifs
 - e.g. logical operators, actuators, **sensors**:

```
(primitive IPTG () boolean  
  :grn-motif ((P high LacI|boolean T)  
    (RXN (IPTG|boolean) represses LacI)  
    (P high R- LacI value T)))
```



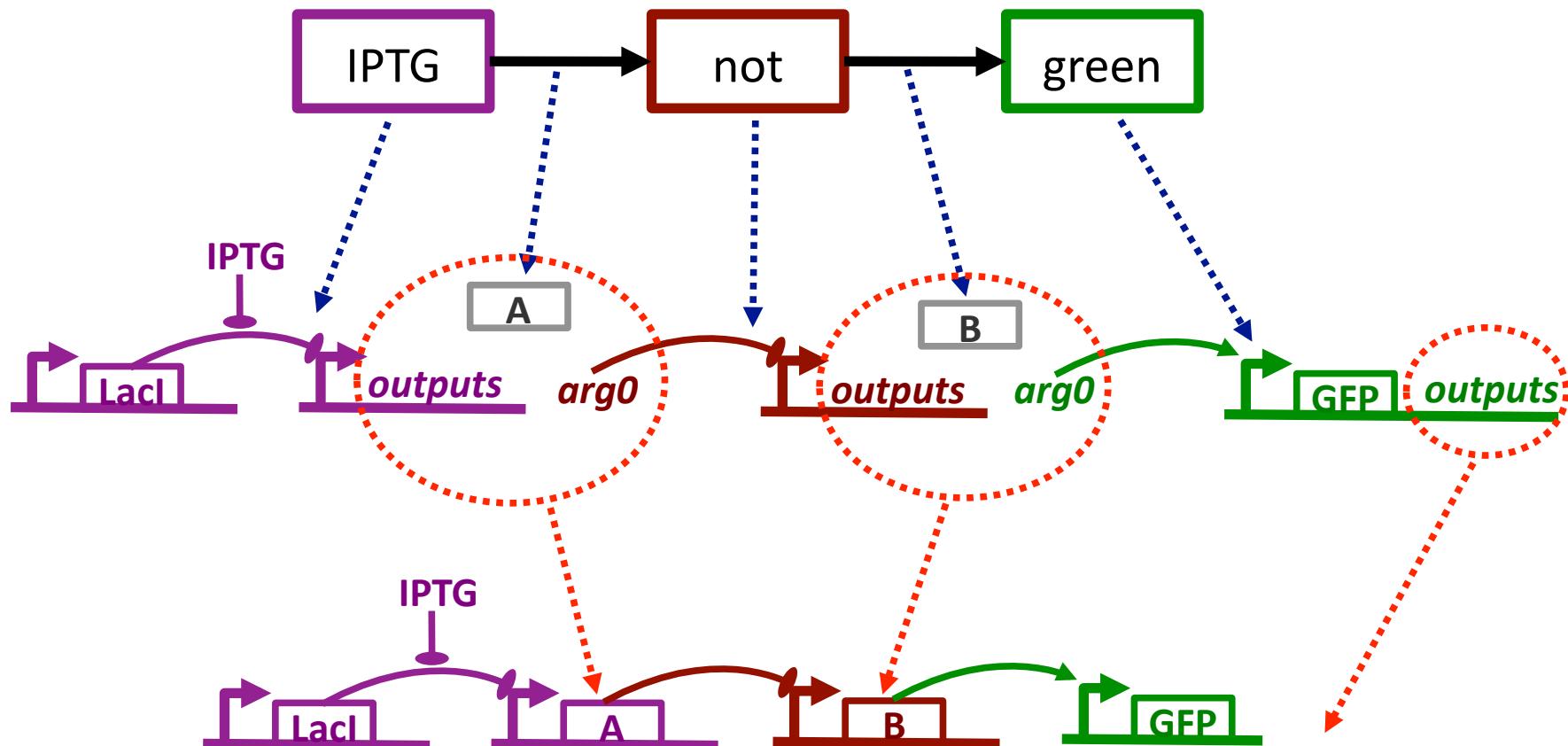
Motif-Based Compilation

- Functional program gives dataflow computation:

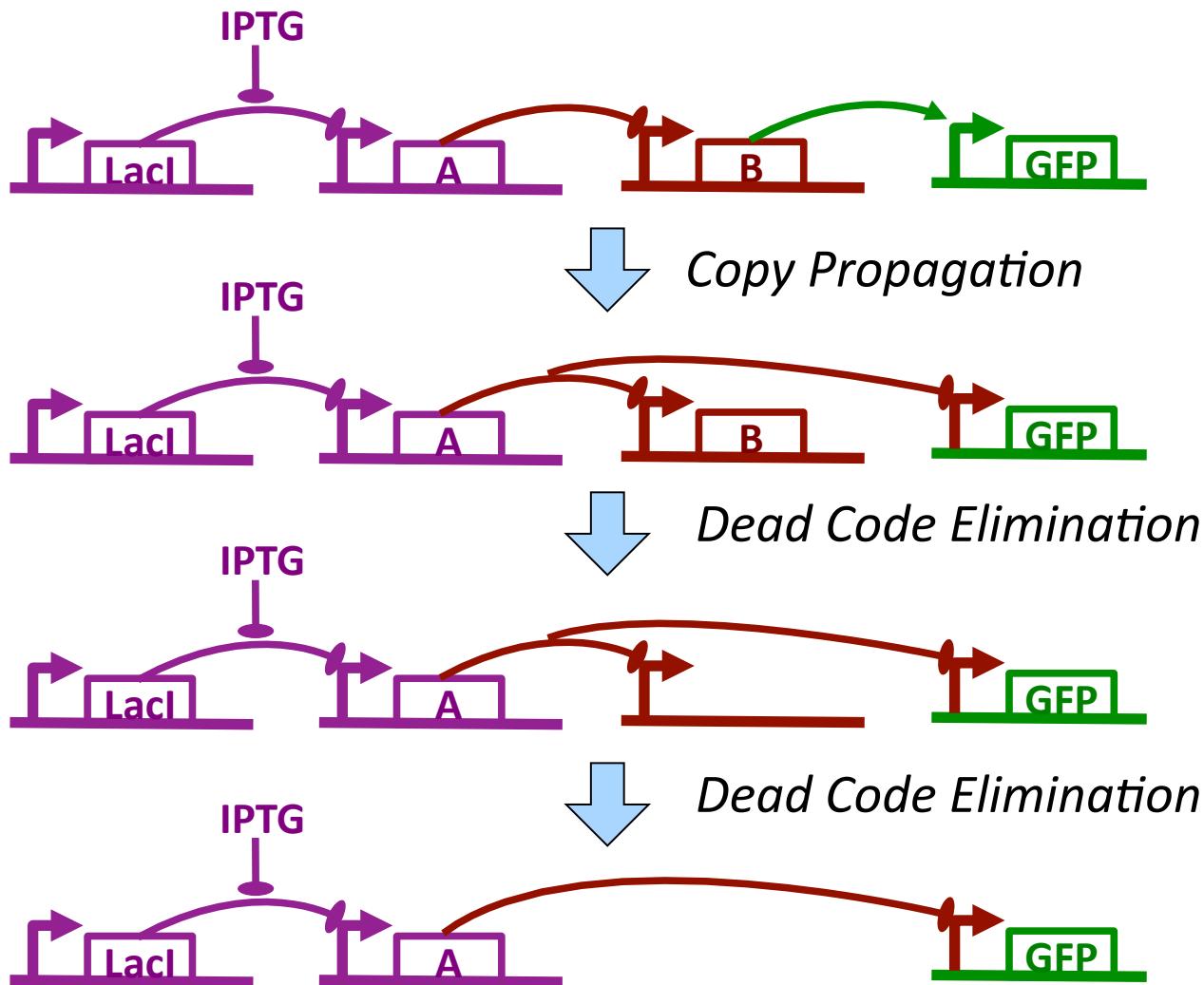


Motif-Based Compilation

- Operators translated to motifs:



Optimization



Complex System: Feedback Latch

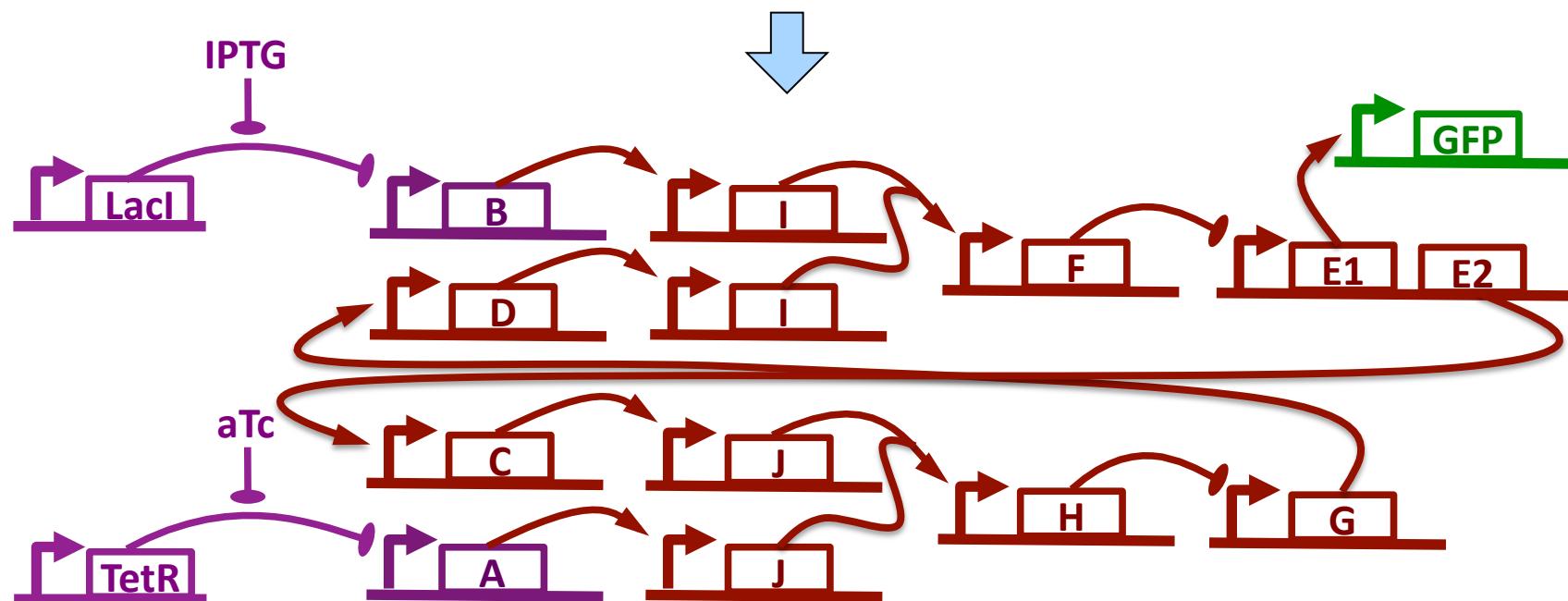
```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar) ))
            (o-bar boolean (not (or s o) )) )
    o) )

(green (sr-latch (aTc) (IPTG)) ) )
```

Complex System: Feedback Latch

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar) ))
            (o-bar boolean (not (or s o) )) )
    o))  

(green (sr-latch (aTc) (IPTG)))
```

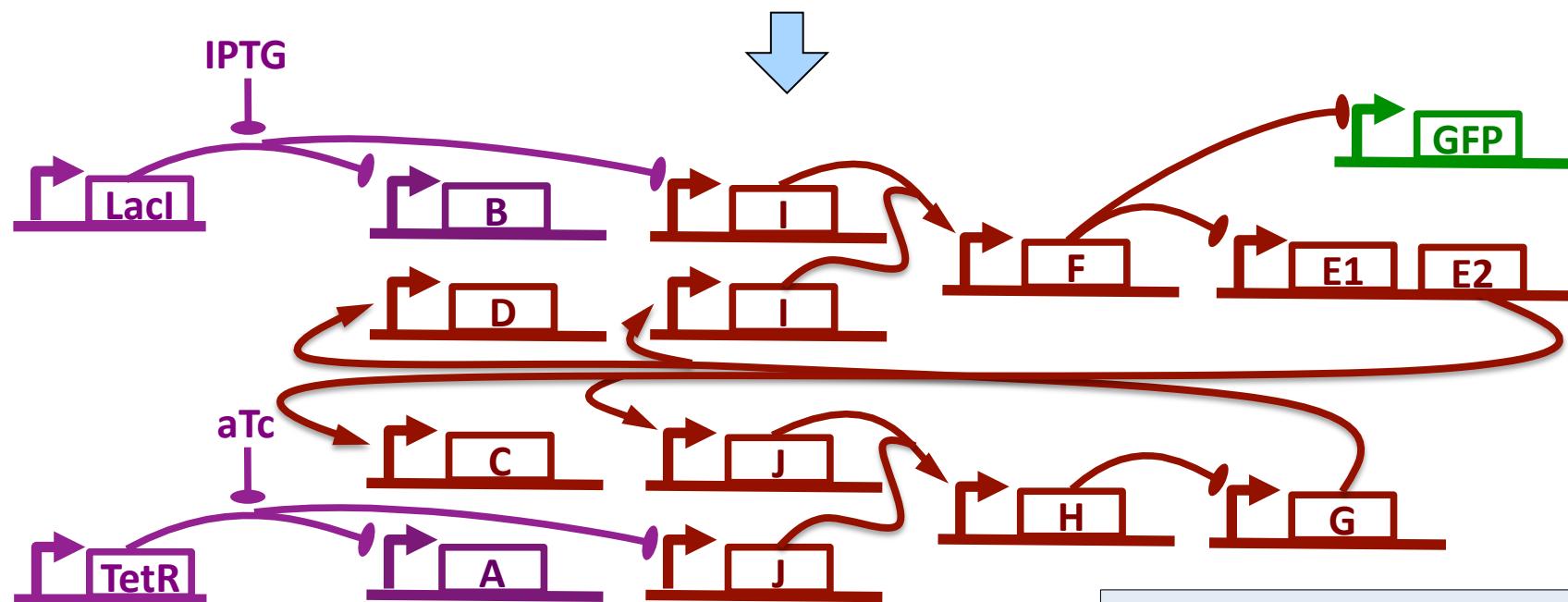


Unoptimized: 15 functional units, 13 transcription factors

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



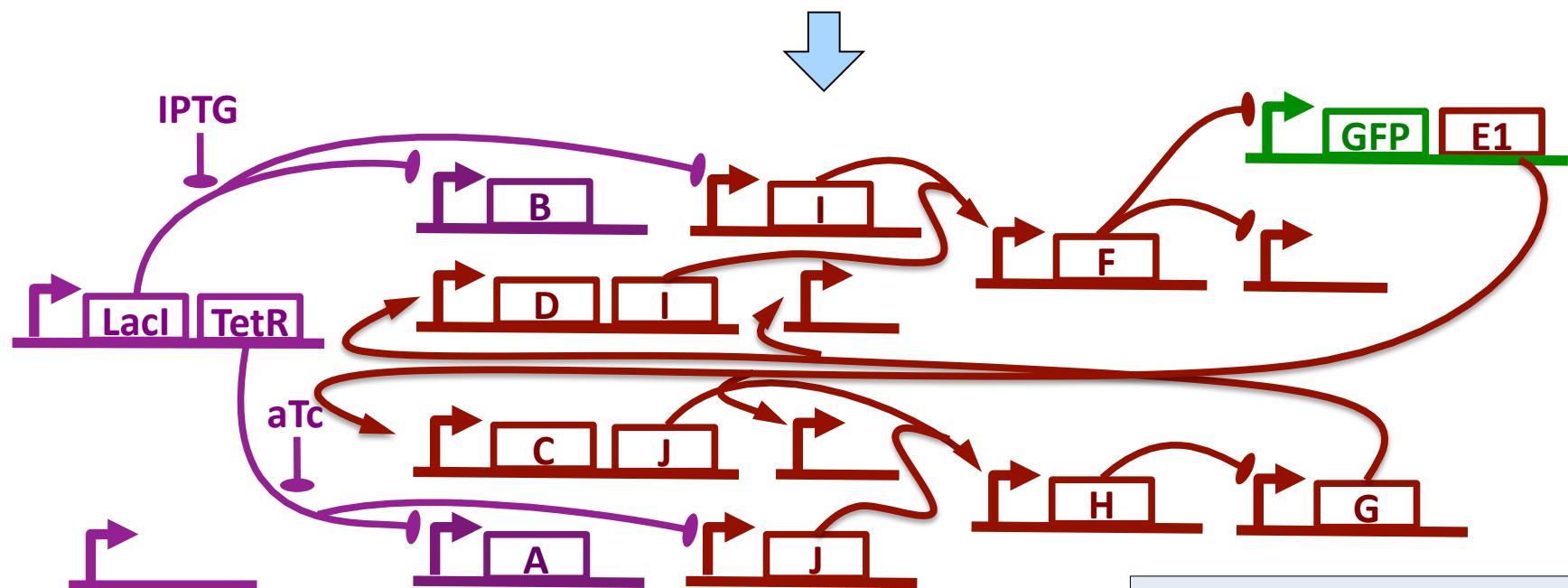
Unoptimized: 15 functional units, 13 transcription factors

*Copy Propagation*₄₇

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



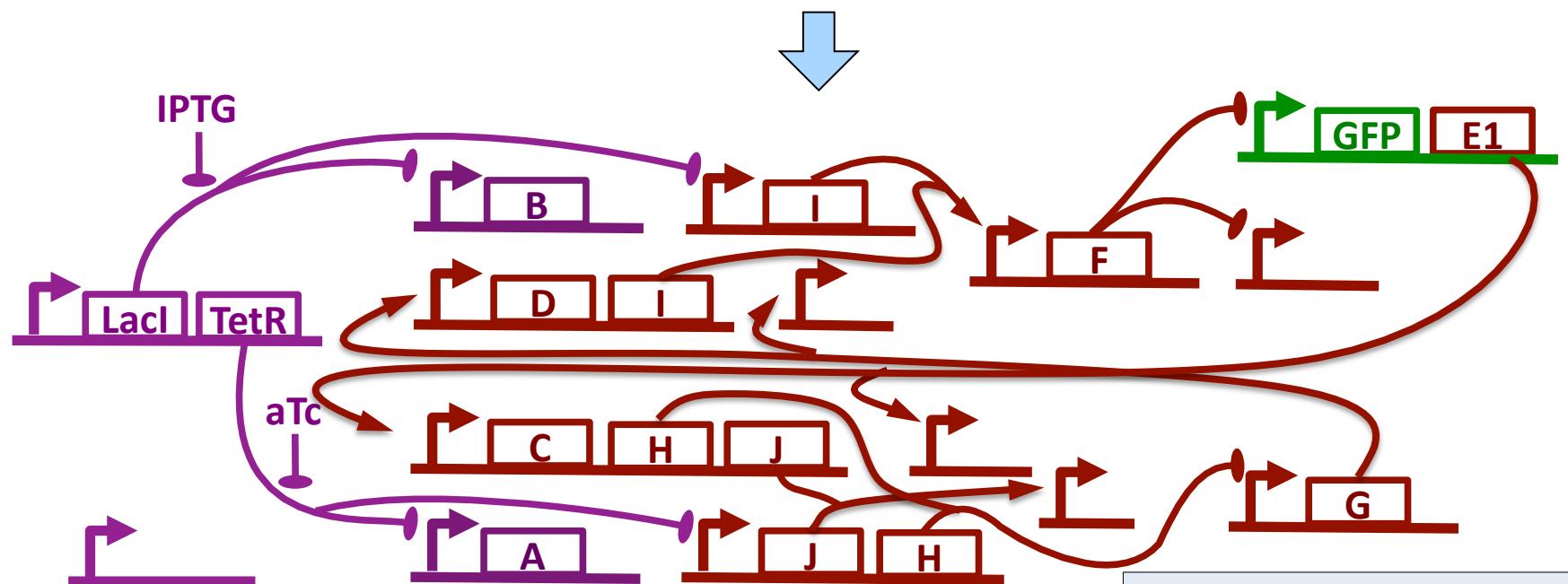
Unoptimized: 15 functional units, 13 transcription factors

Common Subexp. Elim.

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



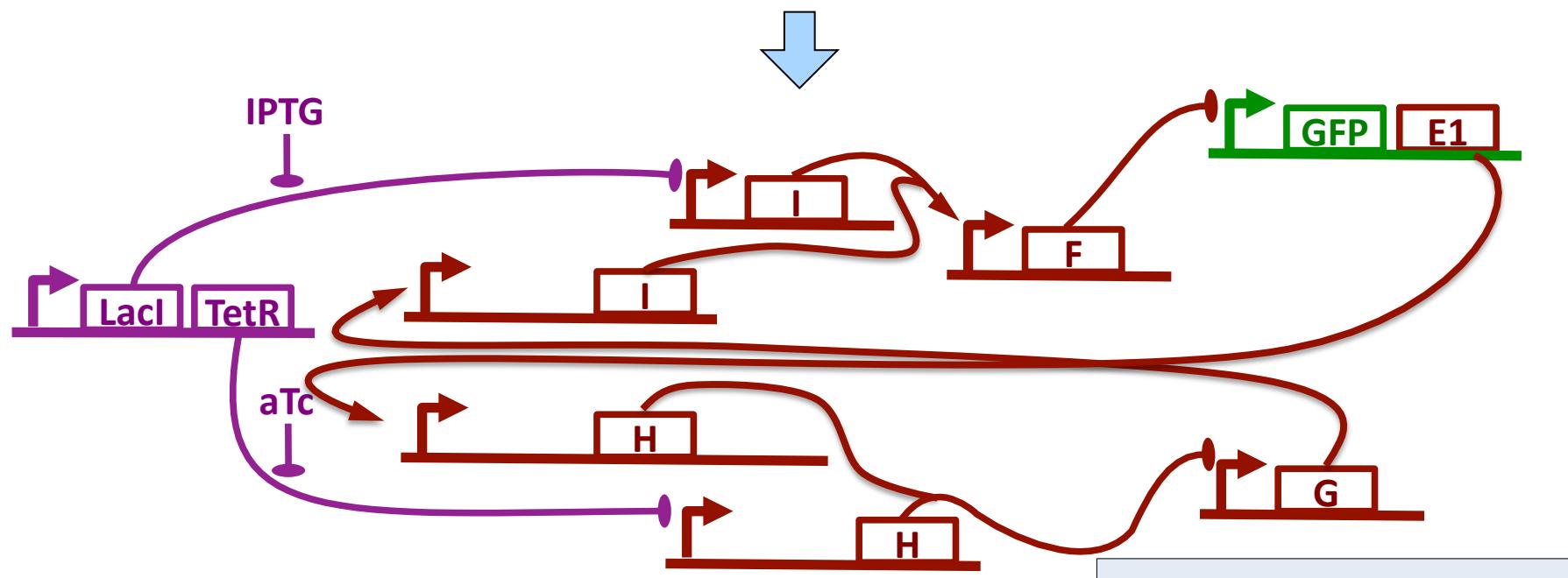
Unoptimized: 15 functional units, 13 transcription factors

NOR Compression 49

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



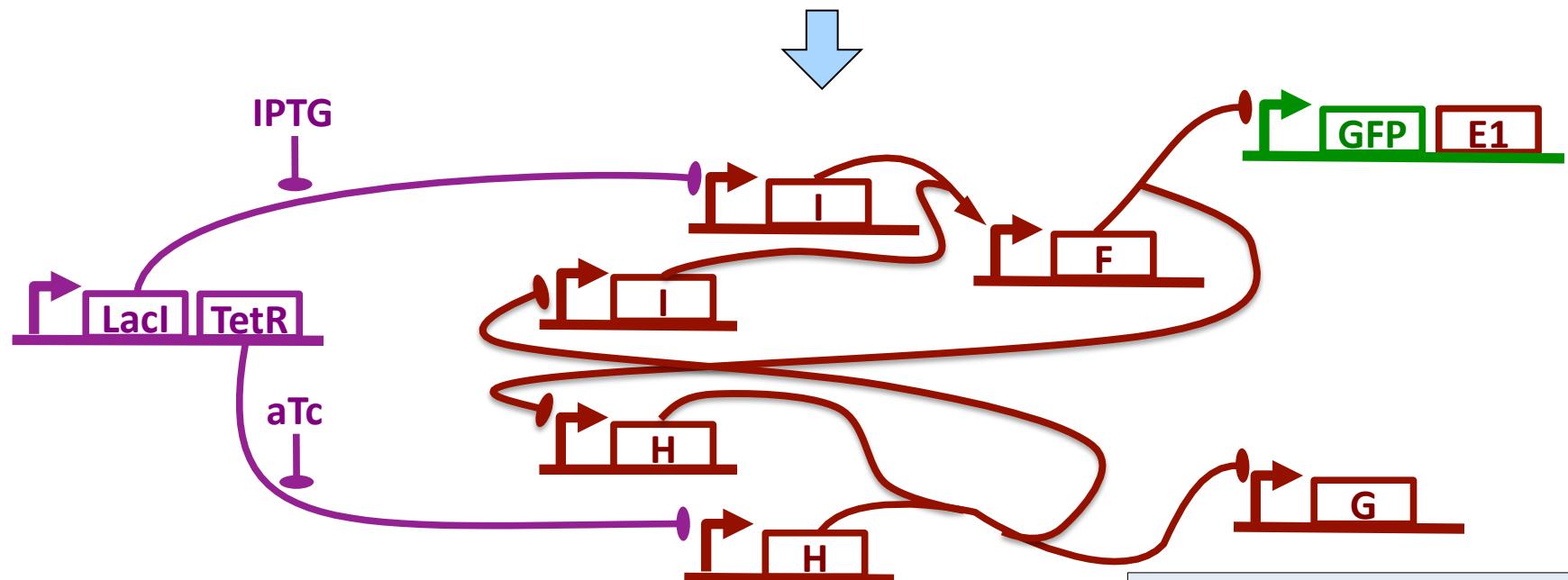
Unoptimized: 15 functional units, 13 transcription factors

Dead Code Elimination 50

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



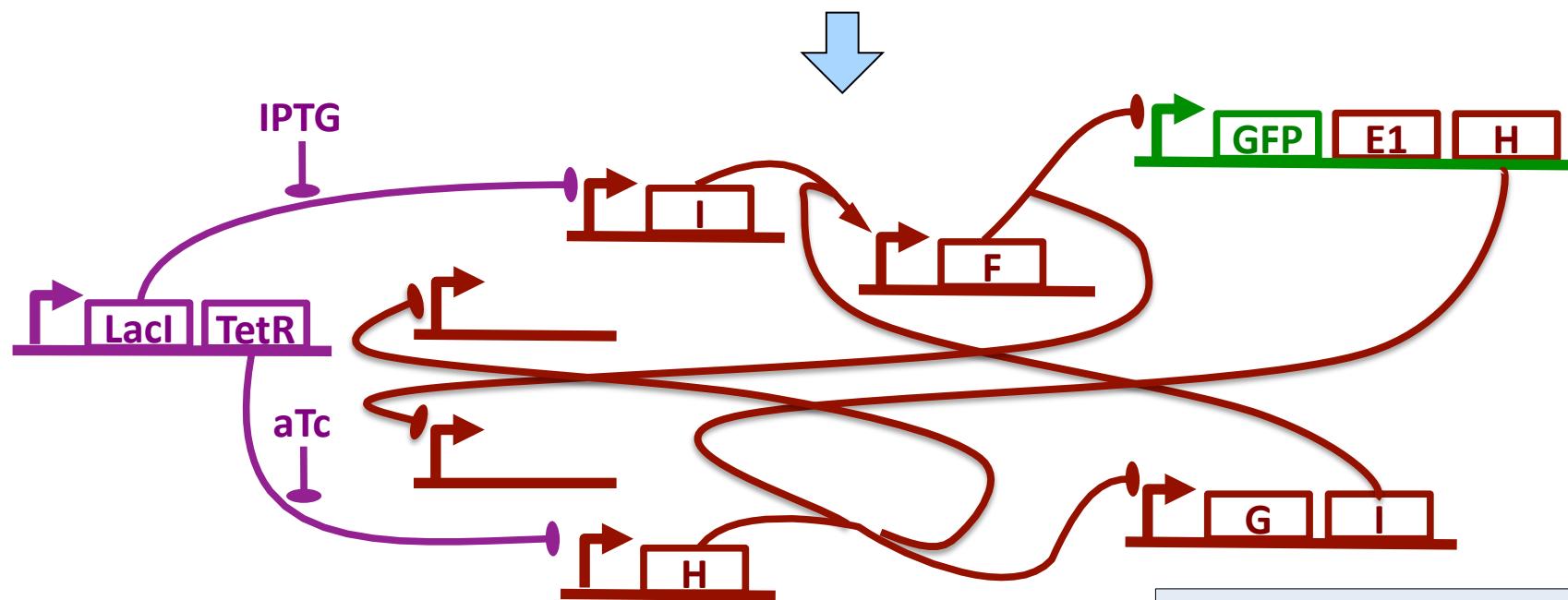
Unoptimized: 15 functional units, 13 transcription factors

Copy Propagation 51

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



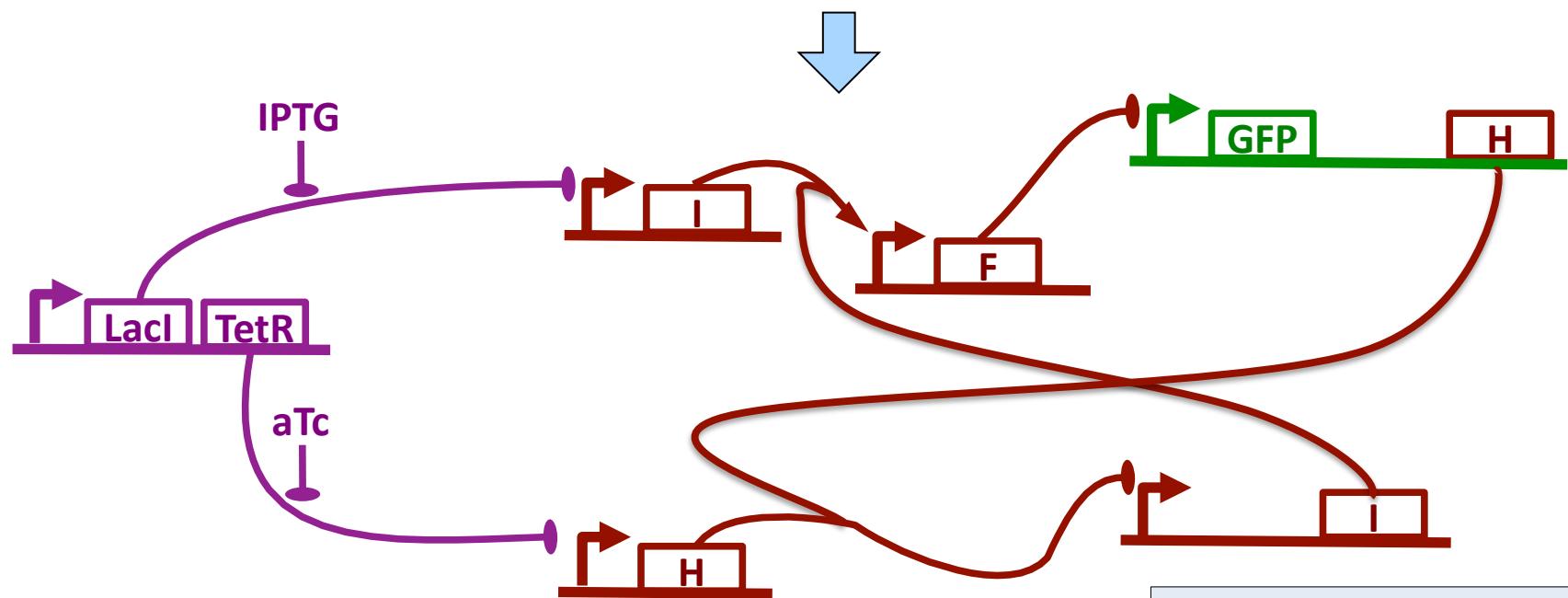
Unoptimized: 15 functional units, 13 transcription factors

Common Subexp. Elim.

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))
```

(**green** (**sr-latch** (**aTc**) (**IPTG**)))



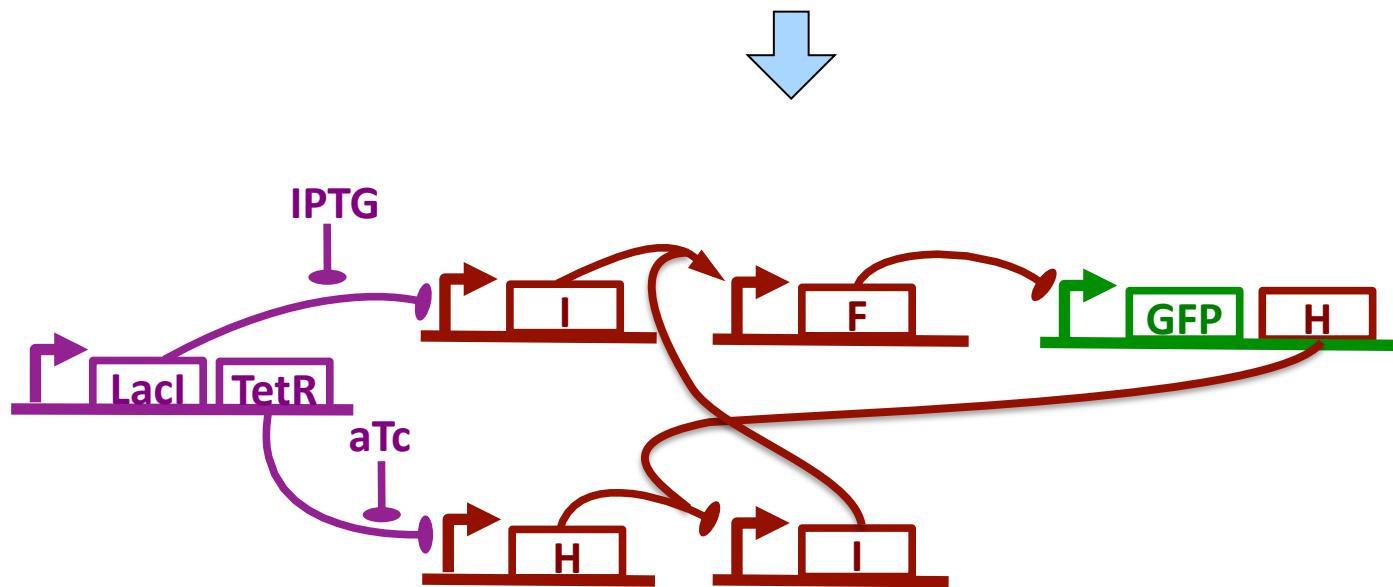
Unoptimized: 15 functional units, 13 transcription factors

Dead Code Elimination 93

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```

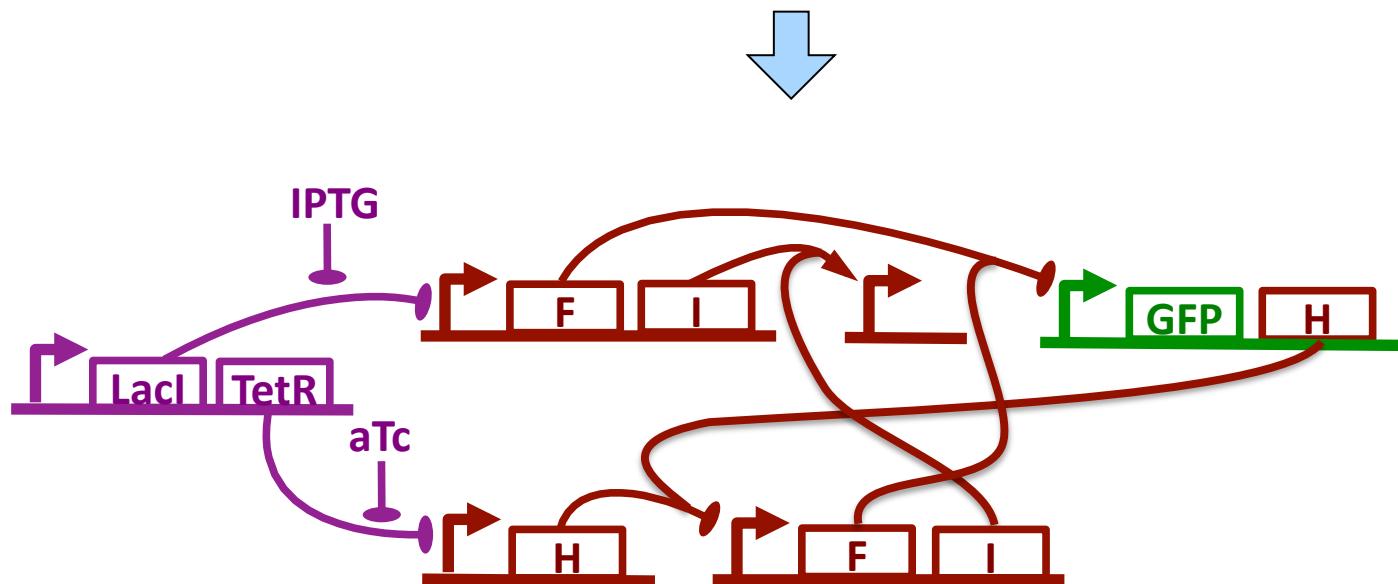


Unoptimized: 15 functional units, 13 transcription factors

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



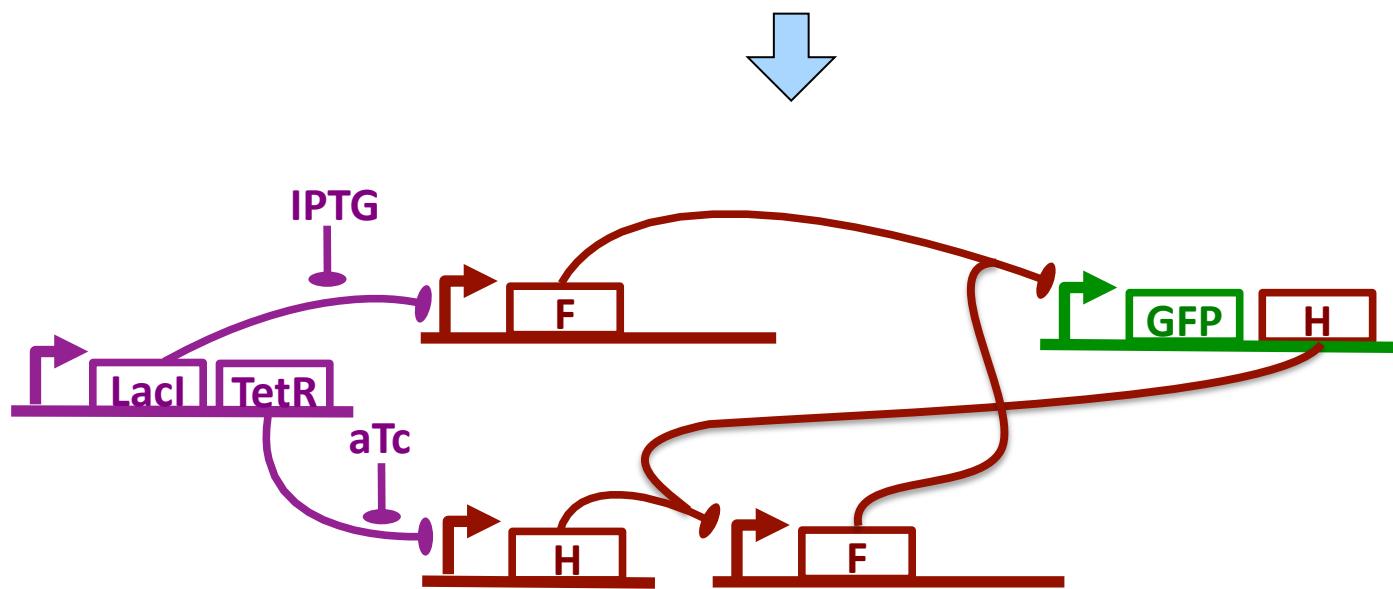
Unoptimized: 15 functional units, 13 transcription factors

NOR Compression₅₅

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



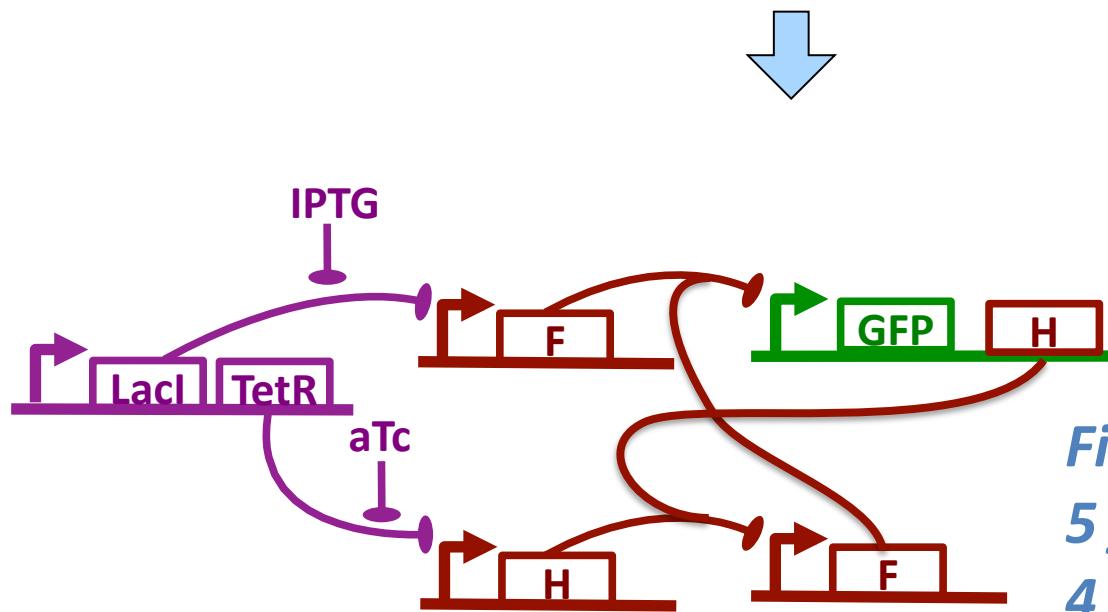
Unoptimized: 15 functional units, 13 transcription factors

Dead Code Elimination 90

Optimization of Complex Designs

```
(def sr-latch (s r)
  (letfed+ ((o boolean (not (or r o-bar)))
            (o-bar boolean (not (or s o))))
    o))  

(green (sr-latch (aTc) (IPTG)))
```



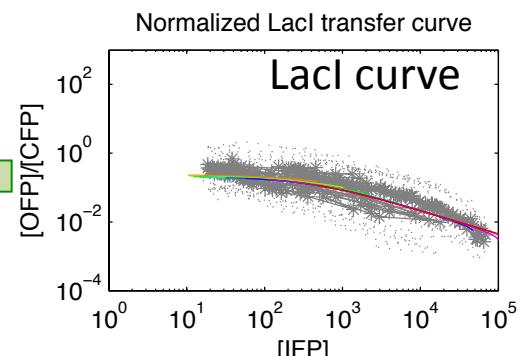
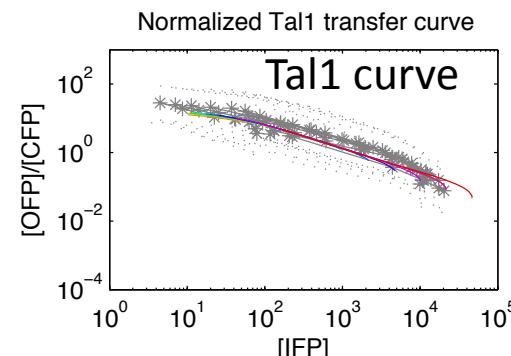
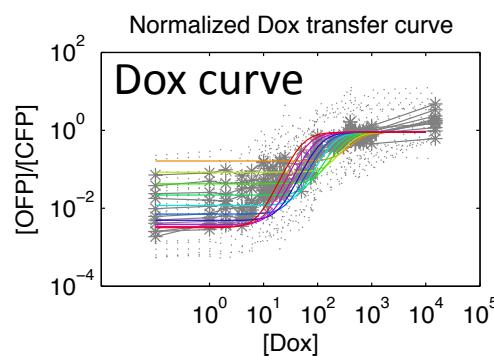
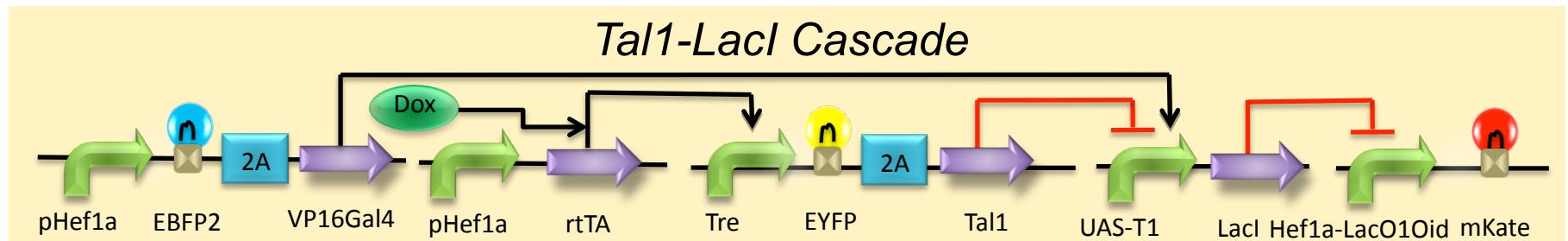
Final Optimized:
5 functional units
4 transcription factors

Unoptimized: 15 functional units, 13 transcription factors

Compilation & Optimization Results:

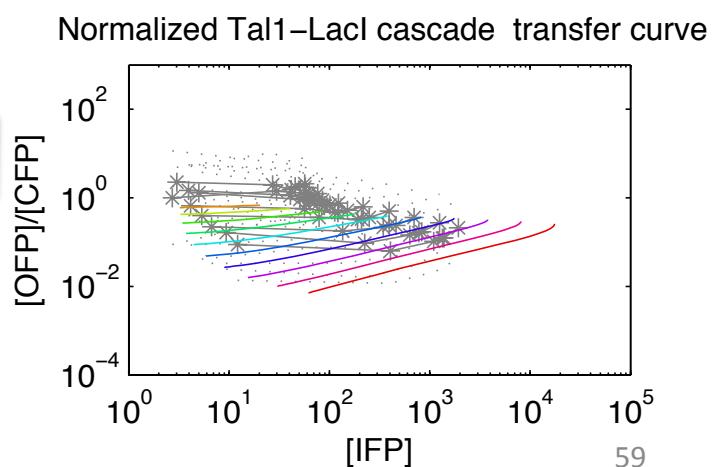
- Automated GRN design for arbitrary boolean logic and feedback systems
 - Verification in ODE simulation
- Optimization competitive with human experts:
 - Test systems have 25% to 71% complexity reduction
 - Optimized systems are often homologous with hand designed networks

Realization with Characterized DNA parts

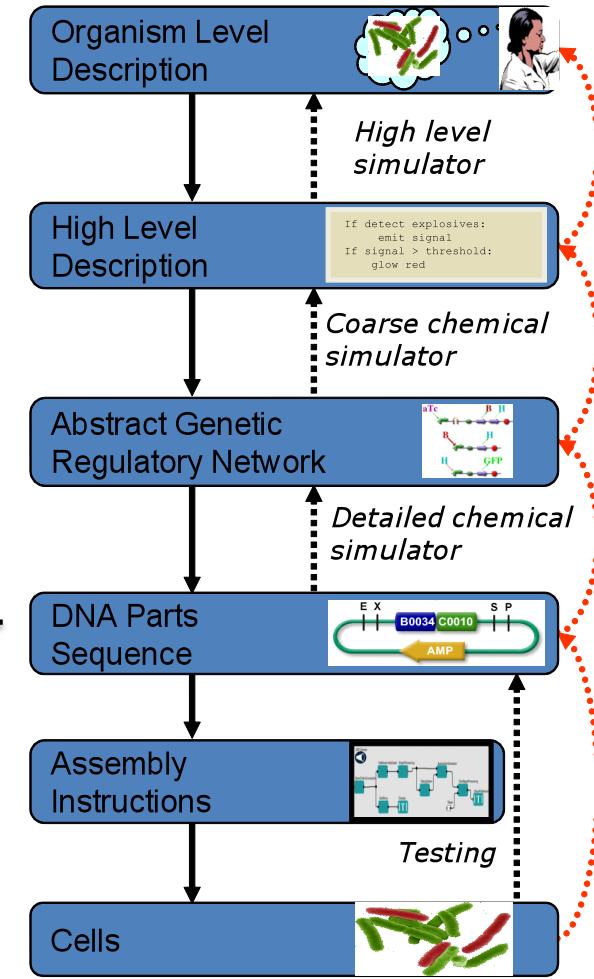
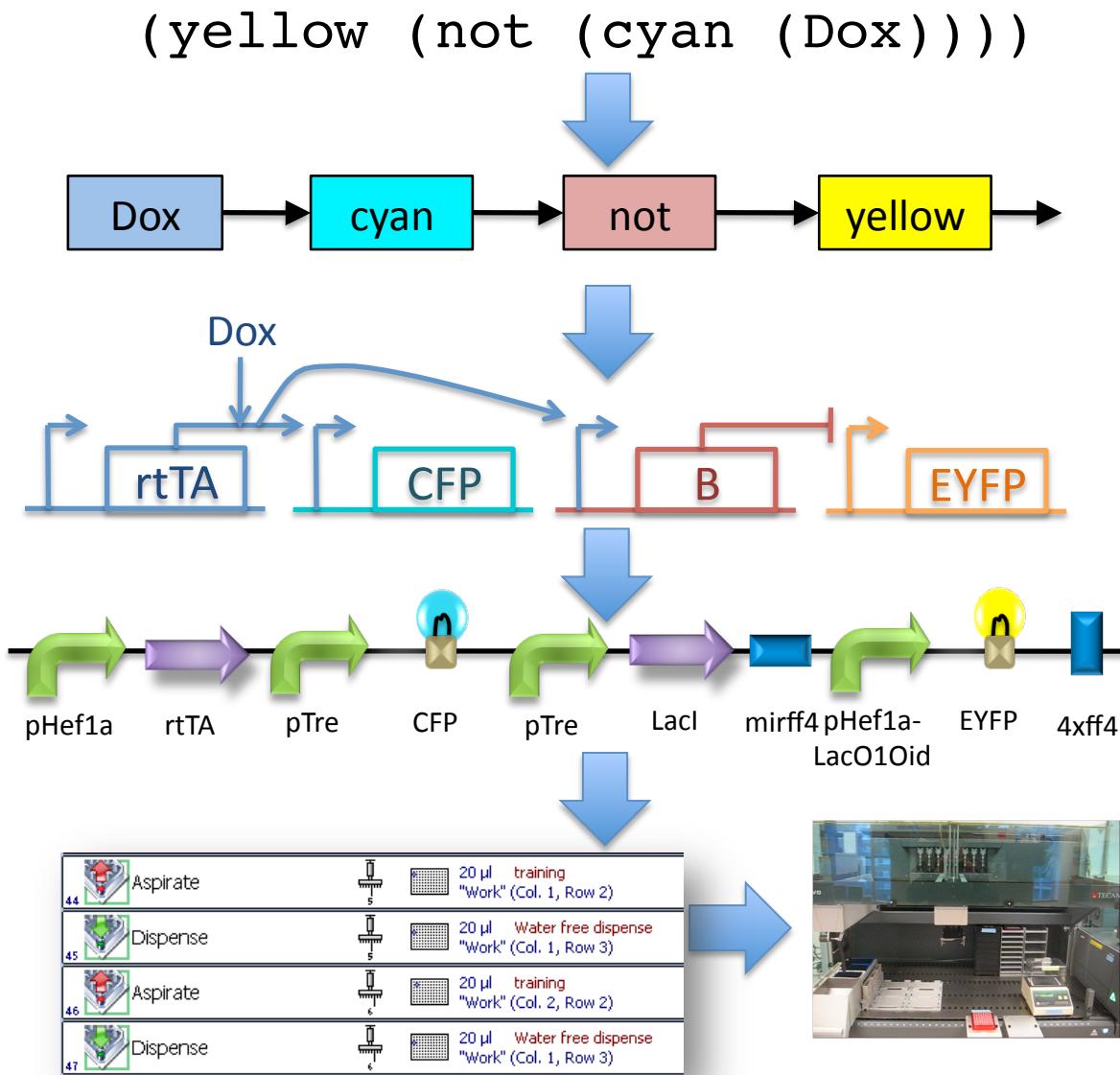


- Model → Experiment:
 - Low copy: no effect → no effect
 - High copy: 30x → 10x
 - Gradual transition → gradual transition
 - Max ~ 10^0 → 3x higher – 2A effect?

Predict

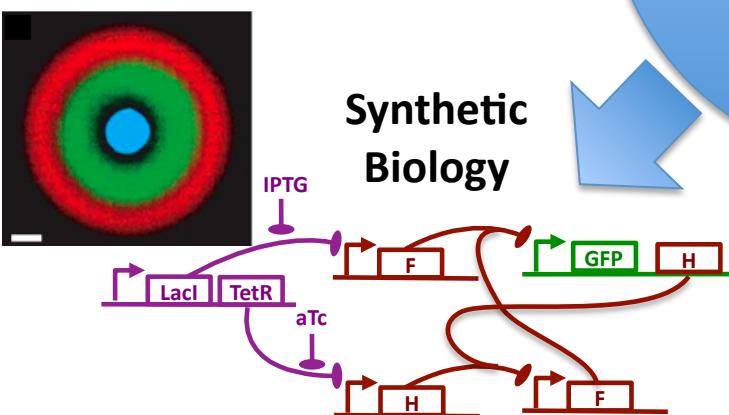
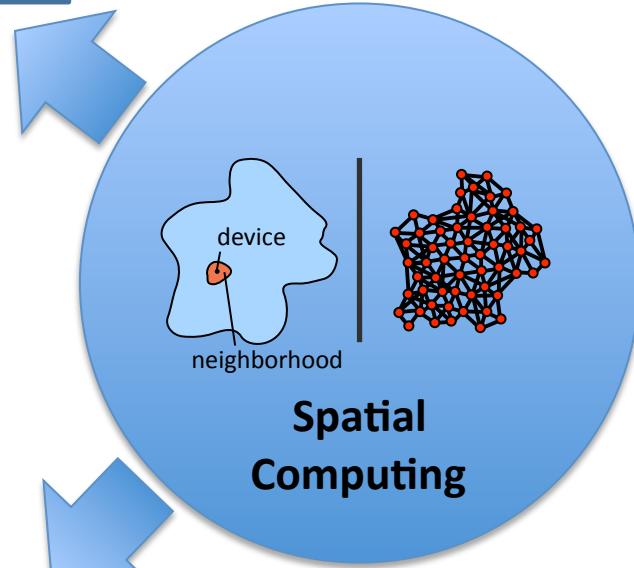


Onward Through the Tool-Chain...



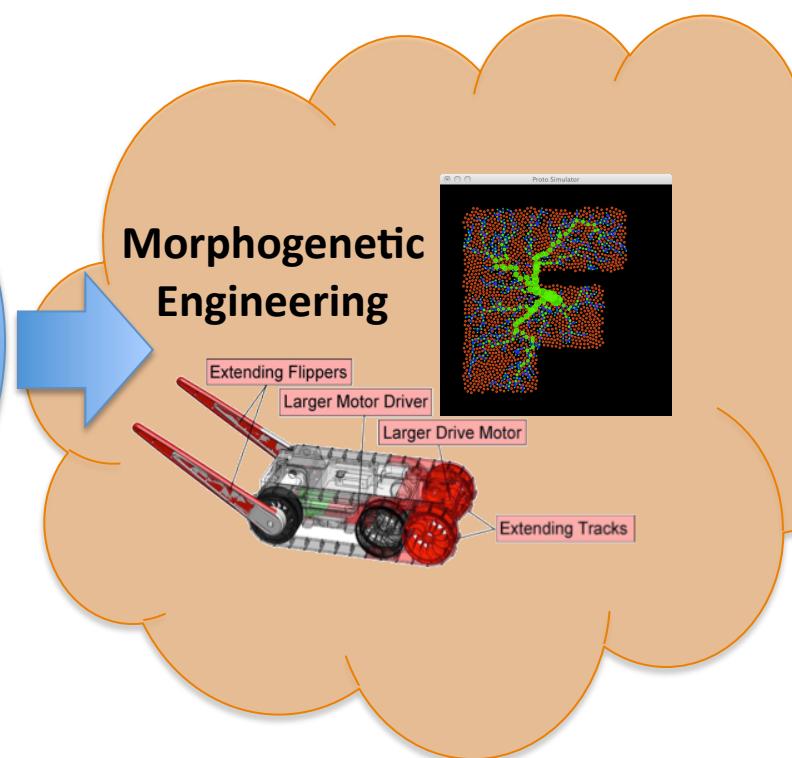


Distributed Power Demand Response

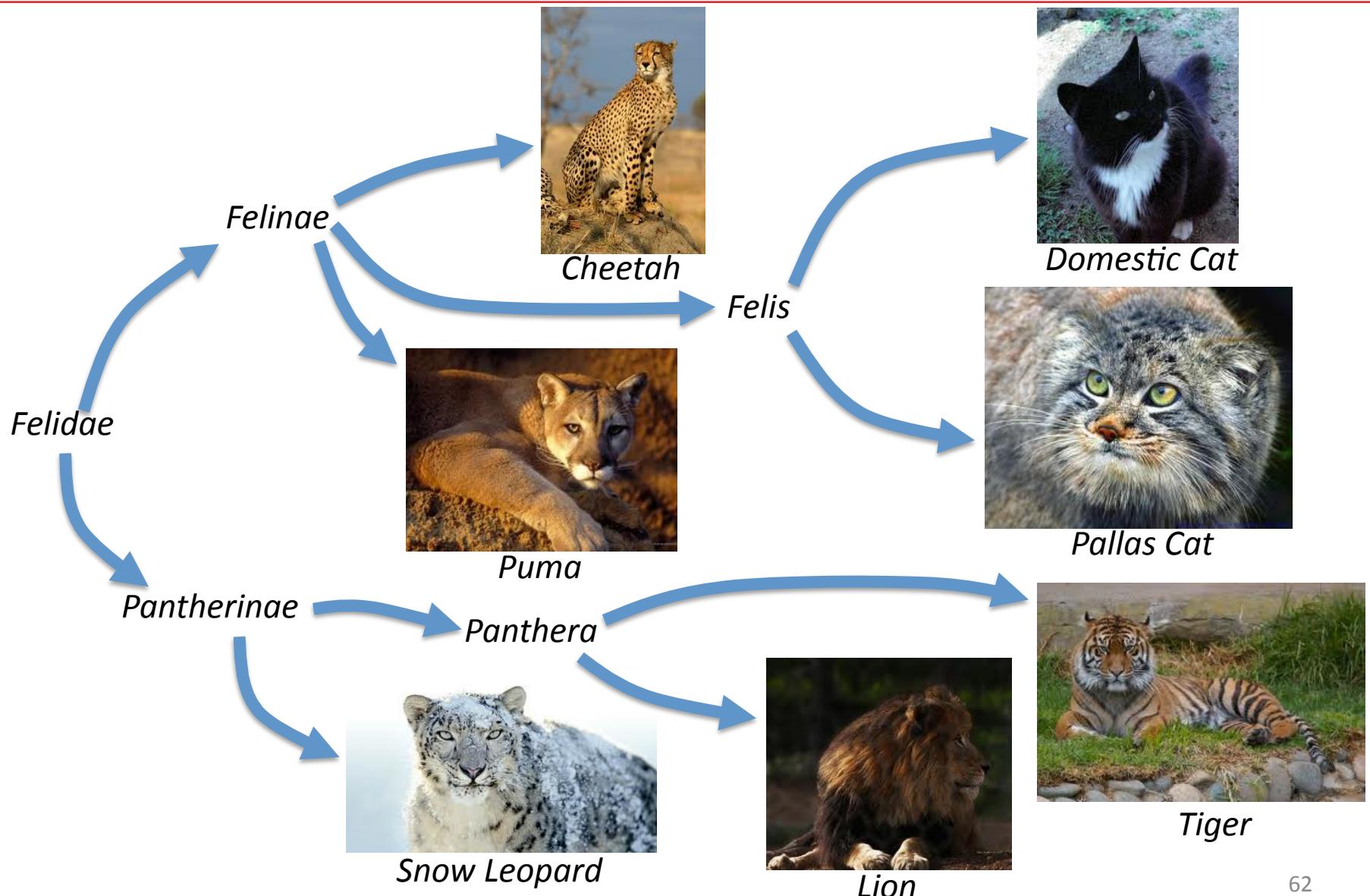


Synthetic Biology

How can the parts of a design work together to adapt it to new uses?



Morphogenesis enables natural variation



A phylogeny of engineered systems?



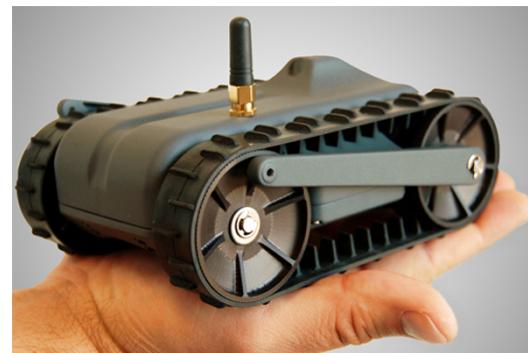
PackBot



SUGV



Warrior



LANdroid



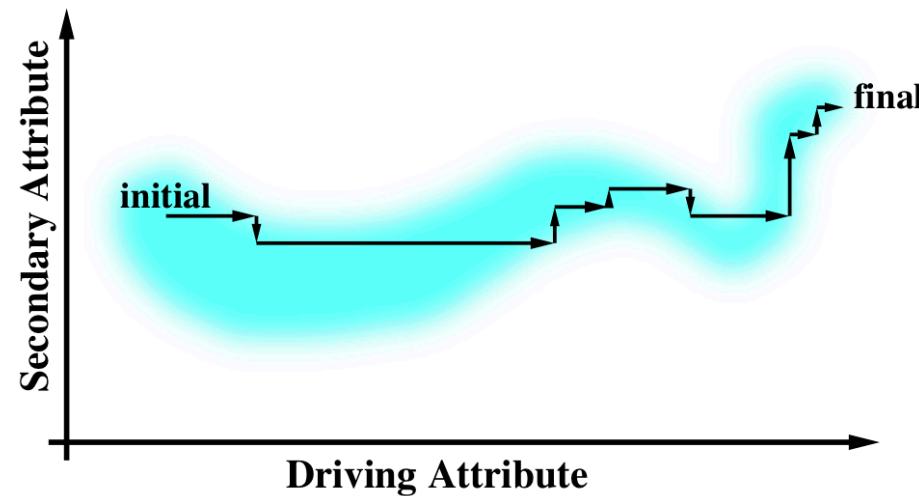
miniDroid

Functional Blueprints

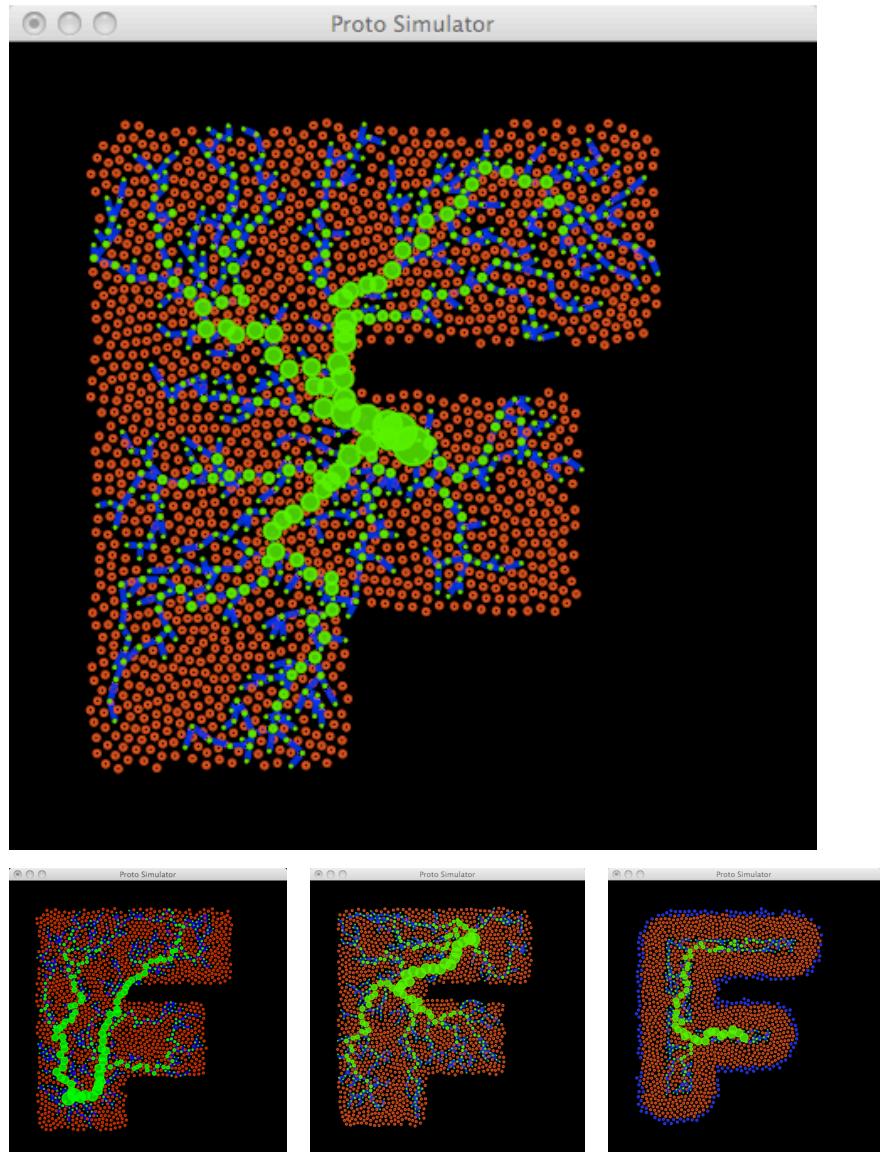
1. Functional behavior that degrades gracefully
2. Metric for degree and direction of stress
3. Incremental adjustment program for stress relief
4. Initial viable system

Idea: incremental motion in viable design space

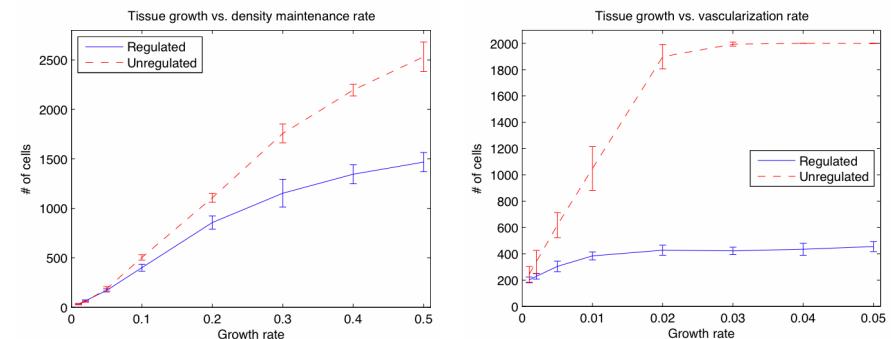
Abstract functional blueprint networks converge reliably and >100x faster than GAs



Example: “Cartoon” Vascularization



- Functional blueprint model of vascularization
 - Stress: oxygen, elastic stress
 - Adjustment: leaking, vessel grow/shrink
- Red cells are healthy, blue cells are oxygen-deficient
- Can model vasculatization and density co-regulation



What if your robot can climb stairs...



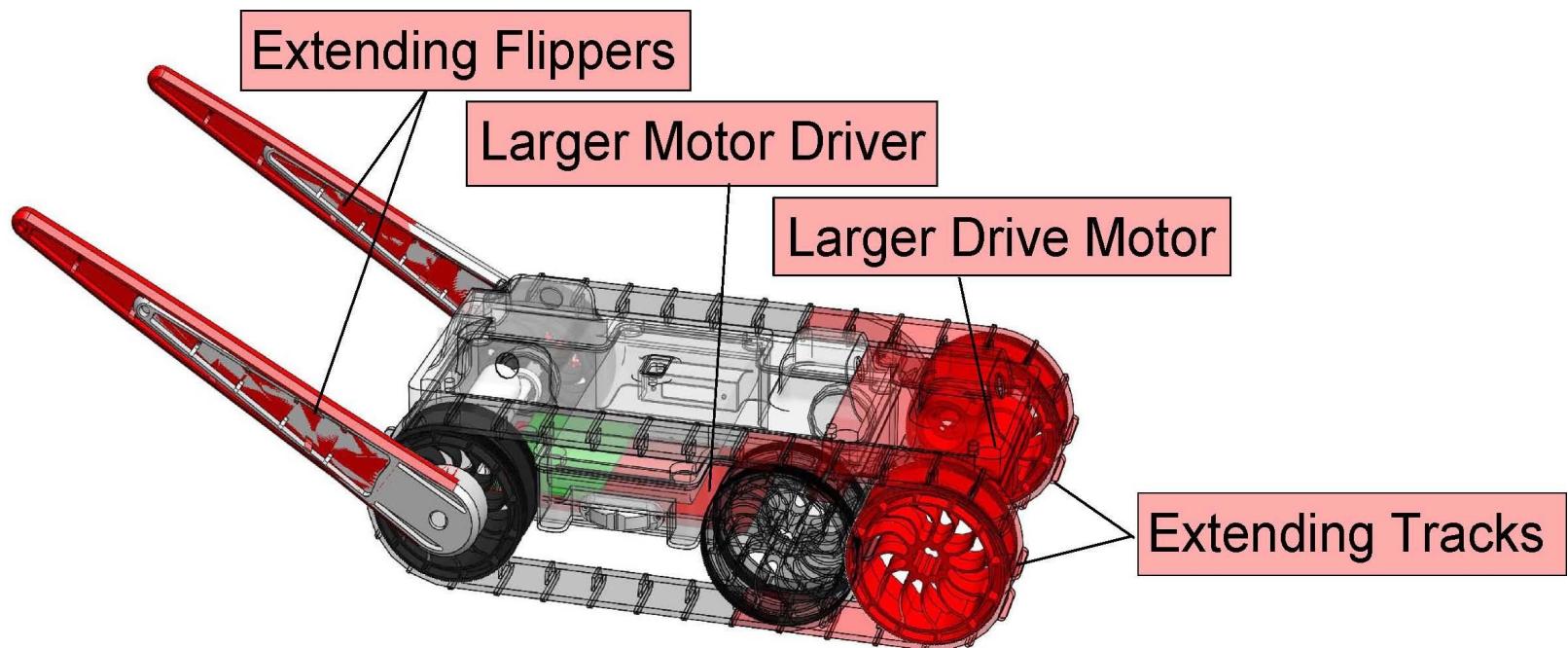
... but you want to check cars for IEDs?

Raytheon
BBN Technologies



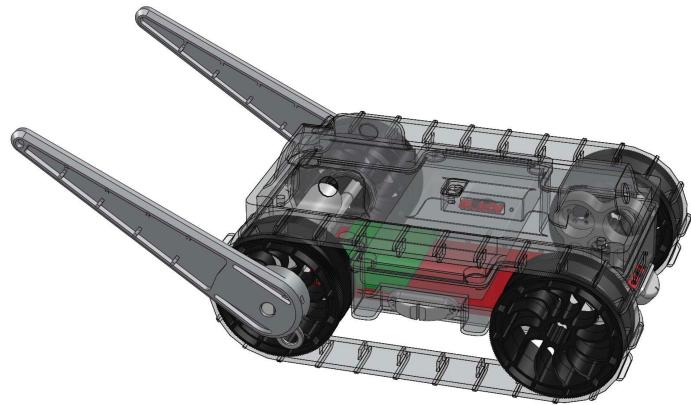
Problem: Redesign Complexity

Even “simple” robots require careful design of many interacting components...



... and small changes have large consequences.

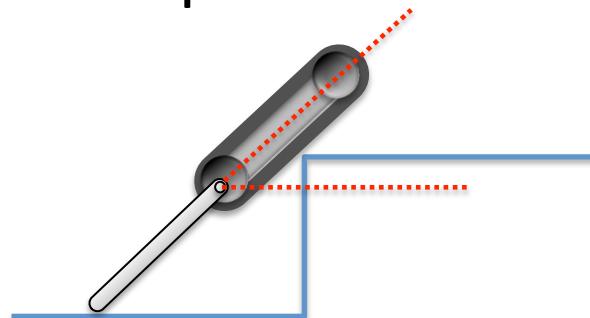
iRobot miniDroid Functional Blueprints



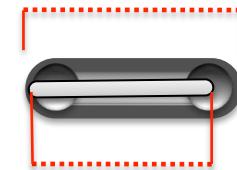
Functional blueprints requirements:

- Behavior that degrades gracefully
- Metric for stress degree & direction
- Incremental adjustment program
- Initial viable design

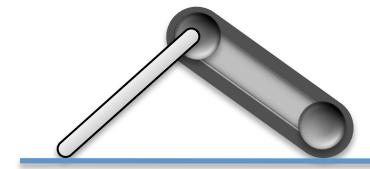
Examples:



Step Climbing
(via ascent angle)

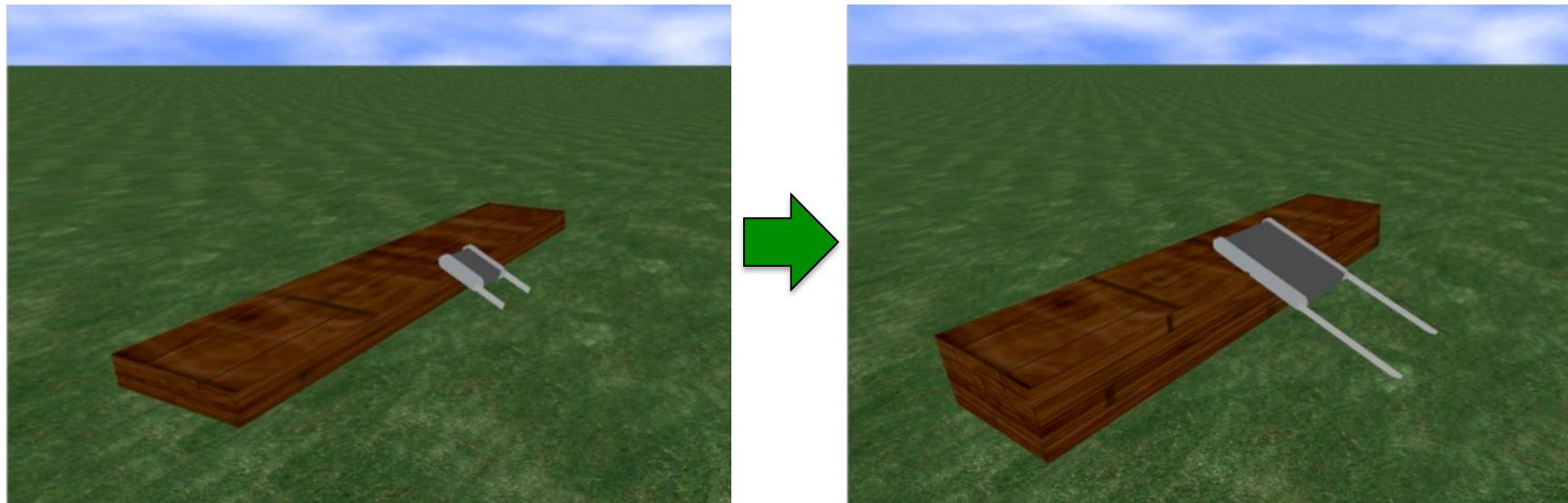


Flipper/Body Ratio



Self-Righting
(via torque/mass)

5x Variation Driven by Step Height



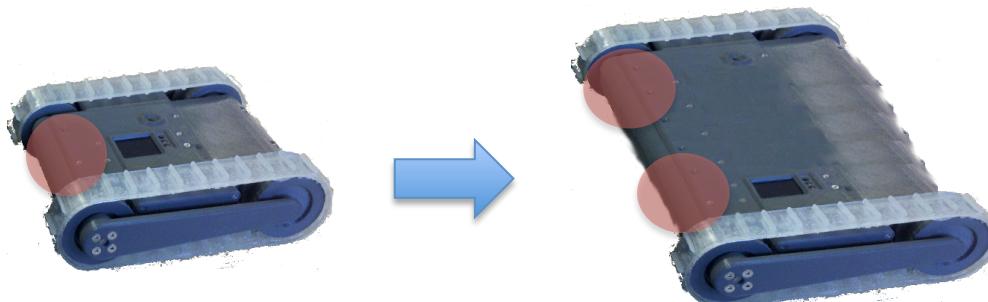
Adjusting Component Layout

- Functional blueprints control key attributes
 - The majority of design parameters are implicit!
- How are components modified and placed?



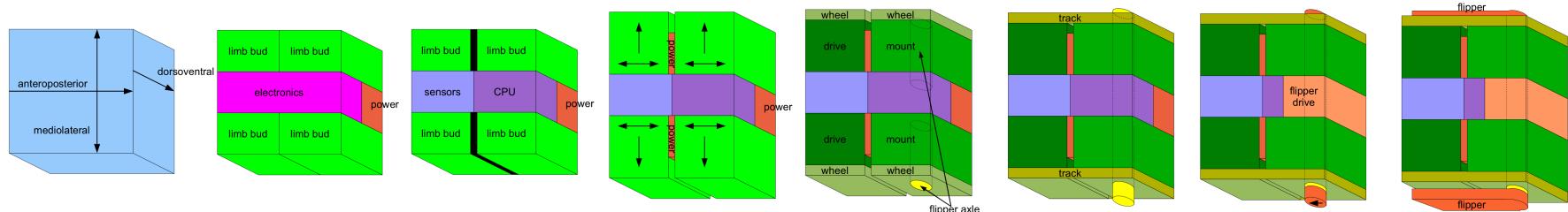
– e.g., Which way does the flipper extend? Where does the CPU go inside the robot body?

- What about changes to number or topology?



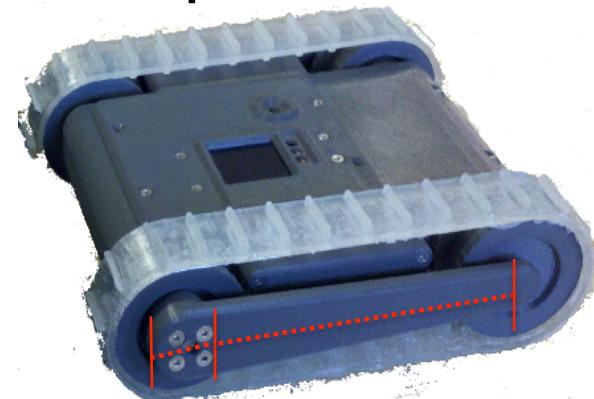
Morphogenesis resolves design constraints

Manifold evolution based on biological development



Implicit parameters specified through geometric relationships developed between components

- Abstraction of components
- Reduced dimensionality
- Greater design flexibility



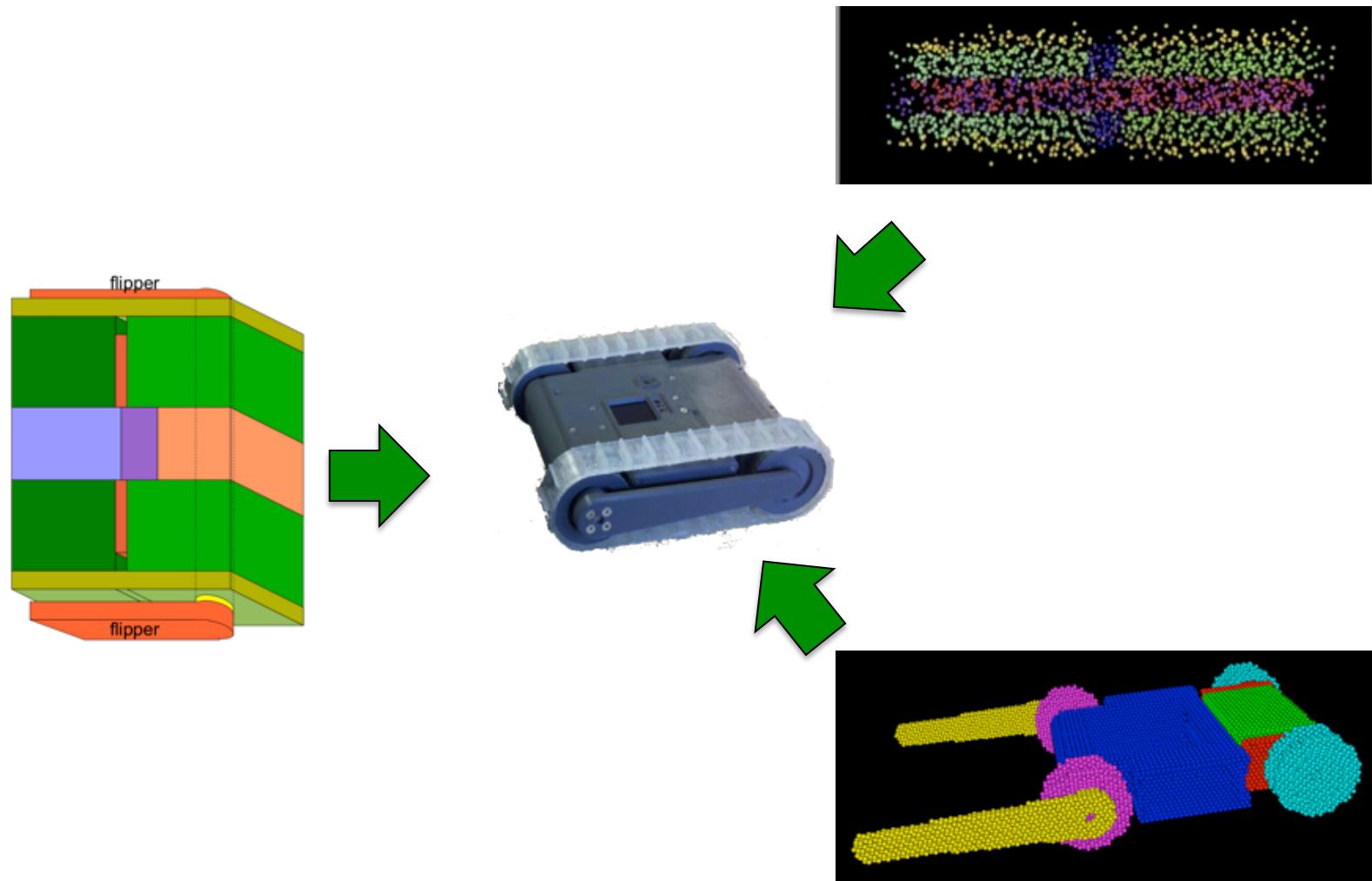
Flipper Length = Axe + Extension

Developmental Programs

Three levels of detail:

- Tissue-level rule definition and evaluation
- Proto cellular models of program plasticity
 - Concurrent & blended execution of rules
 - Manifold operators distort to match the structure of the space on which they execute
- Soft-body physics for tissue interaction
 - Adhesion lets components maintain spatial relations
 - Plasticity lets components “push” around and through each other

Next Step: Model Integration



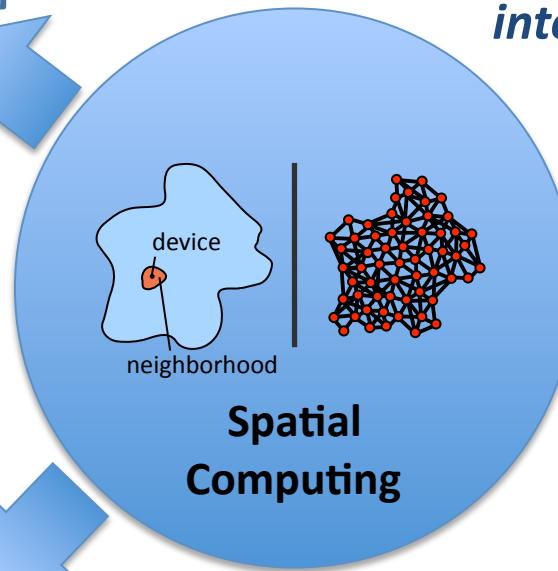
Summary



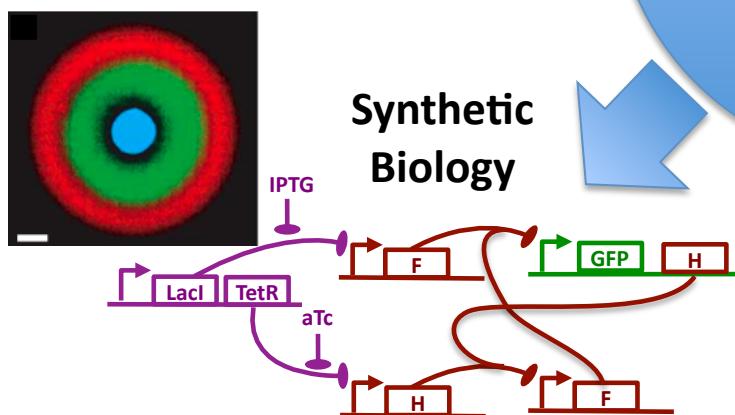
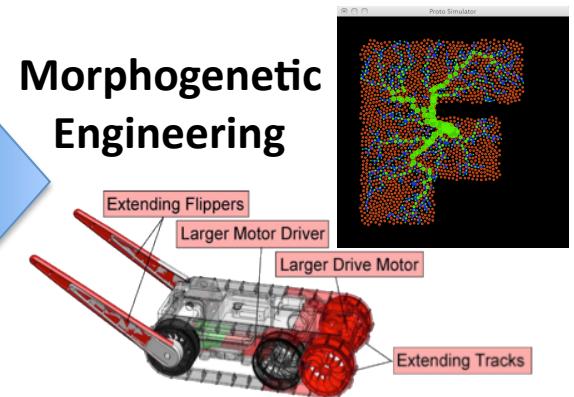
Distributed Power Demand Response

The Amorphous Medium abstraction uses manifolds to simplify programming of scalable, robust behavior on space-filling networks

Proto's four families of space/time operations let geometric aggregate descriptions compile into approximating local actions



Morphogenetic Engineering



Synthetic Biology

Spatial abstractions can help us solve complex problems, from networks to biology and beyond...

Proto is available

<http://proto.bbn.com/>
(or google “MIT Proto”)

- Includes libraries, compiler, kernel, simulator, platforms
- Licensed under GPL with linking exception

Acknowledgements:

Raytheon
BBN Technologies

Aaron Adler (co-PI)
Brett Benyo
Jeff Berliner
Jeff Cleveland
Jessica Lowell
Gretchen Markiewicz
Hala Mostafa
Rick Schantz
Kyle Usbeck
Fusun Yaman
Susan Katz, Joseph Loyall (PMs)

Interns:
Katie McGuire, Taylor Campbell



Ron Weiss (co-PI)
Jonathan Babb
Noah Davidsohn
Tim Shepard
Nelson Elhage

ZOME
ENERGY NETWORKS

Vinayak Ranade

BOSTON
UNIVERSITY

Douglas Densmore (co-PI)
Swapnil Bhatia
Traci Haddock
Viktor Vasilev
Chenkai Liu

iRobot[®]

Annan Mozeika
Ben Axelrod

Sponsors:

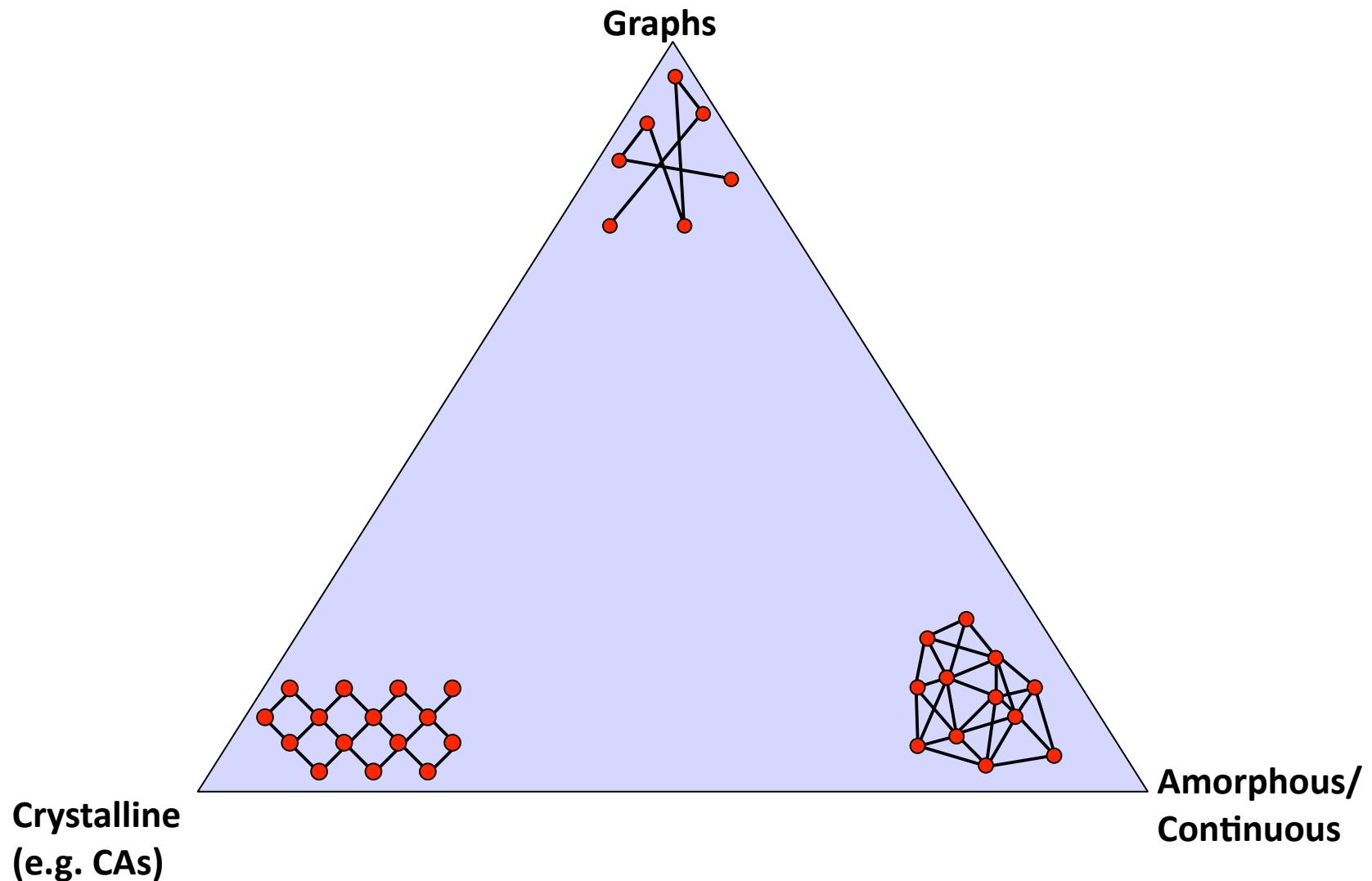


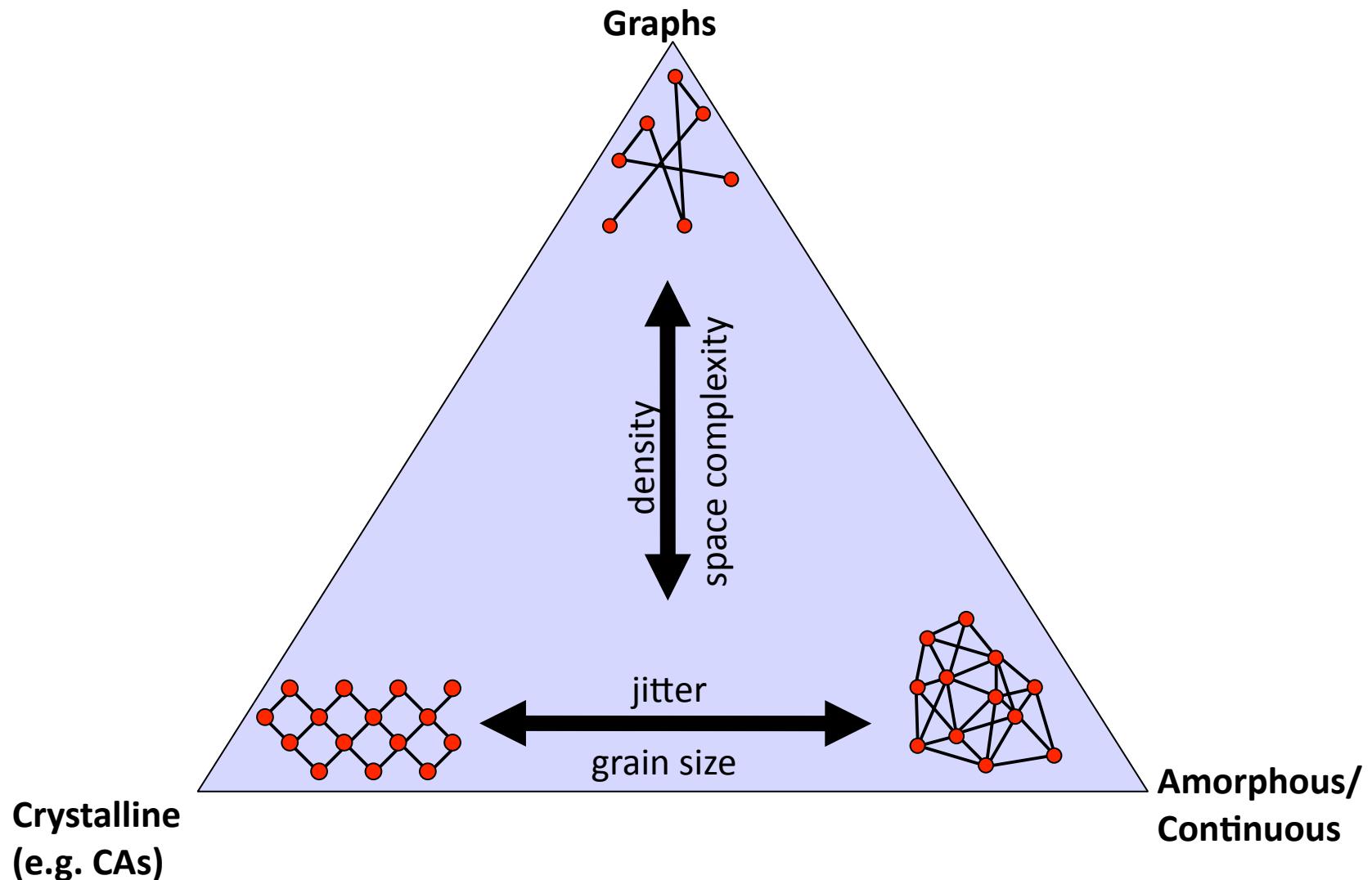
Backup Slides: Spatial Computing

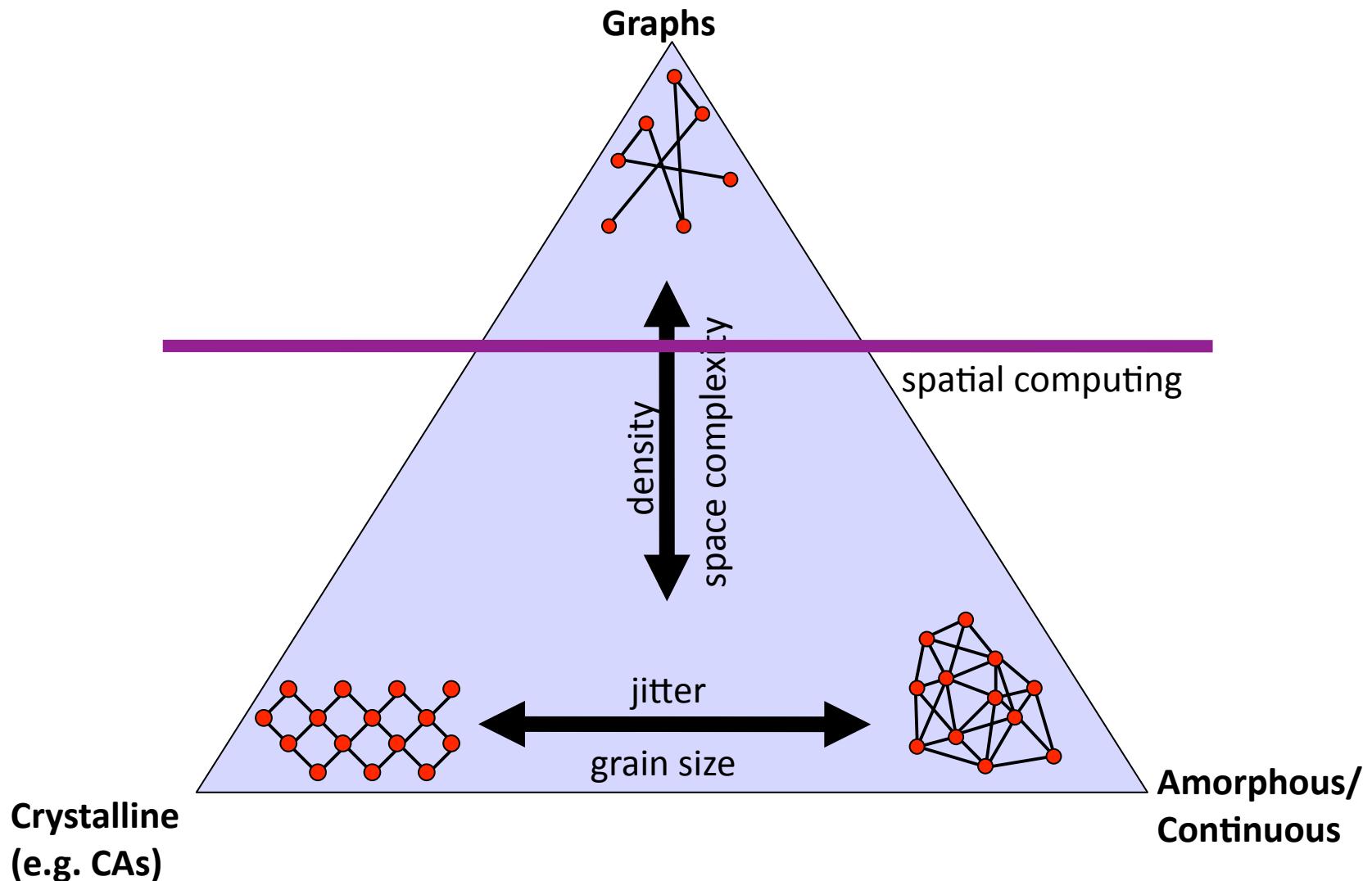
More formally...

- A spatial computer is a collection of computational devices **distributed through** a physical space in which:
 - the difficulty of moving information between any two devices is **strongly dependent** on the distance between them, and
 - the “functional goals” of the system are **generally defined** in terms of the system's spatial structure

Notice the ambiguities in the definition







Backup Slides: Demand Response

Initial ColorPower Prototype Devices

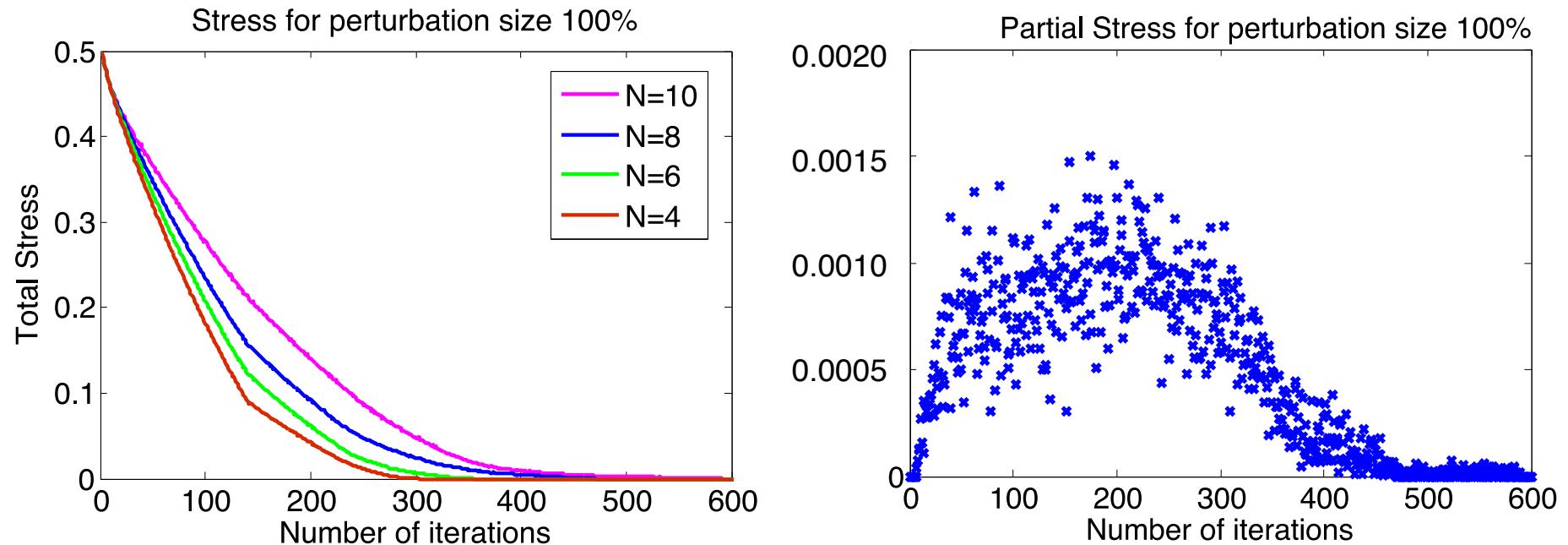
- Atmel AVR Raven
 - 8-bit processor
 - 802.15.4 wireless
- Current sensor
- Power control w. relay
- Button, rotary knob
- BOM: < \$100



Backup Slides: Synthetic Biology

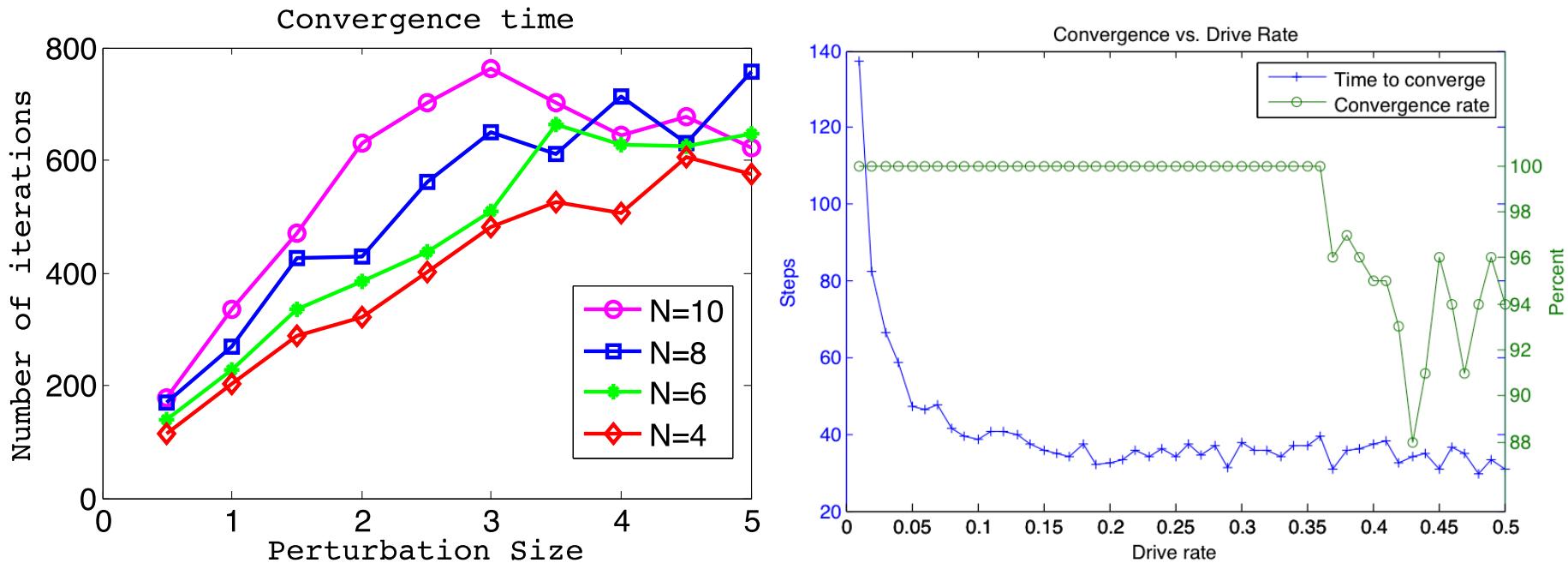
Backup Slides: Morphogenetic Engineering

FB Network Convergence Dynamics



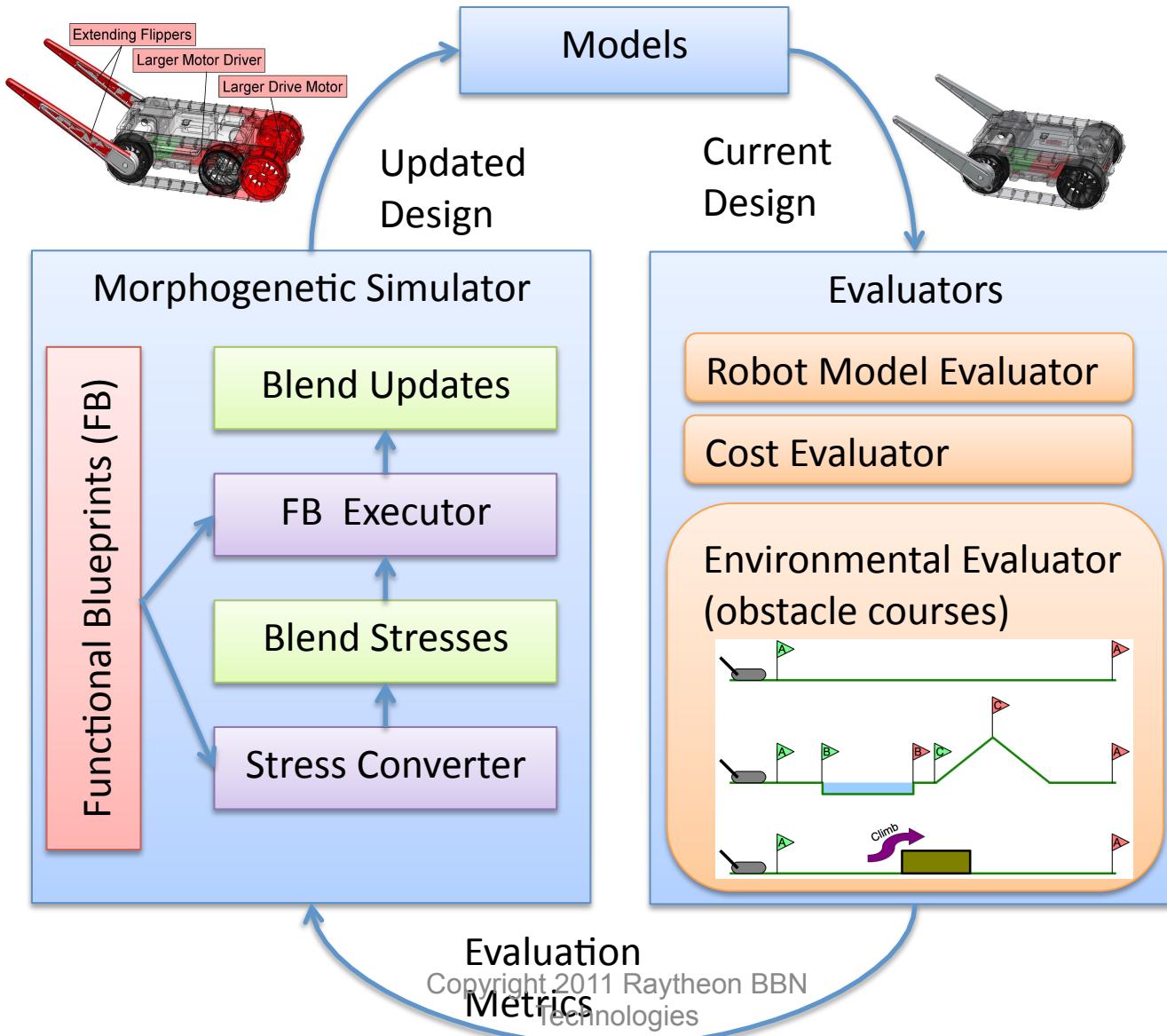
- **Experiment 1:** Induce a 100% perturbation in one of the attributes
 - For all random graphs, total stress decreases exponentially
 - Majority of the stress is the user perturbation (LEFT)
 - The rest of the system disperses the stress efficiently (RIGHT)

Resilience to Parameter Change

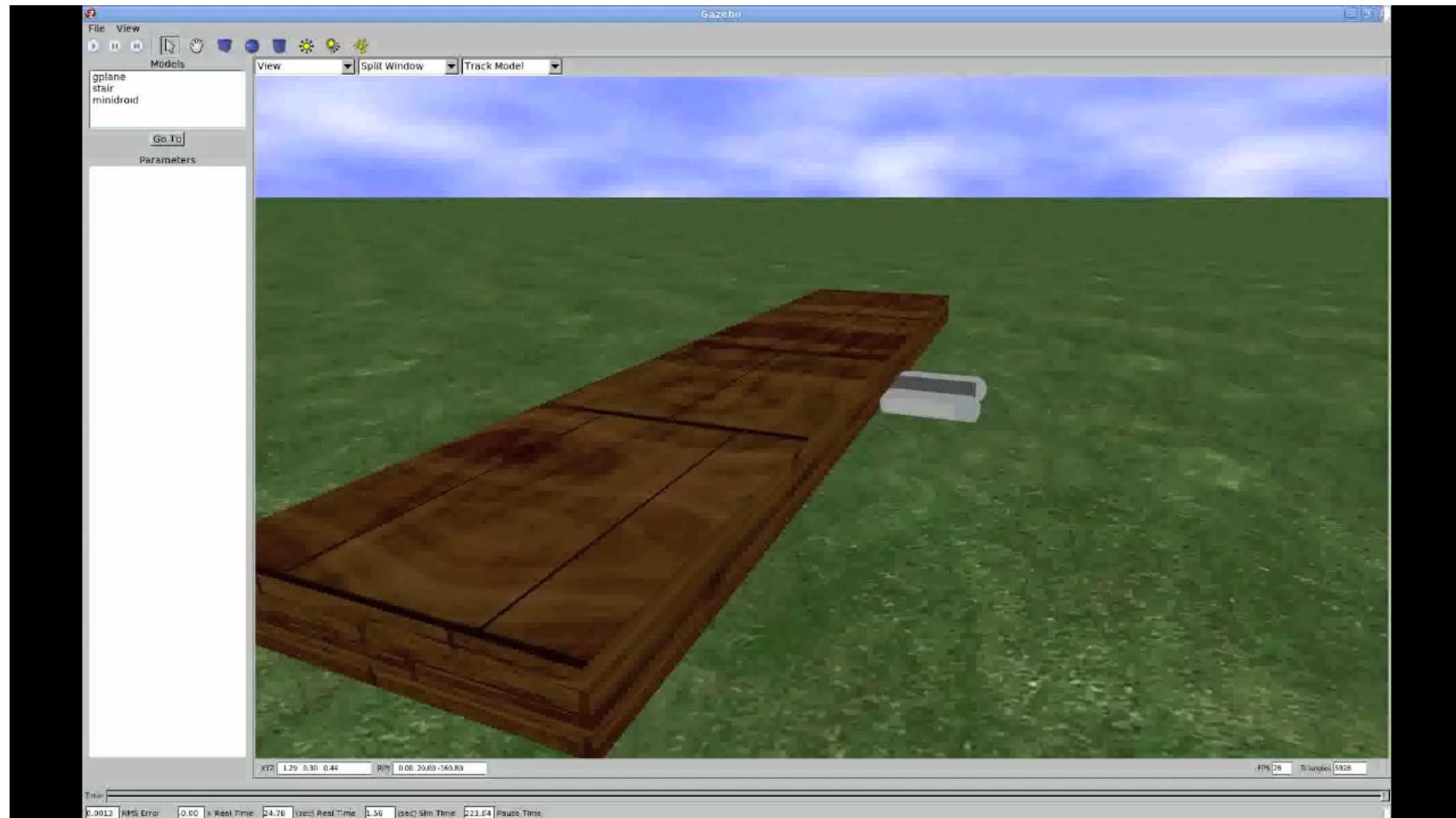


- **Experiment 2:** Vary perturbation from 100% to 500%
 - Regardless of the size of the constraint graph, linear relationship between the perturbation size and convergence time (LEFT)
- **Experiment 3:** Vary update rate (drive rate) from 0.01% to 50%
 - When increment rate is too high the system becomes non-viable (RIGHT)

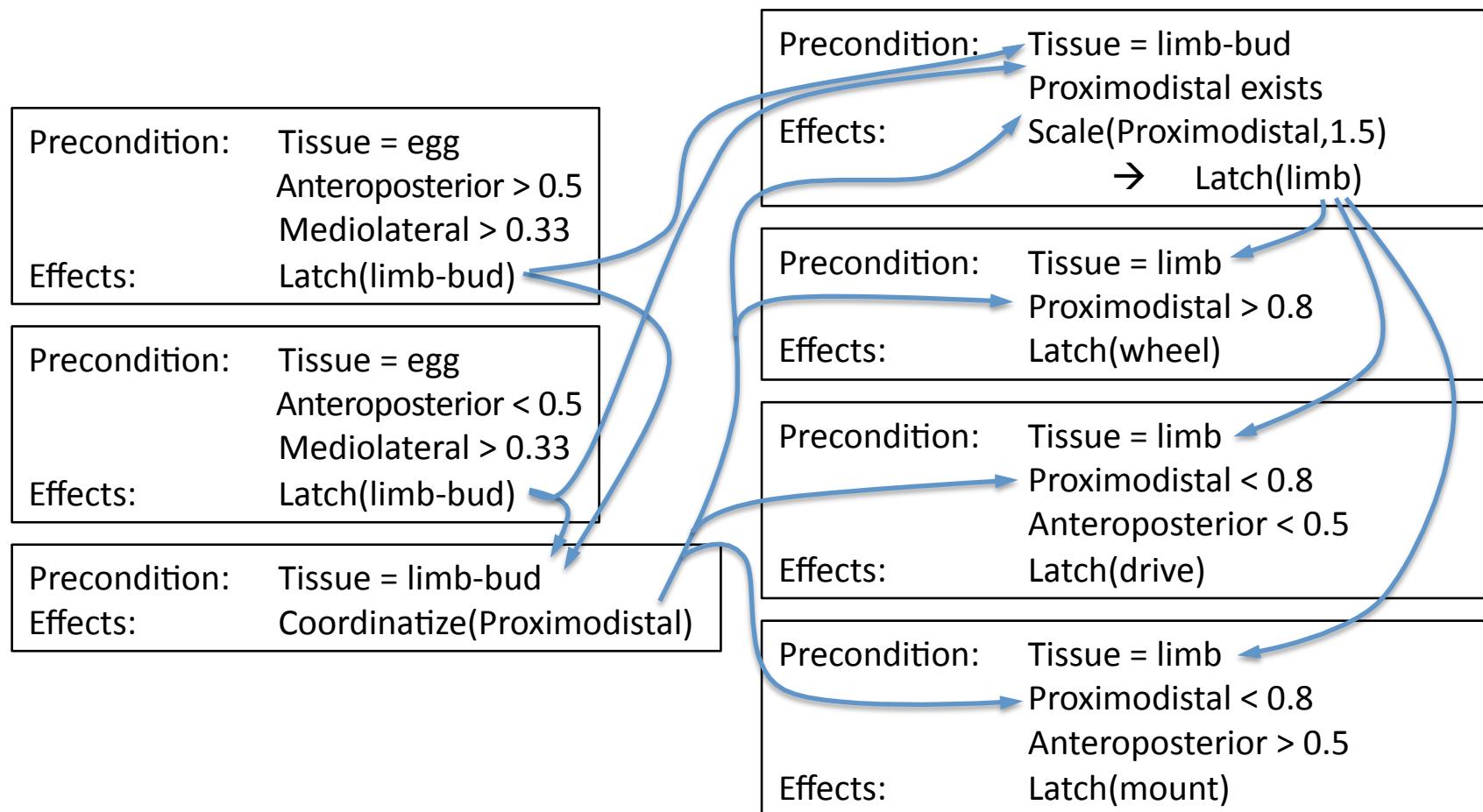
MADV Architecture



5x Variation Driven by Step Height



Program Representation: Manifold Rules



- Parallel application, continuous manifold evolution, conflict resolution by actuator blending
- Benefits: implicit relations, easy to modify/insert

Tissue-Level Developmental Model



Distortable Development in Proto



Soft Body Tissue Development



Jake's internal notes to self

Slide Count Distribution

- Infrastructure: 10 (title, 2 splash, 4 TOC, 3 finale)
- Spatial Computing: 7
- Proto/Amorphous Medium: 8
- Demand Response: 10
- Synthetic Biology: 17 (16 + 13 compile example)
- Morphogenetic Engineering: 11 (9+2x2 build)
- Total: 53 + infrastructure