Embodied Computing

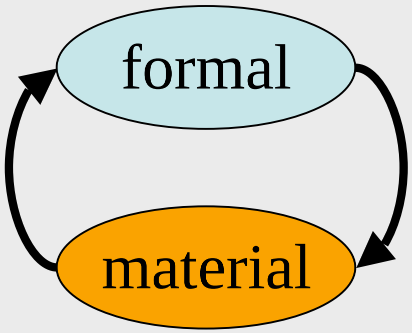
Bruce J. MacLennan
Dept. of Electrical Engineering
& Computer Science
University of Tennessee, Knoxville

Outline

- Recent Research
  - Molecular combinatory computing
  - Generalized computation & the U-machine
  - Artificial morphogenesis
- New Directions
  - Convergence of scales
  - Novel characteristics
  - Need for new models of computation
  - Embodied computing

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 Embodied Computing

- Molecular Combinatory Computing
- Generalized Computation
- Artificial Morphogenesis

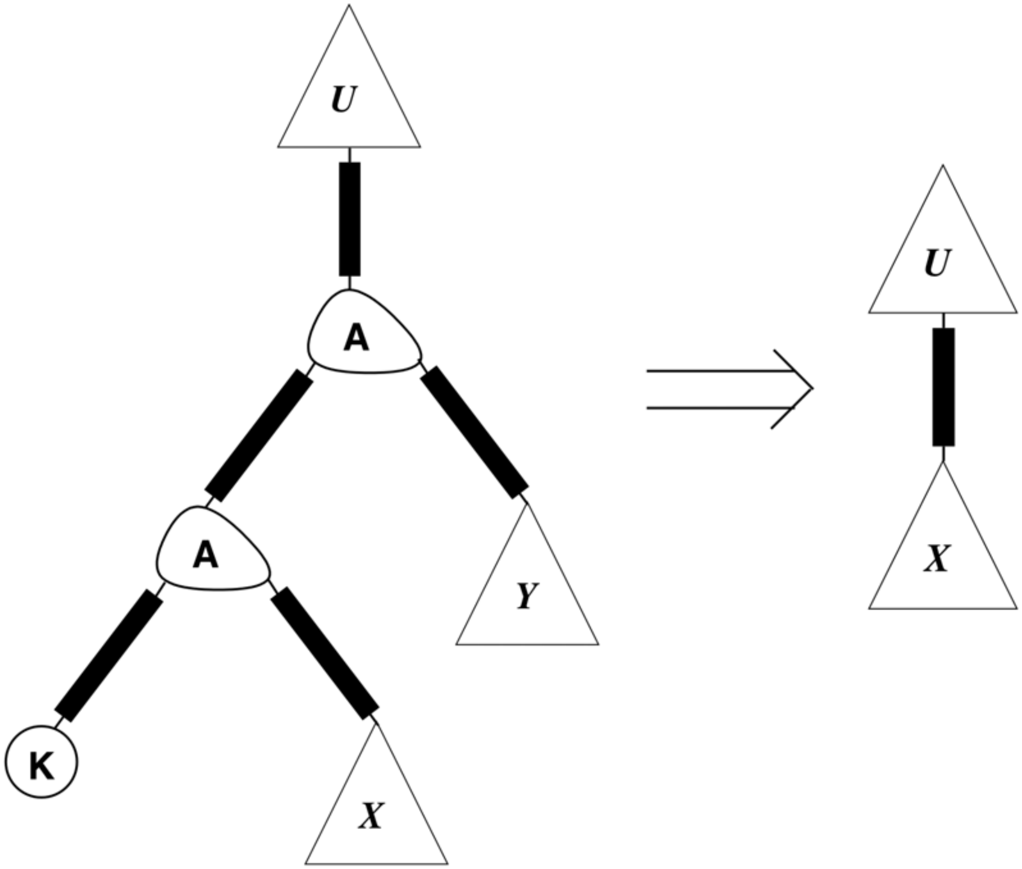
Molecular Combinatory Computing for Nanostructure Synthesis & Control

Supported by NSF
Nanoscale Exploratory Research Grant

Molecular Combinatory Computing

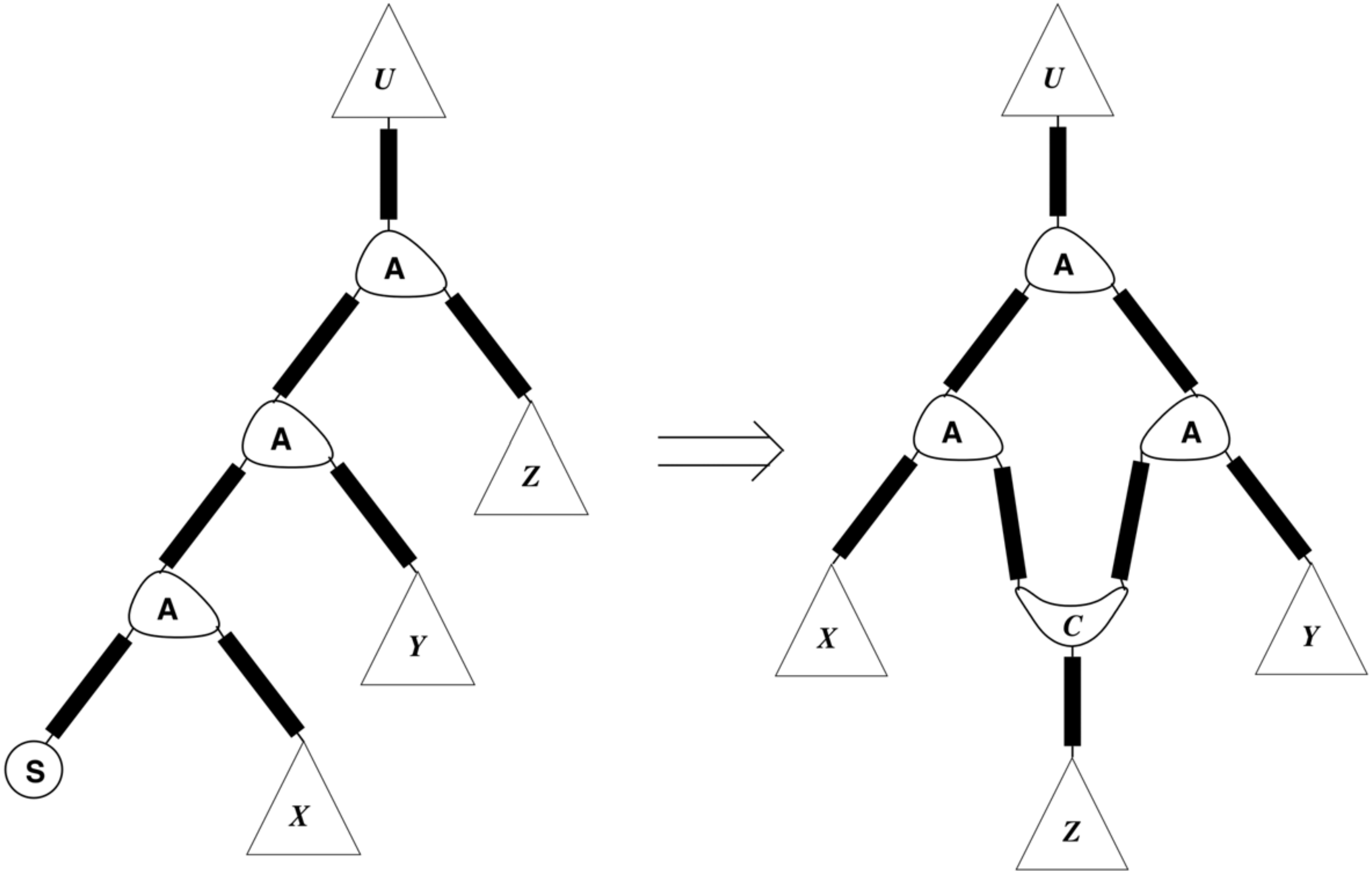
- Systematic approach to nanotechnology based on small set of molecular building blocks
- Combinatory logic
- Computational universality from two substitutions (+ a few more)
- Substitutions may be done in any order or in parallel

K-Substitution



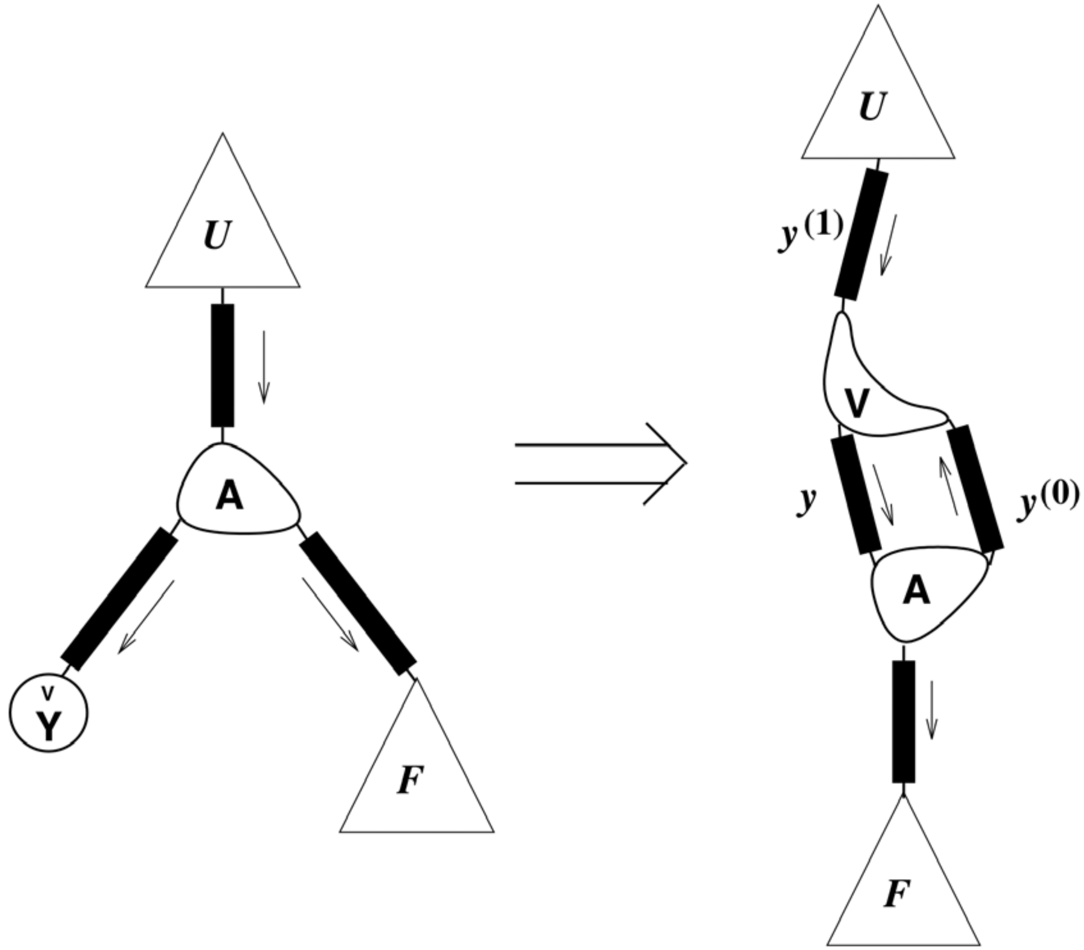
$$((\mathsf{K}\ X)\ Y) \Rightarrow X$$

S-Substitutions



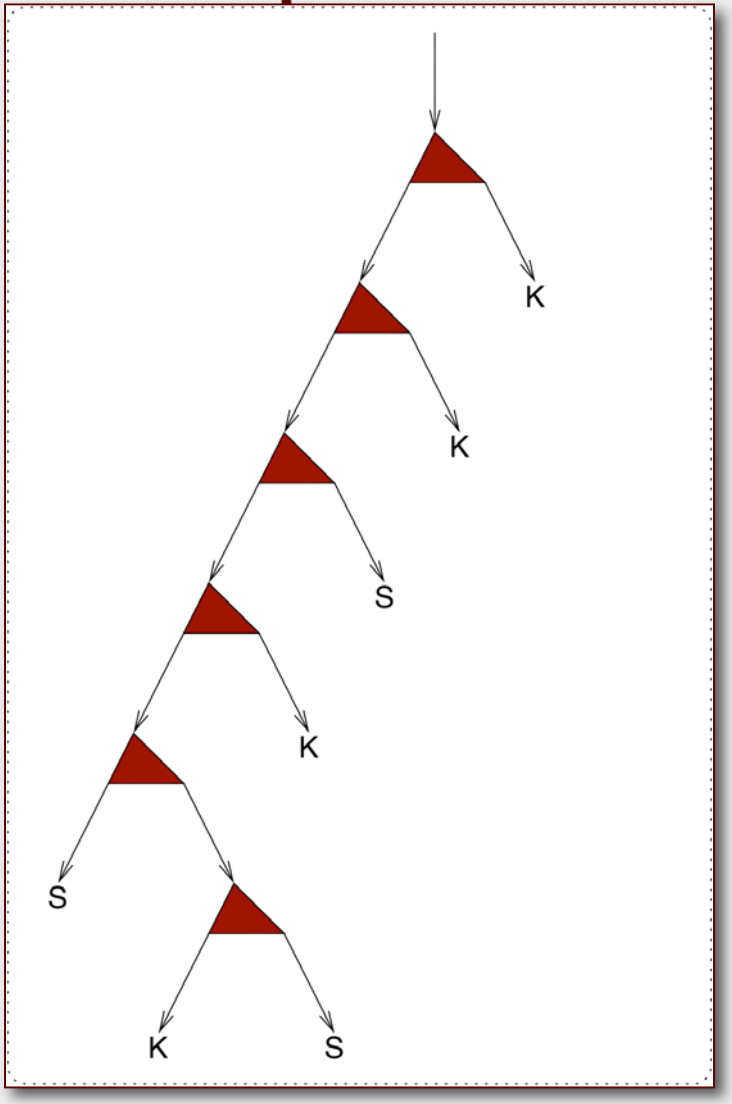
$$(((\mathsf{S}\ X)\ Y)\ Z) \Rightarrow ((X\ Z)\ (Y\ Z'))$$

Ý-Substitution



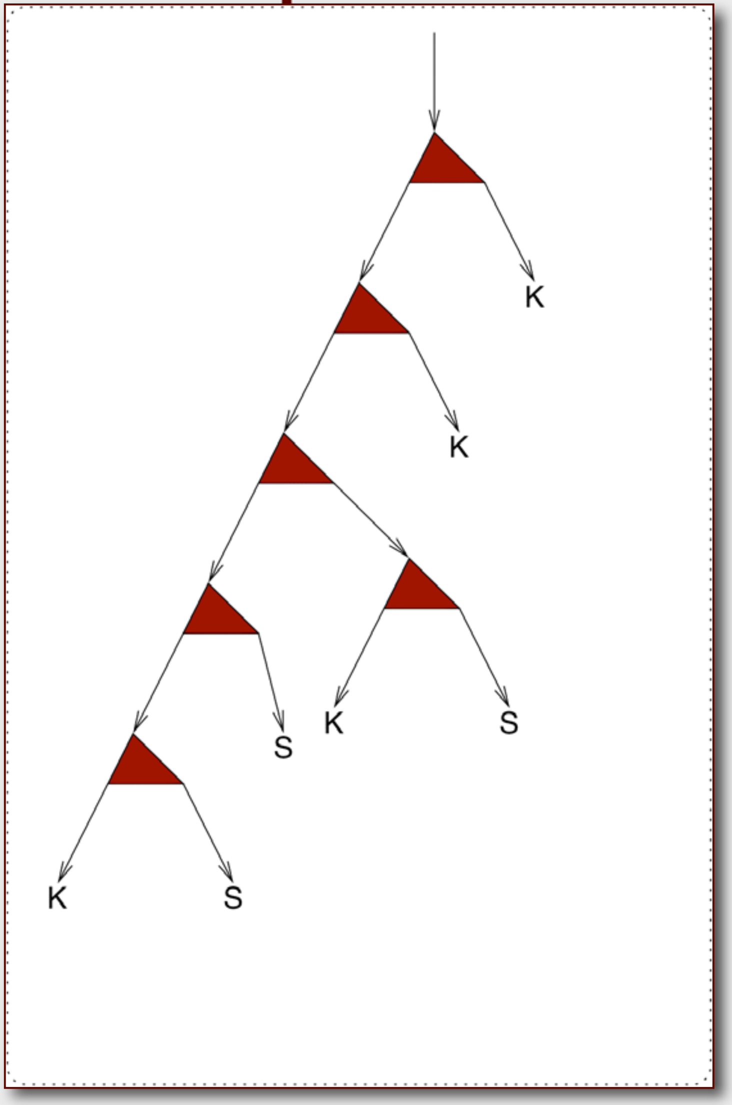
$$(\acute{Y}\ F) \Rightarrow y^{(1)} \quad \text{where } y \equiv (F\ y^{(0)})$$

Example of Simple Computation



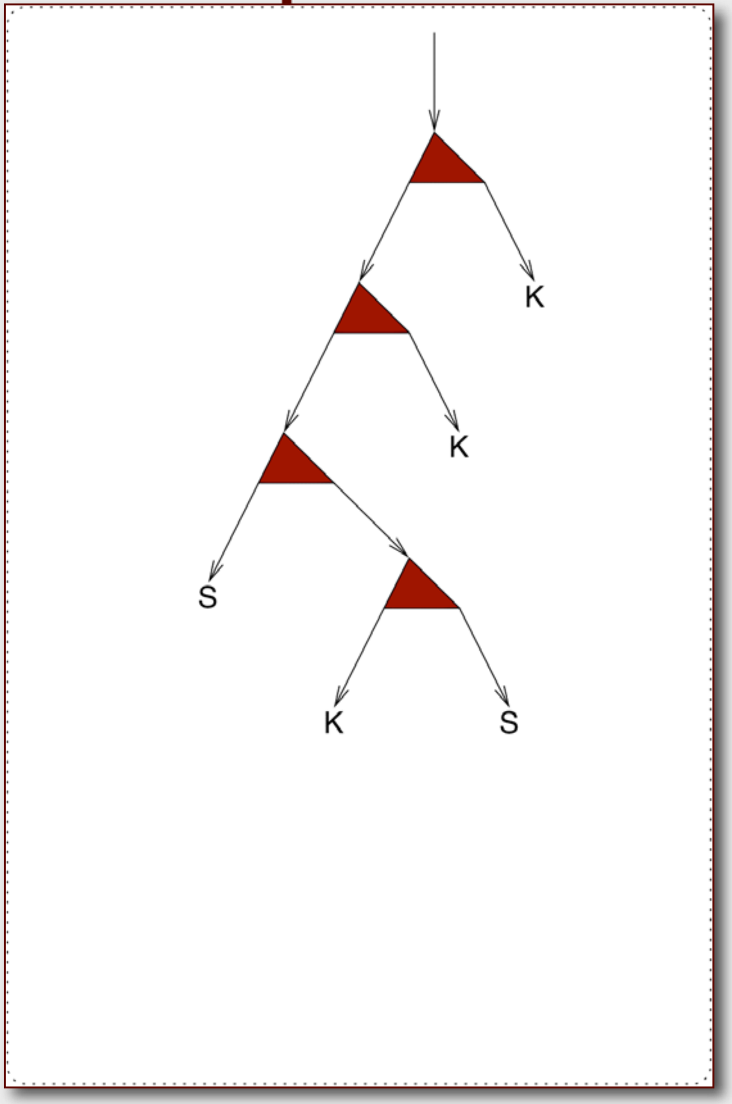
((S(KS)K)S)
$\Rightarrow$ (((KS)S)(KS))

Example of Simple Computation



((KS)S) $\Rightarrow$ S

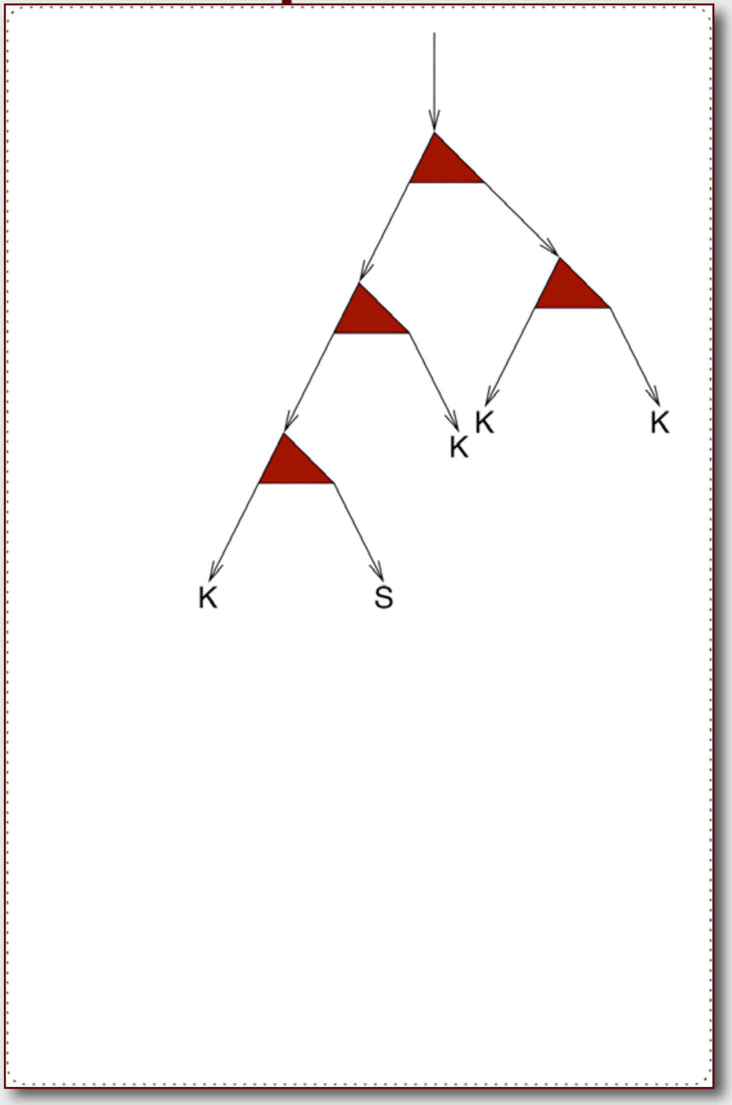
Example of Simple Computation



(((S(KS)K)K)

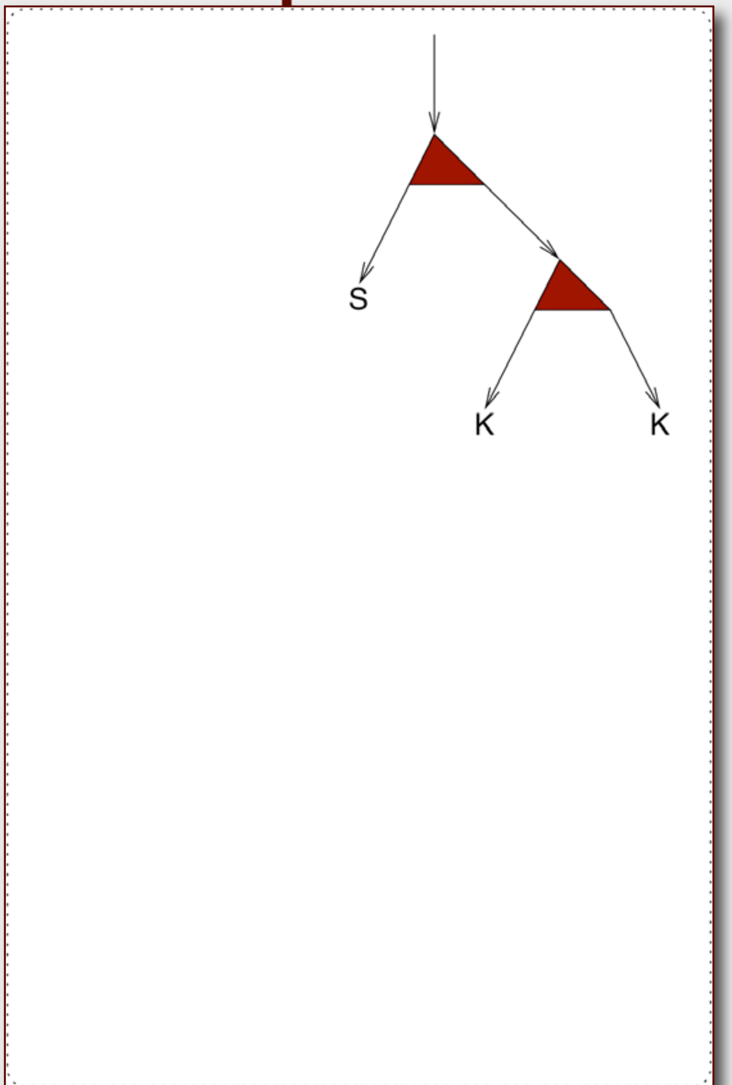
$\Rightarrow$ (((KS)K)(KK))

Example of Simple Computation



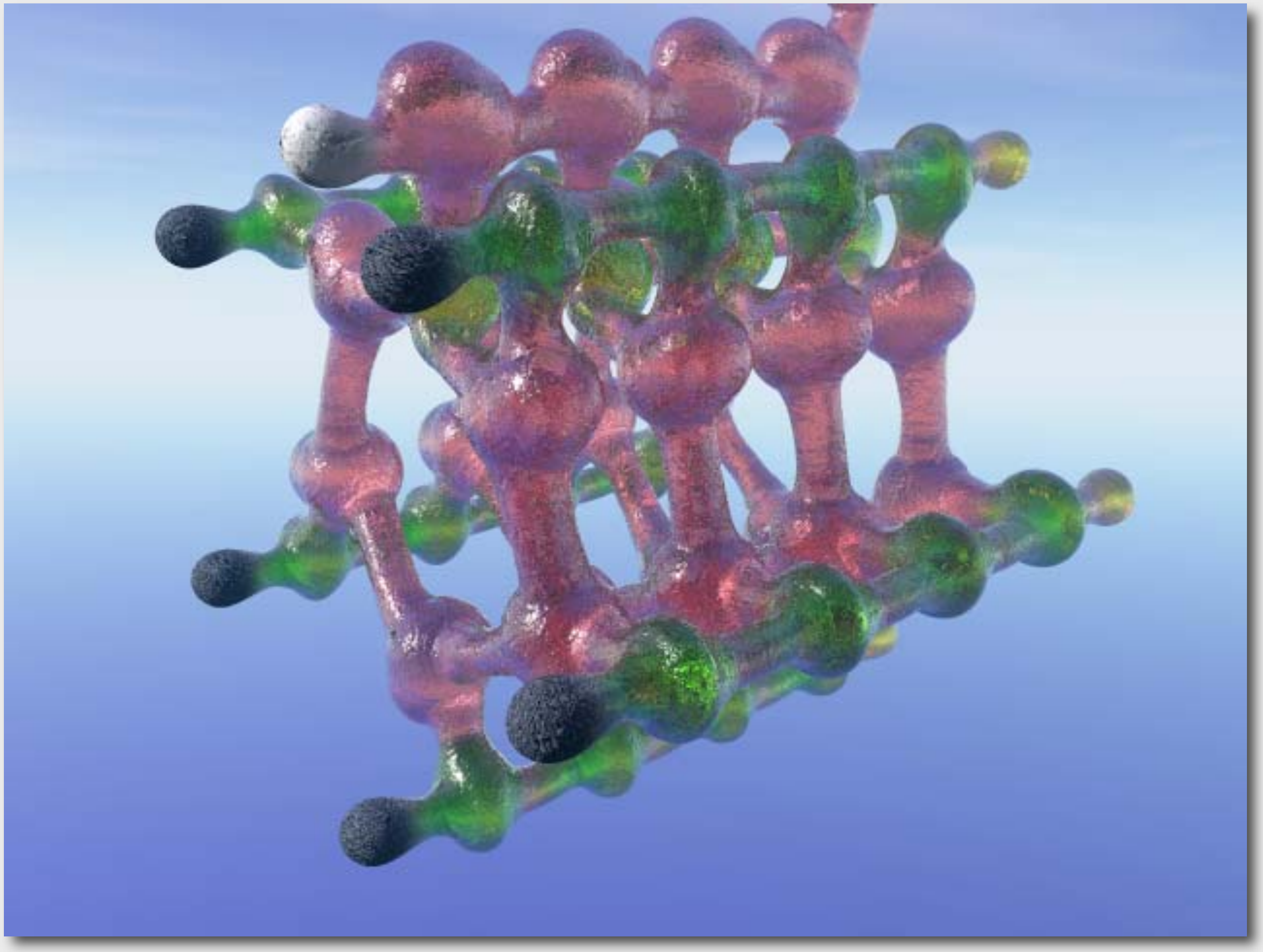
$((KS)K) \Rightarrow S$

Example of Simple Computation



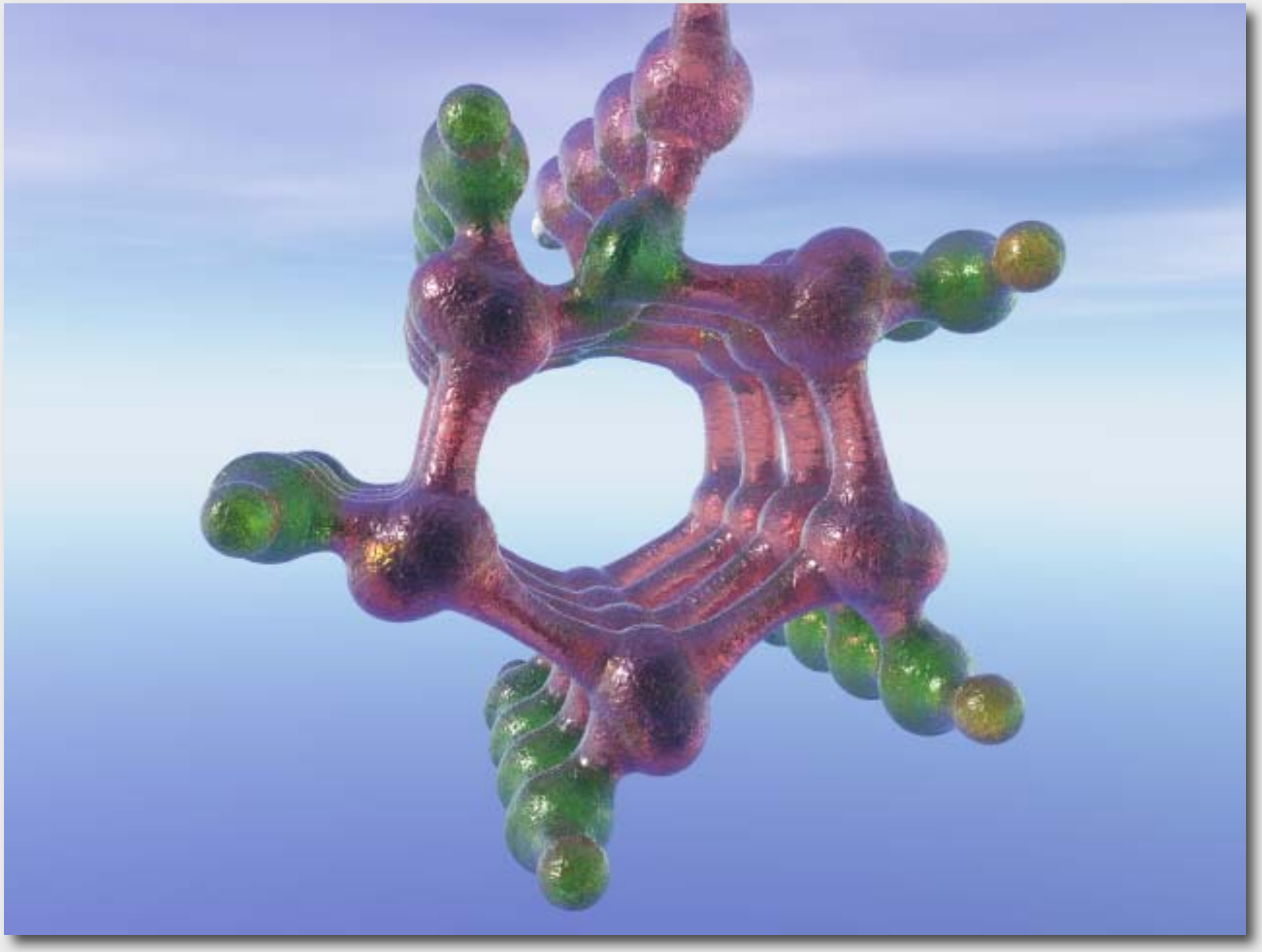
result = (S(KK))

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)

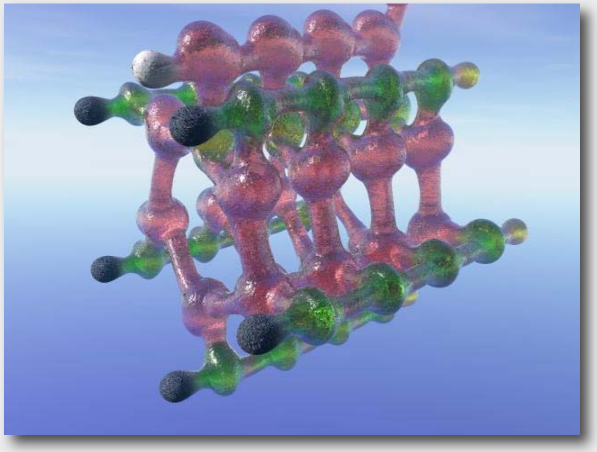
Reduced to SKY Tree

```
(((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)))))
```

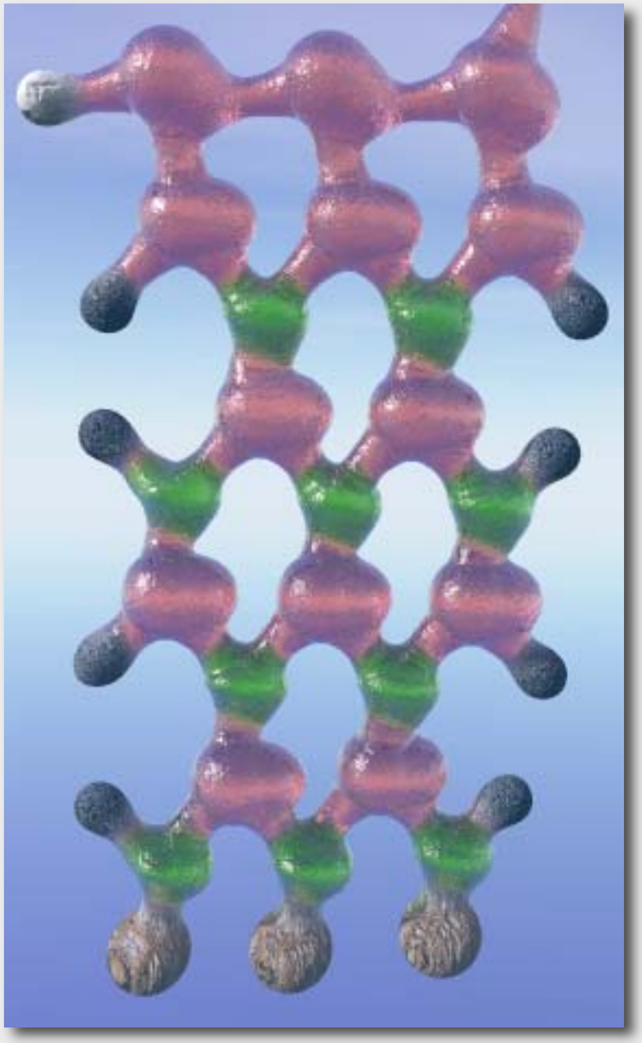
Linearized for Chemical Synthesis & Replication

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



Hexagonal Membrane



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

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

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

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

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

Possible Molecular Implementation

- Covalently-structured molecular building blocks 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)

Pros and Cons

- Pros
  - Turing-universal
  - substitutions can be done in any order or in parallel
- Cons
  - not fault- or error-tolerant
  - computational universality does not imply synthetic/behavioral universality
  - requires aqueous medium

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

Generalized Computation and the U-Machine

Computation in General Sense

- A definition applicable to computation in nature as well as in 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

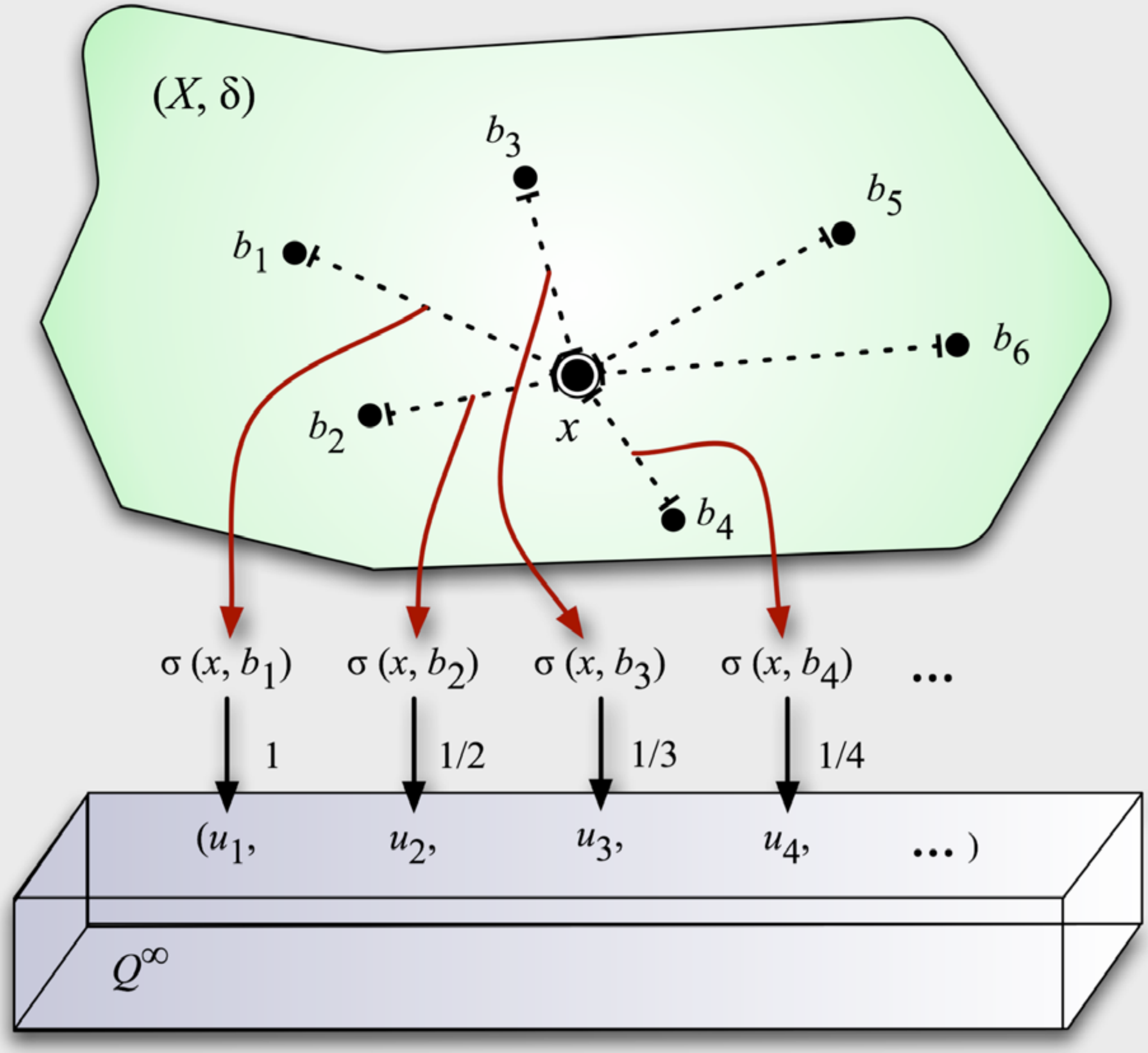
Abstract Spaces

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

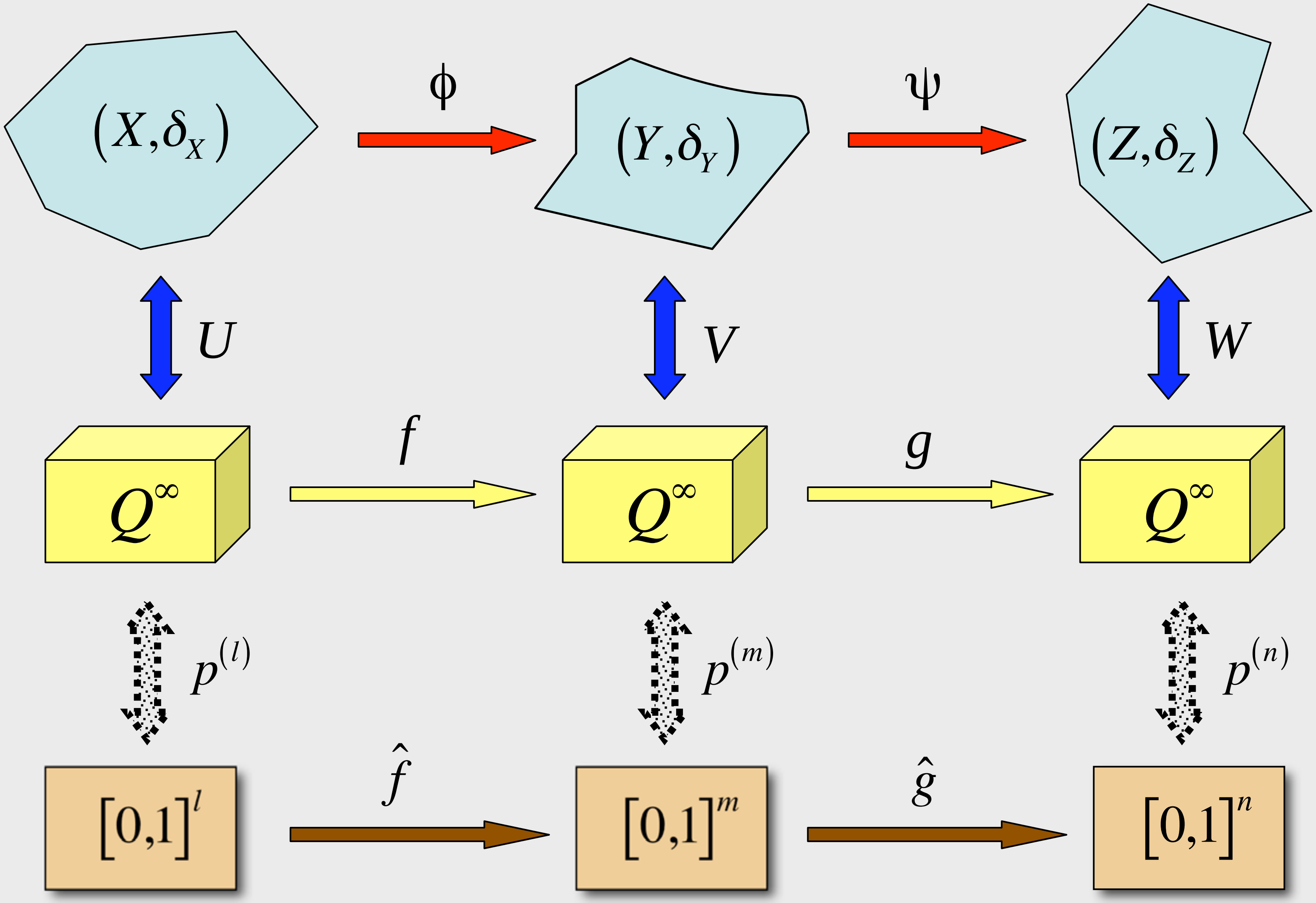
The U-Machine

- 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



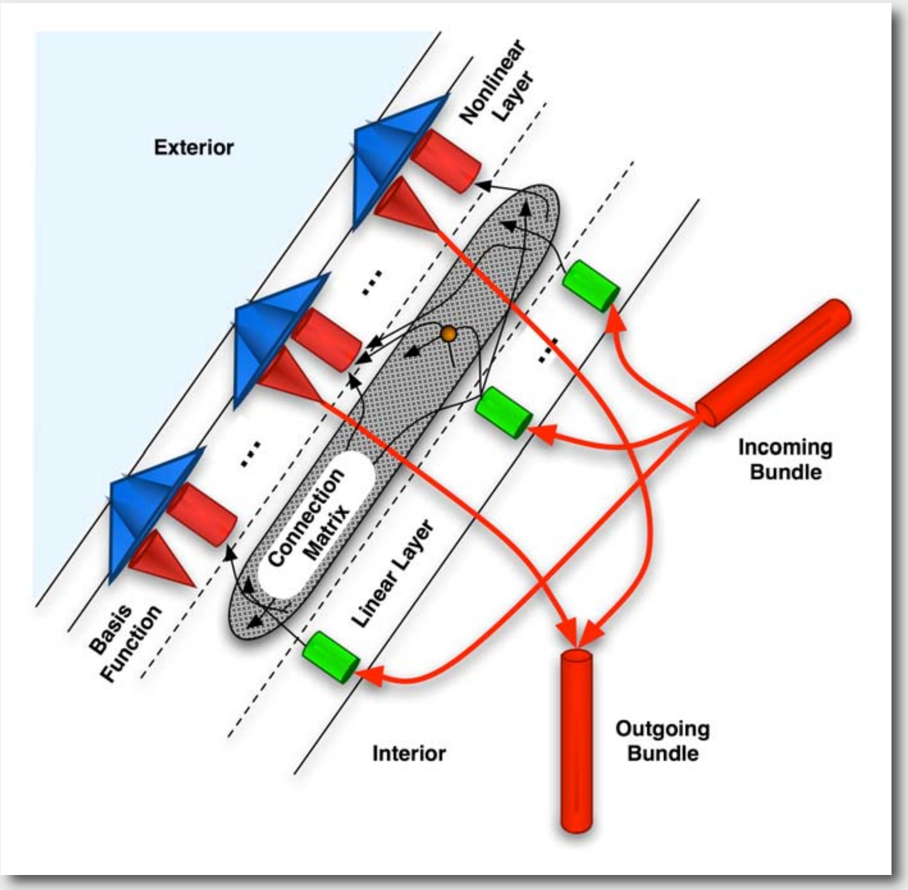
Overall Structure

- 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

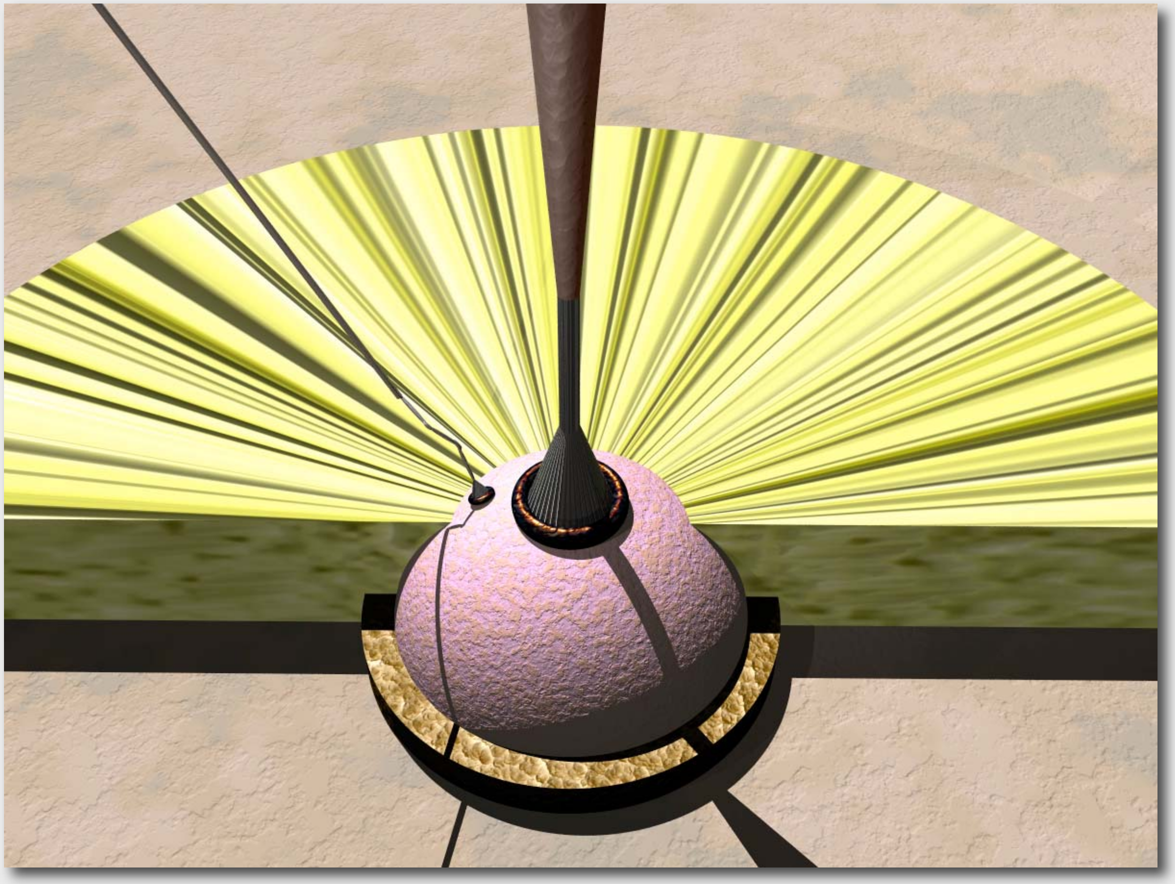
- Shell contains variable areas & computational elements
- Interior filled with solid or liquid *matrix* (not shown)
- Paths formed through or from matrix

Layers in Data Space

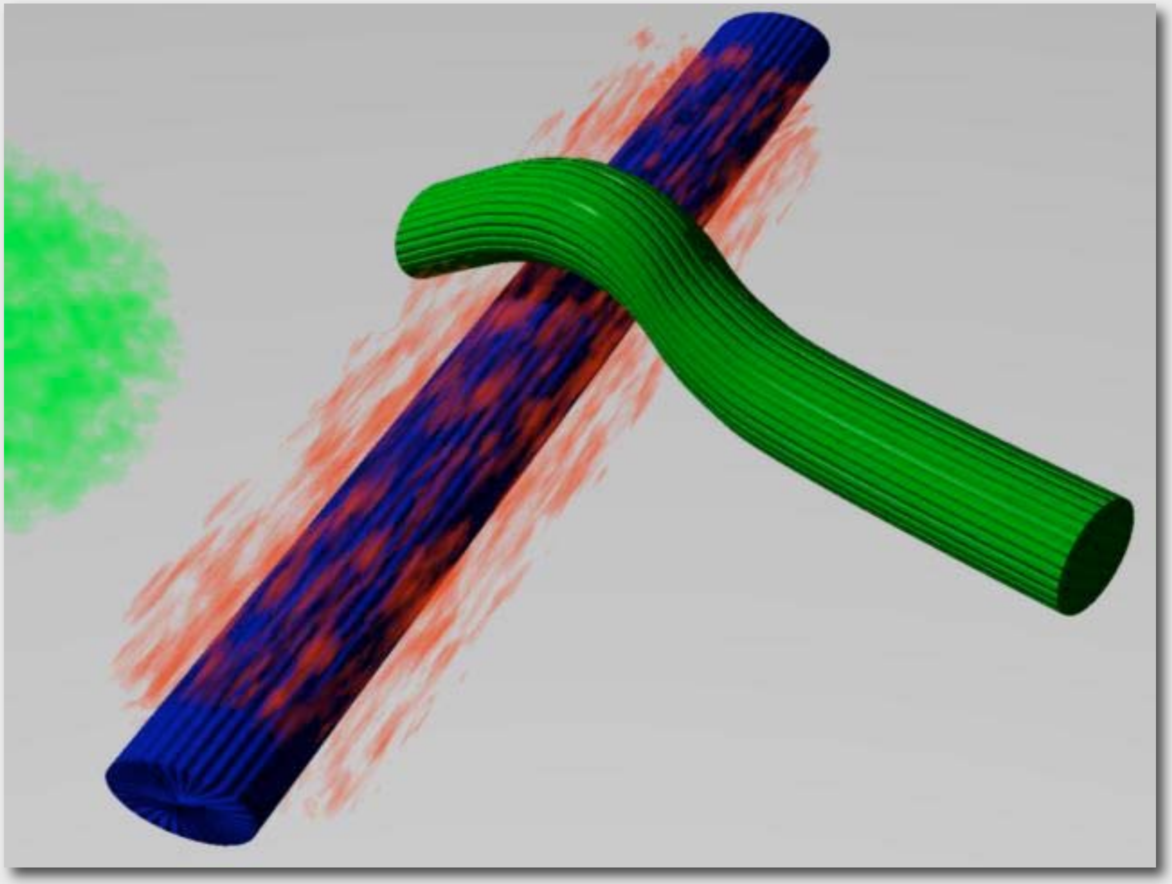


- Connection matrix has programmable weights
- Linear combinations are inputs to nonlinear basis functions
- Exterior access to both sides for programming

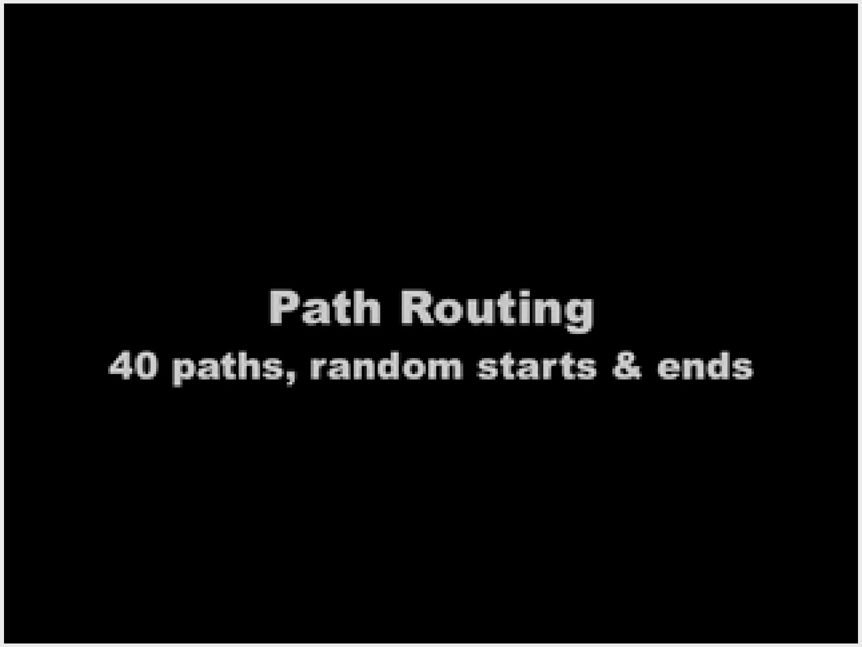
Depiction of UM Exterior



Diffusion-Based Path Routing

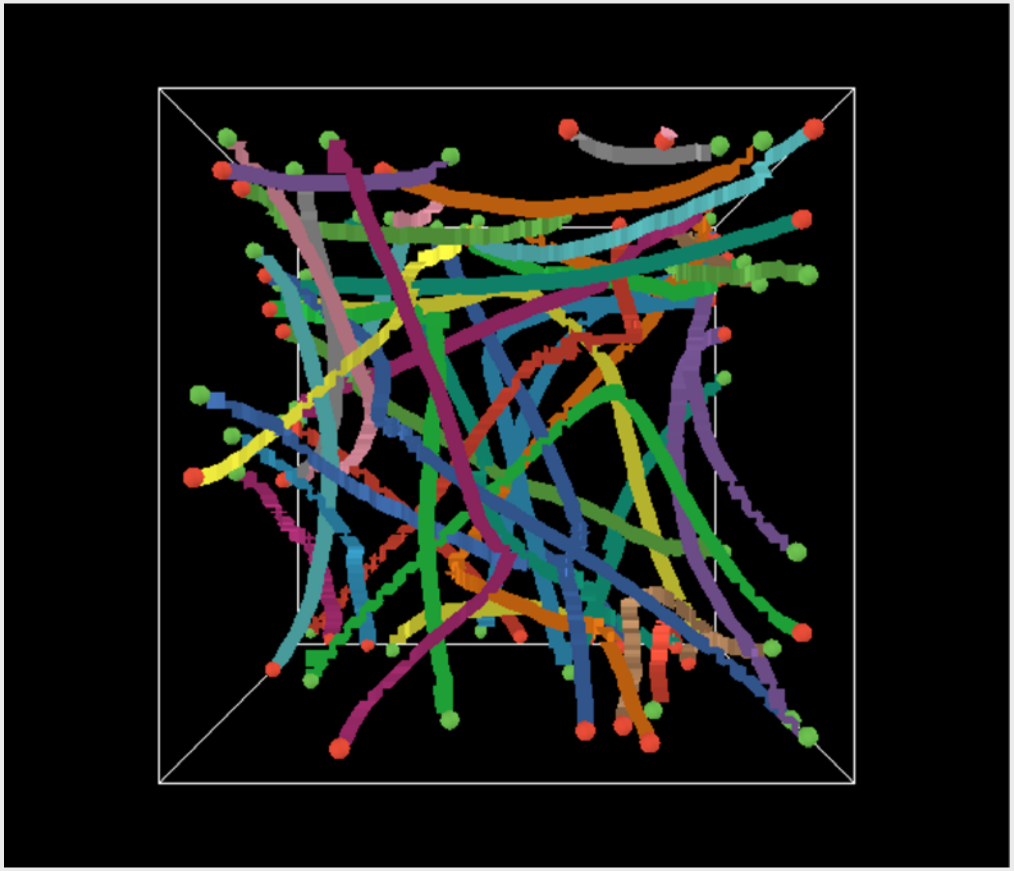


Example of Path Routing



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

Front



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

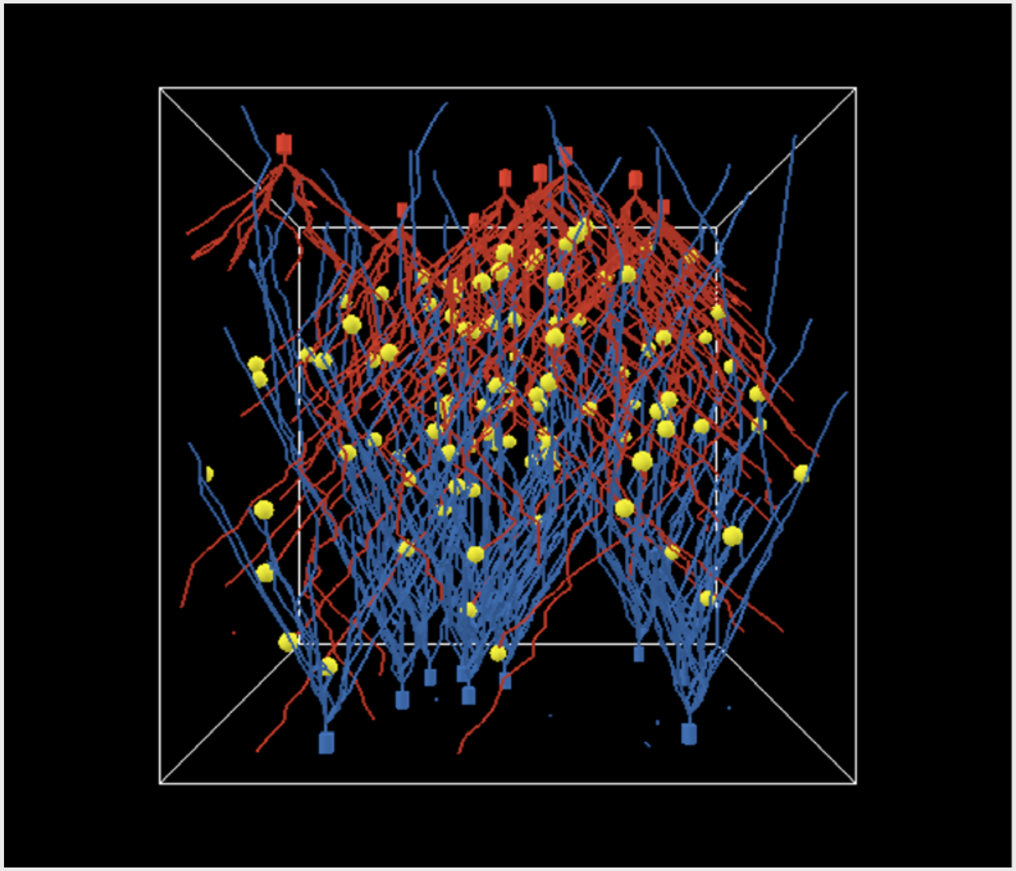
Example Connection-Growth Process

- 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

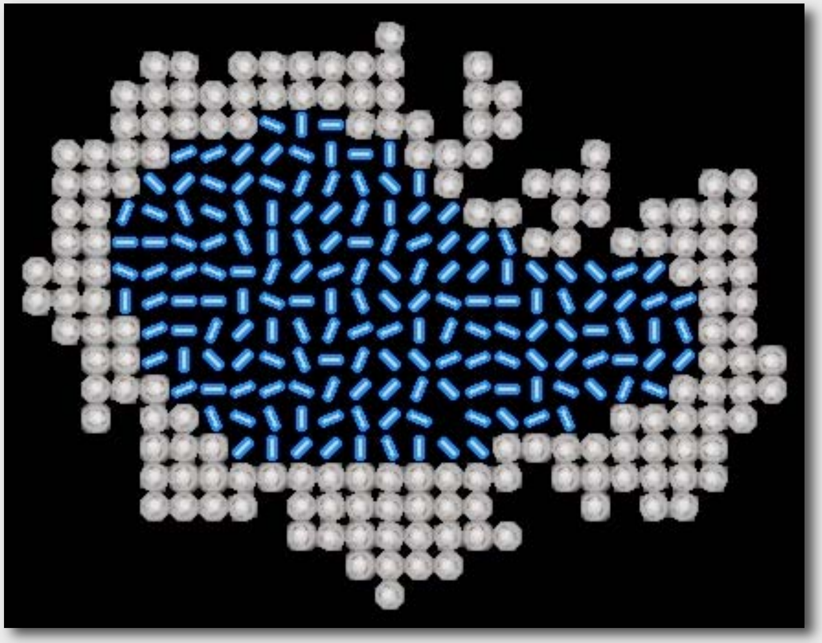


Summary of U-Machine

- 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

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
- But there may be others

Multiscale Assembly

- Assembly from nanoscale to macroscale
- Macroscopic test cases (“killer apps”?):
  - multilayer tissues
  - tubes
  - articulated structures
  - periodic structures
  - branching structures
  - bristles
  - somatic regions

Some Primitives

- Cellular / discrete
  - translation/rotation
  - adhesion/release
  - conformal change
  - differentiation/state change
  - collision/interaction
  - proliferation/apoptosis
- Continuum
  - elasticity
  - diffusion
  - degradation
  - fluid flow
  - gradient ascent

Programmed Morphogenesis

- Libraries of operations
  - diffusion
  - degradation
  - gradient ascent
  - activation-inhibition
  - proliferation/apoptosis
  - elasticity
  - etc.
- Coordinated algorithms

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")

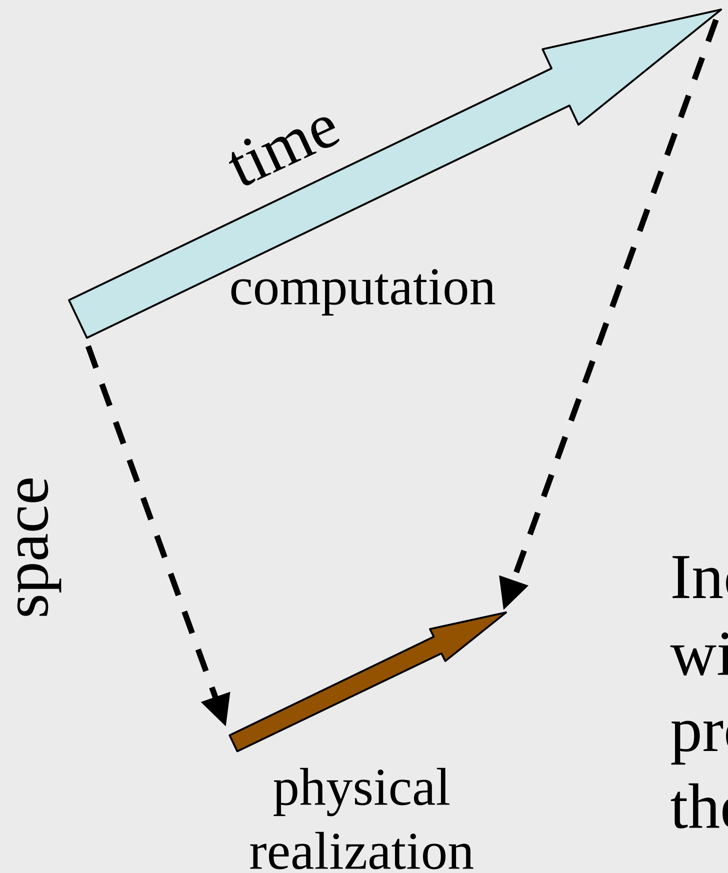
- Noncoding DNA can be used for "genetic circuits"
  - in eukaryotes, 10–70%
  - equivalent of about 3000 genes in yeast
- Equipped with an assortment of generally useful sensors & effectors (especially for self-organization)
- Special-purpose modifications for particular applications
- Research: principles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

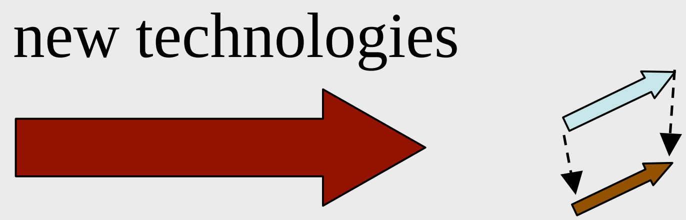
Tentative Capabilities

- 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

New Directions

Convergence of Scales





Increasing speed and density will require computational processes to be more similar to their physical realizations

Novel Characteristics

- Continuous
- Nondeterministic
- Asynchronous parallel
- Reversible
- Nonequilibrium system
- Stationary states
- Spatial factors
- Quantum phenomena
- General-purpose primitives
- Moving more rather than less

Goal: Exploit rather than circumvent

Need for New Models of Computation

- Historical importance of Boolean logic
- Importance of technology-independent models
  - independent of specific implementation
  - generically implementable
- What is the essence of computing?

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

Embodied Computing

- From *embedded* computing to *embodied* computing
- *embodiment* = "the interplay of information and physical processes"
  — Pfeiffer, Lungarella & Iida (*Science* **318**, 1088)

Concept

- Offline computing
  - input—process—output
  - computing is abstract (generically realized)
- Embedded computing
  - sensor—processor—actuator
  - sensors/actuators interface to physical world
  - computing is still abstract
- Embodied computing
  - computation is a physical system among physical systems

Challenges

- Less idealized, less independent of physical realization
- Physical processes cannot be ignored while designing information processes
- Physical morphology, power, etc. are critical
- Previously discussed Novel Characteristics
- Multiscale structural assembly
- General-purpose computing? Universality?

Approach

- Look for common physical processes & view as information processes
  - especially biological processes
- Determine how they exploit physical realization & physical environment
- Determine how generic they are
- Characterize in general terms for multiple instantiation

Benefits

- Information may be implicit in physical representation & environment
- Many information processes come for free
- Examples:
  - diffusion
  - saturation/depletion
  - decay
  - symmetry breaking

Conclusions

- Embodied computation is inherently physical
- This presents challenges
- But also many opportunities
- Nature provides many useful examples

Some Resources

1. Anderson, M.L. (2003), Embodied cognition: A field guide, *Artif. Intell.* **149**: 91–130.
2. Hamann, H., & Wörn, H. (2007), Embodied computation, *Par. Proc. Ltrs.* **17** (3): 287–98.
3. MacLennan, B.J. (2003), Transcending Turing computability, *Minds & Machines* **13** (1): 3–22.
4. MacLennan, B.J. (2004), Natural computation and non-Turing models of computation, *Theor. Comp. Sci.* **317**: 115–145.
5. MacLennan, B.J. (in press), Super-Turing or non-Turing? Extending the concept of computation, *Int'l Journ. Unconvent. Computing*.
6. Pfeifer, R., Lungarella, M., & Iida , F. (2007), Self-organization, embodiment, and biologically inspired robotics, *Science* **318**: 1088–93.
7. Pfeifer, R., & Bongard, J.C. (2007), *How the body shapes the way we think — A new view of intelligence*, MIT.
8. Stepney, S. (2004), Journeys in non-classical computation, *Grand Challenges in Comp. Res.*, ed. Hoare & Milner, BCS, 29–32.
9. Stepney, S. (in press), The neglected pillar of material computation, *Physica D*.