Biochemical Switching Algorithms

Luca Cardelli
Microsoft Research

Joint work with Attila Csikász-Nagy
CoSBi

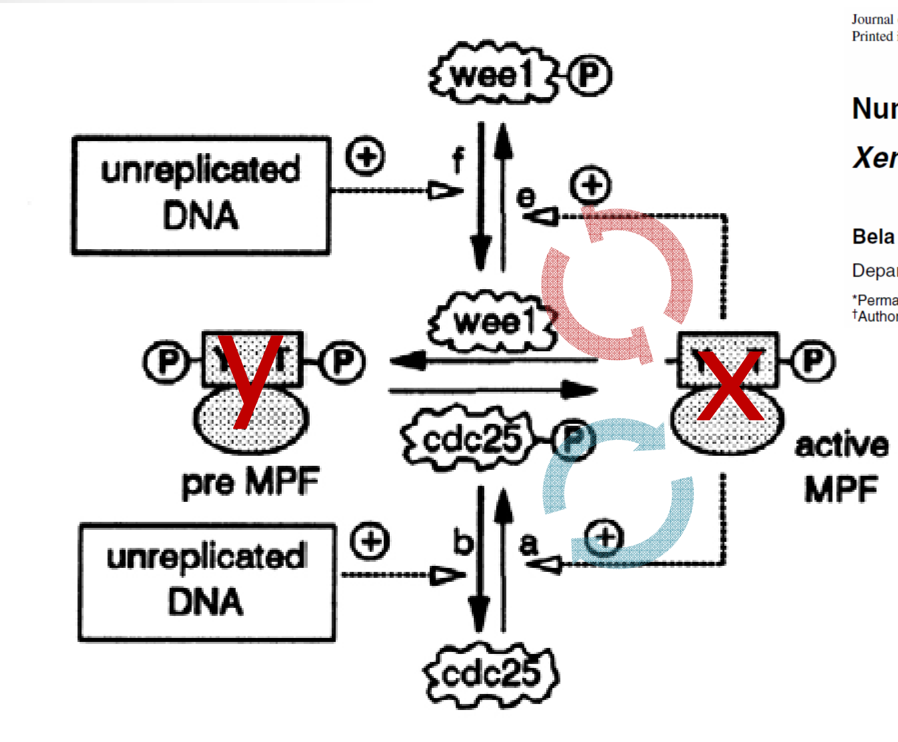
Newton Institute Lecture, Liverpool, 2012-07-16
http://lucacardelli.name

Outline

- Analyzing molecular networks
    - Various biochemical/bioinformatic techniques can tell us something about network structures.
    - We try do discover the function of the network, or to verify hypotheses about its function.
    - We try to understand how the structure is dictated by the function and other natural constraints.
- The Cell-Cycle Switches and Oscillators
    - Some of the best studied molecular networks.
    - Important because of their fundamental function (cell division) and preservation across evolution.

The Cell Cycle Switch

- At the core of the cell-cycled oscillator.
  - This network is universal in all Eukaryotes [P. Nurse].



Journal of Cell Science 106, 1153-1168 (1993)
Printed in Great Britain © The Company of Biologists Limited 1993

Numerical analysis of a comprehensive model of M-phase control in *Xenopus* oocyte extracts and intact embryos

Bela Novak* and John J. Tyson†

Department of Biology, Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24060-0406, USA

*Permanent address: Department of Agricultural Chemical Technology, Technical University of Budapest, 1521 Budapest Gellert Ter 4, Hungary
†Author for correspondence

- Double positive feedback on x
- Double negative feedback on x
- No feedback on y
- What on earth … ???

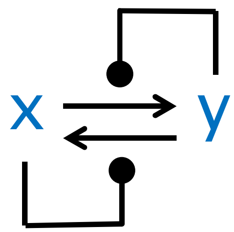
  - Well studied. But *why this structure?*

How to Build a Switch

- What is a "good" switch?
  - We need first a *bistable* system: one that has two *distinct* and *stable* states. I.e., given *any* initial state the system must *settle* into one of two states.
  - The settling must be *fast* (not get stuck in the middle for too long) and *robust* (must not spontaneously switch back).
  - Finally, we need to be able to *flip* the switch: drive the transitions by external inputs.
- "Population" Switches
  - Populations of identical agents (molecules) that switch from one state to another *as a whole*.
  - Highly concurrent (stochastic).

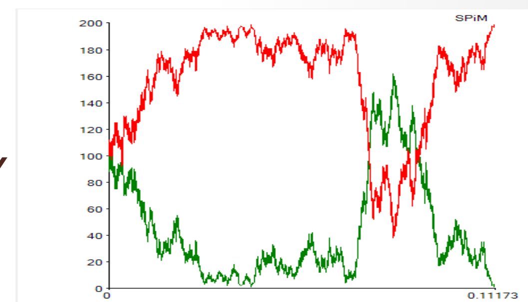
A Bad Algorithm

- Direct x–y competition
  - x catalyzes the transformation of y into x
  - y catalyzes the transformation of x into y



$$y + x \to x + x$$
$$x + y \to y + y$$

- This system is bistable, but
  - Convergence to a stable state is *slow* (a random walk).
  - *Any* perturbation of a stable state can initiate a random walk to the other stable state.




```
directive sample 0.0002
1000
directive plot x(); y(); b()

val rt = 10.0
new xcat@rt:chan
new ycat@rt:chan

let x() =
  do !xcat; x()
  or ?ycat; y()

and y() =
  do !ycat; y()
  or ?xcat; x()

run 100 of x()
run 100 of y()
```

A Very Good Algorithm

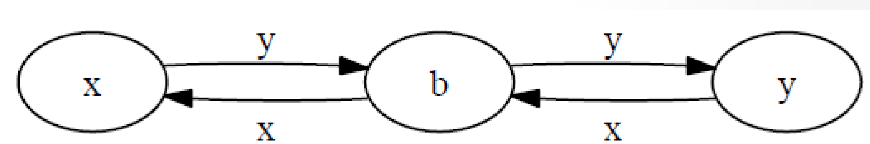
- Approximate Majority
  - Decide which of two populations is in majority
- A fundamental 'population protocol'
  - Agents in a population start in state x or state y.
  - A pair of agents is chosen randomly at each step, they interact ("collide") and change state.
  - The whole population must eventually agree on a majority value (all x or all y) with probability 1.

Dana Angluin · James Aspnes · David Eisenstat

A Simple Population Protocol for Fast Robust Approximate Majority

We analyze the behavior of the following population protocol with states $Q = \{b, x, y\}$. The state b is the **blank** state. Row labels give the initiator's state and column labels the responder's state.

	x	b	y
x	(x, x)	(x, x)	(x, b)
b	(b, x)	(b, b)	(b, y)
y	(y, b)	(y, y)	(y, y)



Third 'undecided' state.

Properties

- With high probability, for n agents
  - The number of state changes before converging is O(n log n)
  - The total number of <u>interactions</u> before converging is O(n log n)
  - The final outcome is correct if the initial disparity is ω(sqrt(n) log n)

[Angluin et al.
http://www.cs.yale.edu/homes/aspnes/papers/disc2007-eisenstat-slides.pdf]

- The algorithm is the fastest possible
  - Must wait Ω(n log n) steps in expectation for all agents to interact

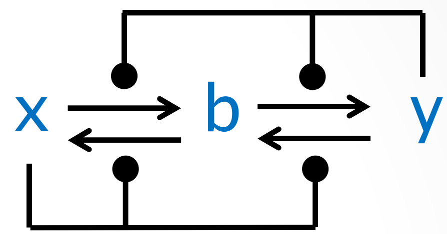
- Logarithmic time bound
  - Parallel time is the number of steps divided by the number of agents.
  - In parallel time the algorithm converges with high probability in O(log n).
  - That is true for any initial conditions, even x=y!

"Although we have described the population protocol model in a sequential light, in which each step is a single pairwise interaction, interactions between pairs involving different agents are independent and may be thought of as occurring in parallel. In measuring the speed of population protocols, then, we define 1 unit of parallel time to be jV j steps. The rationale is that in expectation, each agent initiates 1 interaction per parallel time unit; this corresponds to the chemists' idealized assumption of a well-mixed solution."
Distributed Computing 21(2):87-102.

Chemical Implementation

A programming language for population algorithms!

$x + y \rightarrow y + b$
$y + x \rightarrow x + b$
$b + x \rightarrow x + x$
$b + y \rightarrow y + y$

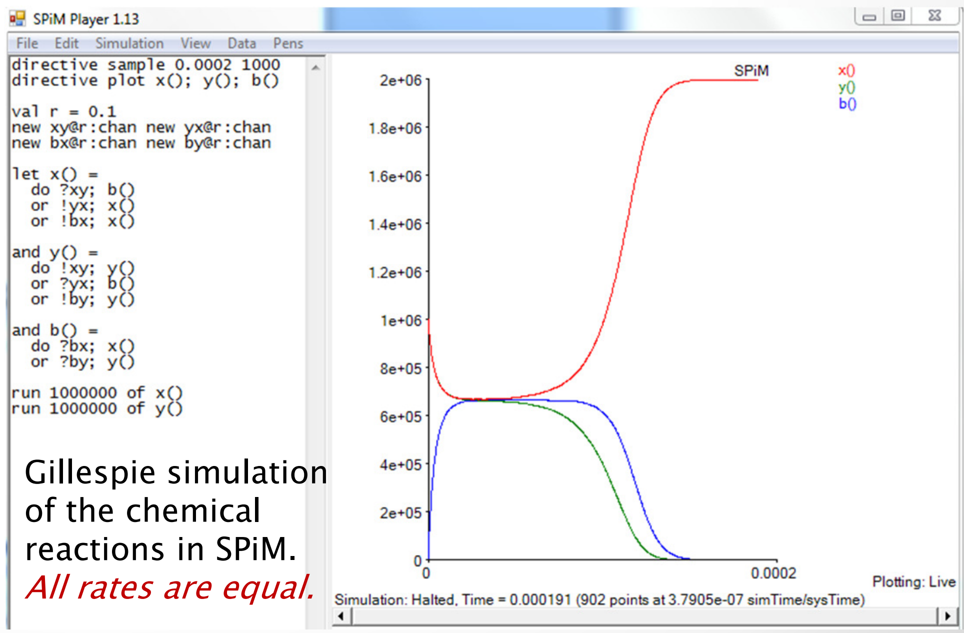


Bistable
   Even when x=y! (stochastically)

Fast
   O(log n) convergence time

Robust
   $\omega(\sqrt{n}\log n)$ majority wins whp

Worse case test: start with x=y.



Gillespie simulation of the chemical reactions in SPiM. *All rates are equal.*

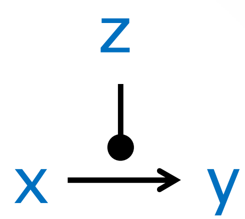
Back to the Cell Cycle

- The AM algorithm has great properties for settling a population into one of two states.
- But that is not what the cell cycle uses to switch its populations of molecules.
- Or is it?

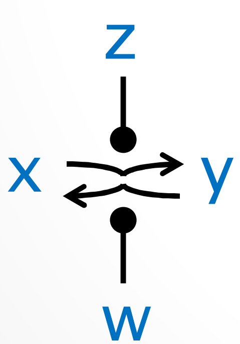
Some Notation

- Catalytic reaction

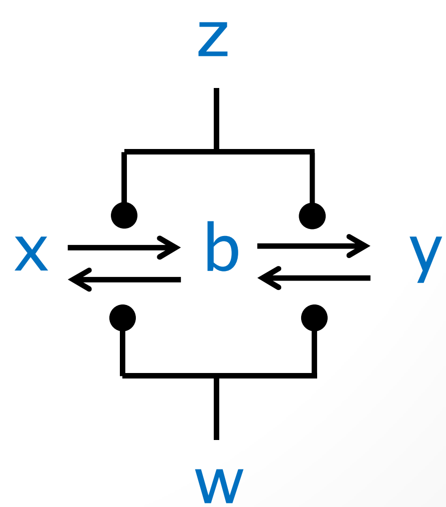
$x + z \rightarrow z + y \quad = $



- Double 'kinase-phosphatase' reactions

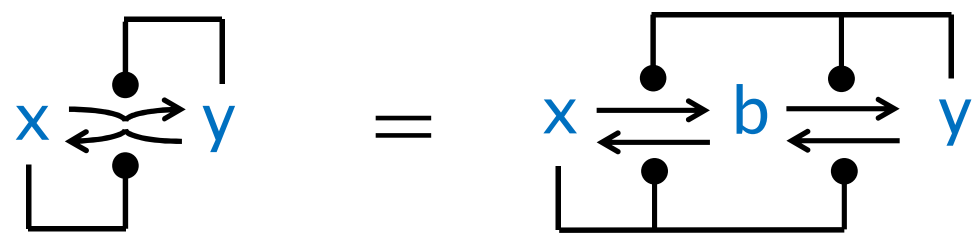


=



Step 1: the AM Network

Abbreviated notation:

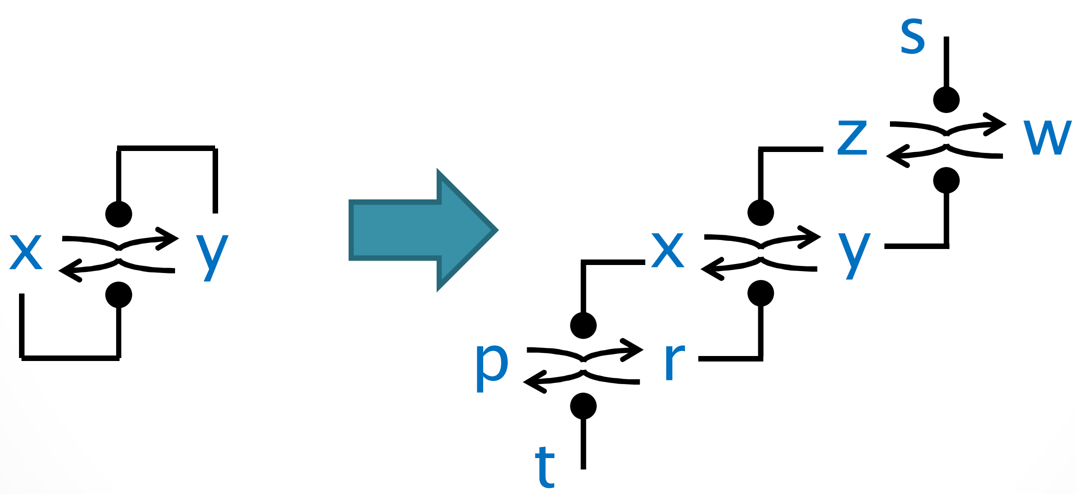


- CONSTRAINT: Autocatalysis, and especially intricate autocatalysis, is not commonly seen in nature.

$$b + x \to x + x$$
$$b + y \to y + y$$

Step 2: remove auto-catalysis

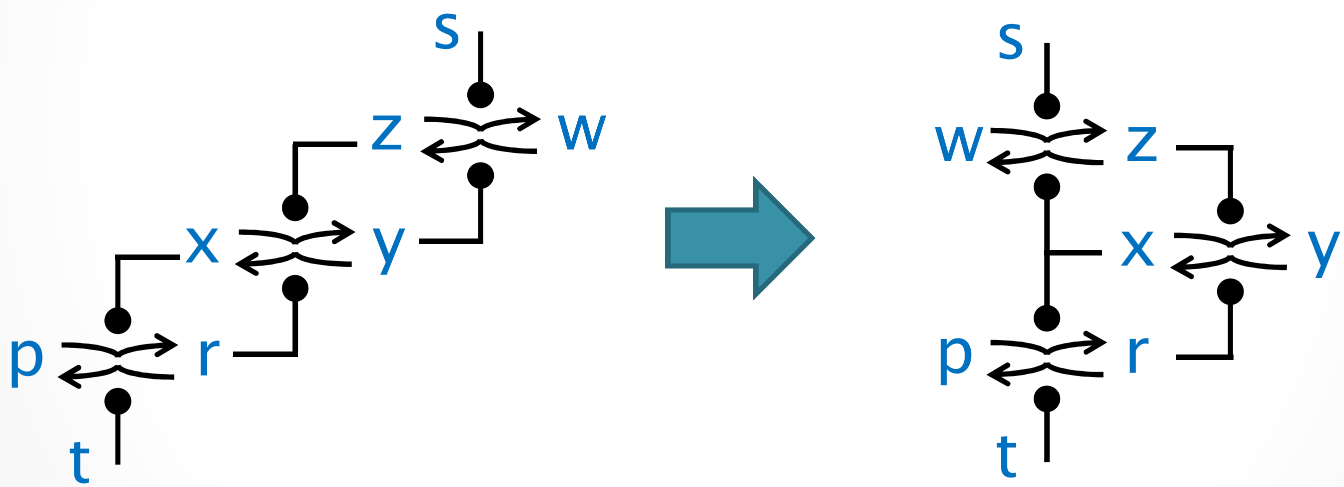
- Replace autocatalysis by mutual (simple) catalysis, introducing intermediate species z, r.
  - Here z breaks the y auto-catalysis, and r breaks the x auto-catalysis, while preserving the feedbacks.
  - z and r need to ‘relax back’ (to w and p) when they are not catalyzed: s and t provide the back pressure.



- CONSTRAINT: x and y (two states of the same molecule) are distinct active catalysts: that is not common in nature.

Step 3: only one active state

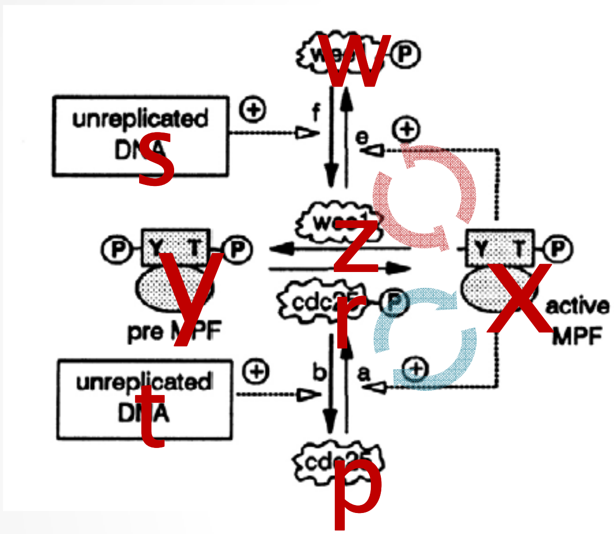
- Remove the catalytic activity of y.
  - Instead of y activating itself through z, we are left with z activating y (which remains passive). Hence, to deactivate y we now need to deactivate z. Since x ‘wants’ to deactivate y, we make x deactivate z.

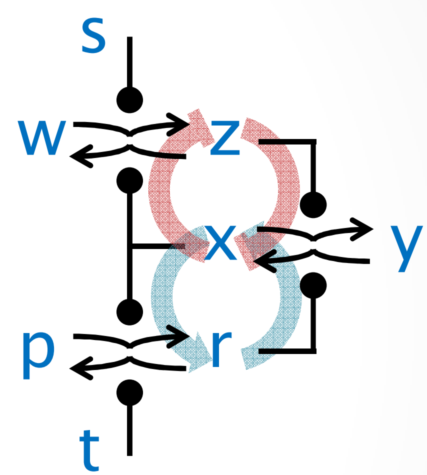


- All species now have one active (x,z,r) and one inactive (y,w,p) form. This is ‘normal’.

Network Structure

- … and that *is* the cell-cycle switch!

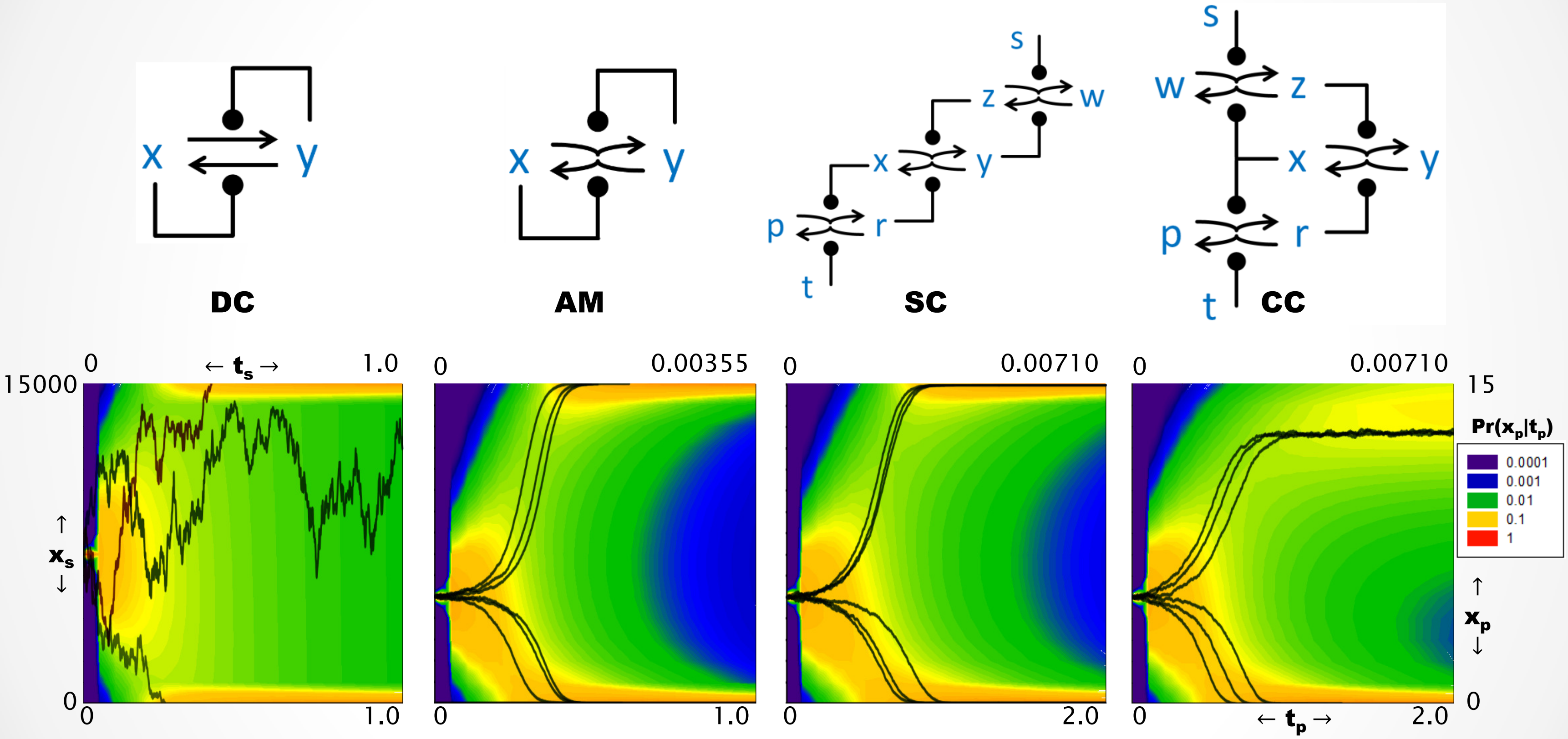




- The question is: did we preserve the AM *function* through our *network transformations*?
  - Ideally: prove either that the networks are ‘contextually equivalent’ or that the transformations are ‘correct’.
  - Practically: compare their ‘typical’ behavior.

Convergence Analysis

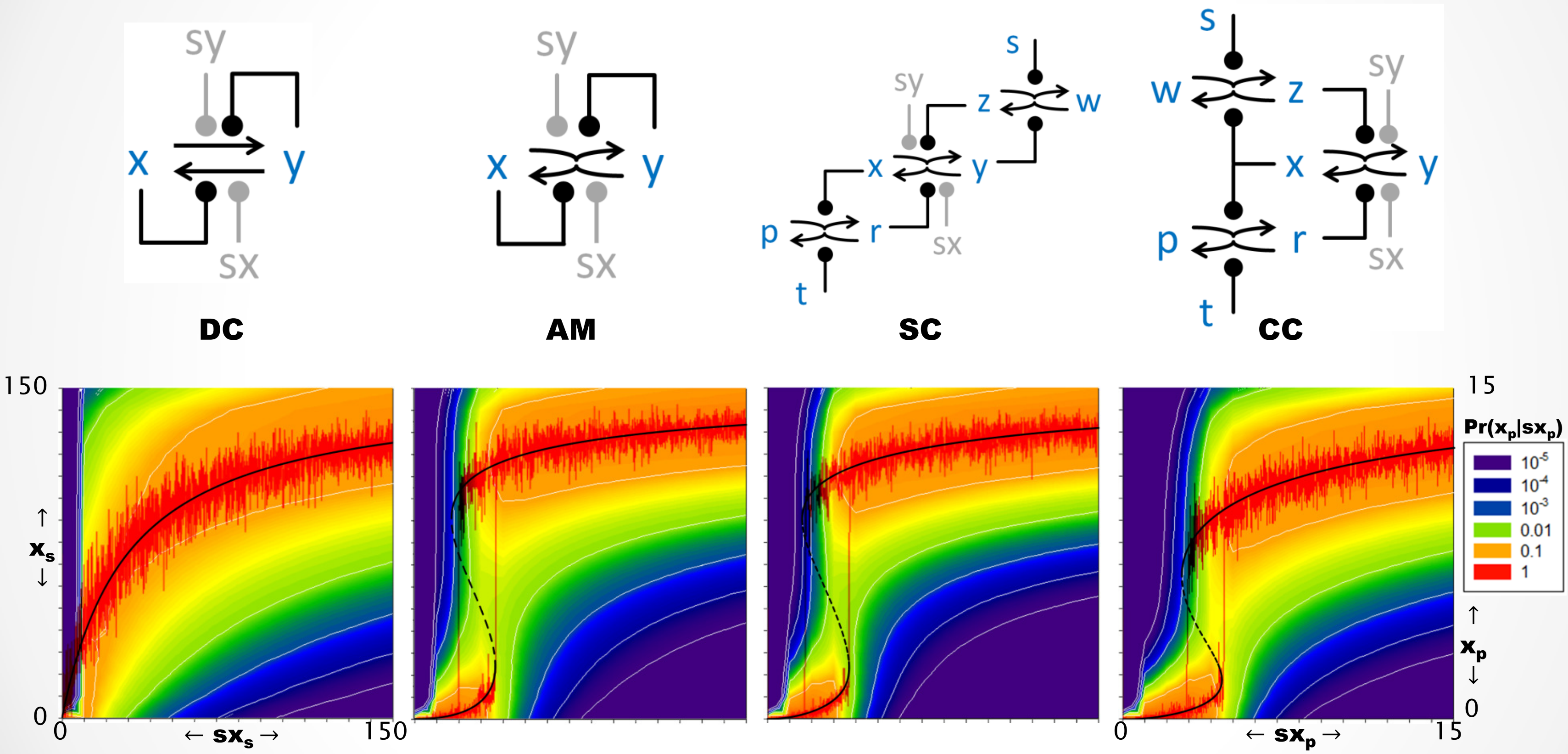
Switches as Computational Systems – Convergence



NEW!
CC
converges
in log time

Steady State Analysis

Switches as Dynamical Systems – Steady State Response



NEW!
AM shows
hysteresis

The Argument So Far

- Relating dynamical and computational systems in isolation (as *closed systems*)
  - The AM algorithm (network) implements an input-driven switching function (in addition to the known majority function).
  - The CC algorithm implements a input-less majority function (in addition to the known switching function).
  - The structures of AM and CC are related, and an intermediate network shares some properties of both.
- But what about the context?
  - Will AM and CC behave similarly in any context (as *open systems*)?
  - That's a hard question, so we look at their intended context: implementing oscillators.

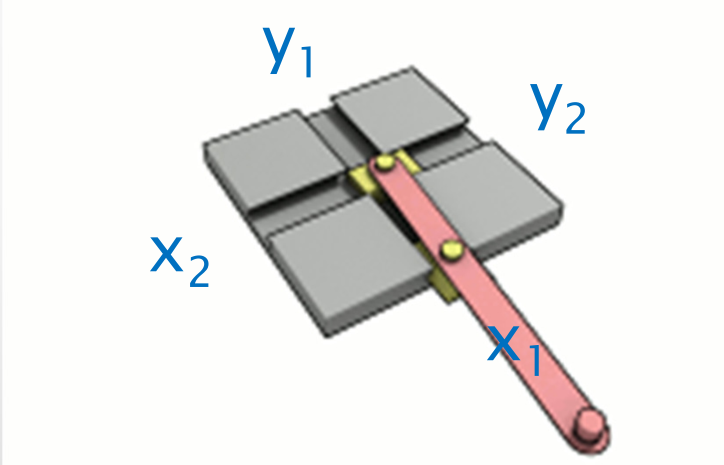
Oscillators

- Basic in Physics, studied by simple *phenomenological* (not structural) ODE models.
- Non-trivial in Chemistry: it was only discovered in the 20's (Lotka) that chemical systems can oscillate: before it was thought impossible in closed systems. Shown experimentally only in the 50's.
- Mechanics (since antiquity) and modern Electronics (as well as Chemistry) must engineer the *network structure* of oscillators.
- Biology: all natural cycles are oscillators. Here we must reverse engineer their network structure.
- Computing: how can populations of agents (read: molecules) interact (network) to achieve oscillations?

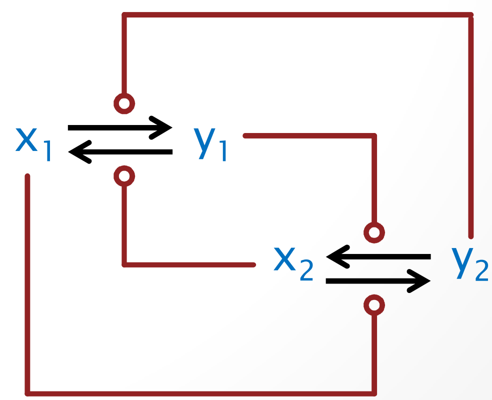
The Trammel of Archimedes

- A device to draw ellipses
  - Two interconnected **switches**.
  - When one switch is on (off) it flips the other switch on (off). When the other switch is on (off) it flips the first switch off (on).
  - The amplitude is kept constant by mechanical constraints.

The function



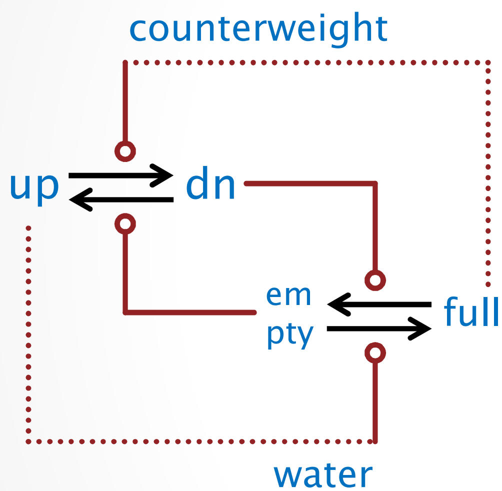
The network



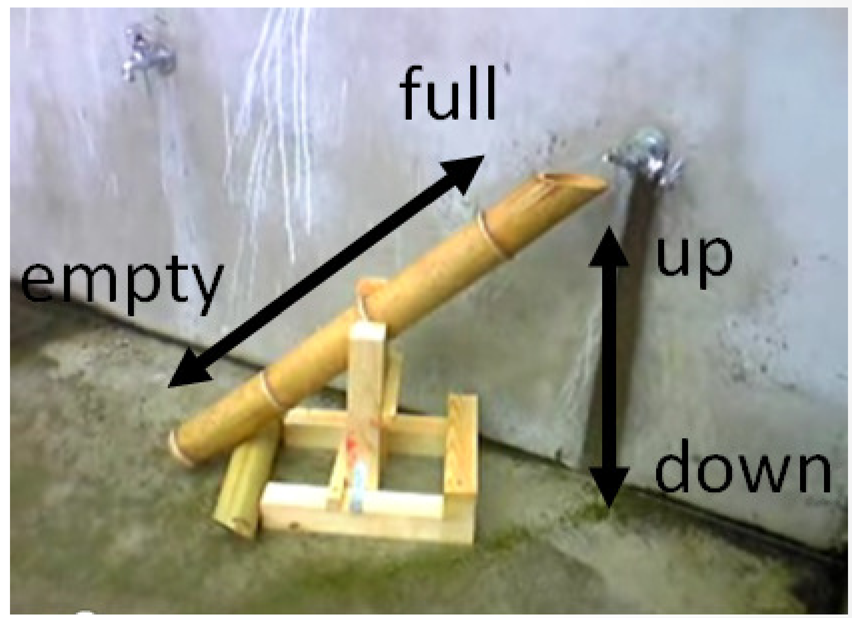
en.wikipedia.org/wiki/Trammel_of_Archimedes

The Shishi Odoshi

- A Japanese scarecrow (*lit.* scare-deer)
  - Used by Bela Novak to illustrate the cell cycle switch.



empty + up → up + full
up + full → full + dn
full + dn → dn + empty
dn + empty → empty + up

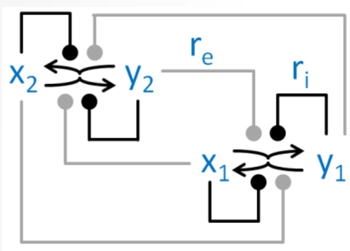


http://www.youtube.com/watch?v=VbvecTIftcE&NR=1&feature=fvwp

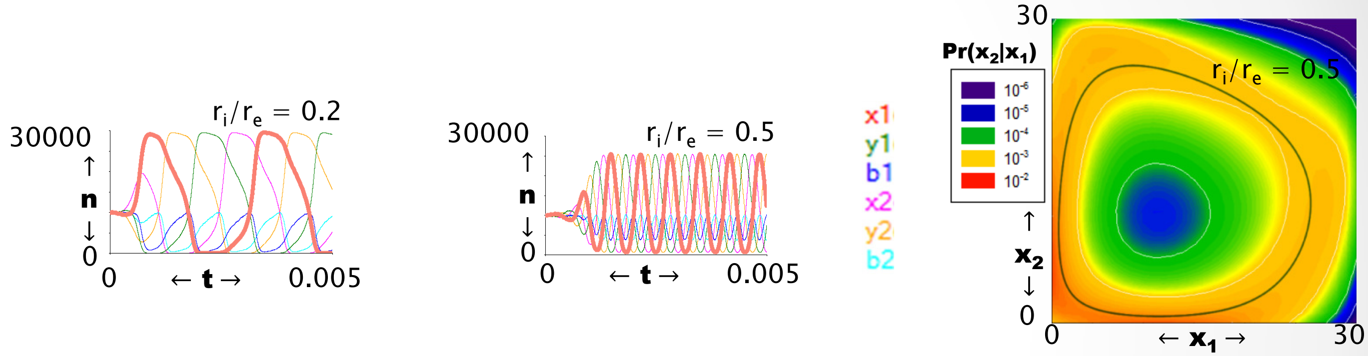
Outer switched connections replaced by constant influxes: tap water and gravity.

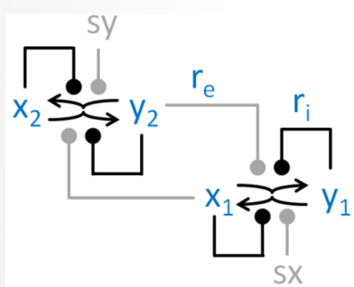
Contextual Analysis

AM switches in the context of larger networks (oscillators).

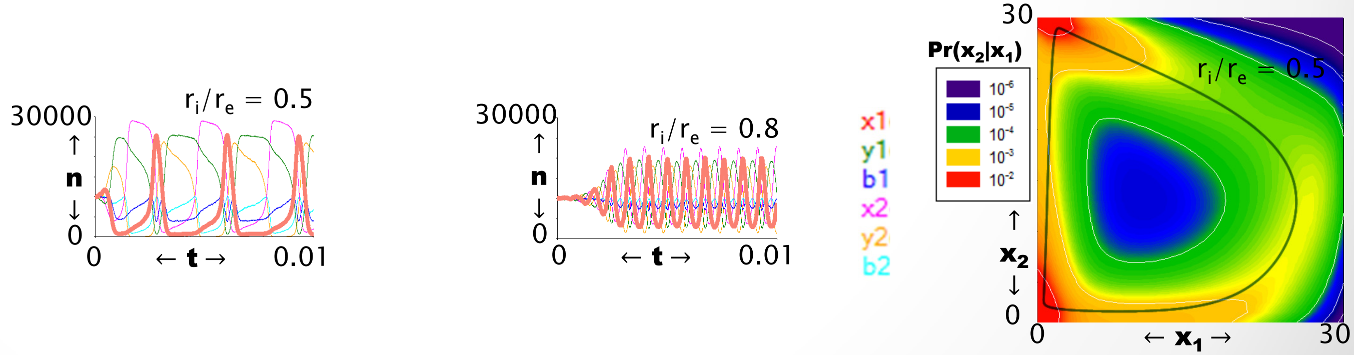


Trammel



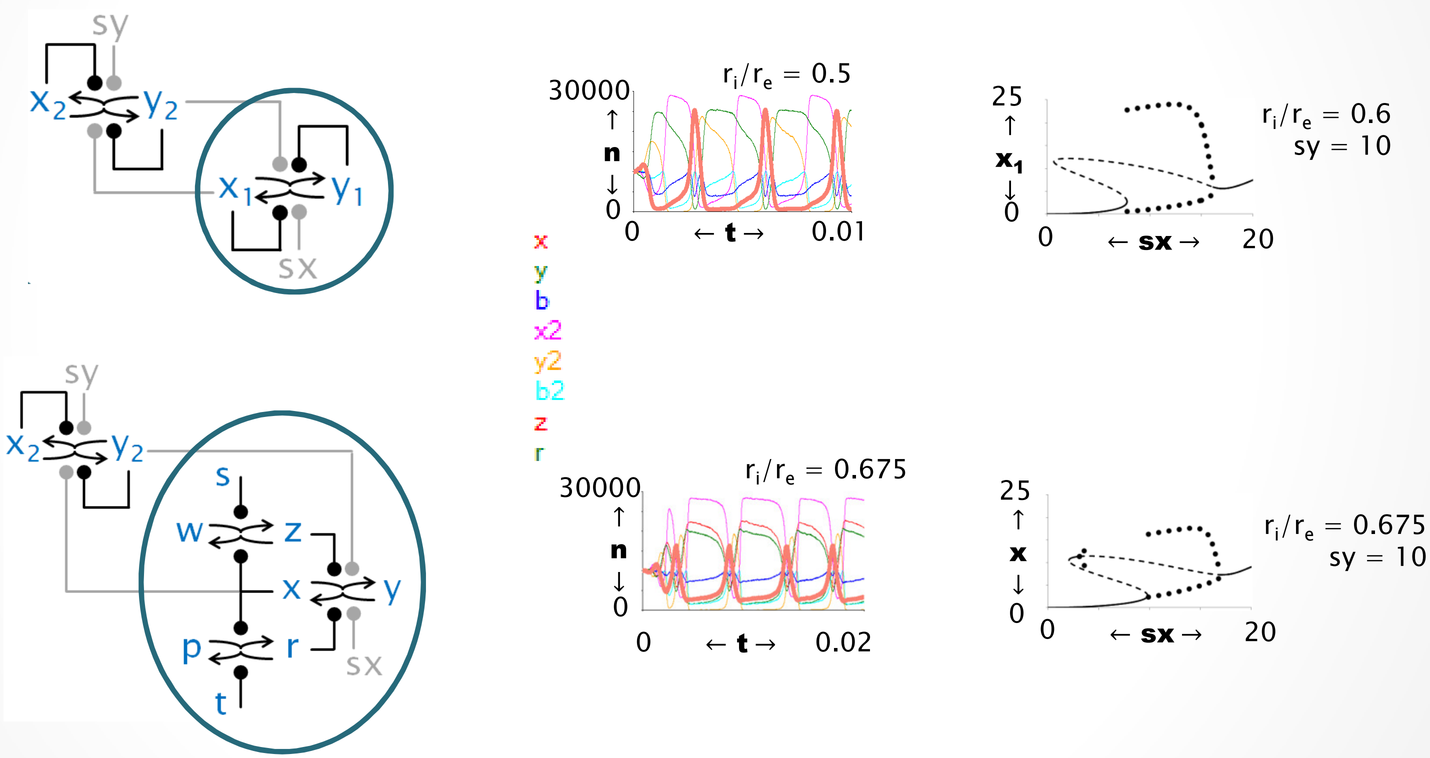


Shishi Odoshi



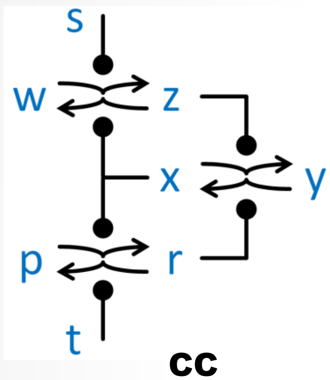
Modularity Analysis

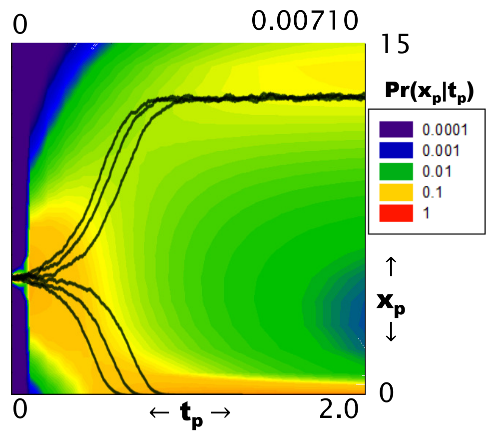
CC can be swapped in for AM.



CC does not “fully switch”

We have seen that the output of CC does not go ‘fully on’ like AM:



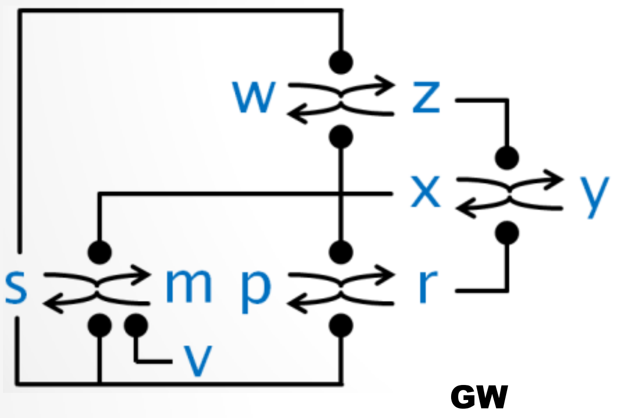


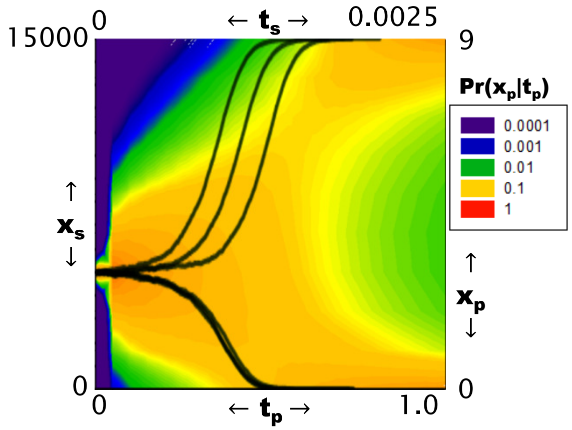
because s continuously inhibits s so that x cannot fully express.
This could be solved if x would inhibit s in retaliation.

Q: How would *you* fix this problem?

Nature fixed it!

There is another known feedback loop in real cell cycle switches by which x suppresses s:



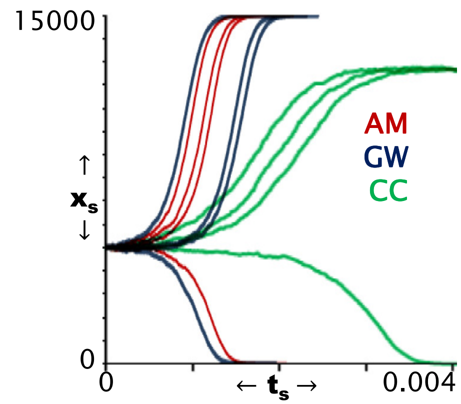


Full activation!

(Also, s and t happen to be the same molecule)

And made it fast too!

More surprising: the extra feedback also speeds up the decision time of the switch, making it about as good as the ‘optimal’ AM switch:



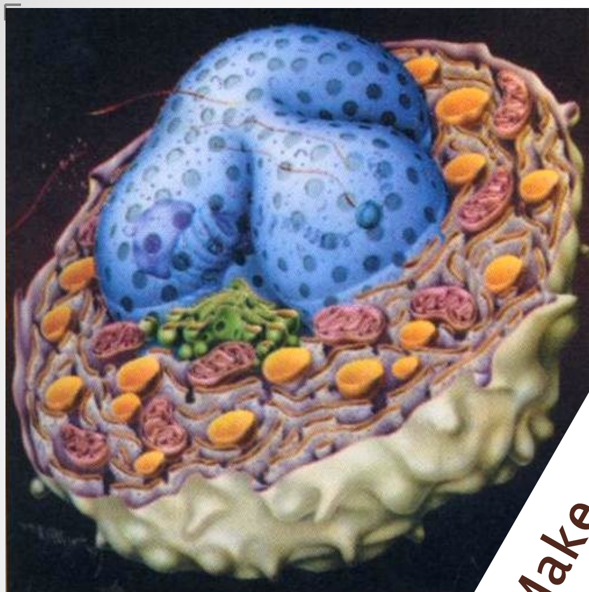
Conclusion:
Nature is trying as hard as it can to implement an AM-class algorithm!

Summary

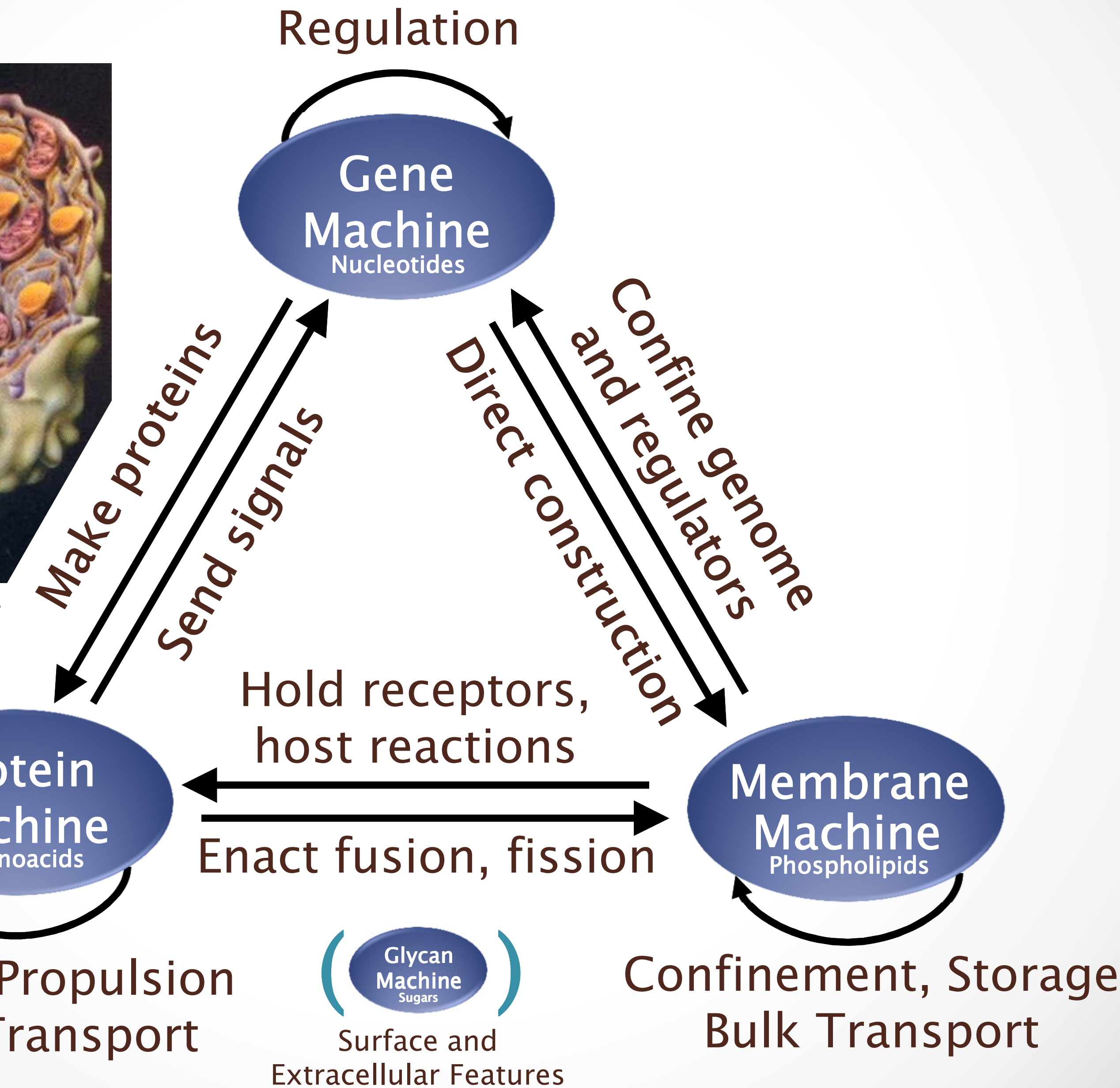
- The structure of AM implements an input-driven switching function (in addition to the known majority function).
- The structure of CC implements a input-less majority function (in addition to the known switching function).
- The structures of AM and CC are related, and an intermediate network shares the properties of both.
- The behaviors of AM and CC in isolation are related.
- The behaviors of AM and CC in oscillator contexts are related.
- A refinement of the core CC network, known to occur in nature, improves switching performance and brings it in line with AM performance.

Computational Outlook

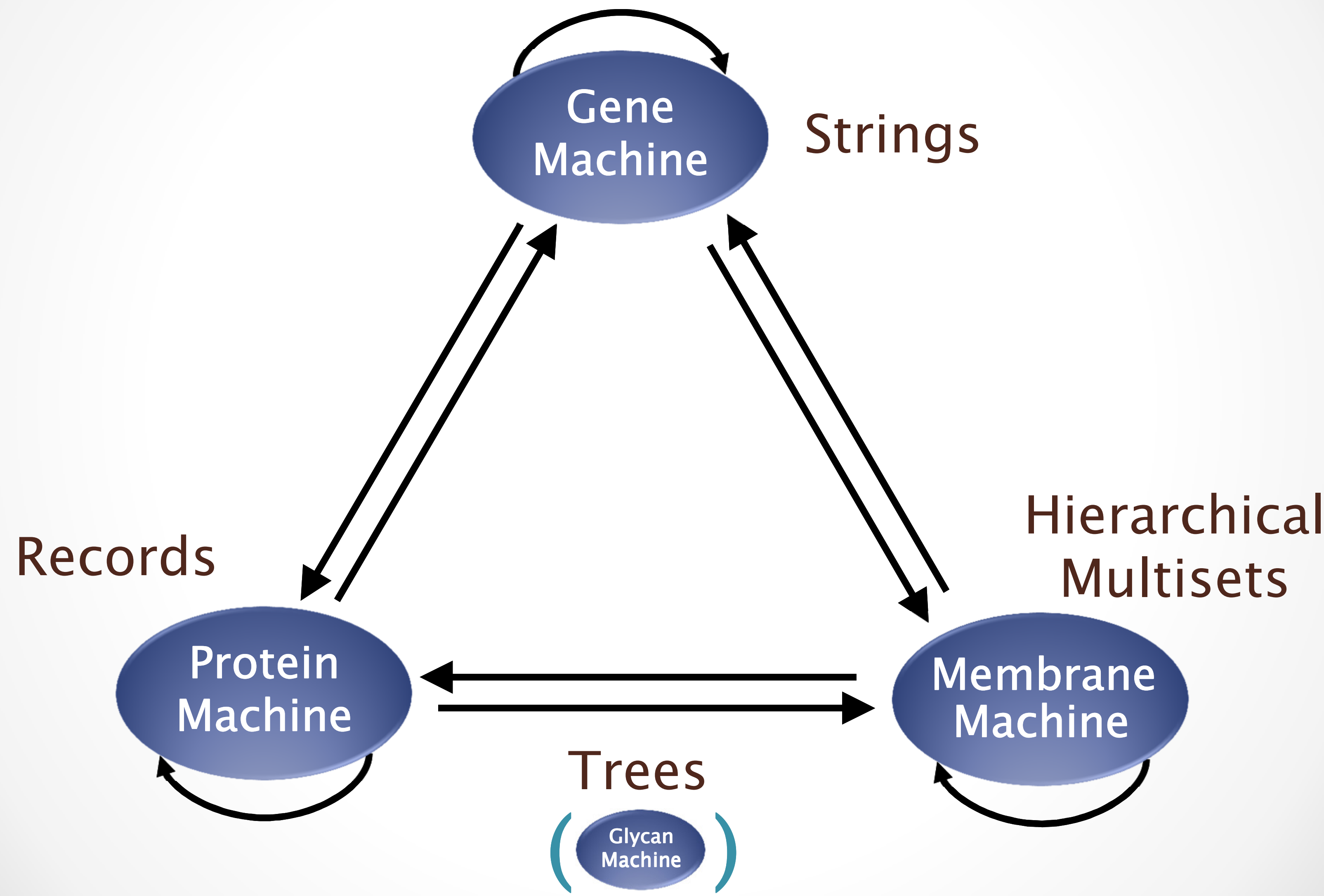
Abstract Machines of Biochemistry



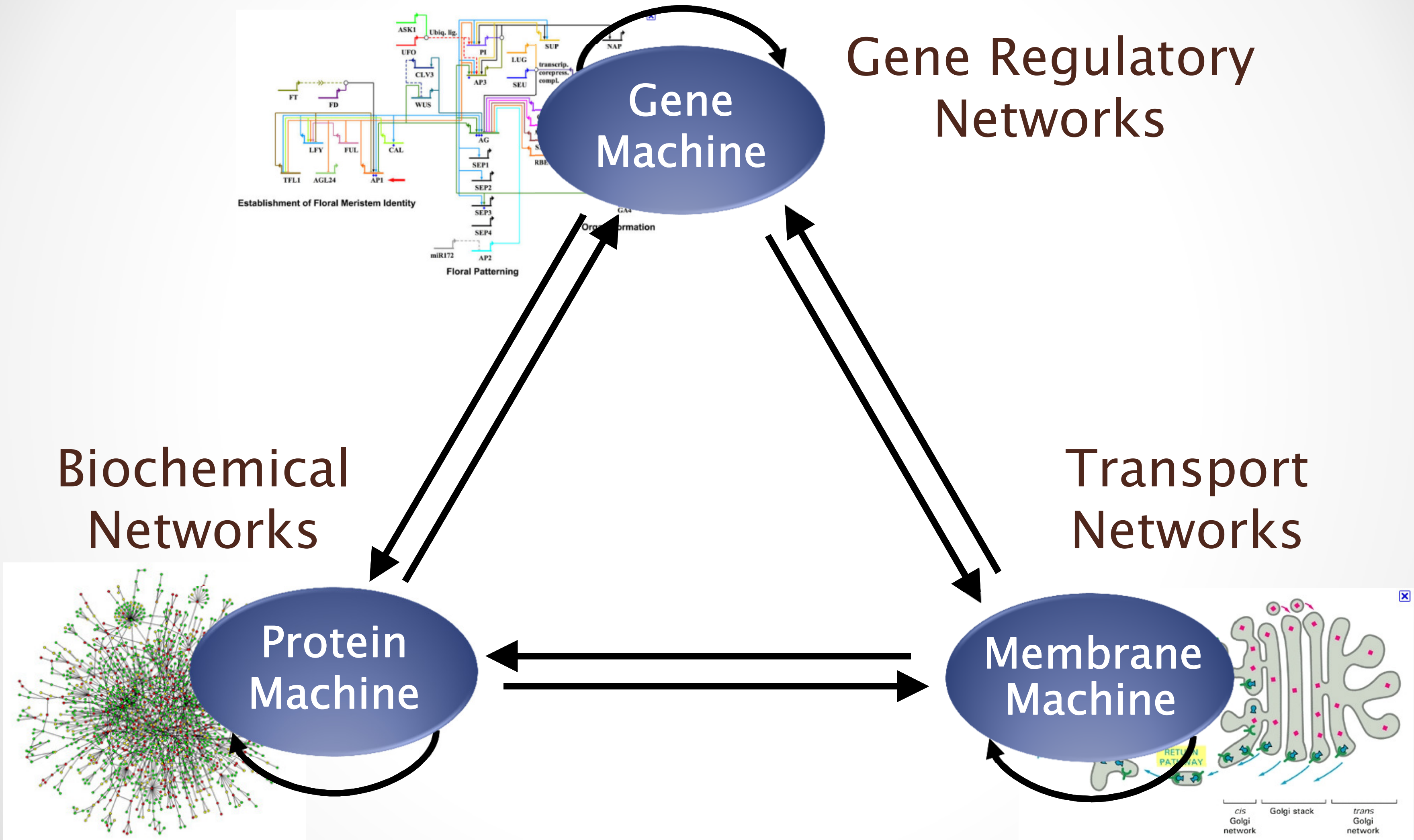
H.Lodish & al. Molecular Cell Biology 4th ed.



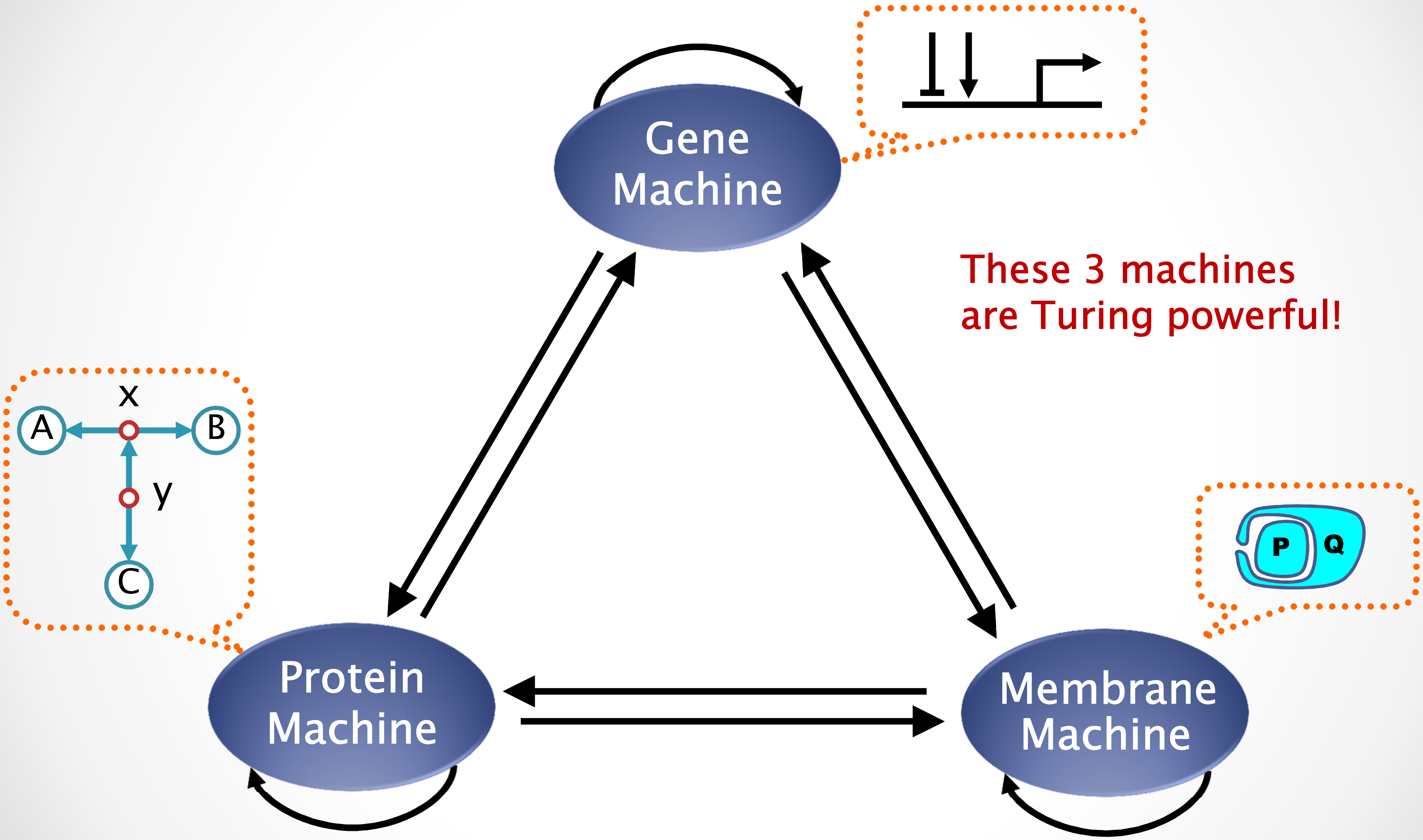
Bioinformatic View (Data Structures)



Systems Biology View (Networks)



Algorithmic View



These 3 machines
are Turing powerful!

Computational viewpoint

- Cells are computational engines
  - Their *primary* function is information processing
    - Which controls feeding, escape, and reproduction.
    - Without properly processing information cells soon die (by starvation or predation).
    - Hence a strong pressure to process information better.
  - Which *happens* to be implemented by chemistry
    - Fundamental is not the 'hardware' (proteins etc.) which easily varies between organisms but the 'software' the runs on the hardware.
- So, what algorithms do they run?

Reverse Engineering

- Q (traditional): What kind of **dynamical system** is the cell-cycle switch?
- A (traditional): Bistability – ultrasensitivity – hysteresis … Focused on how unstructured sub-populations change over time.

- Q: What kind of **algorithmic system** is the cell-cylce switch?
- A: Interaction – complexity – convergence ... Focused on individual molecules as programmable, structured, algorithmic entities.

- Leading to a better understanding of not just the *function* but also the *network* (algorithm).

Direct Engineering

- The AM algorithm was not learned from nature
  - CC was invented ~2.7 billions years ago.
  - AM was invented ~6 years ago (but independently).

- But nature may have more tricks
  - If there is some clever population algorithm out there, how will we recognize it?
  - We need to understand how nature operates.