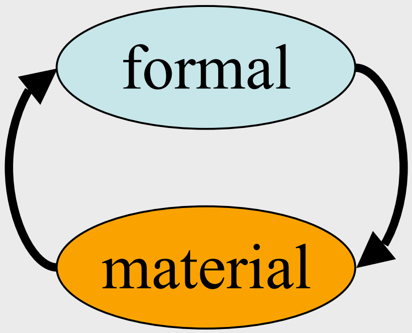
Self-Organization for Nano-Computation and Nano-Assembly

Bruce J. MacLennan
Dept. of Computer Science
University of Tennessee, Knoxville

Central Themes

- Computation and physics
  - novel computational models to exploit novel technologies
  - computational control of matter
- Natural computation
  - computation occurring in nature, or
  - inspired by that occurring in nature



Current Research in Self-Organization

- Synthetic Ethology & the Emergence of Communication
- Molecular Computing for Nanostructure Synthesis & Control
- Radical Reconfiguration of Computing Systems
- Generalized Computation (U-Machine)
- Programmable Microorganisms for Artificial Morphogenesis
- Applications in Command, Control & Coordination

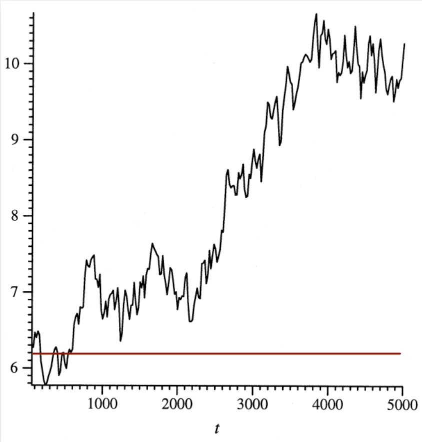
Some Principles of Adaptive Self-Organization

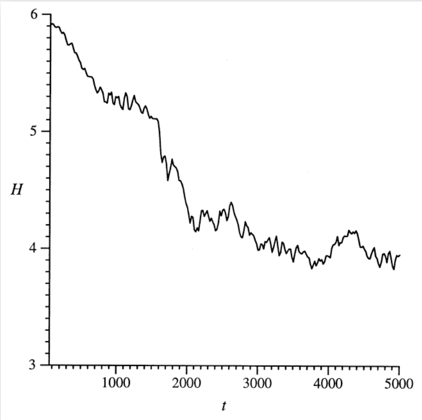
- Positive & negative feedback
- Noise, randomness, imperfection
- Amplification of random fluctuations
- Symmetry breaking
- Diffusion
- Stigmergy
- Simple, local microdecisions
- Multiple interactions
- Circular causality
- Excitable media
- Local nonlinear interactions
- Adaptive stationary states
- Nonconvergence, diversity & suboptimal solutions
- Developmental cascades
- Entrainment & distributed synchronization

Synthetic Ethology and the Emergence of Communication

Evolution of Communication (1990)

- Experiments to demonstrate self-organized emergence of communication among simple agents
- Selective pressure in favor of *cooperation*
- Agents can modify or sense state of shared global environment
- GA selects for best cooperators
- Agents evolve to *communicate* using a simple code





The U-Machine

Computation in General Sense

- A definition applicable to computation in nature as well as computers
- Computation is a *physical* process, the purpose of which is *abstract* operation on *abstract* objects
- A computation must be implemented by *some* physical system, but it may be implemented by *any* physical system with the appropriate abstract structure

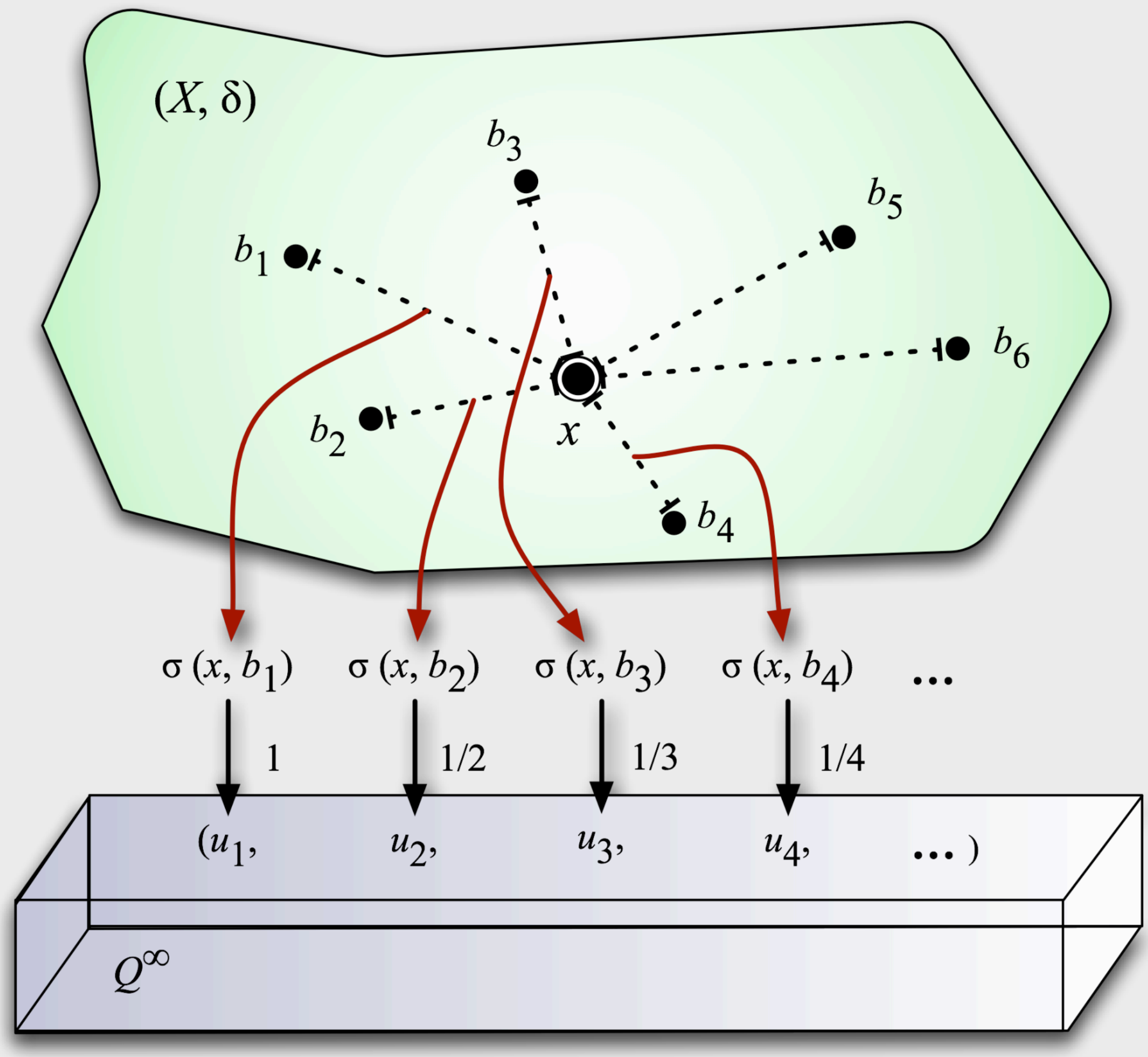
Abstract Spaces

- Should be general enough to include continuous & discrete spaces
- Hypothesis: *separable metric spaces*
- Include continua & countable discrete spaces
- separable ⇒ approximating sequences

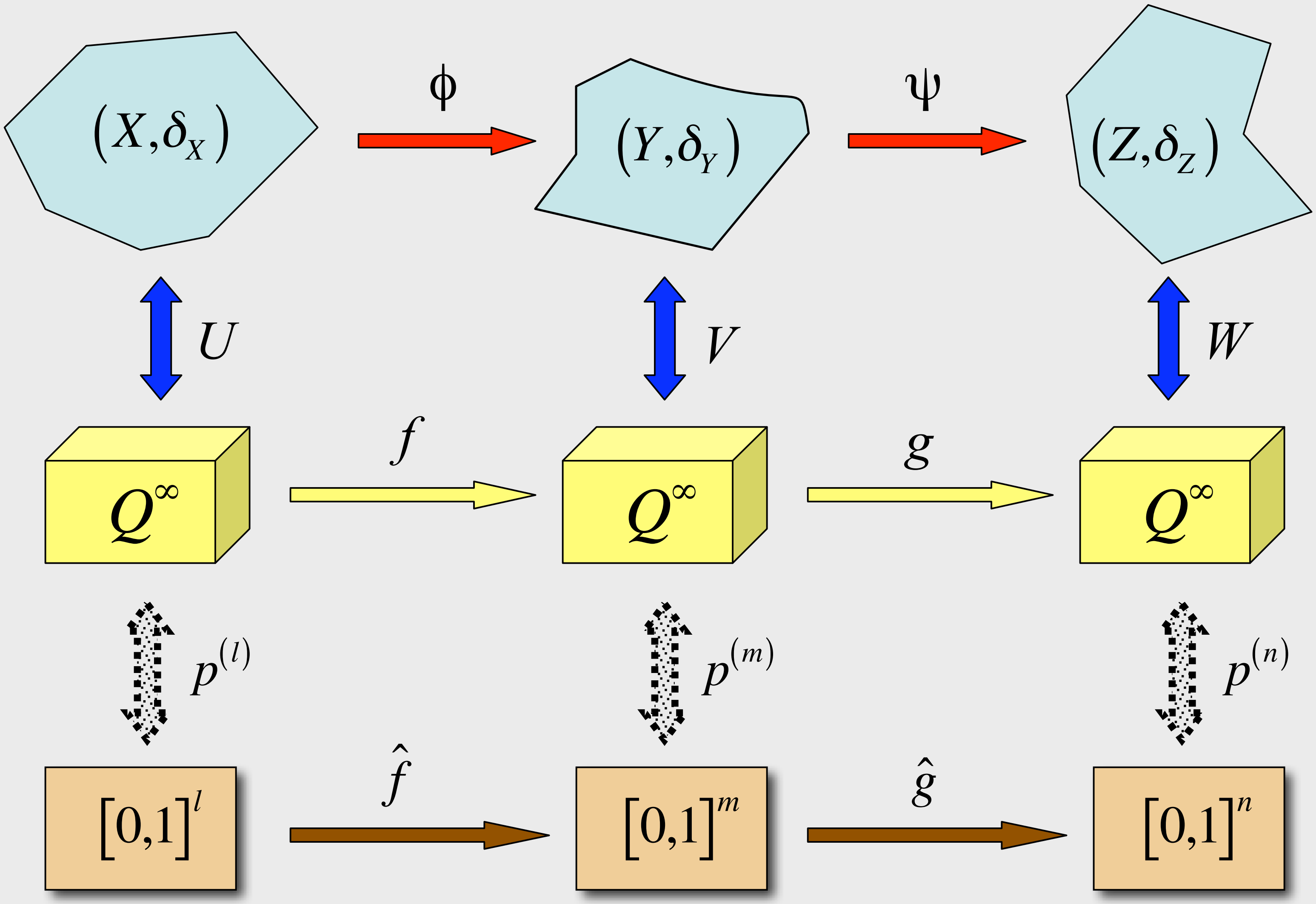
The U-Machine

- Goal: a model of computation over abstract spaces that can be implemented in a variety of physical media
- In particular, bulk nanostructured materials in which:
  - access to interior is limited
  - detailed control of structure is difficult
  - structural defects and other imperfections are unavoidable

Urysohn Embedding

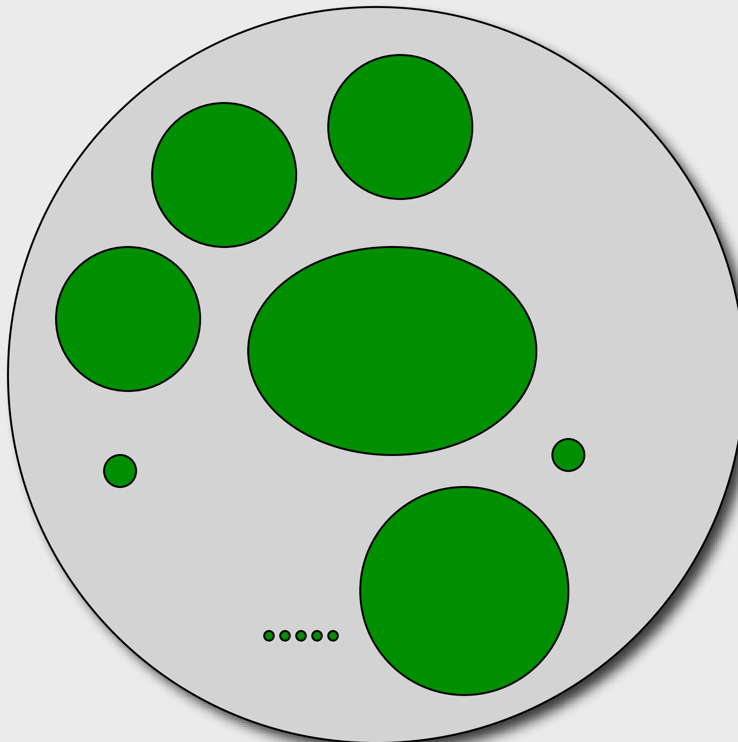


Computation in Hilbert Space



An Abstract Cortex

- Finite-dimensional representations of abstract spaces can be allocated disjoint regions in data space
- Field representations can be allocated to separated regions
- Analogous to regions in neural cortex



Decomposition of Computations

- Complex computations may be decomposed into simpler ones
- Variable regions provide interfaces between constituent computational processes
- For maximum flexibility: don't build in specific primitive processes
- How are primitive processes implemented?

Implementation of Primitive Computations

- There are several "universal approximation theorems" that make use of approximations of the form:

$$\mathbf{v} = \mathbf{F}(\mathbf{u}) \approx \sum_{j=1}^{H} \mathbf{a}_j r_j(\mathbf{u})$$

- Works for a variety of simple nonlinear "basis functions" r_j

(Re-)Configuration Methods

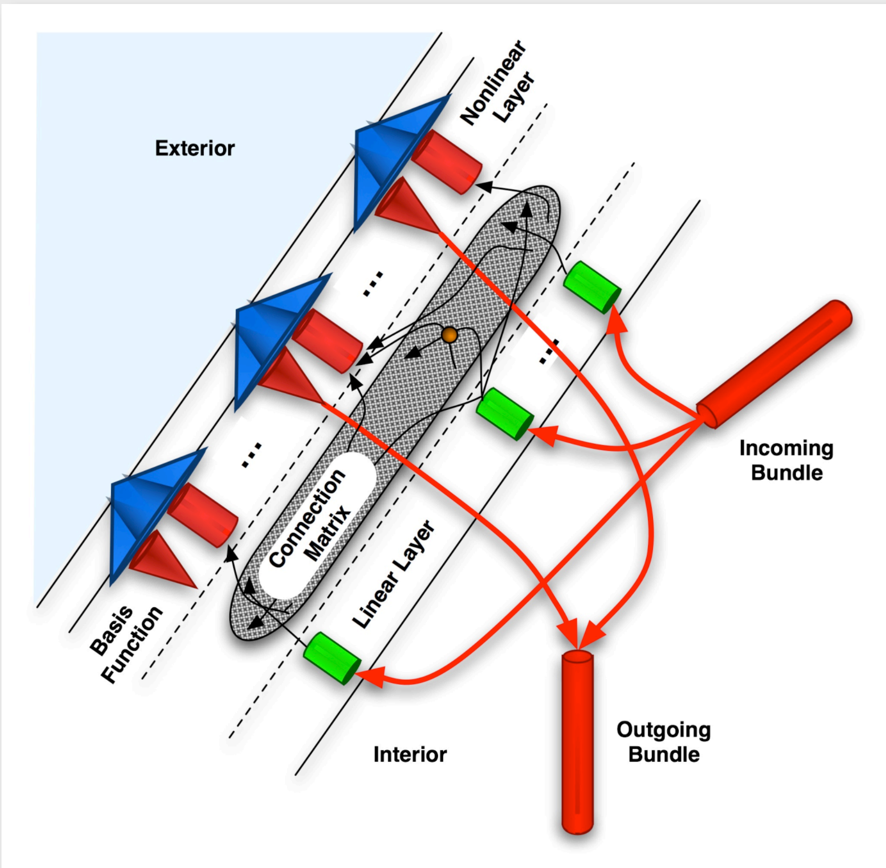
Overall Structure

- Variable (data) space
  - Large number of scalar variables for Hilbert coefficients
  - Partitioned into regions representing abstract spaces
- Function (program) space
  - Flexible interconnection (∴ 3D)
  - Programmable linear combinations
  - Application of basis functions

Depiction of UM Interior

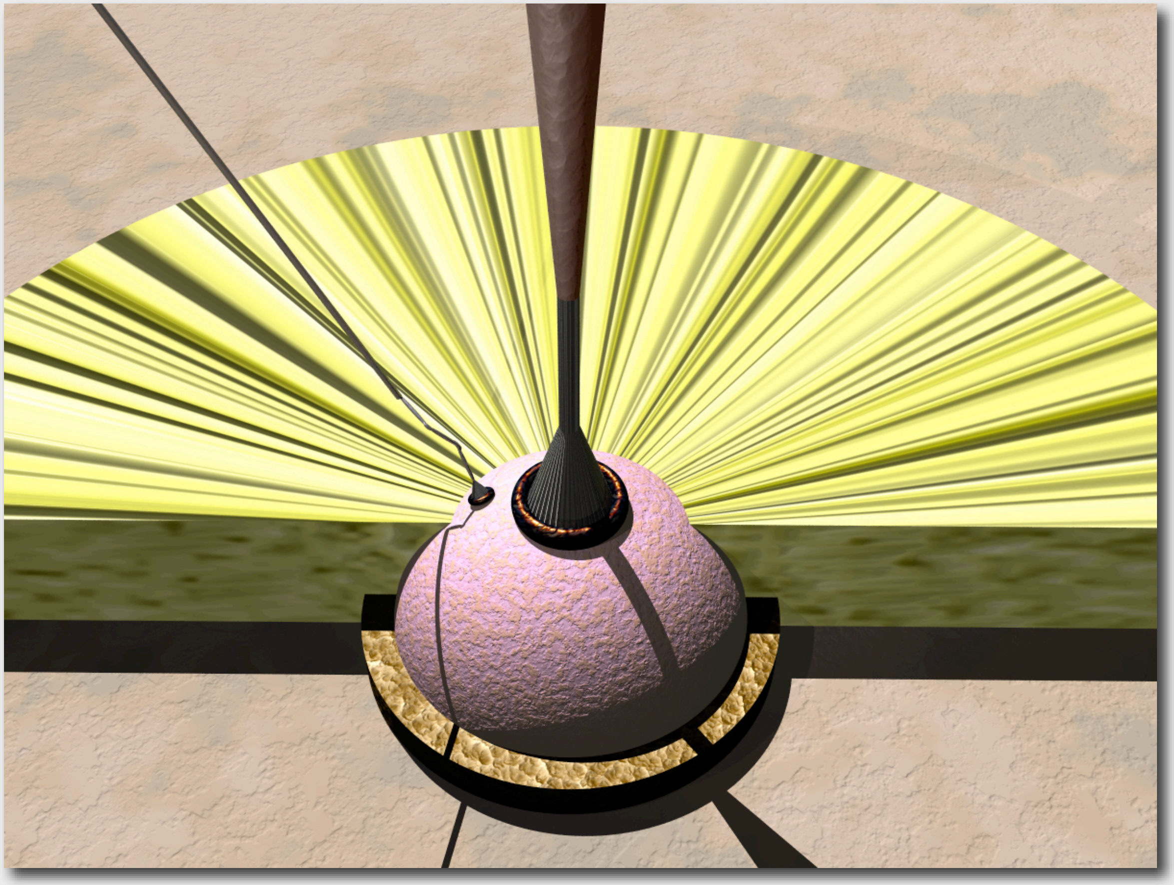
- Shell contains variable areas & computational elements
- Interior filled with solid or liquid *matrix* (not shown)
- Paths formed through or from matrix

Layers in Data Space

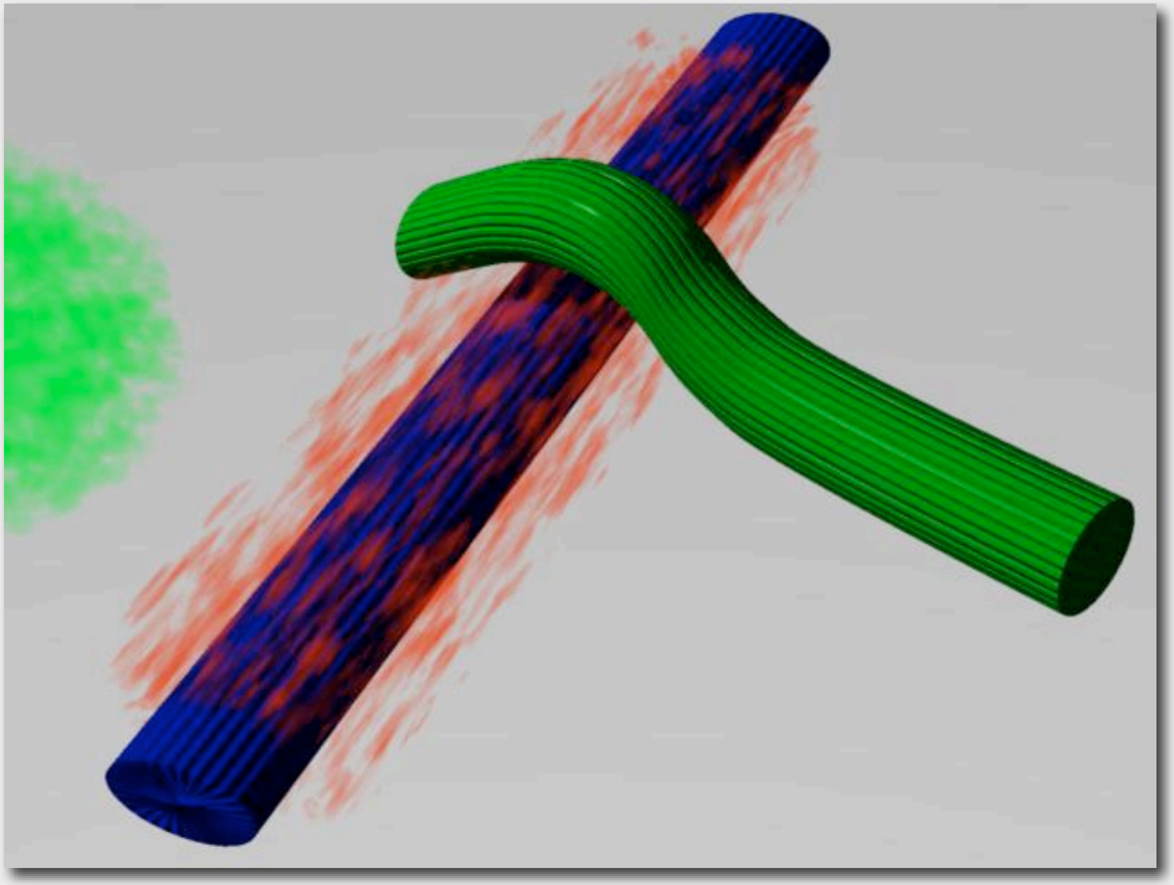


- Connection matrix has programmable weights
- Linear combinations are inputs to nonlinear basis functions
- Exterior access to both sides for programming

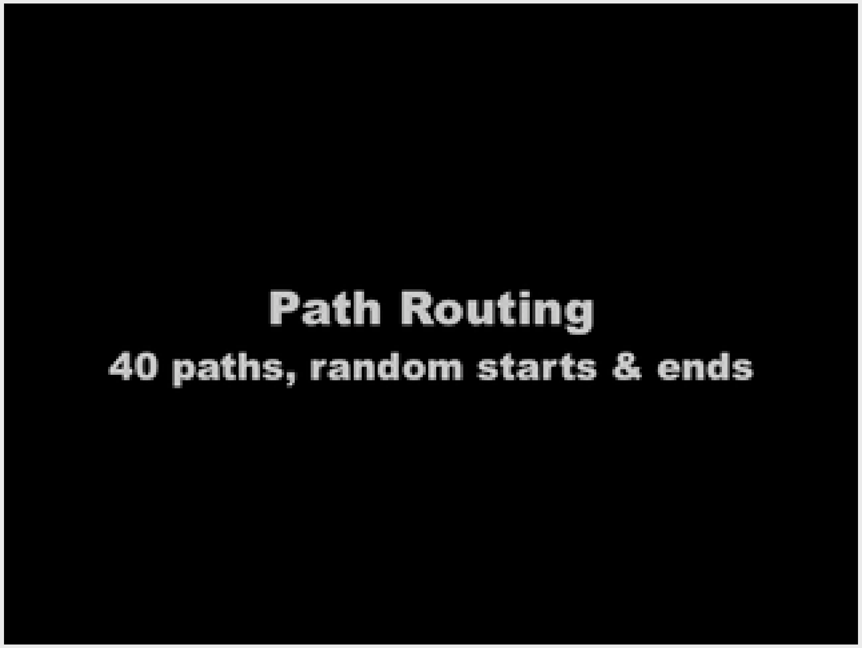
Depiction of UM Exterior



Diffusion-Based Path Routing

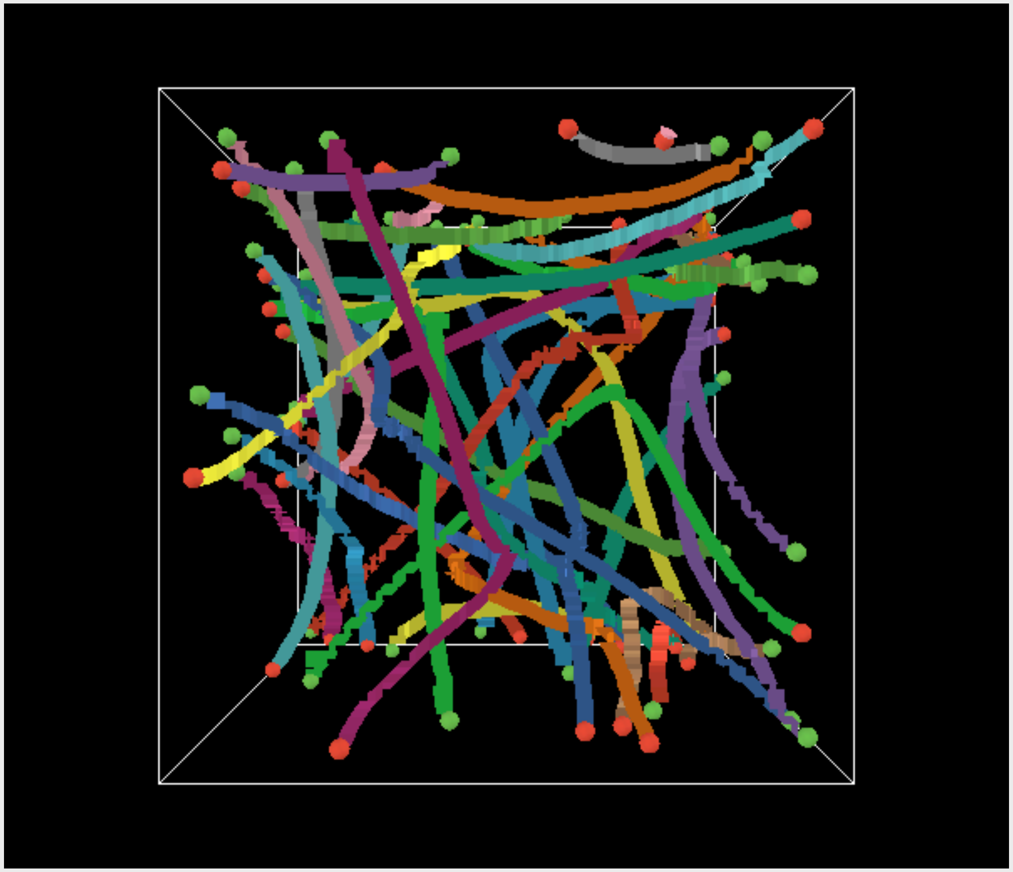


Example of Path Routing

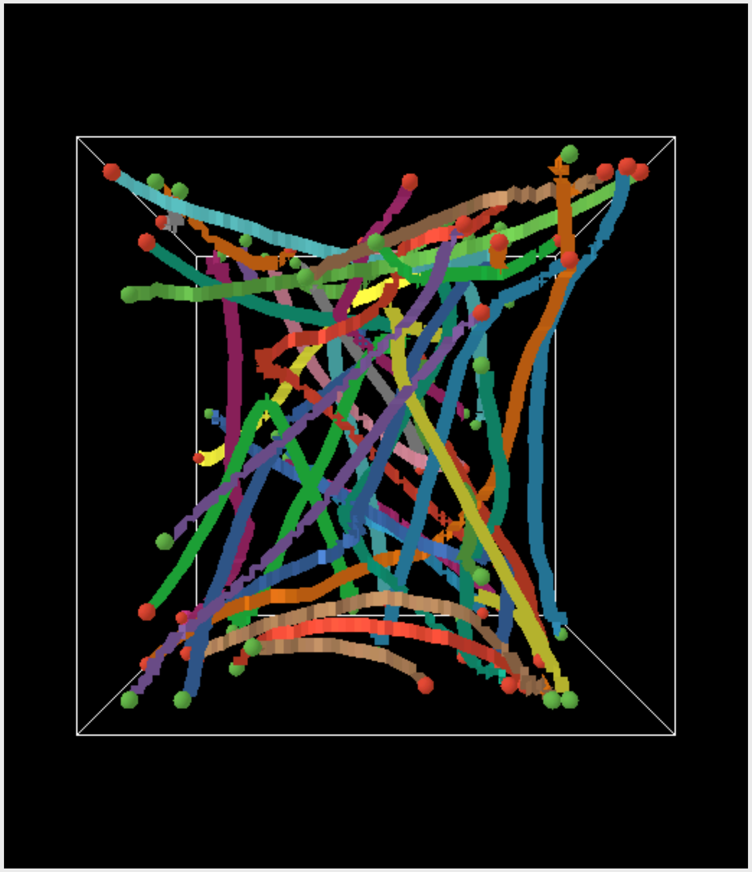


- Starts and ends chosen randomly
- Quiescent interval (for attractant decay) omitted from video
- Each path occupies ~0.1% of space
- Total: ~4%

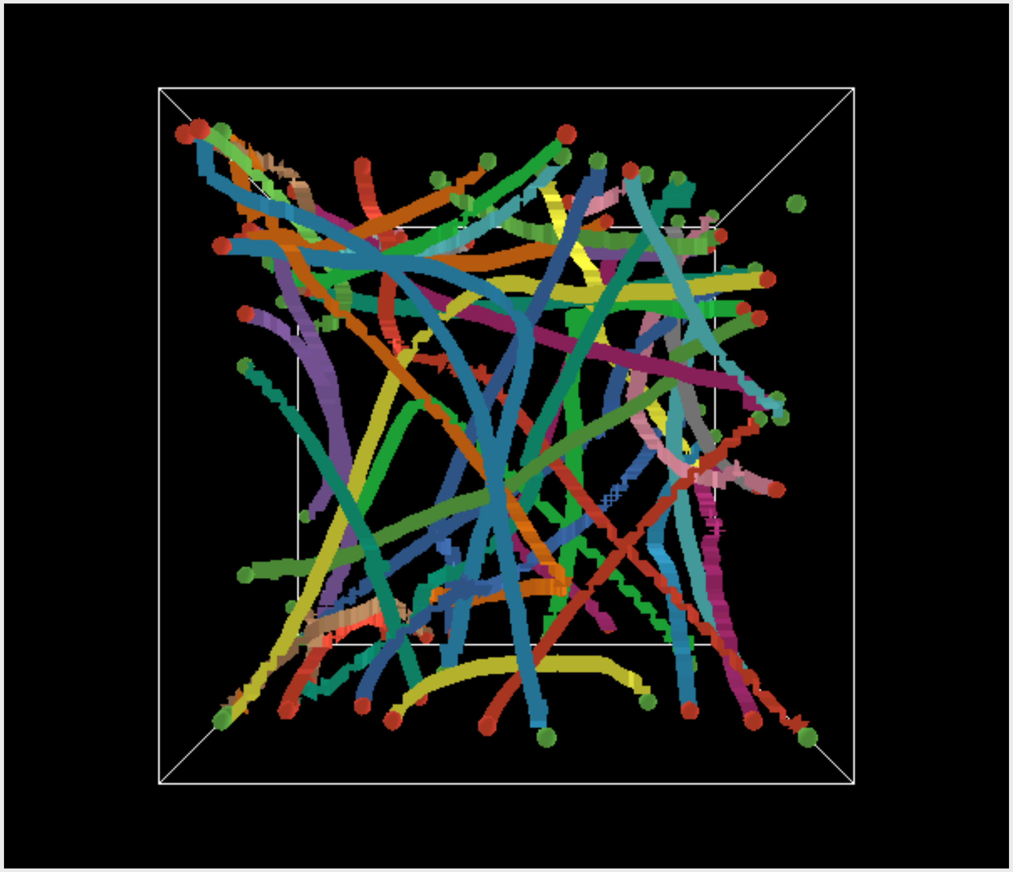
Front



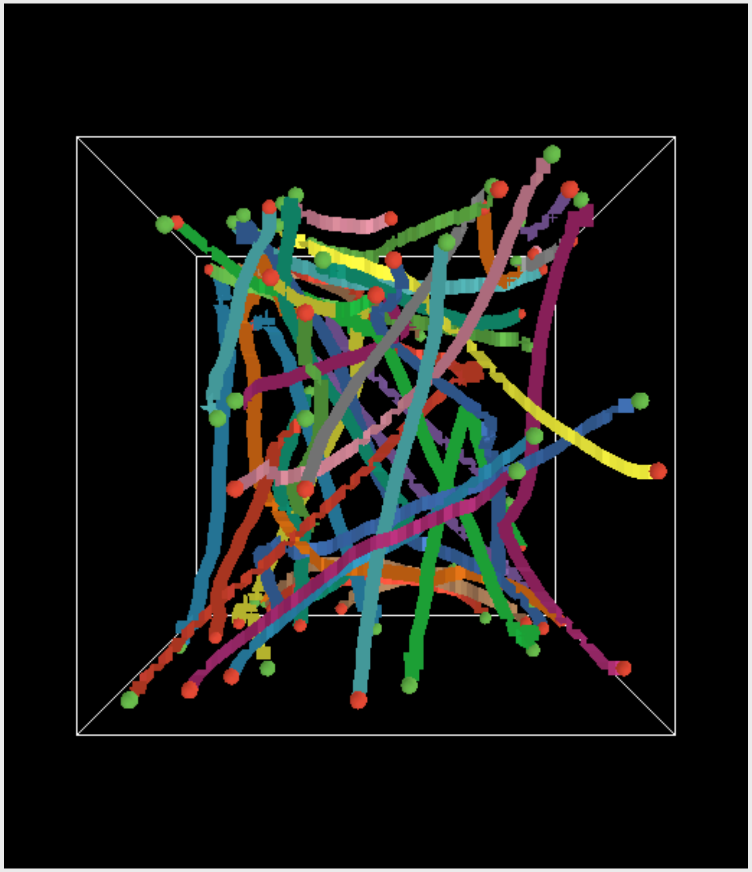
Right



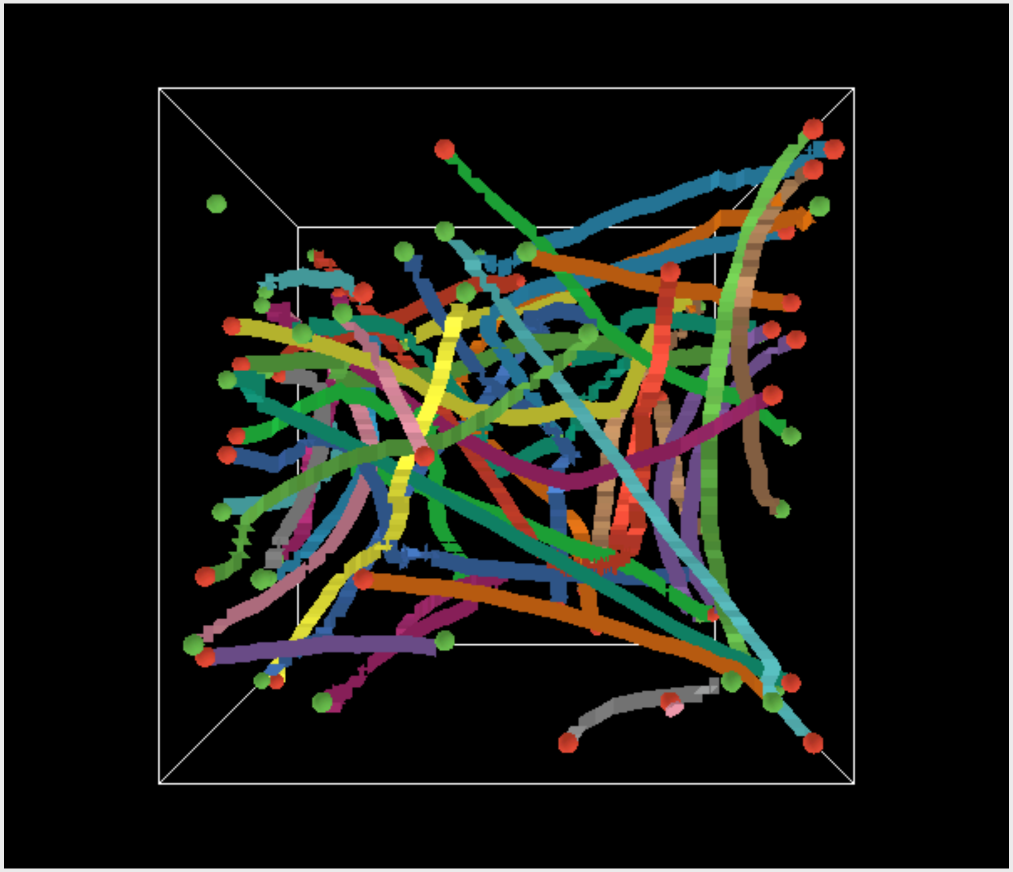
Back



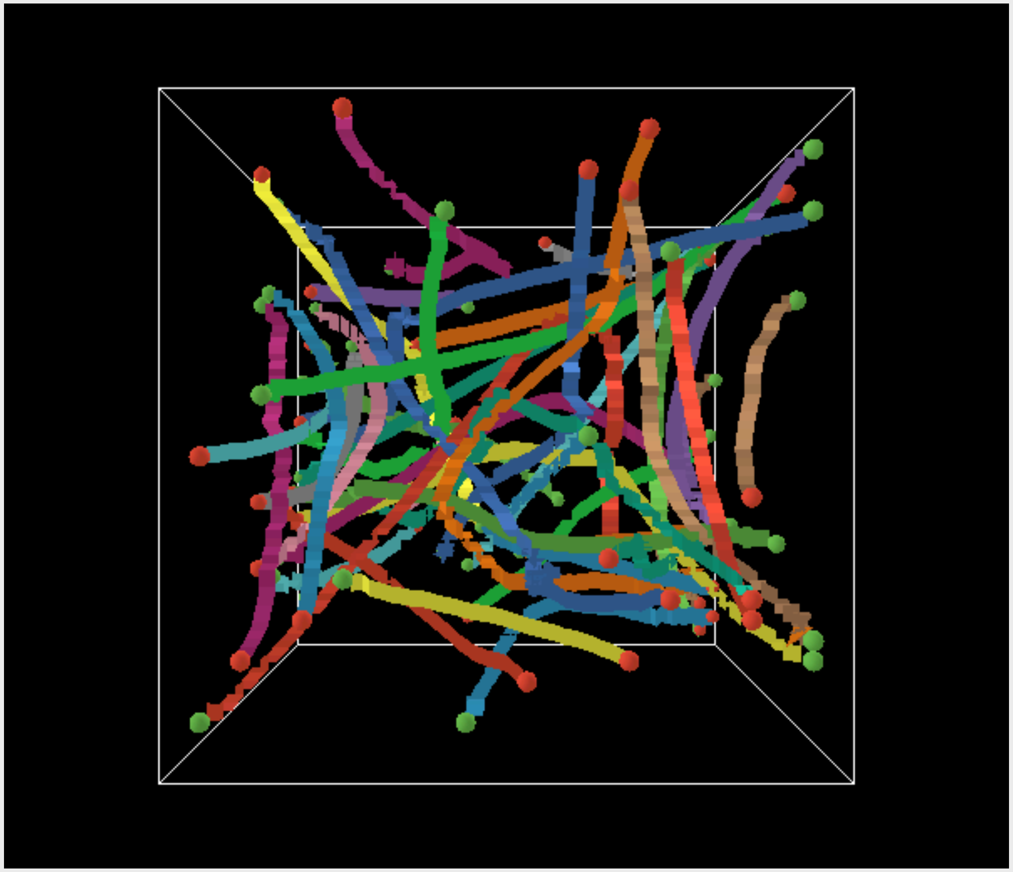
Left



Top



Bottom



Remarks

- More realistic procedure:
  - Systematic placement of regions
  - Order of path growth
  - Control of diffusion & growth phases
- General approach is robust (many variations work about as well)
- Paths could be formed by:
  - Migration of molecules etc.
  - Change of state of immobile molecules

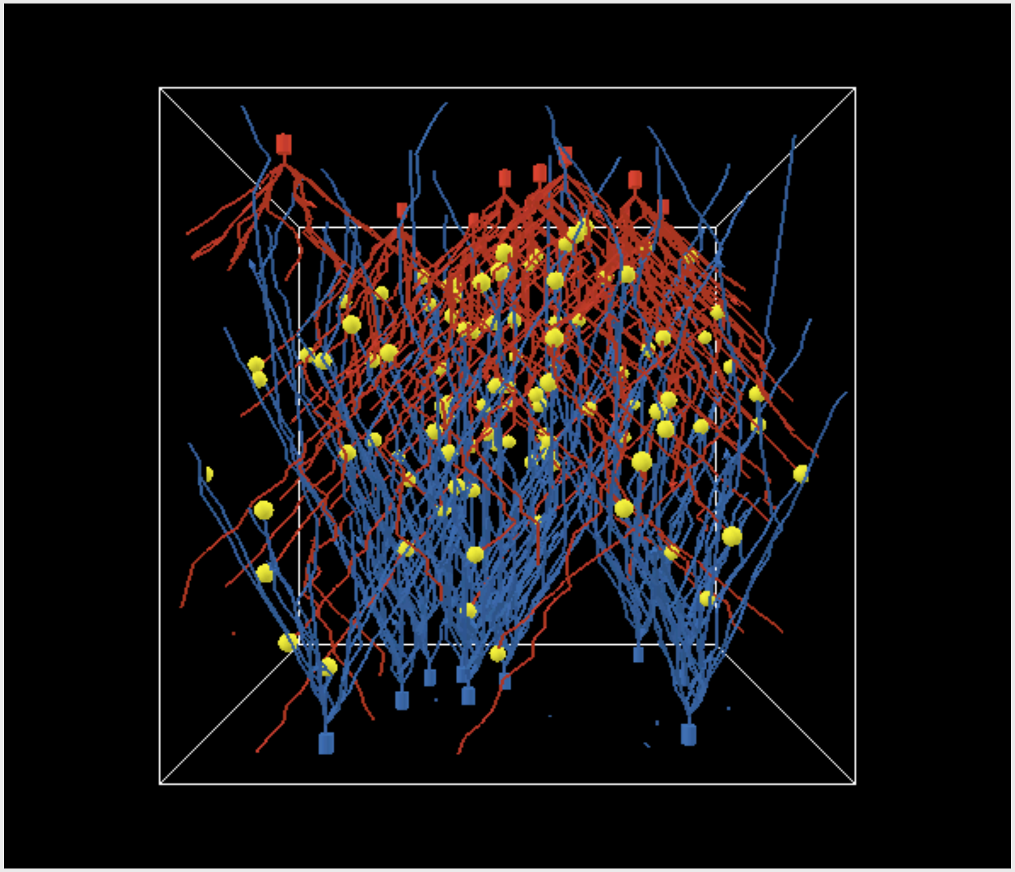
Example Connection-Growth Process

- Goal: approximately full interconnection between incoming "axons" (A) and "dendrites" (D) of basis functions
  - Doesn't have to be perfect
- Each A & D periodically initiates fiber growth
  - Growth is approximately away from source
- Fibers repel others of same kind
  - Diffusible, degradable repellant
  - Fibers follow decreasing gradient (in XZ plane)
- Contact formed when A and D fibers meet

Example of Connection Formation

- 10 random “axons” (red) and “dendrites” (blue)
- Simulation stopped after 100 connections (yellow) formed

Resulting Connections



Summary of U-Machine

- Permits computation on quite general abstract spaces (separable metric spaces)
  - Includes analog & digital computation
- Computation by linear combinations & simple nonlinear basis functions
- Simple computational medium can be reconfigured for different computations
- Potentially implementable in a variety of materials

Computational (Re-)Configuration of Systems

“Radical Reconfiguration”

- Ordinary reconfiguration changes connections among fixed components
- Radical reconfiguration of <u>transducers</u>
  - to create new sensors & actuators
- Radical reconfiguration of <u>processors</u>
  - to reallocate matter to different components
- Also for repair & damage recovery
- Requires rearrangement of atoms and molecules into new components
- Requires “molar parallelism”

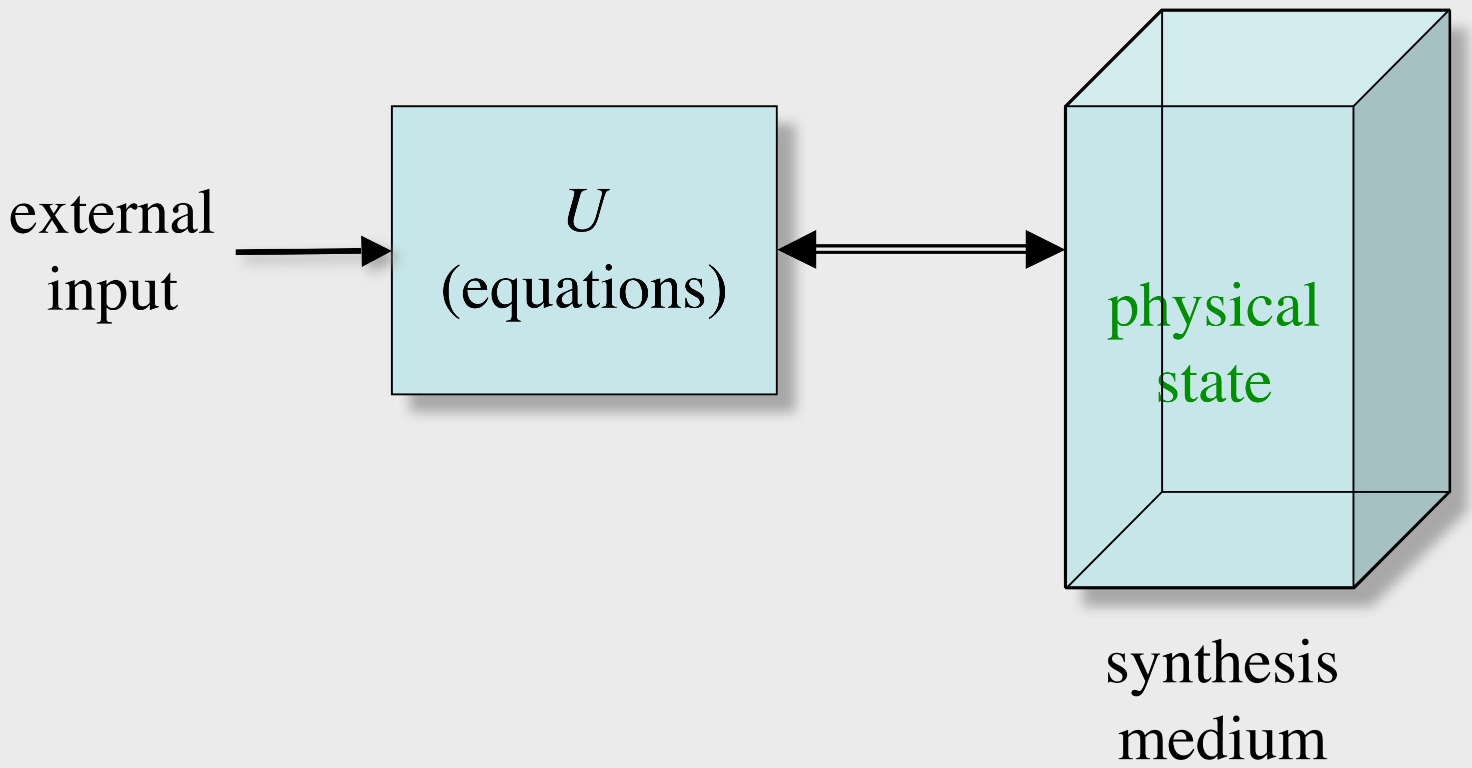
Computational Control of Matter

- A material process may be used as a substrate for formal computation
- Formal computation may be used to control a material process
- A material process may be a substrate for universal computation, controlled by a formal program
- A formal program may be used to govern a material process

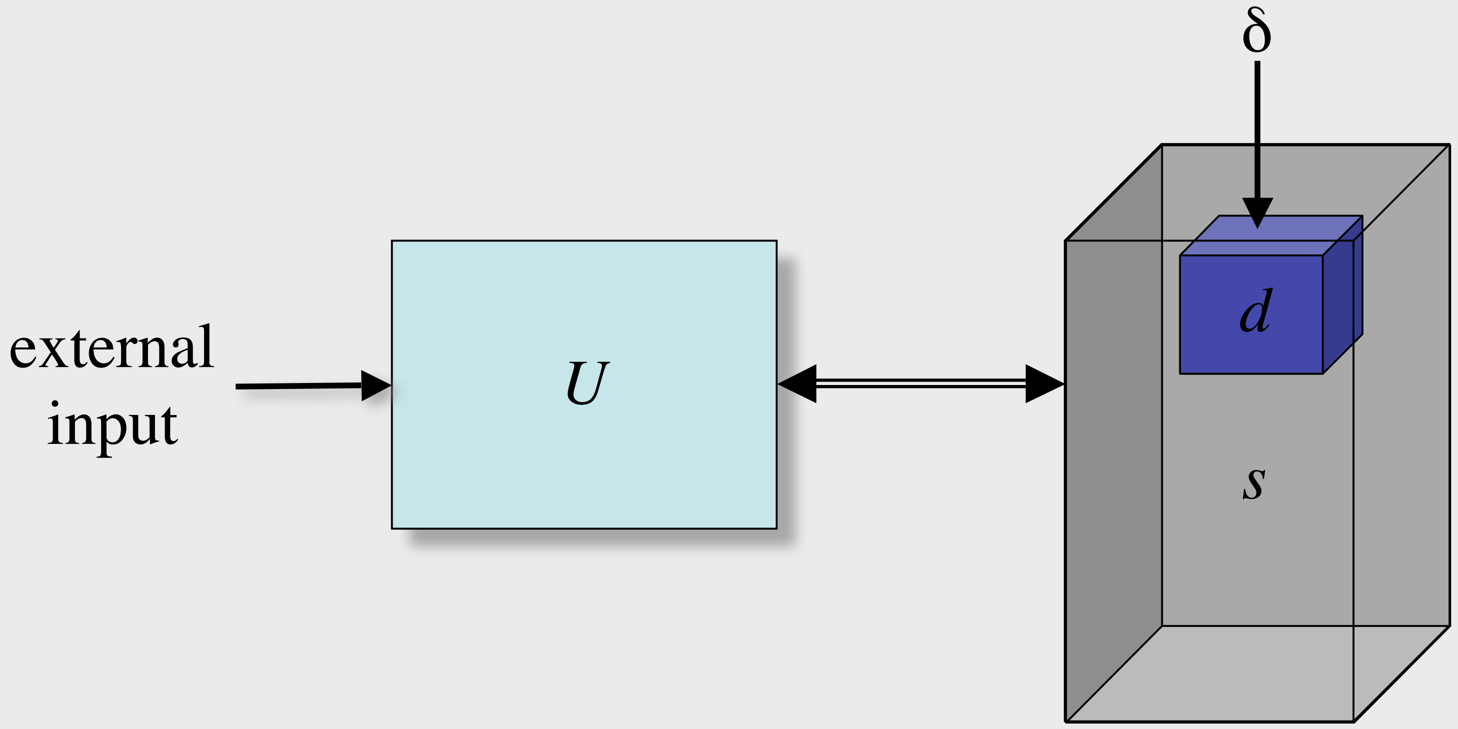
The Physical State as Synthetic Medium

- Computation controls physical state (as synthesis medium)
- Reconfigured computer is embodied in physical state
- Computation must be able to distinguish synthetically relevant physical states

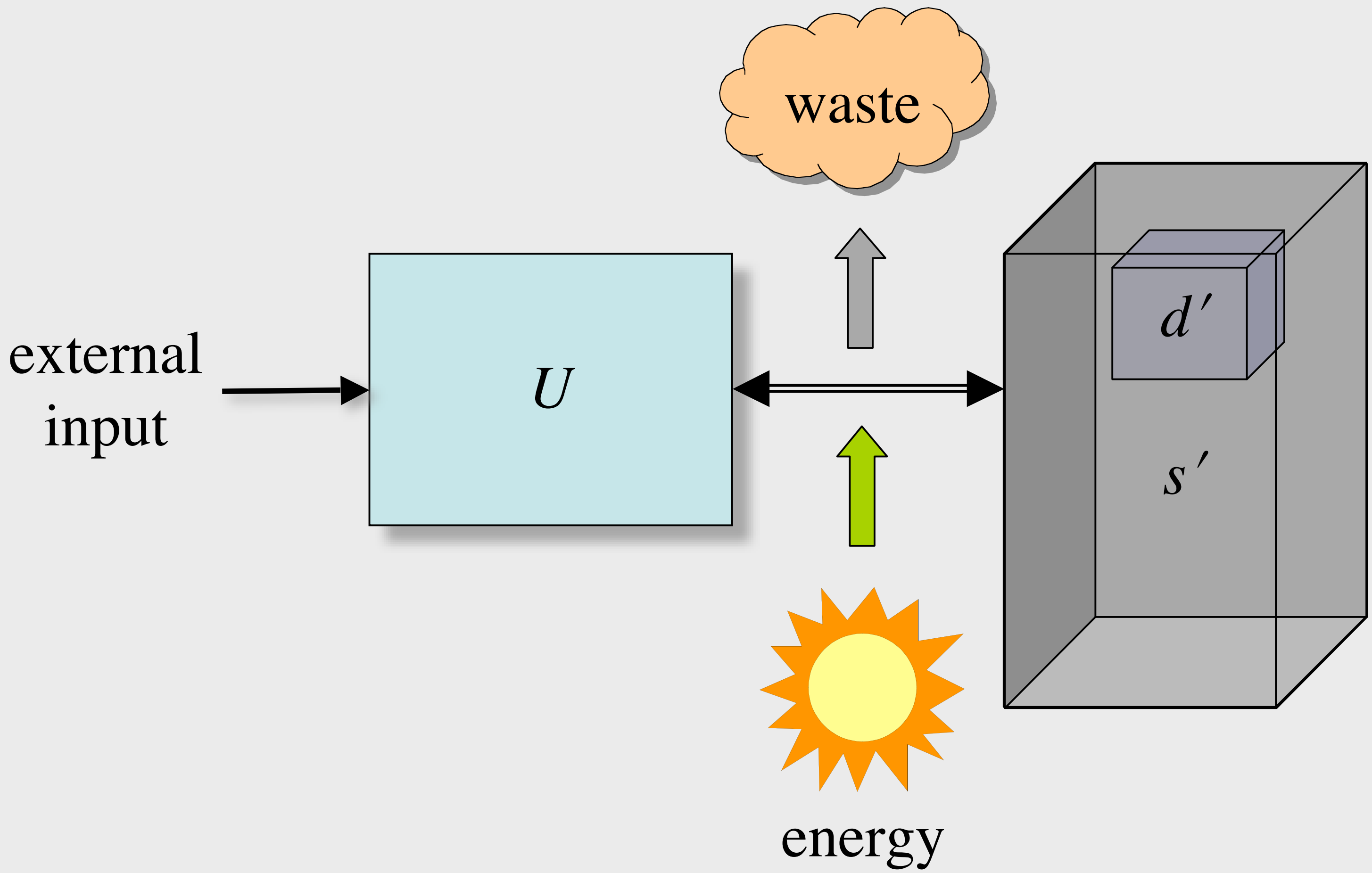
Universal Computer



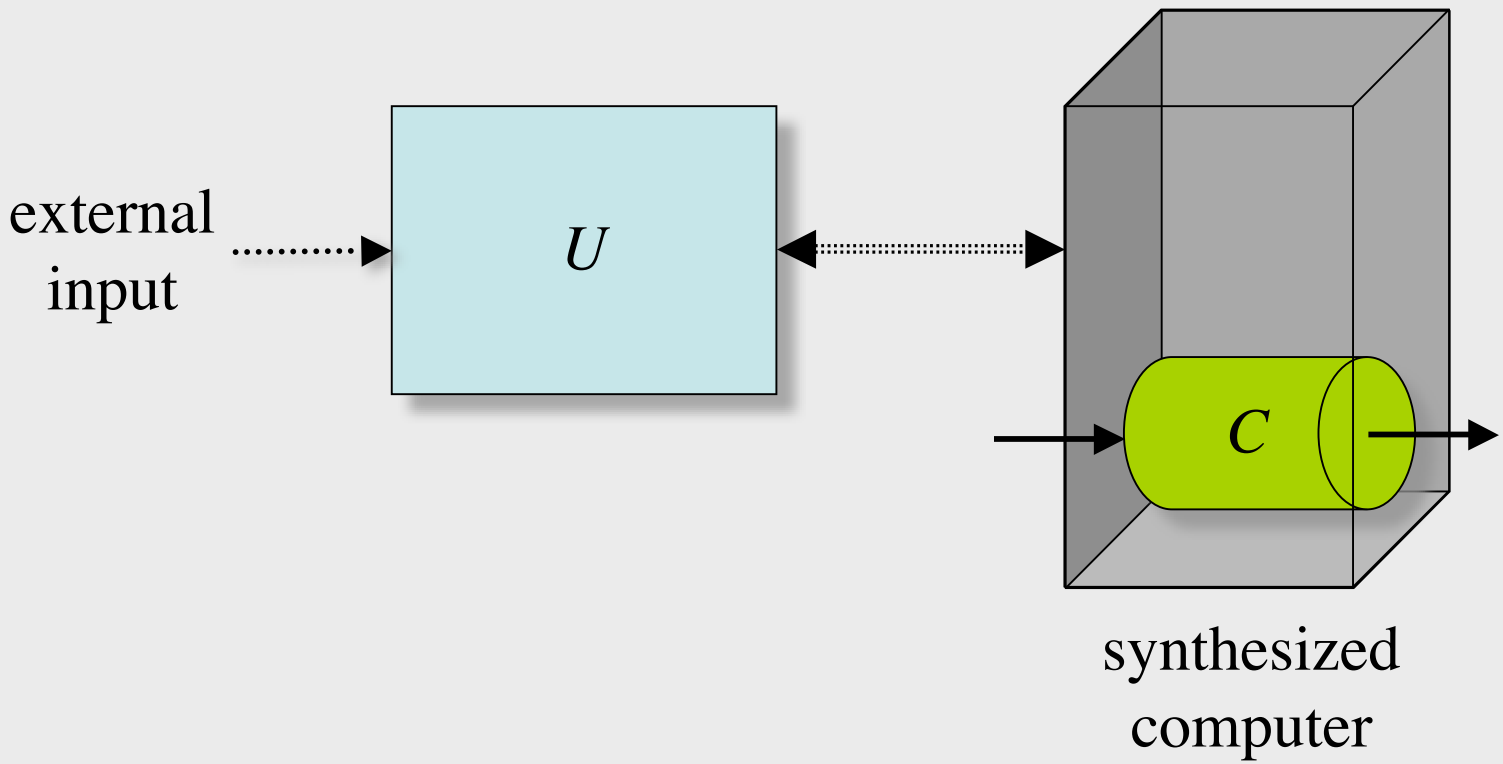
Initialization



Computation

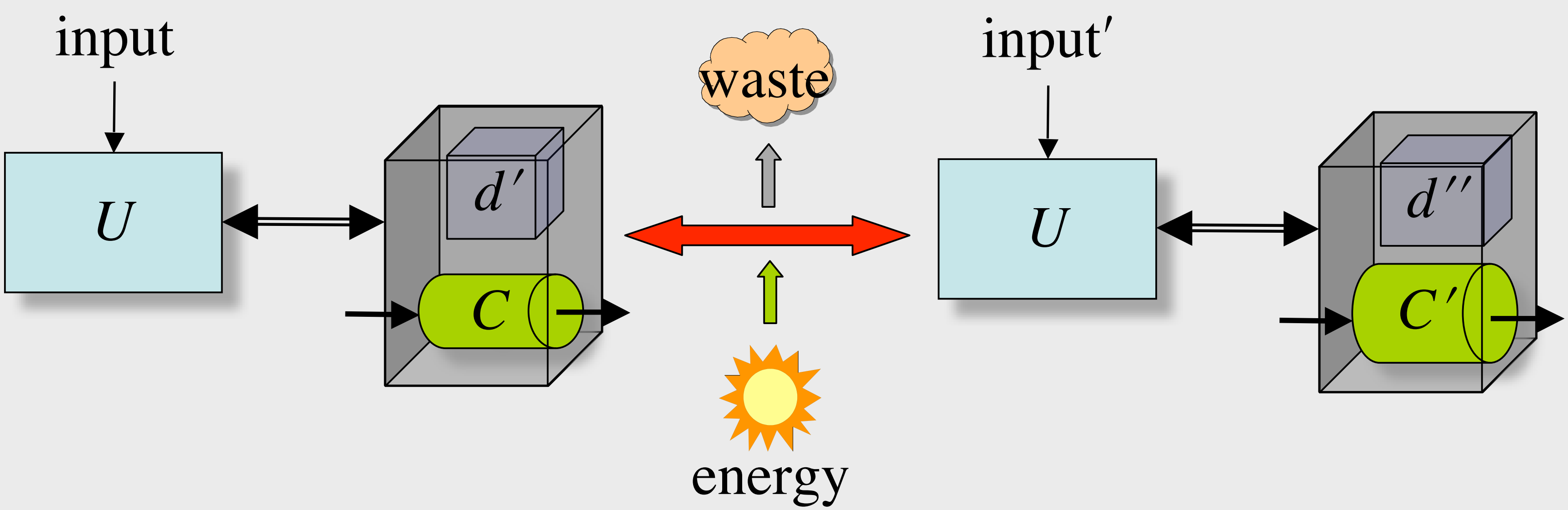


Completion



Equilibrium vs. Stationary Configurations

- Program terminates for equilibrium config.
- Program continues to run for stationary config.



Thermodynamics of a Configuration

- Either, configuration is a <u>stable state</u>
  - damage may shift to undesirable equilibrium
- Or, configuration is a <u>stationary state</u> of a non-equilibrium system
  - continuously reconfigures self
  - self-repair as return to original stationary state
  - adaptation & damage recovery as move to different stationary state

Useful Media for Computational Synthesis

- For pure computation, move as little matter & energy as possible
- For synthesis, need to control atoms & molecules as well as electrons
- Need sufficiently wide variety of controllable atoms & molecules
- Goal: structures on the order of optical wavelengths (100s of nm)

Models of Computation for Synthesis

- Need massive parallelism to control detailed organization of state
- Need tolerance to errors in state
  - synthesis program should be tolerant
  - configured computer should be tolerant

Locus of Control of Detailed Organization

- Reorganizing atoms & molecules
  ⇒ vast amount of detailed control
- *Heterosynthesis*
  - external configuration controller determines fine structure of medium (high bandwidth)
- *Autosynthesis*
  - external configuration controller determines general boundary conditions (low BW)
  - fine structure results from self-organization

General Model of Radical Reconfiguration

- Synthesis controller
  - low bandwidth to outside world
  - bandwidth to medium:
    - high for heterosynthesis
    - low for autosynthesis
- Synthetic medium
  - molar parallelism of interactions
    - simple for heterosynthesis
    - complex for autosynthesis
  - what are suitable synthetic media?

Simple Example: Reaction-Diffusion System

photos ©2000, S. Cazamine

- Many natural patterns can be explained by reaction-diffusion equations
- $\partial \mathbf{c} / \partial t = D\nabla^2\mathbf{c} + \mathbf{F}(\mathbf{c})$
- where **c** is a vector of concentrations,
  and D is a diagonal matrix of diffusion rates,
  and **F** is a nonlinear vector function

Example: Activation-Inhibition System

- Let σ be the logistic sigmoid function
- Activator *A* and inhibitor *I* may diffuse at different rates in *x* and *y* directions
- Cell is "on" if activator + bias exceeds inhibitor

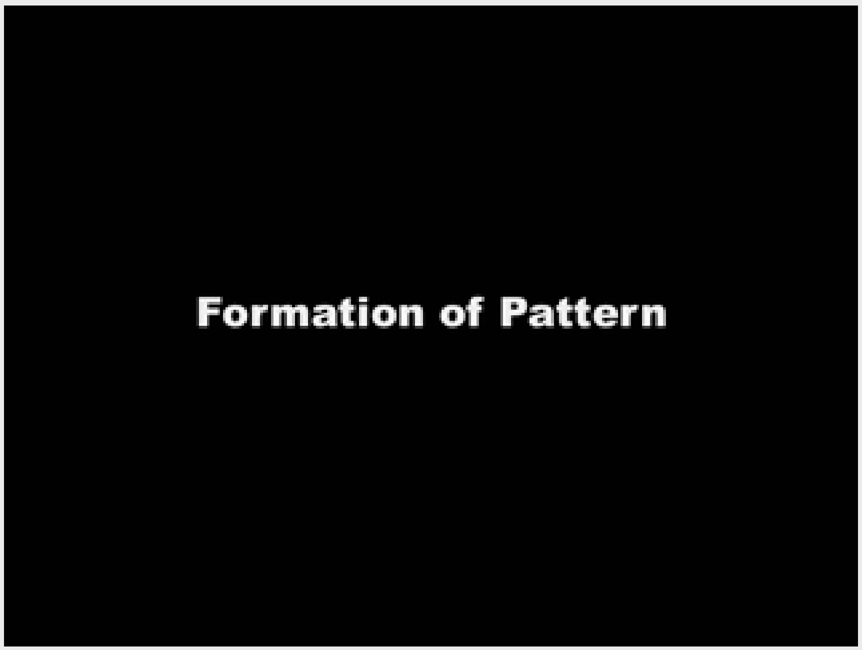
$$\frac{\partial A}{\partial t} = d_{\mathrm{Ax}} \frac{\partial^2 A}{\partial x^2} + d_{\mathrm{Ay}} \frac{\partial^2 A}{\partial y^2} + k_{\mathrm{A}} \sigma\left[m_{\mathrm{A}}\left(A + B - I\right)\right]$$

$$\frac{\partial I}{\partial t} = d_{\mathrm{Ix}} \frac{\partial^2 I}{\partial x^2} + d_{\mathrm{Iy}} \frac{\partial^2 I}{\partial y^2} + k_{\mathrm{I}} \sigma\left[m_{\mathrm{I}}\left(A + B - I\right)\right]$$

Double Activation-Inhibition System

- Two independently diffusing activation-inhibition pairs
- May have different diffusion rates in X and Y directions
  - In this example, $I_{1y} >> I_{1x}$ and $I_{2x} >> I_{2y}$
- Colors in simulation:
  - green = system 1 active
  - red = system 2 active
  - yellow = both active
  - black = neither active

Formation of Pattern



- Random initial state
- System stabilizes to < 1% cell changes
- Modest noise (annealing noise) improves regularity

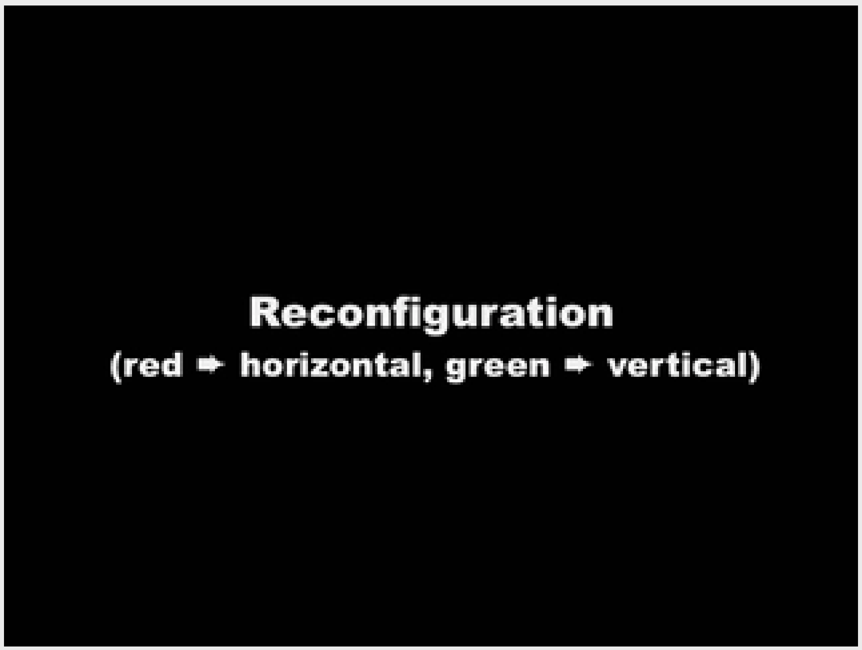
Stationary State

- System is being continually maintained in a stationary state
- Continuing change < 1%

Recovery from Damage

- Simulated damage
- Damage destroys activators & inhibitors as well as structure
- System repairs self by returning to stationary state
- No explicit repair signal

Reconfiguration: Orthogonal Structure



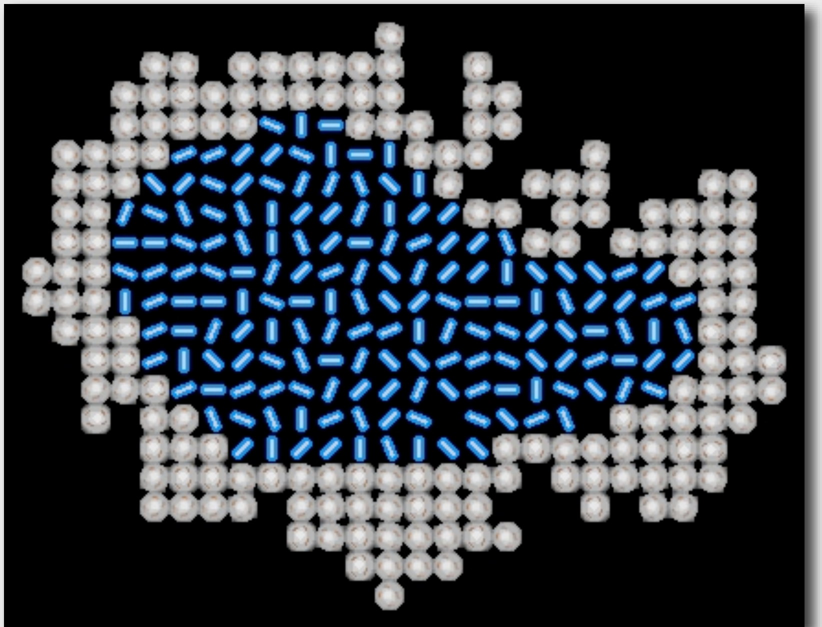
- Exchange inhibitor diffusion rates for systems 1 & 2
- Vertical stripes become horizontal
- Horizontal stripes become vertical
- No explicit reconfiguration signal

Summary of Radical Reconfiguration

- Computation can be used to rearrange matter
- External control of initial and boundary conditions
- Detailed structure by self-organization with molar parallelism
- Stationary states can be used for self-repair and adaptation

Programmable Microorganisms
for
Artificial Morphogenesis

Artificial Morphogenesis

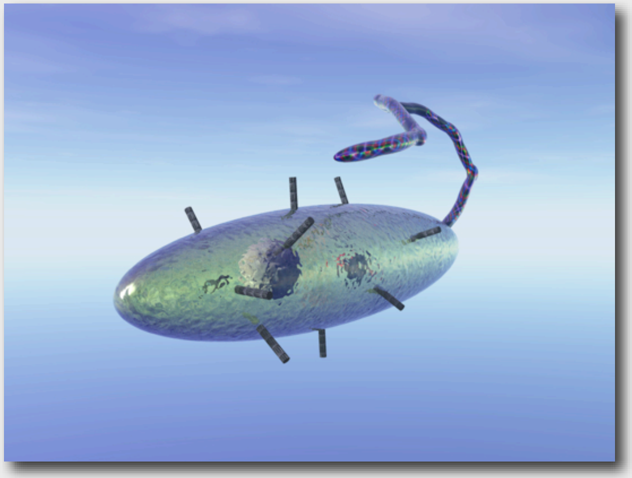


- Based on models of embryological development
- Cells migrate by local interaction & chemical signals
- Possible implemen-tation: "programmable" micro-organisms

Why Micro-Organisms?

- Micro-organisms can be viewed a micro-robots with capabilities for:
  - locomotion
  - sensing
  - control
  - simple (low-precision analog) computation
  - assembly
  - collective, coordinated behavior
  - reproduction
  - self-defense
  - metabolism (matter/energy acquisition, growth, repair)
- Can be genetically-engineered for our purposes

The Programmable Microorganism ("Promorg")



- Noncoding DNA can be used for "genetic circuits"
  - in eukaryotes, 10–70%
  - equivalent of about 3000 genes in yeast
- Equipped with an assortment of generally useful sensors & receptors (especially for self-organization)
- Special-purpose modifications for particular applications
- Research: principles of design & self-organization

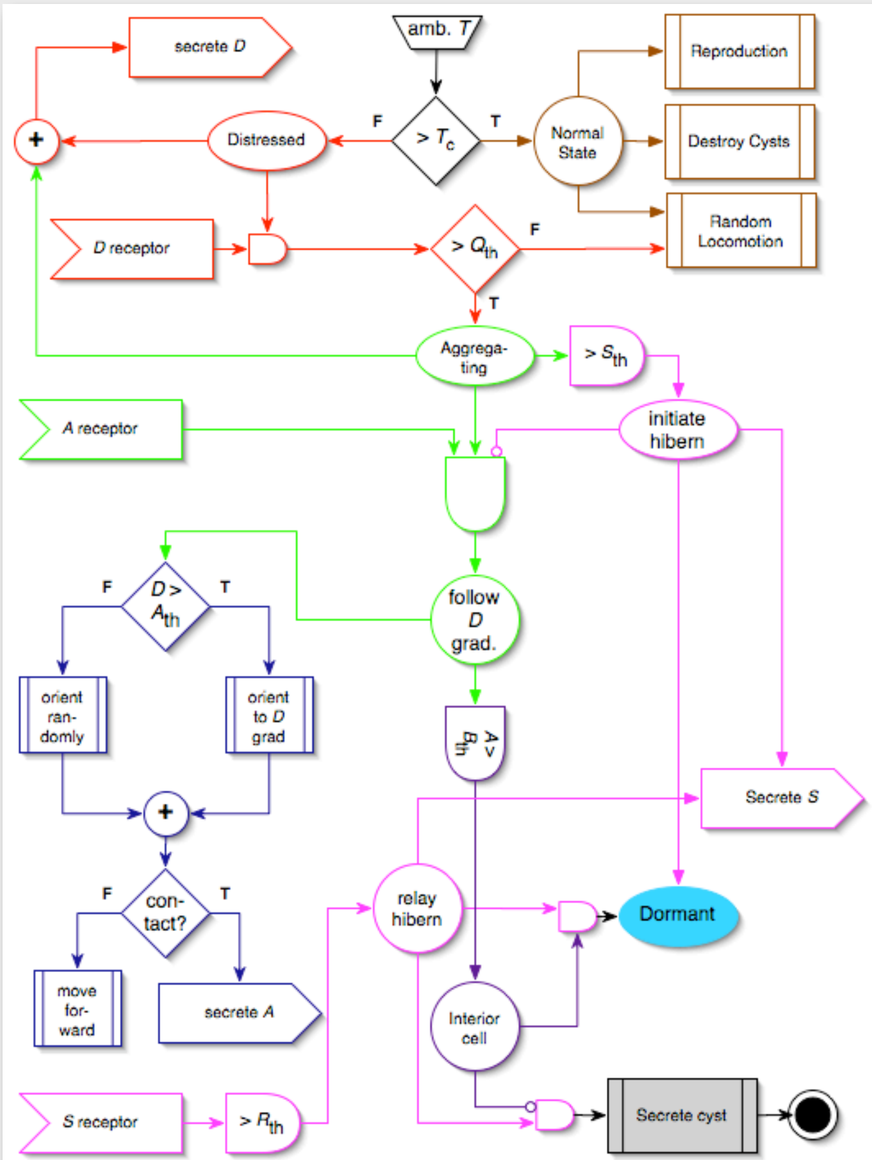
Tentative Capabilities

- Neutral proteins for programmable control
  - gene regulatory & coding sequences
  - connections can be regulated by external signals
- Membrane & cytoplasm receptors for:
  - chemical signals
  - light, etc.
- Effectors
  - cilia for locomotion
  - cell adhesion
  - exocytosis
  - programmed cell death

Simulation of
Self-Organized Aggregation &
Protective Differentiation of
Simple Autonomous Agents

Goal

- Adverse conditions aggregate into dense colonies
- Differentiate into outer “boundary” and inner “interior” cells
- Interior cells become dormant until favorable conditions return
- Boundary cells secrete protective cyst material and die
- When favorable conditions return, dormant cells reanimate and break out of protective case



Control Diagram for Cells

Normal State

- Organisms wander
- & reproduce
- Until some environmental condition ("temperature") becomes unfavorable

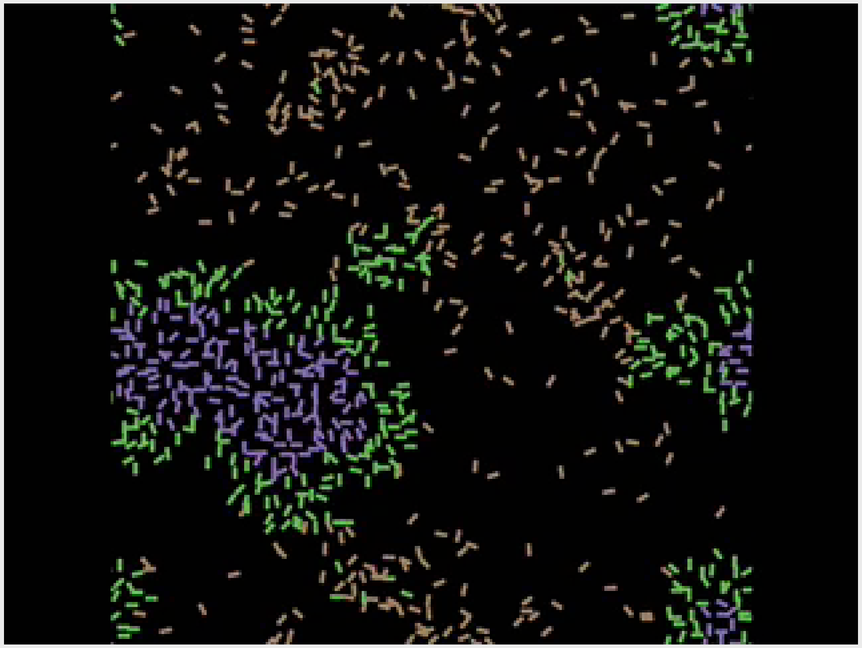
Distressed State

- Distressed cells emit Distress signal (red)
- If concentration exceeds quorum threshold, cells climb gradient
- Concentration of distress signal shown in red

Aggregated State

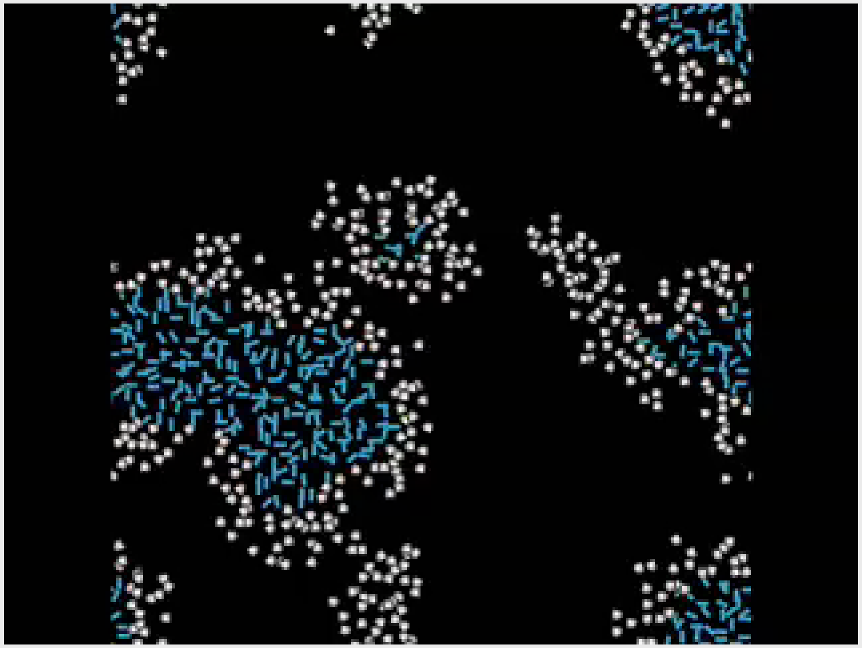
- Cells in contact emit Aggregated signal (green)
- If concentration is above Boundary threshold, cell enters Interior state (purple)
- Else remains in Boundary state (green)

Dormant State



- When Distress signal exceeds signaling threshold:
- Cell emits rapidly diffusing Spore-formation signal (magenta)
- Interior cells enter dormant state (blue)
- Boundary cells form cysts (grey) & die

Return of Favorable Conditions



- Surviving cells reanimate
- Destroy cyst material
- Return to Normal state

Conclusions

- Demonstrates useful collective behavior based on:
  - simple control mechanisms
  - diffusible chemical signals
- Quorum sensing & aggregation of cells
- Differentiation of function
- Assembly of a simple protective structure

Molecular Combinatory Computing for Nanostructure Synthesis & Control

Supported by NSF
Nanoscale Exploratory Research Grant

skip

Definition

- *Intelligent* Matter
  - a material in which individual molecules or supra-molecular clusters function as agents to accomplish a purpose
- *Programmable* Intelligent Matter
  - a program controls the behavior of the material at the molecular level
- *Universally* Programmable Intelligent Matter
  - small set of molecular building blocks that can be rearranged to accomplish any purpose describable by a computer program
  - power of Universal Turing Machine

Non-Traditional Models of Computation

- Need to explore non-traditional models of computation more closely matched to physical processes
- Discrete (digital) computation
- Continuous (analog) computation
- Hybrid computation

Continuous (Analog) Computation

- Exploit continuous physical processes for computation
- Need to identify small set of widely useful systems of DEs & PDEs
- Research in universal analog computers is relevant
- Field computation

Discrete (Digital) Computation

- Alternatives to Boolean logic
- Need information representation more closely matched to molecular & sub-molecular structures
- Need elementary operations more closely matched to molecular & sub-molecular processes

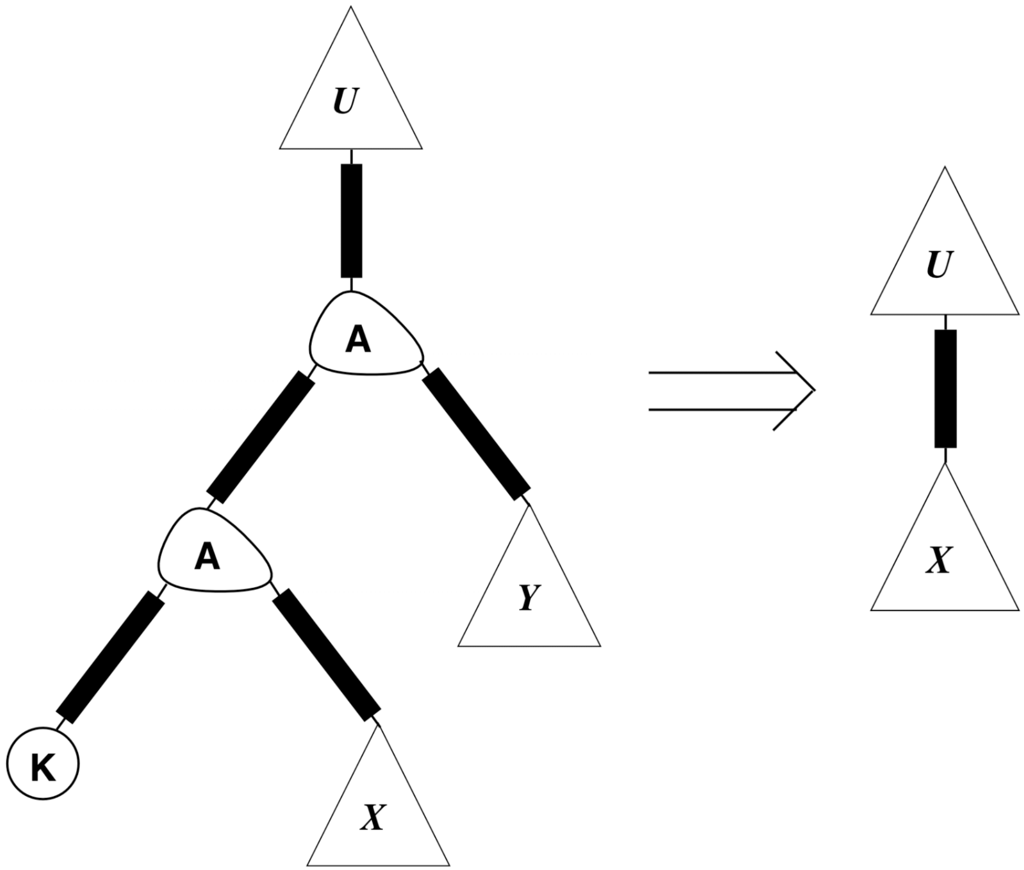
Alternative Models of Discrete Computation

- Many Turing-equivalent models were developed in the early 20th century
  - e.g., Post productions, Markov algorithms, lambda calculus, combinatory logic, McCulloch-Pitts cells
- Cellular Automata are promising
  - need to be universal in a relevant way

Molecular Combinatory Computing

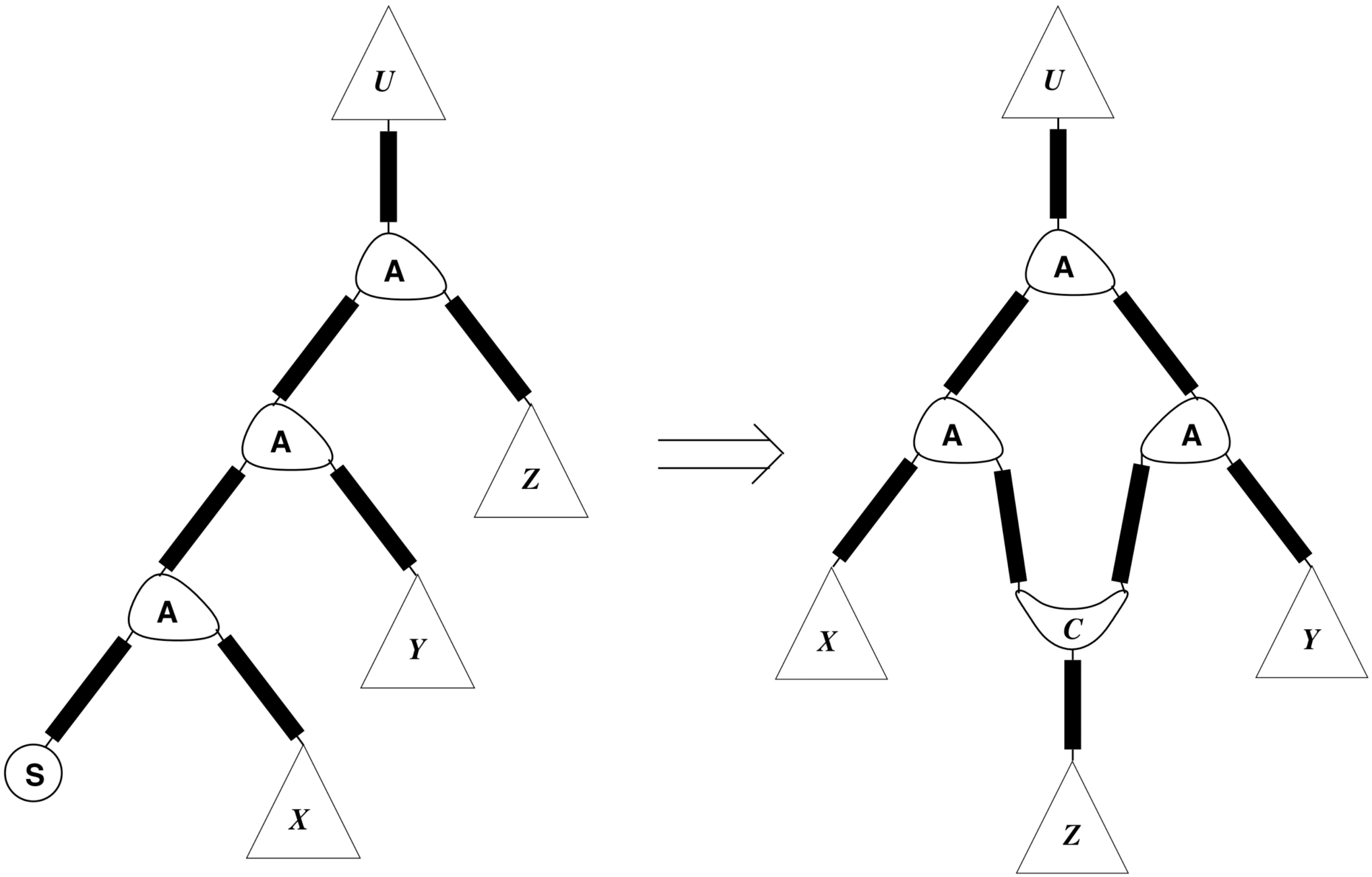
- Systematic approach to nanotechnology based on small set of MBBs
- Combinatory logic
- Computational universality from two substitutions (+ a few more)
- Substitutions may be done in any order or in parallel

K-Substitution



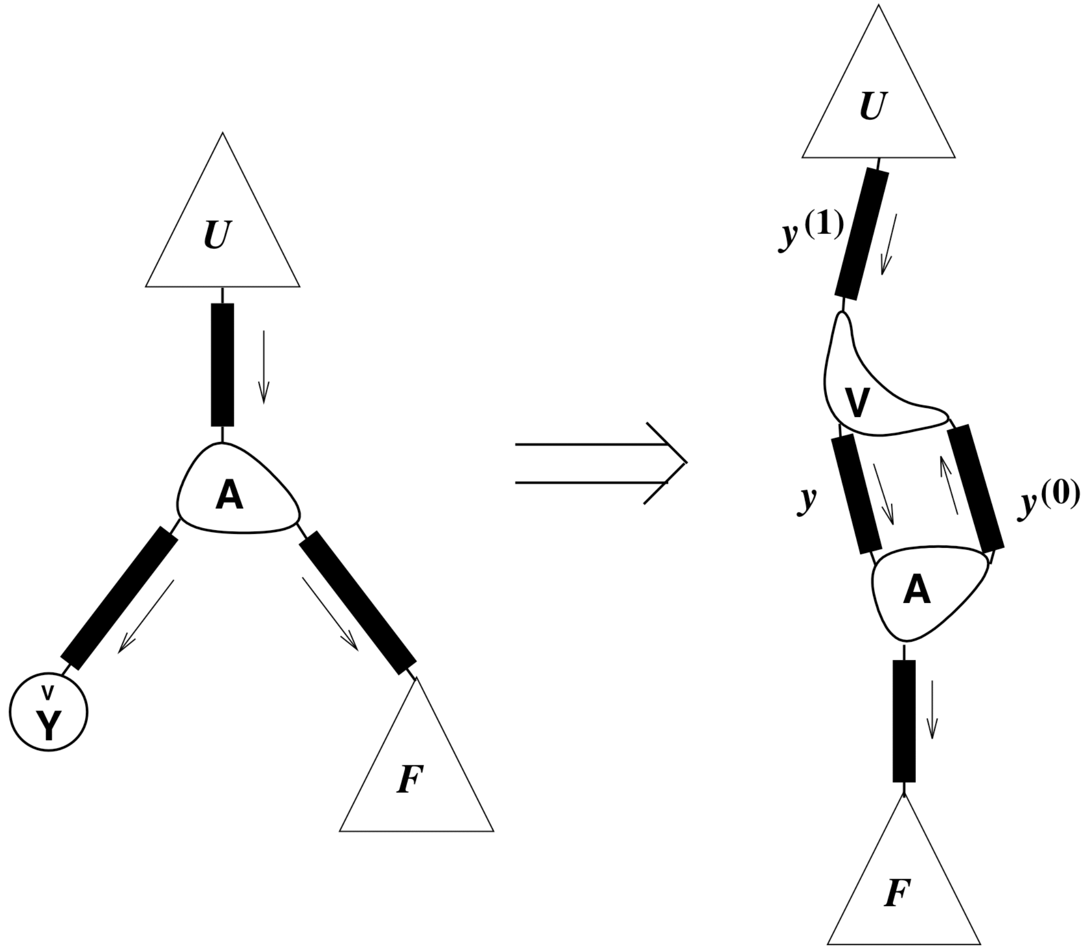
$$((\mathsf{K}\ X)\ Y) \Rightarrow X$$

S-Substitutions



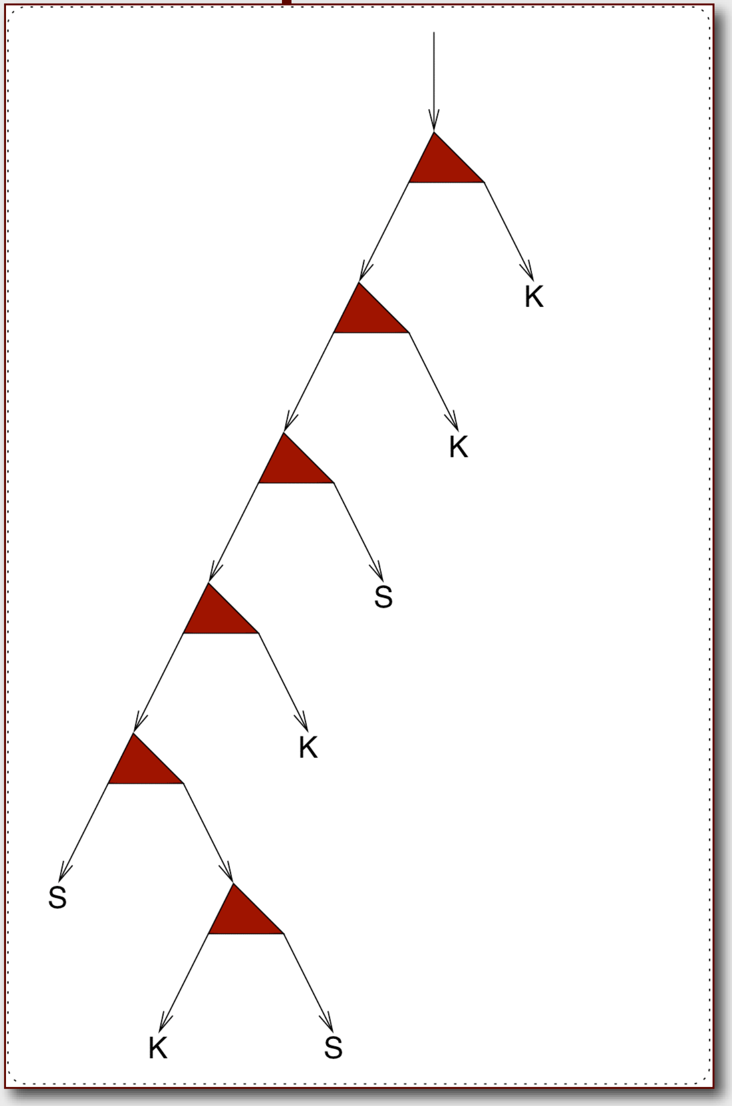
$$(((\mathsf{S}\ X)\ Y)\ Z) \Rightarrow ((X\ Z)\ (Y\ Z'))$$

Ý-Substitution



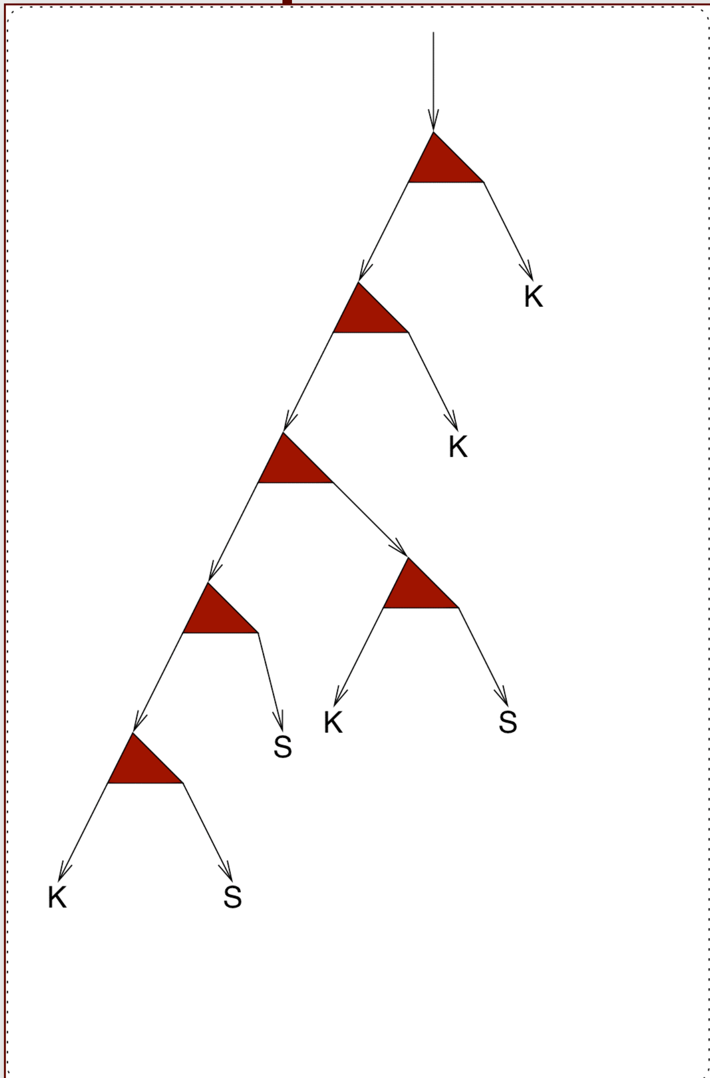
$$(\acute{Y}\ F) \Rightarrow y^{(1)} \quad \text{where } y \equiv (F\ y^{(0)})$$

Example of Simple Computation



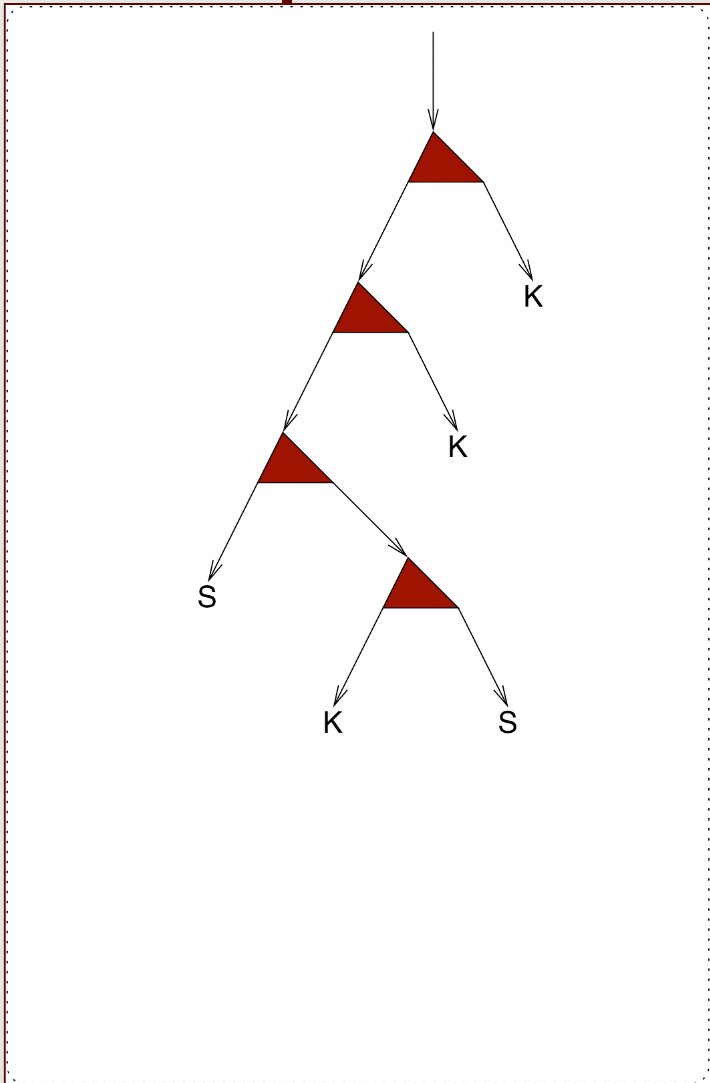
((S(KS)K)S)
⇒ (((KS)S)(KS))

Example of Simple Computation



((KS)S) ⇒ S

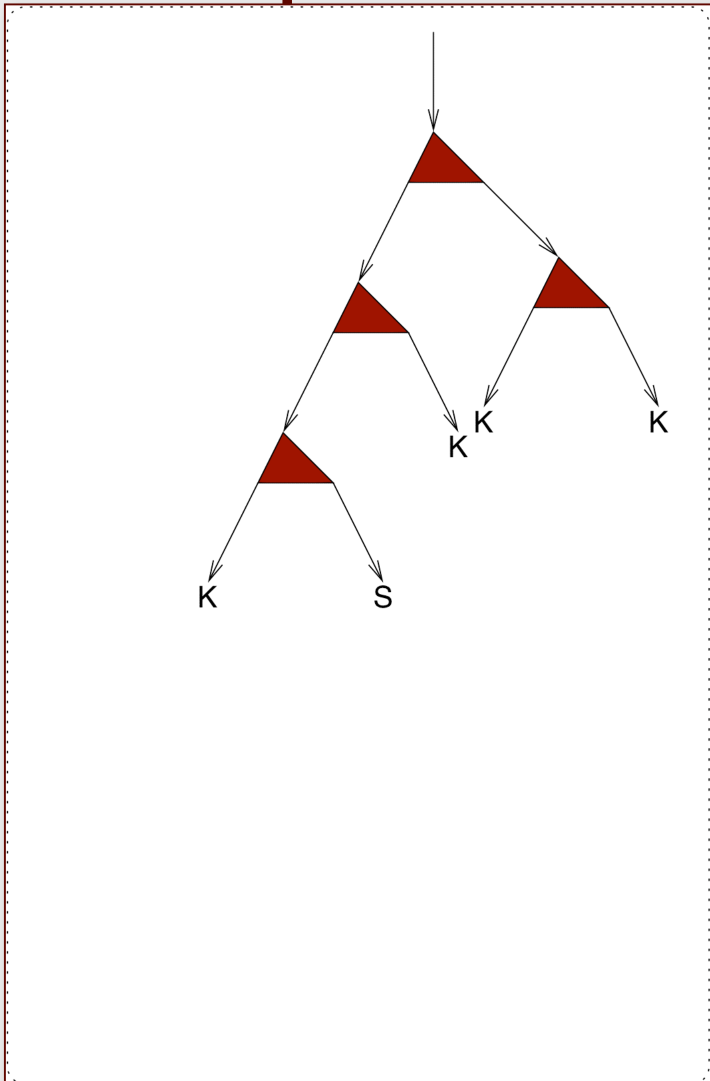
Example of Simple Computation



(((S(KS)K)K)

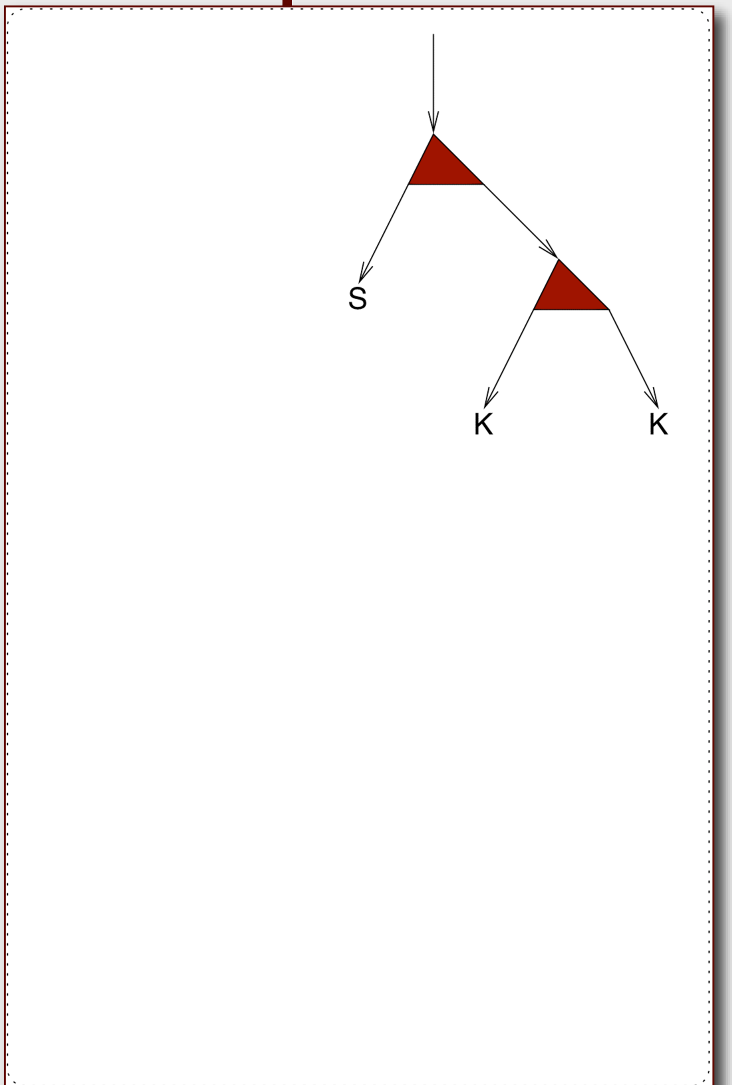
$\Rightarrow$ (((KS)K)(KK))

Example of Simple Computation



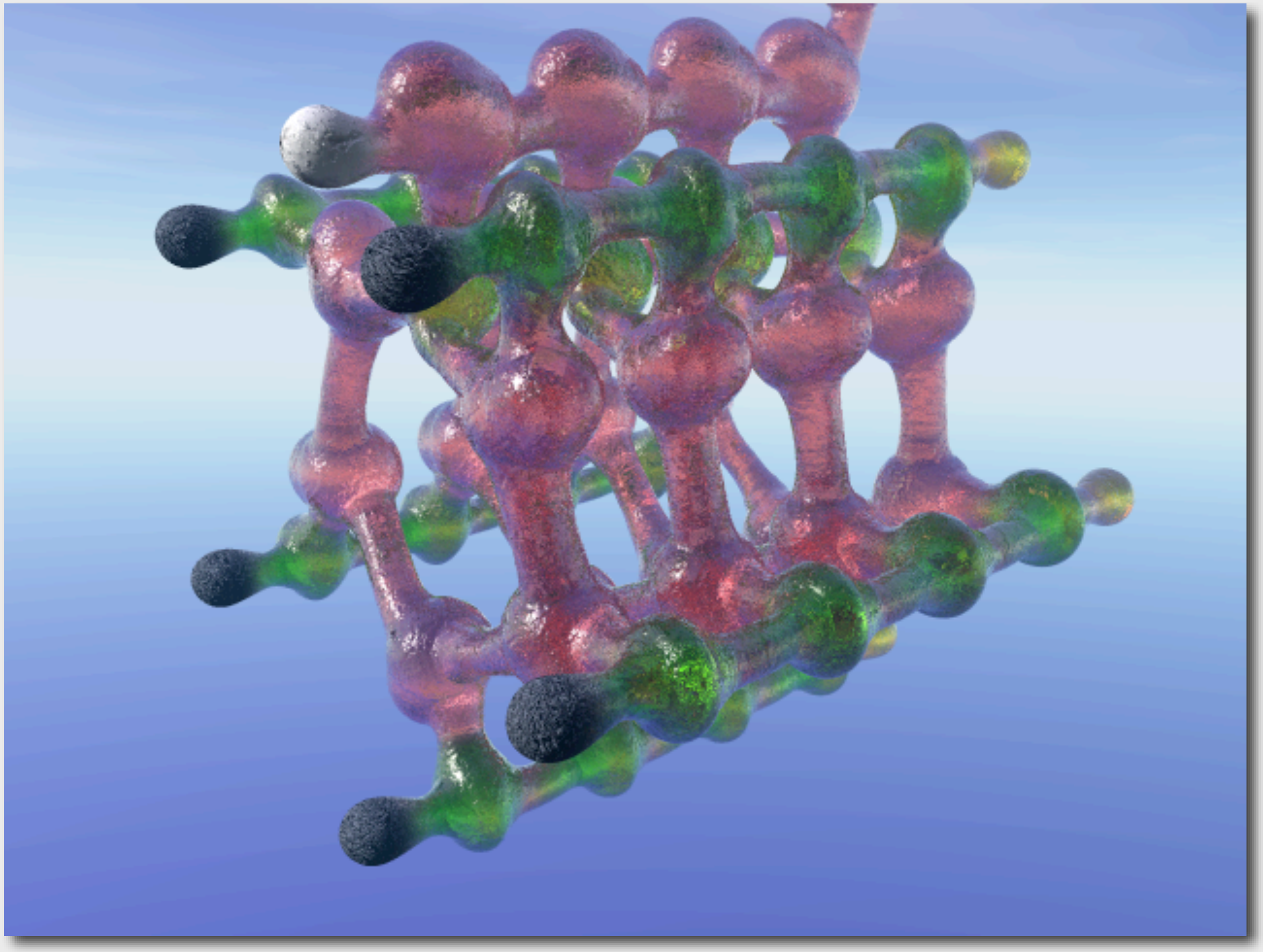
((KS)K) $\Rightarrow$ S

Example of Simple Computation



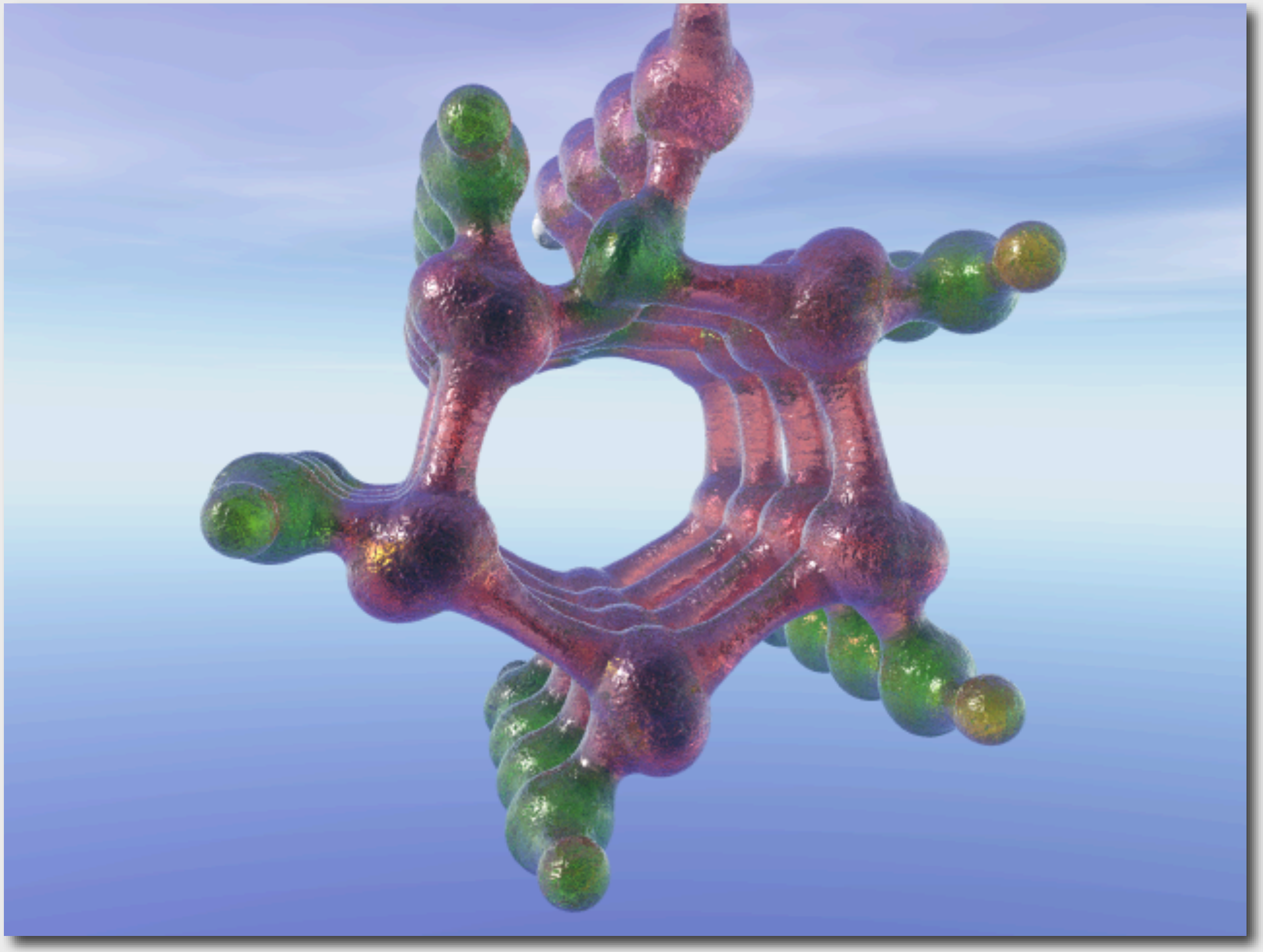
result = (S(KK))

Example: Nanotube



Visualization of nanotube produced by $\text{ptube}_{5,4}$

Example: Nanotube



In Functional Programming Language

let prib(*m*) =

compose (polyextend shared-formalize *m*) rib

where rib = polyextend compose *m* cycle

(reduce permute *m* identity)

in let ptube(*m*, *n*) = iterate *n* prib(*m*)

in ptube(5, 4)

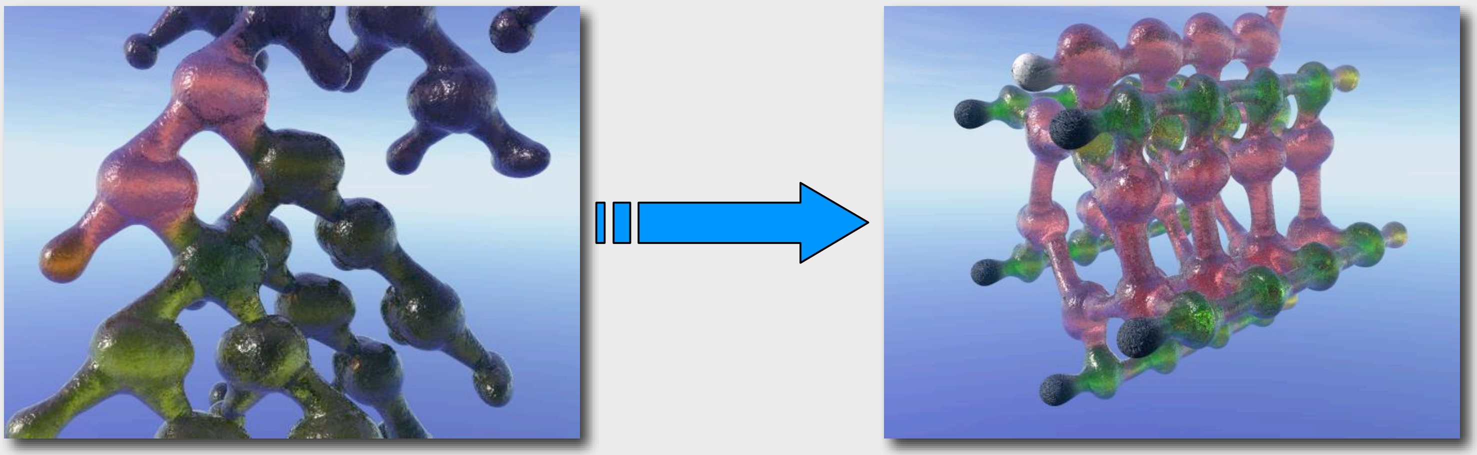
Reduced to SKY Tree

```
(((S ((S (K S)) K))
  ((S ((S (K S)) K)) ((S ((S (K S)) K)) ((S ((S (K S)) K)) (K ((S K) K))))))
 ((((S ((((S (K S)) K) ((S (K S)) K)) S)) (K K))
   ((((S (K S)) K) S)
    (((S (K S)) K)
     ((((S (K S)) K) S)
      (((S (K S)) K)
       ((((S (K S)) K) S)
        (((S (K S)) K) ((((S (K S)) K) S) (((S (K S)) K) S)))))))))
  ((((((S (K S)) K) ((S (K S)) K))
     ((((S (K S)) K) ((S (K S)) K))
      ((((S (K S)) K) ((S (K S)) K))
       ((((S (K S)) K) ((S (K S)) K)) ((S (K S)) K)))))
    Y)
   (((((S (K S)) K)
      (((S (K S)) K)
       ((((S (K S)) K)
         (((S (K S)) K)
          ((((S (K S)) K)
            (((S (K S)) K)
             ((((S (K S)) K)
               (((S (K S)) K) ((S ((((S (K S)) K) ((S (K S)) K)) S)) (K K))))
              ((S ((((S (K S)) K) ((S (K S)) K)) S)) (K K)))))
           ((S ((((S (K S)) K) ((S (K S)) K)) S)) (K K)))))
        ((S ((((S (K S)) K) ((S (K S)) K)) S)) (K K)))))
     ((S ((((S (K S)) K) ((S (K S)) K)) S)) (K K)))
    ((S K) K)))))
```

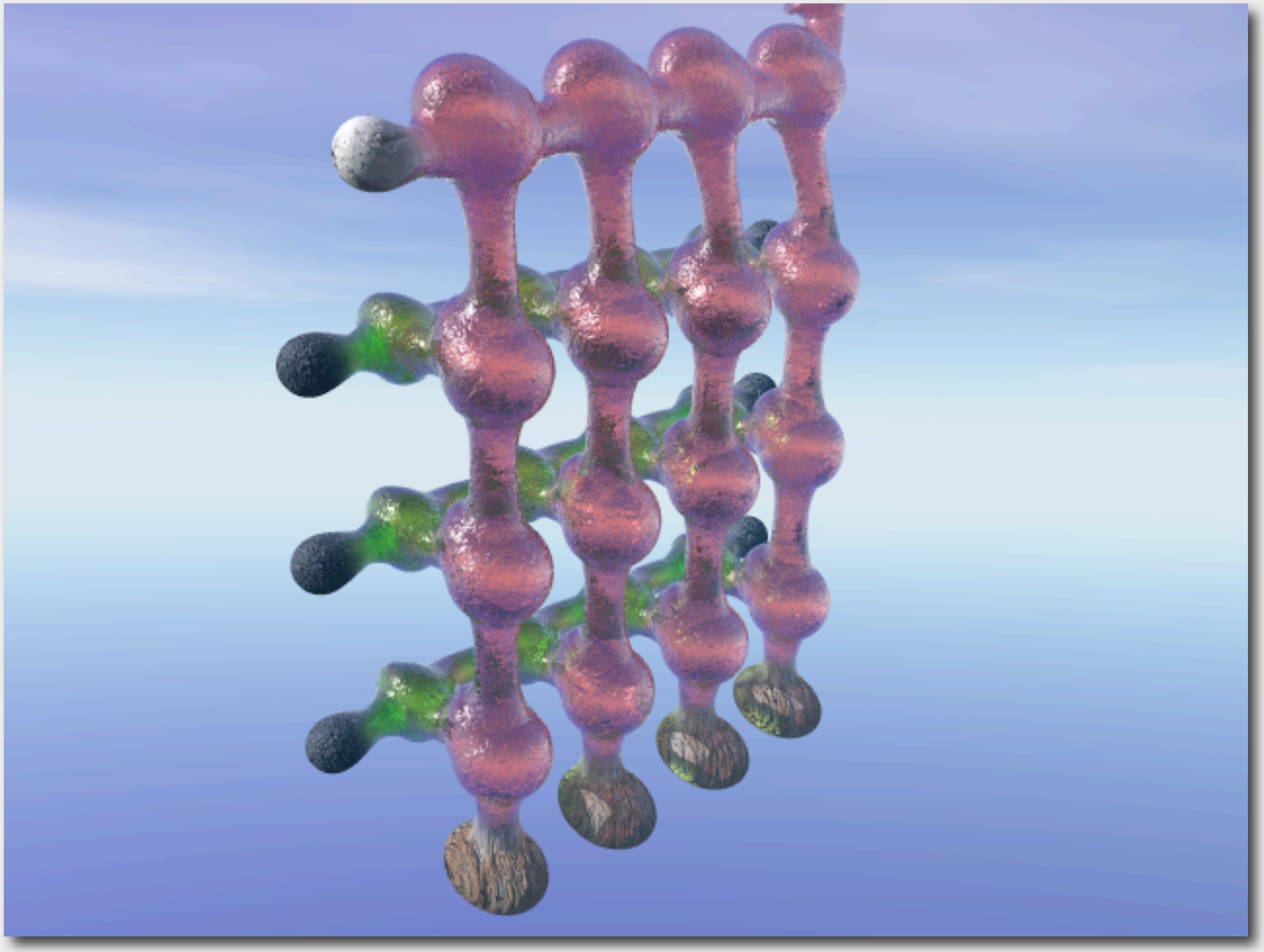
Linearized for Chemical Synthesis & Replication

PPPS PPS PK SQQ KQQ PPS PPS PK SQQ KQQ PPS PPS PK SQQ KQQ PPS PPS PK SQQ KQQ PK PPS KQ KQQQQQ PPPPS PPPPS PK SQQ KQ PPS PK SQQ KQQ SQQ PK KQQ PPPPS PK SQQ KQ SQ PPPS PK SQQ KQ PPPPS PK SQQ KQ SQ PPPS PK SQQ KQ PPPPS PK SQQ KQ SQ PPPS PK SQQ KQ PPPPS PK SQQ KQ SQ PPPS PK SQQ KQ SQQQQQQQQ PPPPPPS PK SQQ KQ PPS PK SQQ KQQ PPPPS PK SQQ KQ PPS PK SQQ KQQ PPPPS PK SQQ KQ PPS PK SQQ KQQ PPPPS PK SQQ KQ PPS PK SQQ KQQ PPS PK SQQ KQQQQQ YQ PPPPPS PK SQQ KQ PPPS PK SQQ KQ PPPPS PK SQQ KQ PPPS PK SQQ KQ PPPPS PK SQQ KQ PPPS PK SQQ KQ PPPPS PK SQQ KQ PPPS PK SQQ KQ PPS PPPPS PK SQQ KQ PPS PK SQQ KQQ SQQ PK KQQQQ PPS PPPPS PK SQQ KQ PPS PK SQQ KQQ SQQ PK KQQQQQ PPS PPPPS PK SQQ KQ PPS PK SQQ KQQ SQQ PK KQQQQQ PPS PPPPS PK SQQ KQ PPS PK SQQ KQQ SQQ PK KQQQQQ PPS PPPPS PK SQQ KQ PPS PK SQQ KQQ SQQ PK KQQQ PPS KQ KQQQQQ

Molecular Computation

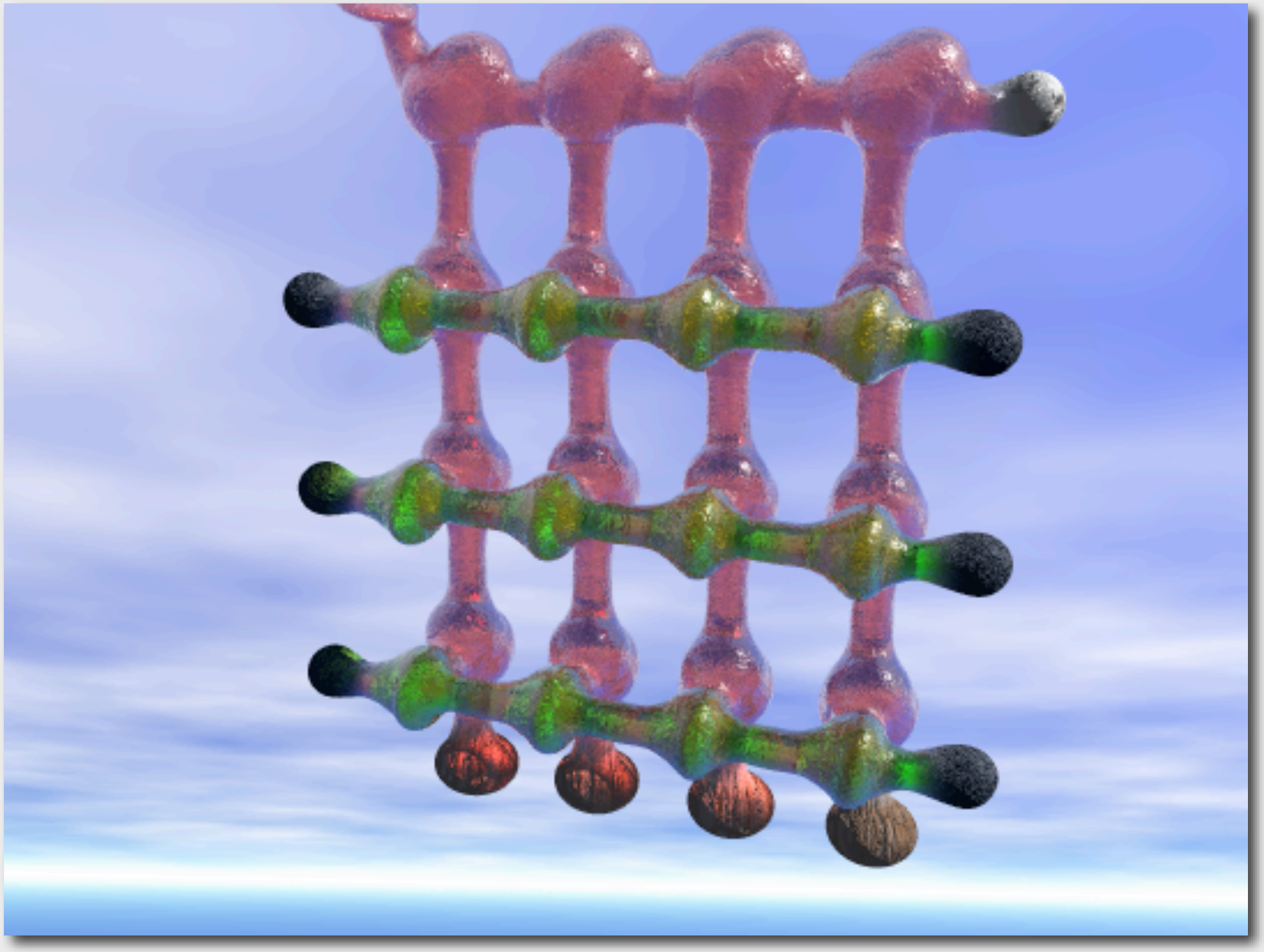


Cross-Linked Membrane

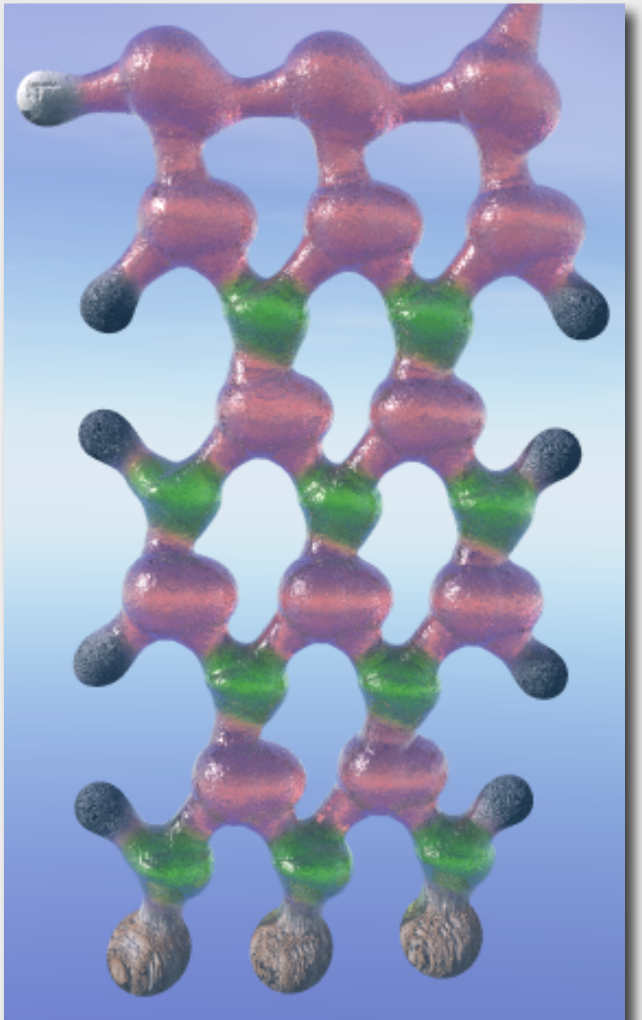


Visualization of membrane produced by $\text{xgrid}_{3,4}$ NNN

Cross-linked Membrane



Hexagonal Membrane



Produced by $\mathrm{hgridt}_{2,3}\ \mathrm{N}$

$\mathrm{Arow}_n = \mathrm{B}\check{\mathrm{W}}_{[n\text{-}1]}{\circ}\mathrm{B}^{[n]}$

$\mathrm{Vrow}_n = \check{\mathrm{W}}_{[n]}{\circ}\mathrm{KI}{\circ}\mathrm{K}_{(2n\text{-}2)}{\circ}$
$\quad \mathrm{B}^{[n\text{-}1]}{\circ}\mathrm{C}^{[n]}\mathrm{IN}{\circ}\mathrm{CIN}$

$\mathrm{drowt}_n = \mathrm{Vrowt}_n{\circ}\mathrm{Arow}_n$

$\mathrm{hgridt}_{m,n} =$
$\quad \mathrm{Z}_{n\text{-}1}\mathrm{W}(\mathrm{Z}_m\mathrm{drowt}_n\mathrm{N})$

Possible Molecular Implementation

- Covalently-structured MBBs for nodes and linking groups
- H-bonds for interconnections
- H-bonds for identification
- Synthetic components appended
- Substitutions controlled by enzyme-like covalently-structured molecules

Progress to Date

- Simulation & theoretical studies:
  - ways of assembling hierarchical heterogeneous structures from patches
  - membranes, pores, sensor interface, one-shot channels, simple actuators, nanotubes
- In progress:
  - recyclable channels, cilia, rotary motion
  - molecular implementations (including DNA)

Summary of MCC

- Concept of molecular combinatory computing
  - molecular networks self-organize by simple substitution reactions
  - computationally universal
- Simulated synthesis applications
- Synthesis of large, heterogeneous structures
- Possible molecular implementation based on H-bonded, covalently-structured building blocks

Applications in
Command, Control & Coordination

Potential Application Domains

- Robots & autonomous vehicles
  - contemporary robots & AVs
  - microrobots
  - nanobots
- Informational agents
- Command, control & coordination of human agents

- Allocation of resources
- Exploration vs. exploitation
- Communication
- Distributed synchronization
- Information storage
- Construction

Conclusions

- Computation can be used to control matter
  - for reconfiguration of computers, transducers, etc.
  - for nano-assembly and control
- Detailed structure determined by self-organization
- Natural systems provide good models and possible implementation technologies
- Artificial systems with the robustness of natural systems should be achievable
- For more information: www.cs.utk.edu/~mclennan