Highly Programmable Matter & Generalized Computation

Research in Reconfigurable Analog & Digital Computation In Bulk Materials

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

“Radical Reconfiguration”

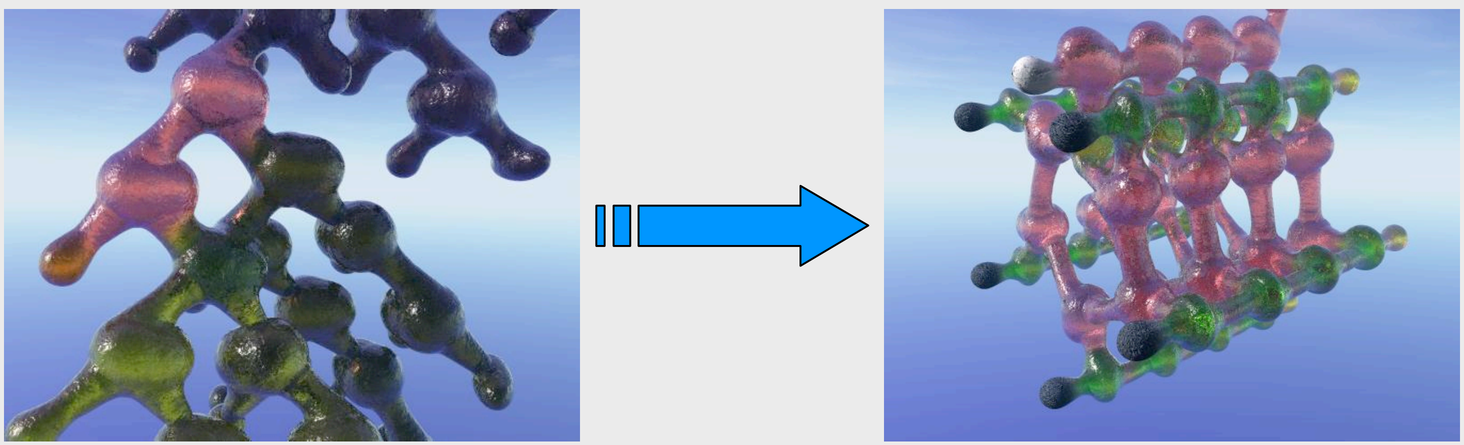
- Ordinary reconfiguration changes connections among fixed components
- Radical reconfiguration of <u>transducers</u>
  - to create new sensors & actuators
- Radical reconfiguration of <u>processors</u>
  - to reallocate matter to different components
- Also for repair & damage recovery
- Requires rearrangement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

Our Ongoing Research in Radical Reconfiguration

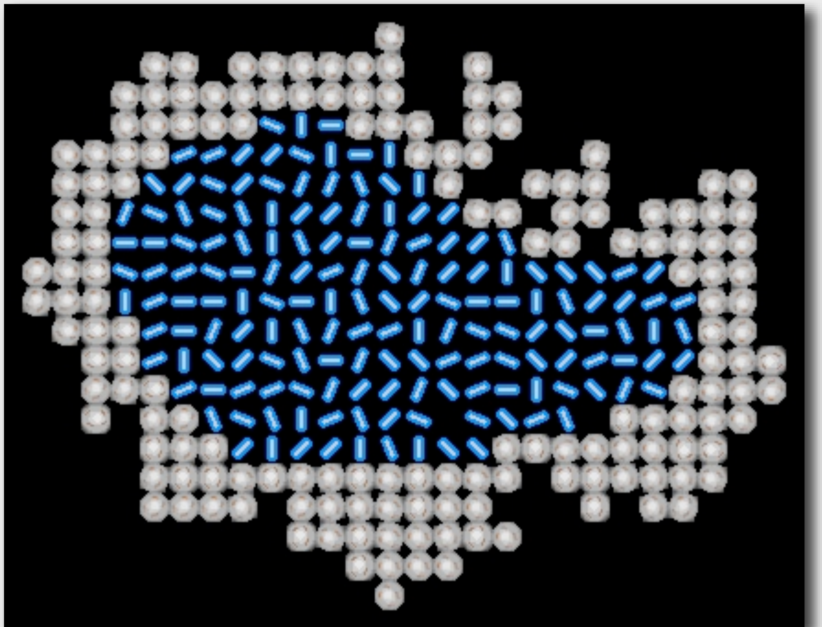
- Programmable matter
  - Computational control of matter
  - Molecular combinatory computing
  - Morphogenetic approaches
- Generalized computation
  - Flexible general computing medium
  - Encompassing both analog & digital
  - U-machine

Molecular Combinatory Computing

- Comb. comp. based on two graph substitutions
  - computationally universal
  - can compile programs into these computational graphs
- Can proceed asynchronously & in parallel
- Program computes into structure represented in molecules
- Supported by NSF (Nanoscale Exploratory Research)



Morphogenetic Approaches



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

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 $\Rightarrow$ approximating sequences

The U-Machine

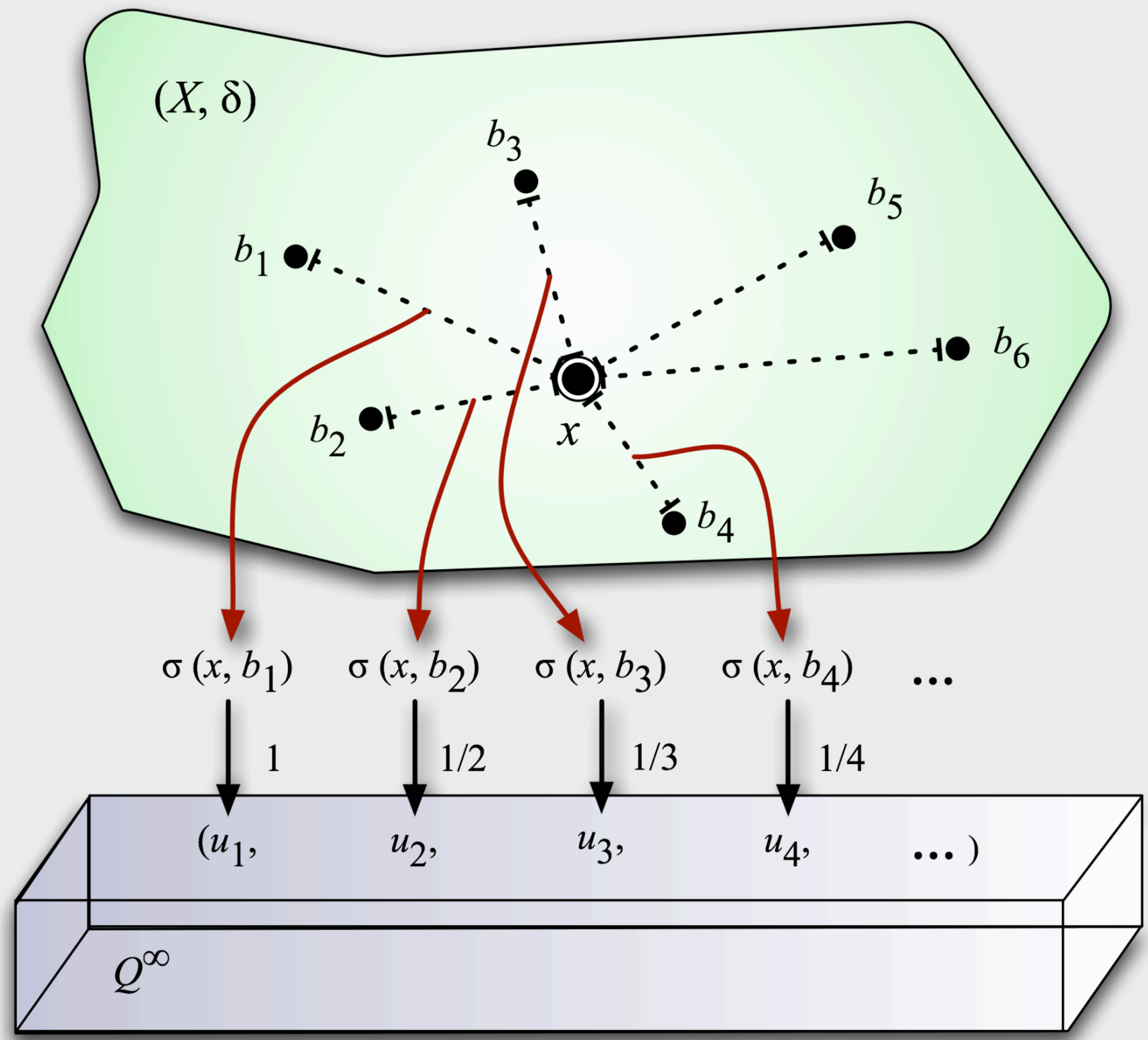
- Goal: a model of computation over abstract spaces that can be implemented in a variety of physical media
- In particular, bulk nanostructured materials in which:
  - access to interior is limited
  - detailed control of structure is difficult
  - structural defects and other imperfections are unavoidable

Urysohn Embedding

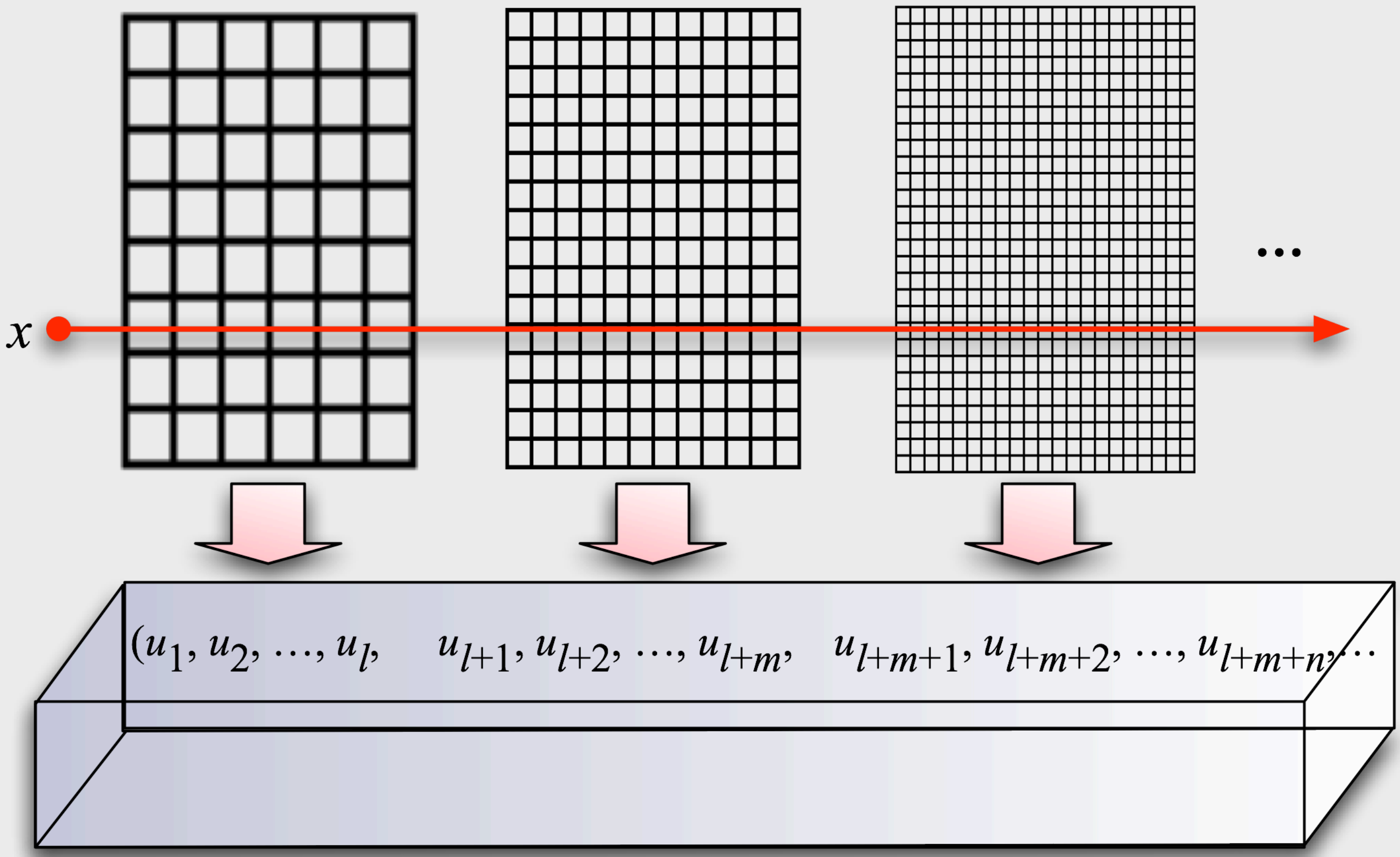
- A separable metric space is homeomorphic to a subset of a Hilbert space
- Let (X, δ) be separable metric space
- Let $b_1, b_2, \ldots \in X$ be a ctbl dense subset
- WLOG suppose δ is bounded in [0, 1]
- Let similarity $\sigma(x, y) = 1 - \delta(x, y)$
- Define:

$$U(x) = \left(\frac{\sigma(x, b_1)}{1}, \frac{\sigma(x, b_2)}{2}, \frac{\sigma(x, b_3)}{3}, \ldots \right)$$

Urysohn Embedding



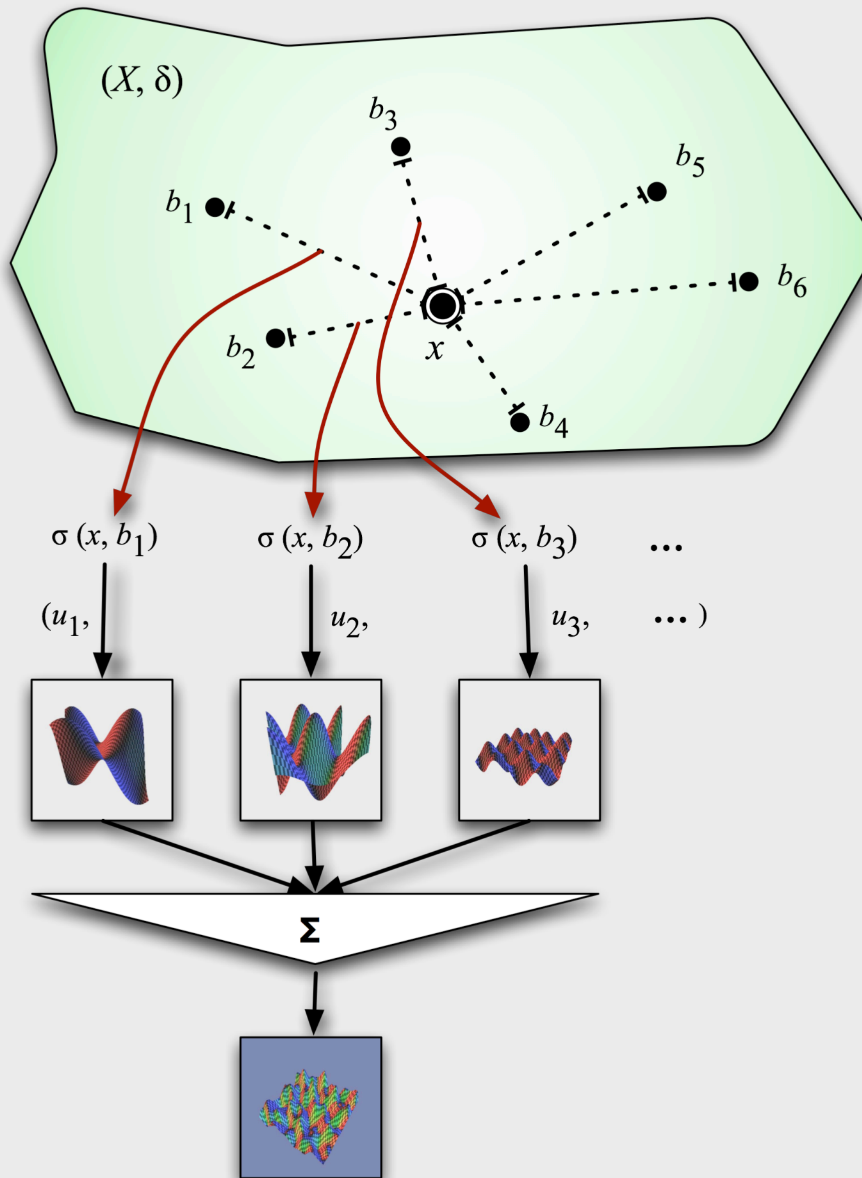
ε-Nets



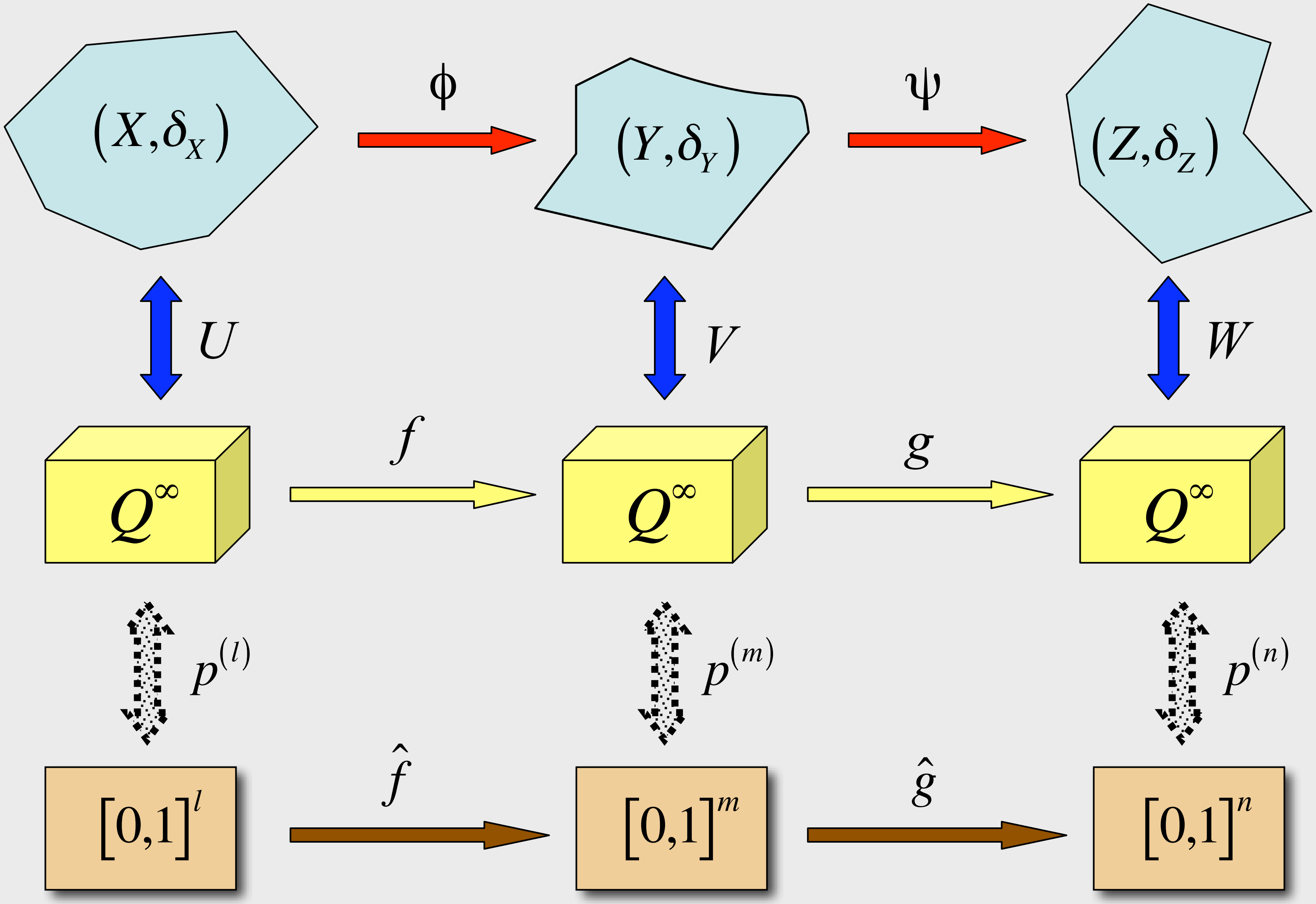
...
x
$(u_1, u_2, \ldots, u_l, \quad u_{l+1}, u_{l+2}, \ldots, u_{l+m}, \quad u_{l+m+1}, u_{l+m+2}, \ldots, u_{l+m+n}, \ldots$

Field Computation

- Field = continuous distribution of continuous quantity = element of Hilbert function space
- u_k used to scale basis functions
- Linear superposition represents element of abstract space

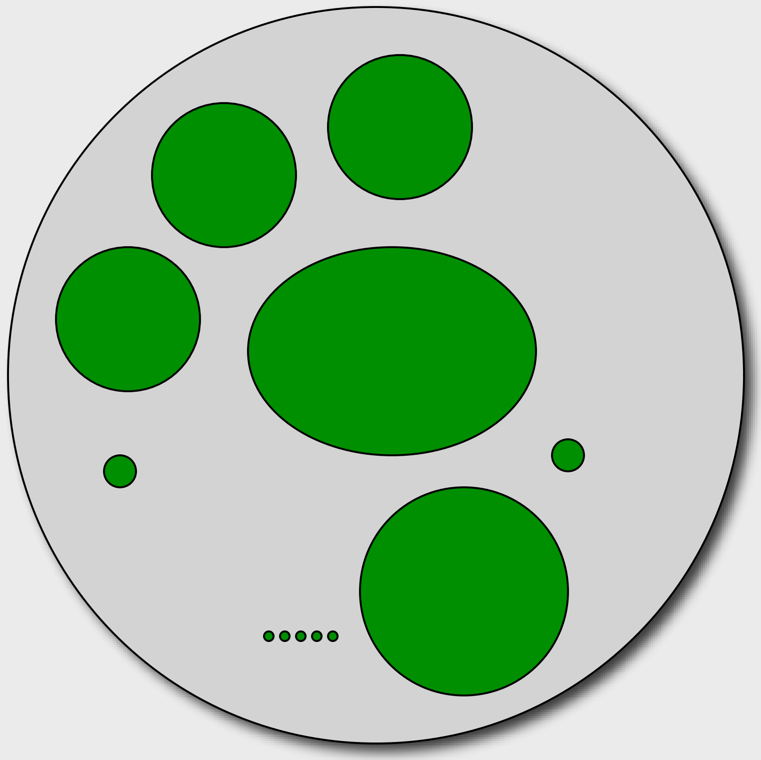


Computation in Hilbert Space



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 “radical reconfiguration”: 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:

$$\mathbf{v} = \mathbf{F}(\mathbf{u}) \approx \sum_{j=1}^{H} \vec{\alpha}^{\,j} r_j(\mathbf{u})$$

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

Example Interpolation Method

- Training samples: $(x_1, y_1), \ldots, (x_P, y_P)$
- Hilbert space reprs.: $\mathbf{u}^k = U(x_k)$, $\mathbf{v}^k = V(y_k)$
- Interpolation conditions: $$\mathbf{v}^k = \sum_j \vec{\alpha}^j r_j(\mathbf{u}^k)$$
- Let $V_{ki} = v_i^k, \quad R_{kj} = r_j(\mathbf{u}^k), \quad A_{ji} = \alpha_i^j$
- Exact interpolation: $\mathbf{V} = \mathbf{RA}$
- Best (least squares) approx.: $\mathbf{A} \approx \mathbf{R}^+\mathbf{V}$ where $\mathbf{R}^+ = (\mathbf{R}^\mathrm{T}\mathbf{R})^{-1}\mathbf{R}^\mathrm{T}$

Typical Basis Functions

- Perceptron-style: $r_j(\mathbf{u}) = r(\mathbf{w}_j \cdot \mathbf{u} + b_j)$
- Radial-basis functions:

$$r_j(\mathbf{u}) = r\left(\left\|\mathbf{u} - \mathbf{c}_j\right\|\right)$$

- Green's function for stabilizer:

$$r_j(\mathbf{u}) = G(\mathbf{u}, \mathbf{u}^j)$$

- Key point: <u>lots of choices</u>

Determination of Interconnection Matrices

- Unknown functions: neural-net training
- Known function — compute offline by:
  - Generating sufficient interpolation samples, or:
  - Determining matrices analytically
- Typical primitives could include those found typically on analog or digital computers

Merging Linear Transforms

- General form of approximation:
  $\mathbf{v} = \mathbf{A}\ \mathbf{r}(\mathbf{Bu})$, where $[\mathbf{r}(\mathbf{Bu})]_j = r([\mathbf{Bu}]_j)$
- Suppose: $\mathbf{w} = \mathbf{A}'\ \mathbf{r}'(\mathbf{B}'\mathbf{v})$
- Linear parts can be combined:
  $\mathbf{w} = \mathbf{A}'\ \mathbf{r}'[\mathbf{C}\ \mathbf{r}(\mathbf{Bu})]$, where $\mathbf{C} = \mathbf{B}'\mathbf{A}$
- Two basic operations:
  - Matrix-vector multiplication
  - Simple point-wise nonlinear function **r**

Input Transduction

- Input space: (X, δ)
- Suitable ε-net: $(b_1, b_2, \ldots, b_n)$
- Hilbert-space representation $\mathbf{u} = U(x)$:
  $u_k = \sigma\,(x, b_k)$,
  where $\sigma\,(x, b_k) = 1 - \delta\,(x, b_k)$
- In effect have fuzzy feature detectors:
  $\sigma_k(x) = \sigma\,(x, b_k)$
- Thus $\mathbf{u} = (\sigma_1(x), \sigma_2(x), \ldots, \sigma_n(x)\,)^T$

Output Transduction

- Physical output spaces are usually vector spaces
- Approximate by: $y = V^{-1}(\mathbf{v}) \approx \sum_{j=1}^{H} \mathbf{a}^j r_j(\mathbf{v})$
- This summation is a physical superposition of physical vectors $\mathbf{a}^j$
- Compute approximation parameters in any of the usual ways

Real-time Computation

- Time-varying input & outputs $x(t)$ are represented by time varying vectors $\mathbf{u}(t)$
- Differential changes are computed like other functions
- Differential changes are integrated into variable regions

(Re-)Configuration Methods

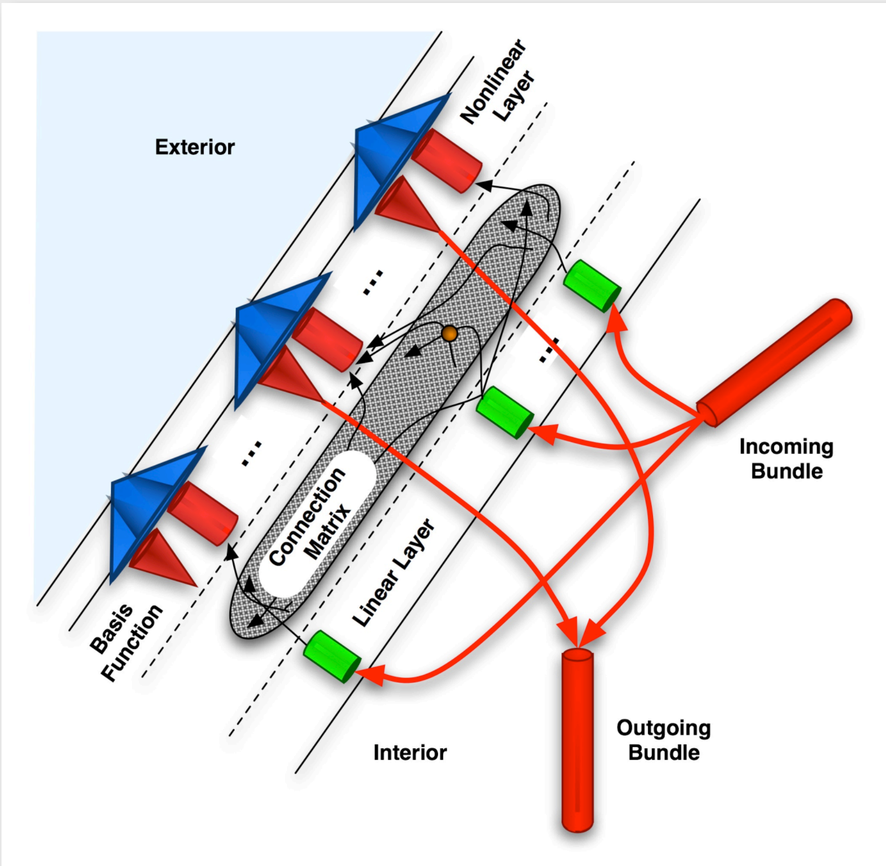
Overall Structure

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

Depiction of UM Interior

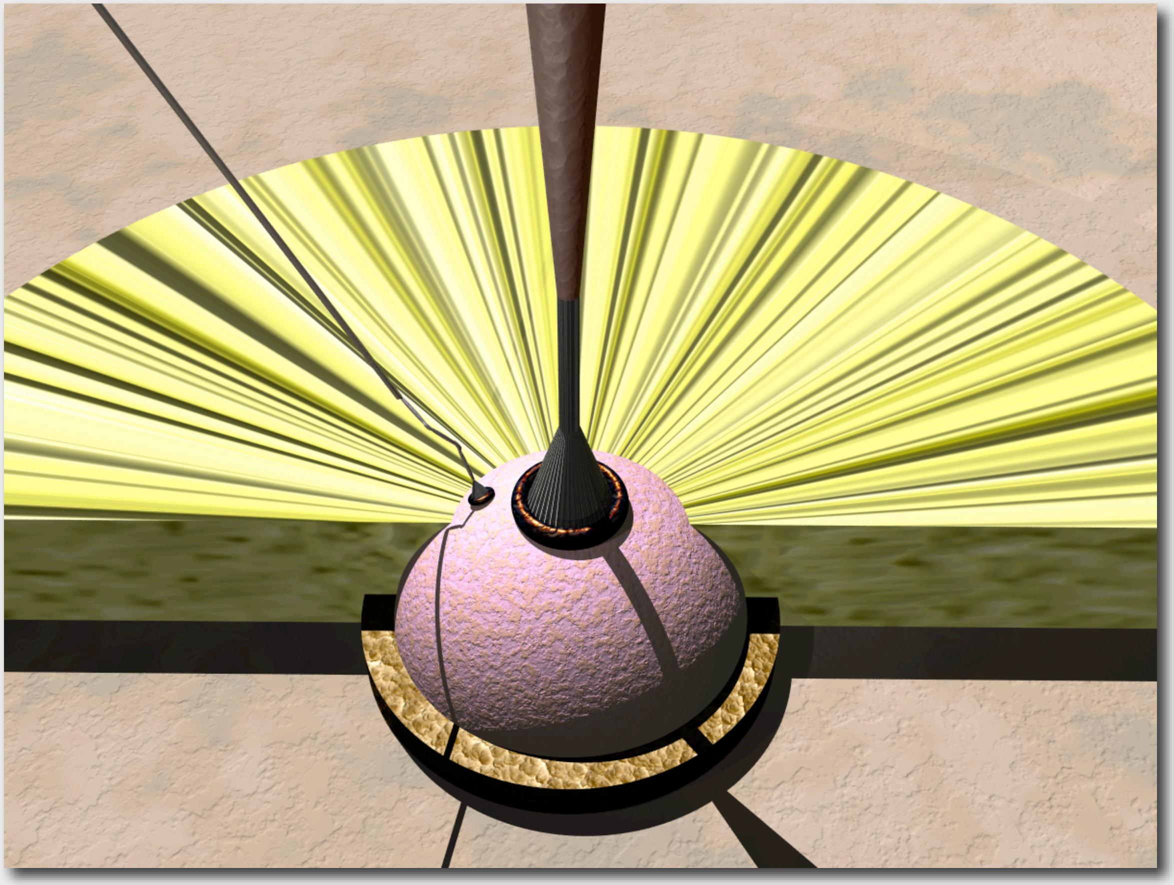
- Shell contains variable areas & computational elements
- Interior filled with solid or liquid *matrix* (not shown)
- Paths formed through or from matrix

Layers in Data Space

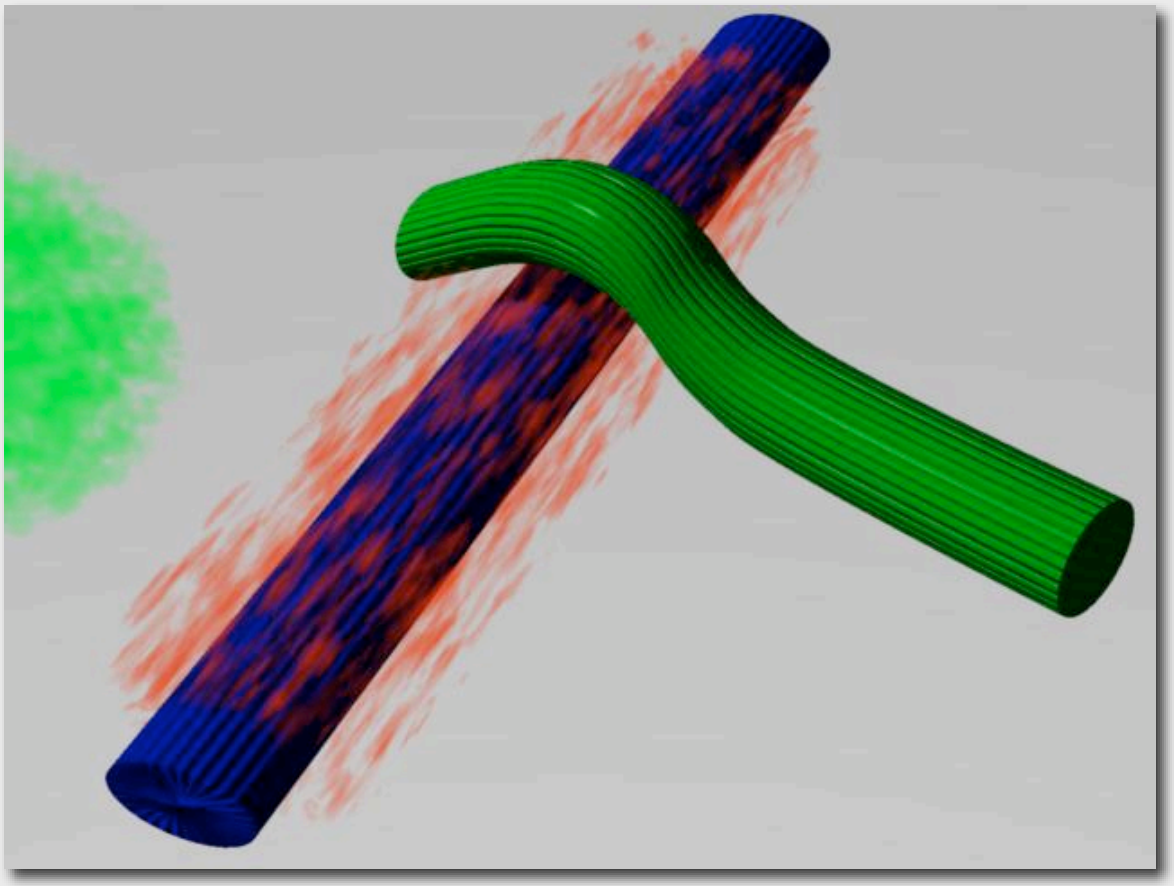


- Connection matrix has programmable weights
- Linear combinations are inputs to nonlinear basis functions
- Exterior access to both sides for programming

Depiction of UM Exterior



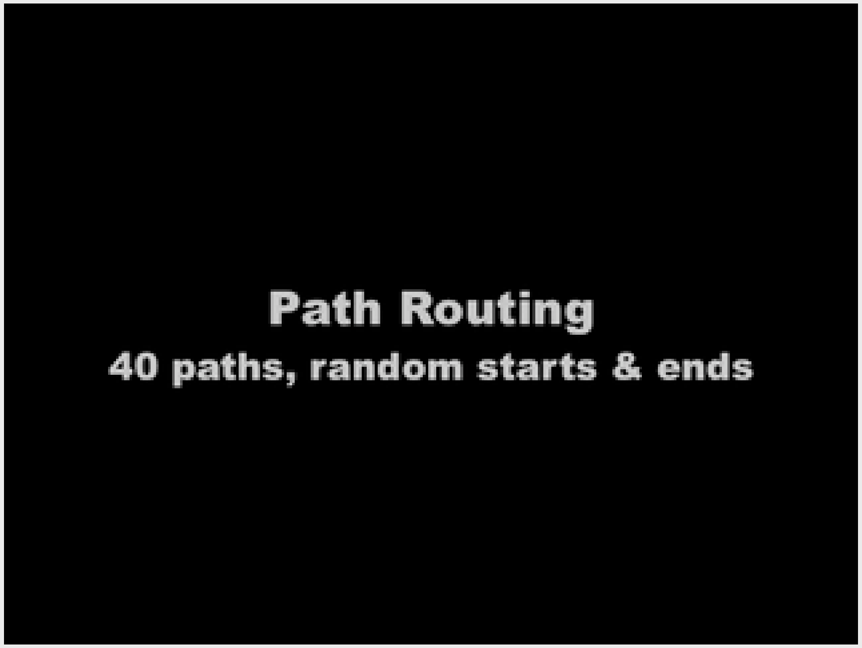
Diffusion-Based Path Routing



Example Path-Routing Process

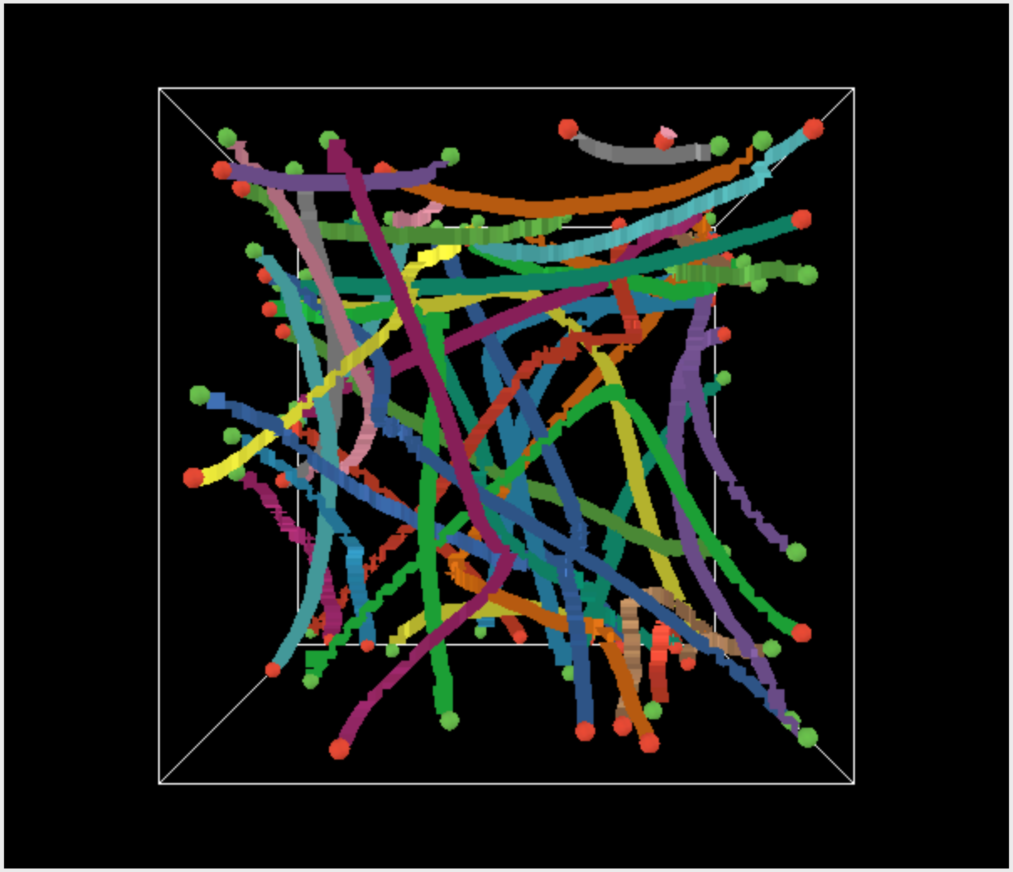
- Attractant diffuses from destination
  - Could be chemical, electrons, molecular state
  - Attractant degrades
- Existing paths clamp attractant to 0
  - Effectively repel new path
- Path "grows" from source by climbing attractant gradient
  - Attractant injection rate ramped up
- After connection made, attractant allowed to decay before routing next path

Example of Path Routing

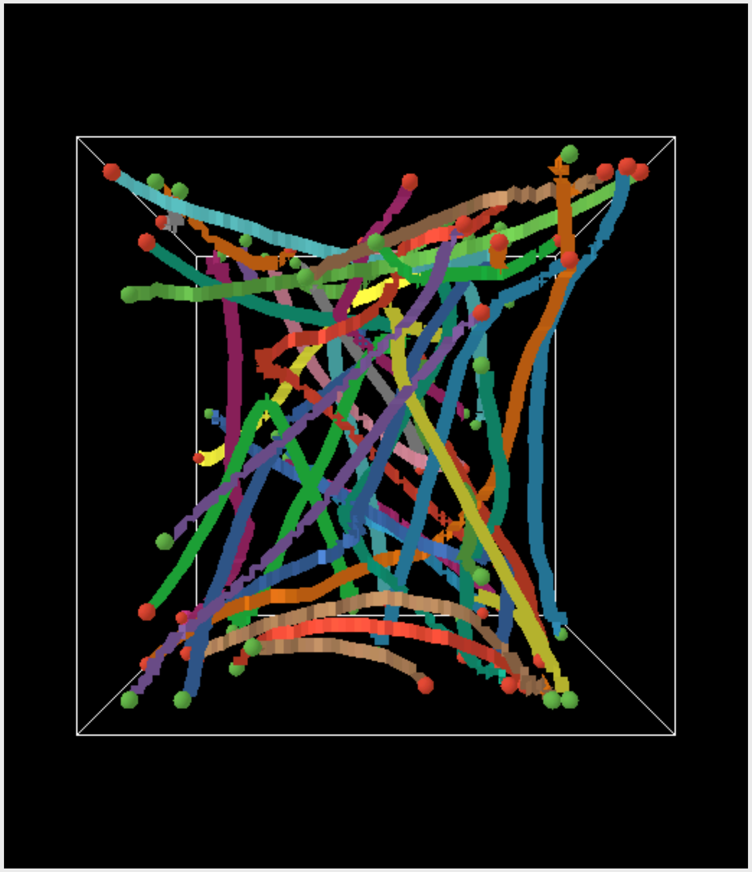


- Starts and ends chosen randomly
- Quiescent interval (for attractant decay) omitted from video
- Each path occupies ~0.1% of space
- Total: ~4%

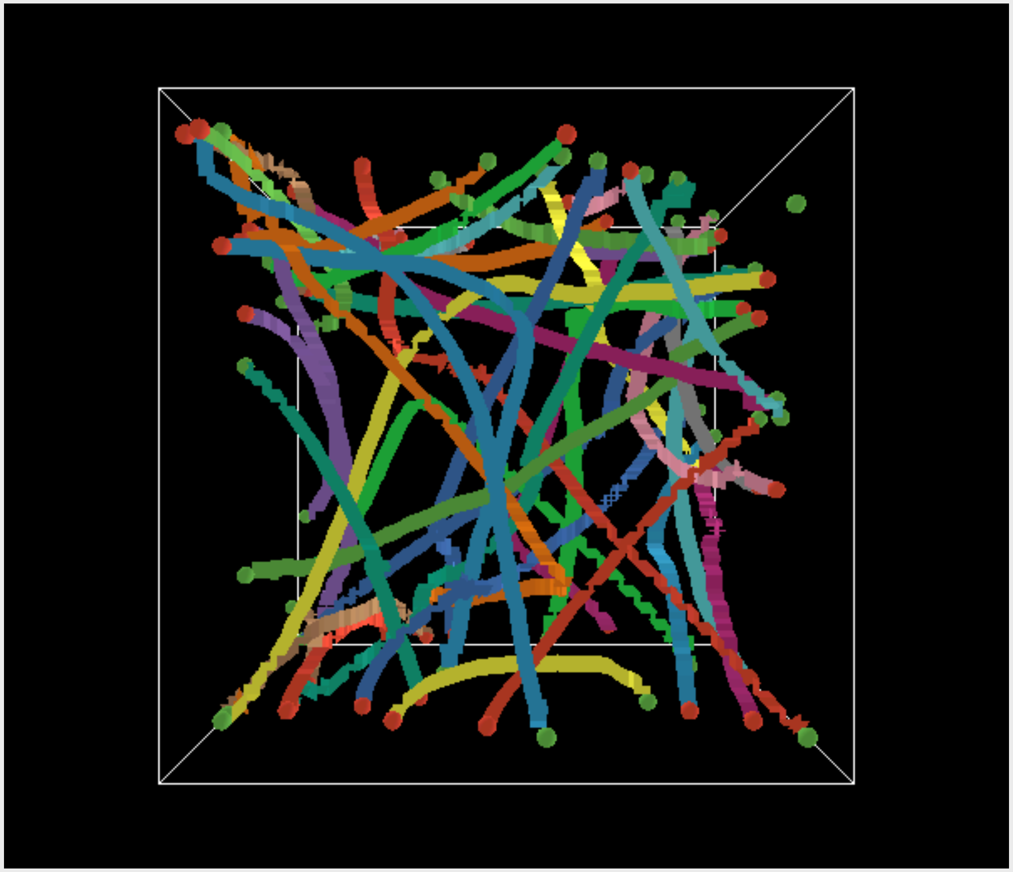
Front



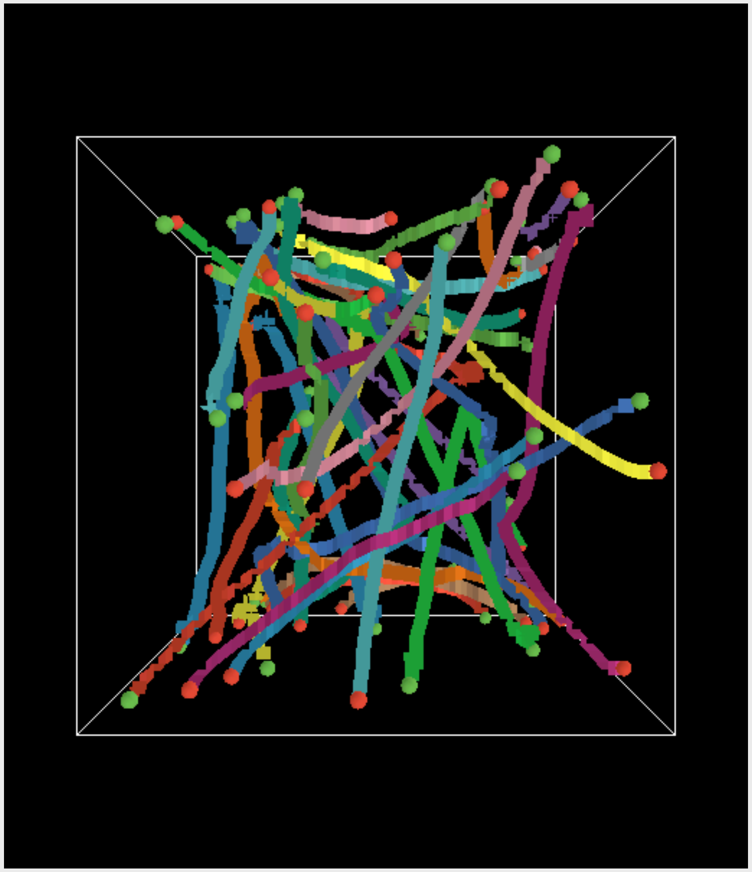
Right



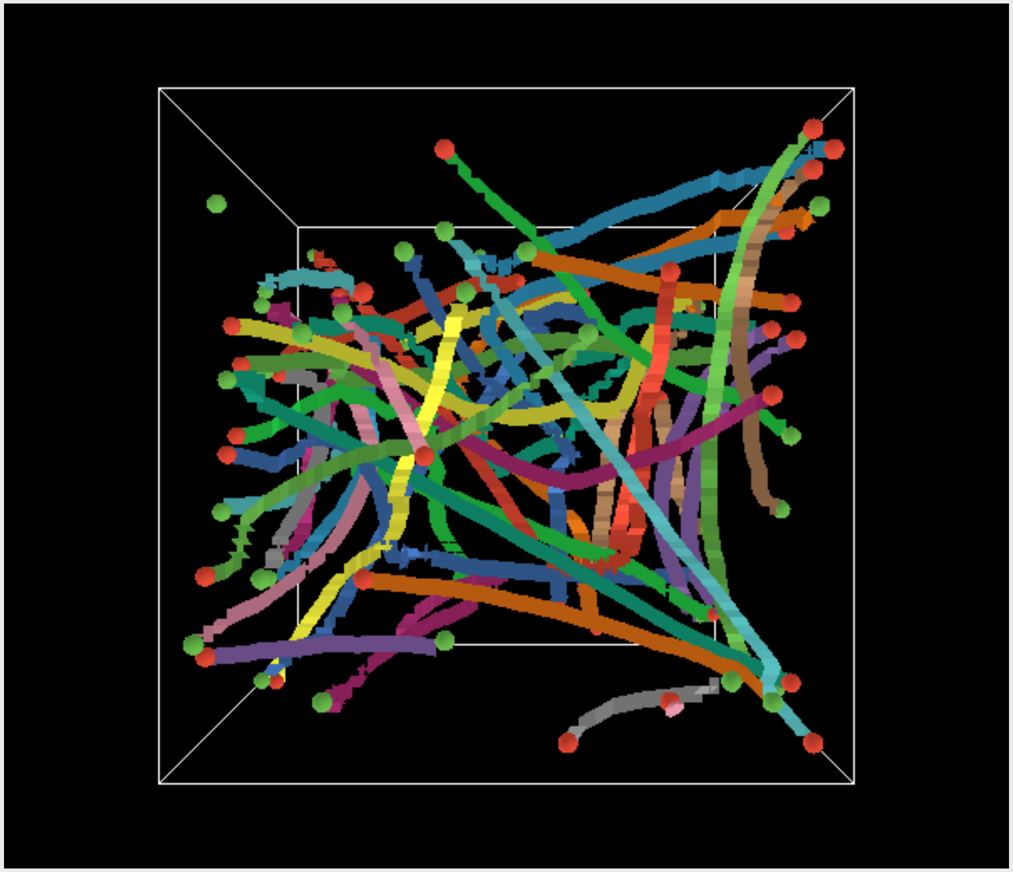
Back



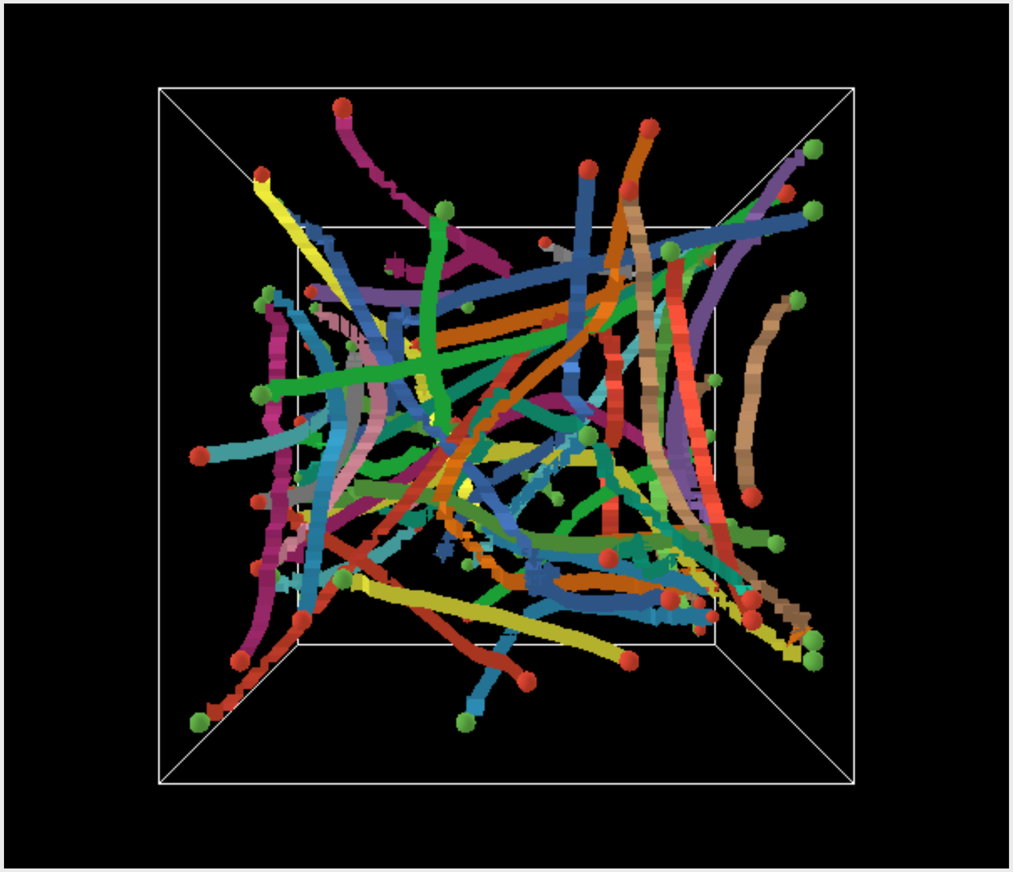
Left



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

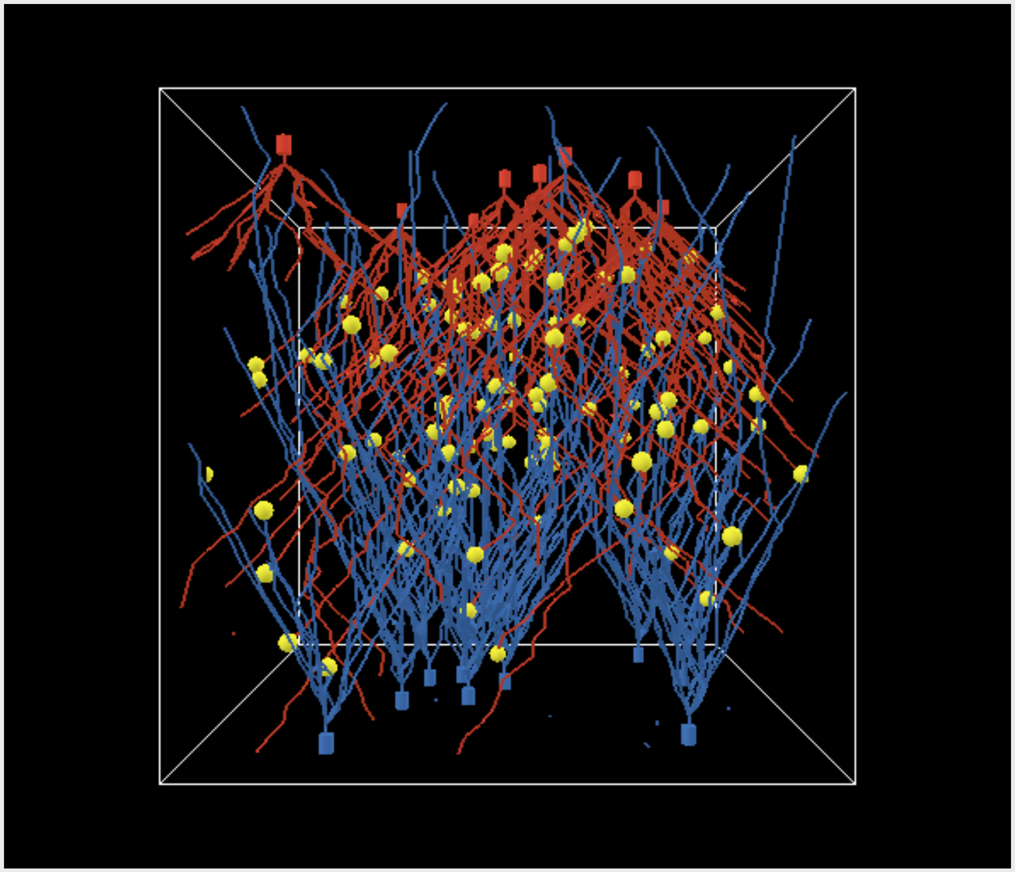
Example Connection-Growth Process

- 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



Setting Connection Strengths by SVD

- Let $m \times n$ connection matrix $\mathbf{M} = \mathbf{U} \Sigma \mathbf{V}^{\mathrm{T}}$, where

$$\Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}$$

  and $\Sigma_r = \operatorname{diag}(s_1, \ldots, s_r)$
- Let $\mathbf{u}_k$ and $\mathbf{v}_k$ be columns of $\mathbf{U}$ and $\mathbf{V}$:

  $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$, $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$
- Then,

$$\mathbf{M} = \sum_{k=1}^{r} s_k \mathbf{u}_k \mathbf{v}_k^{\mathrm{T}}$$

- For each k, apply $\mathbf{v}_k$ to input and $\mathbf{u}_k$ to output and program with strength s_k

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

Highly Programmable Matter

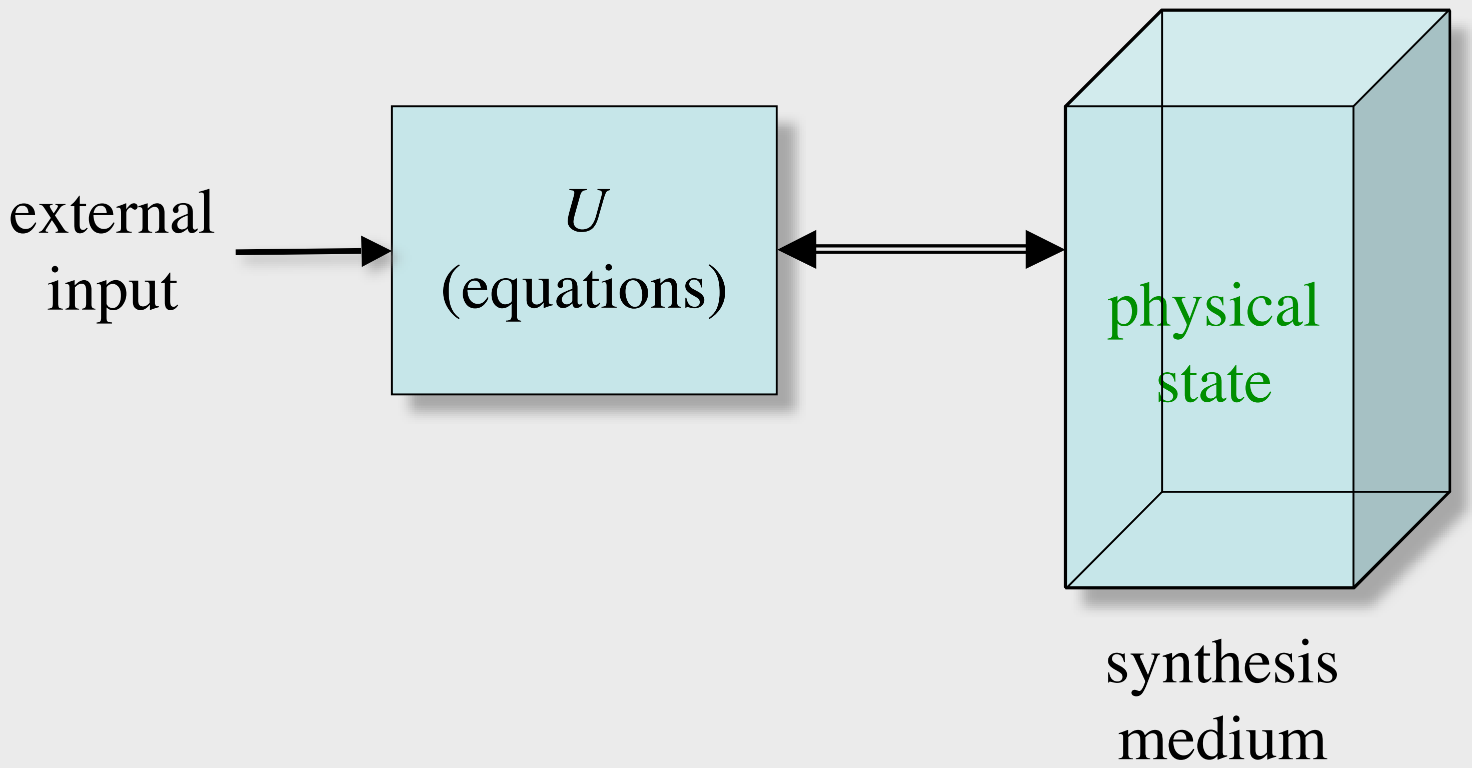
Computational Control of Matter

- 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

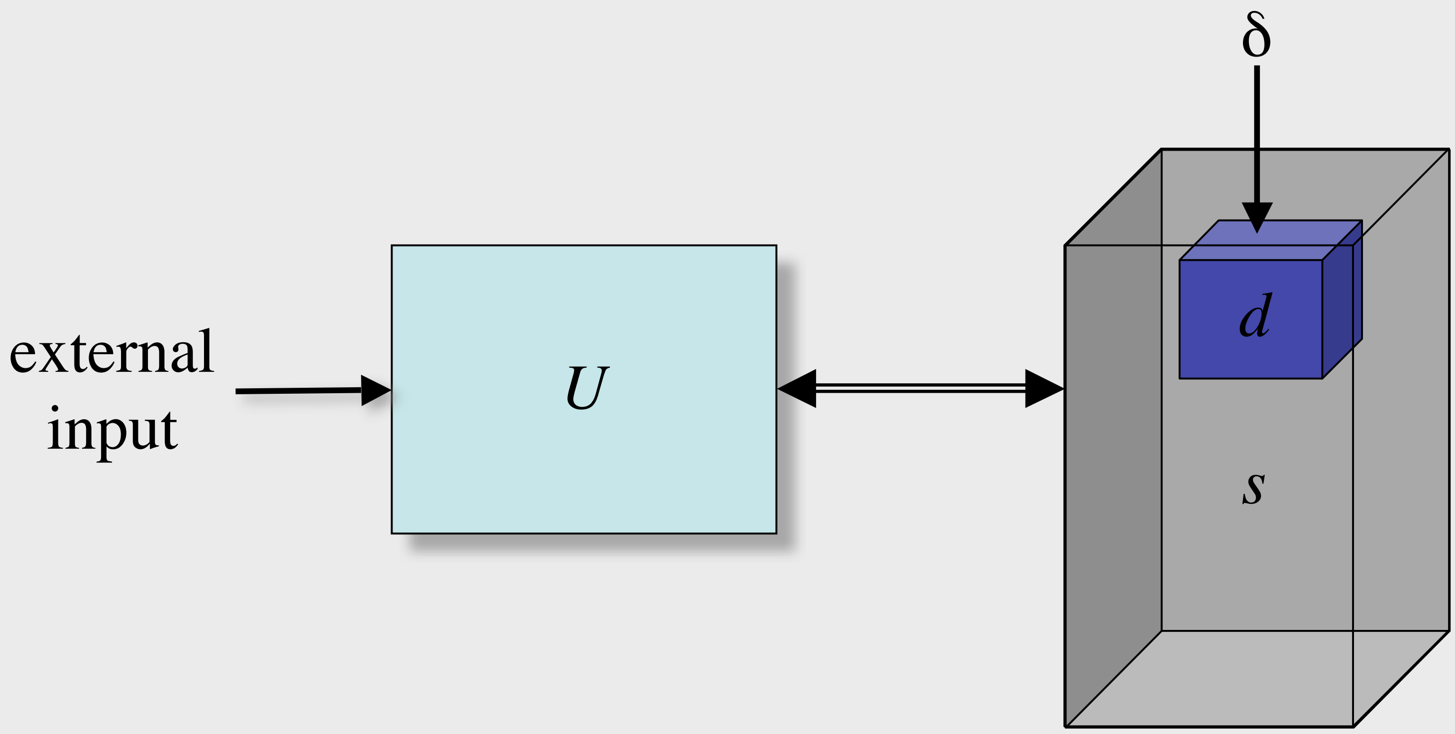
The Physical State as Synthetic Medium

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

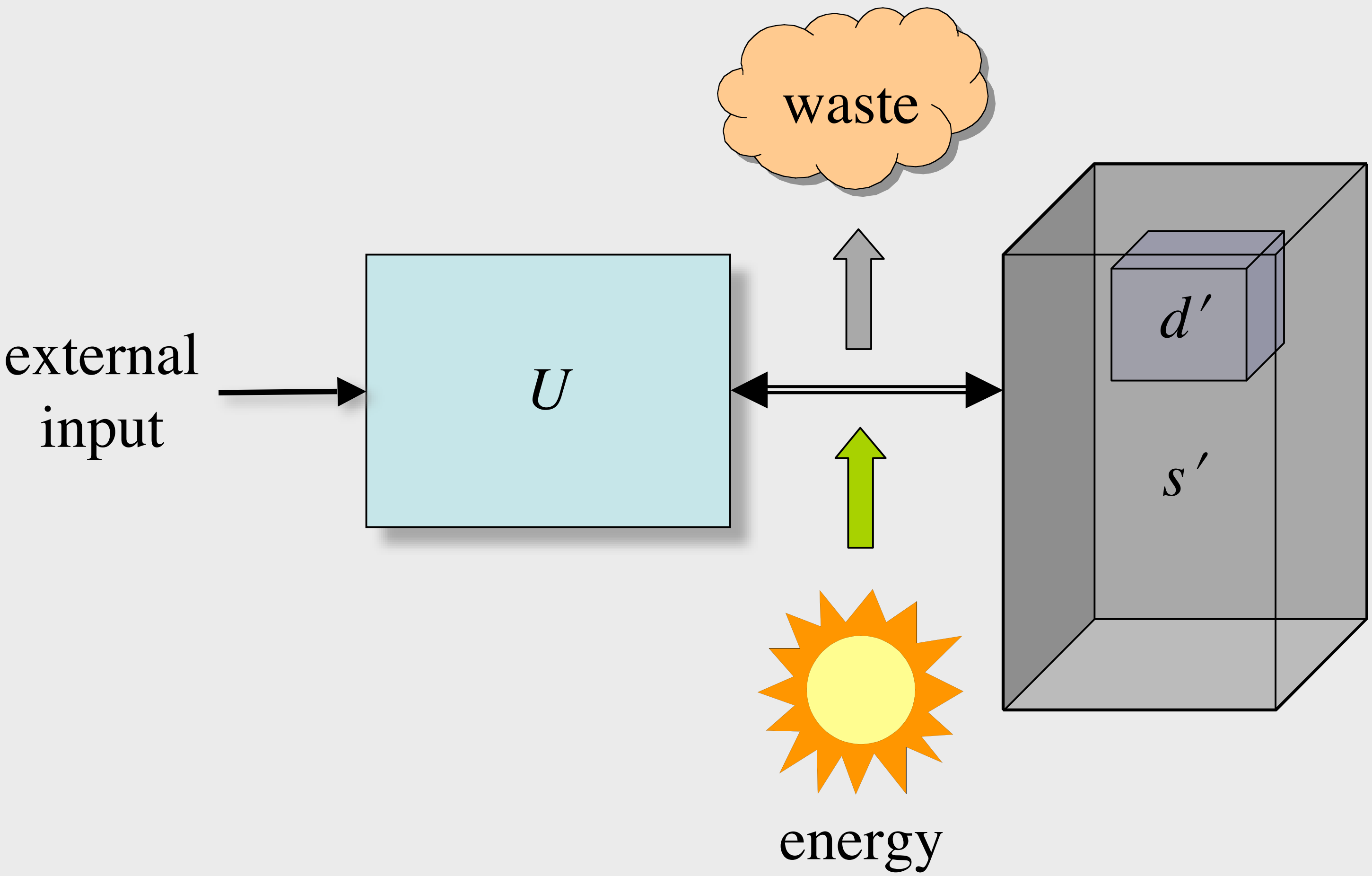
Universal Computer



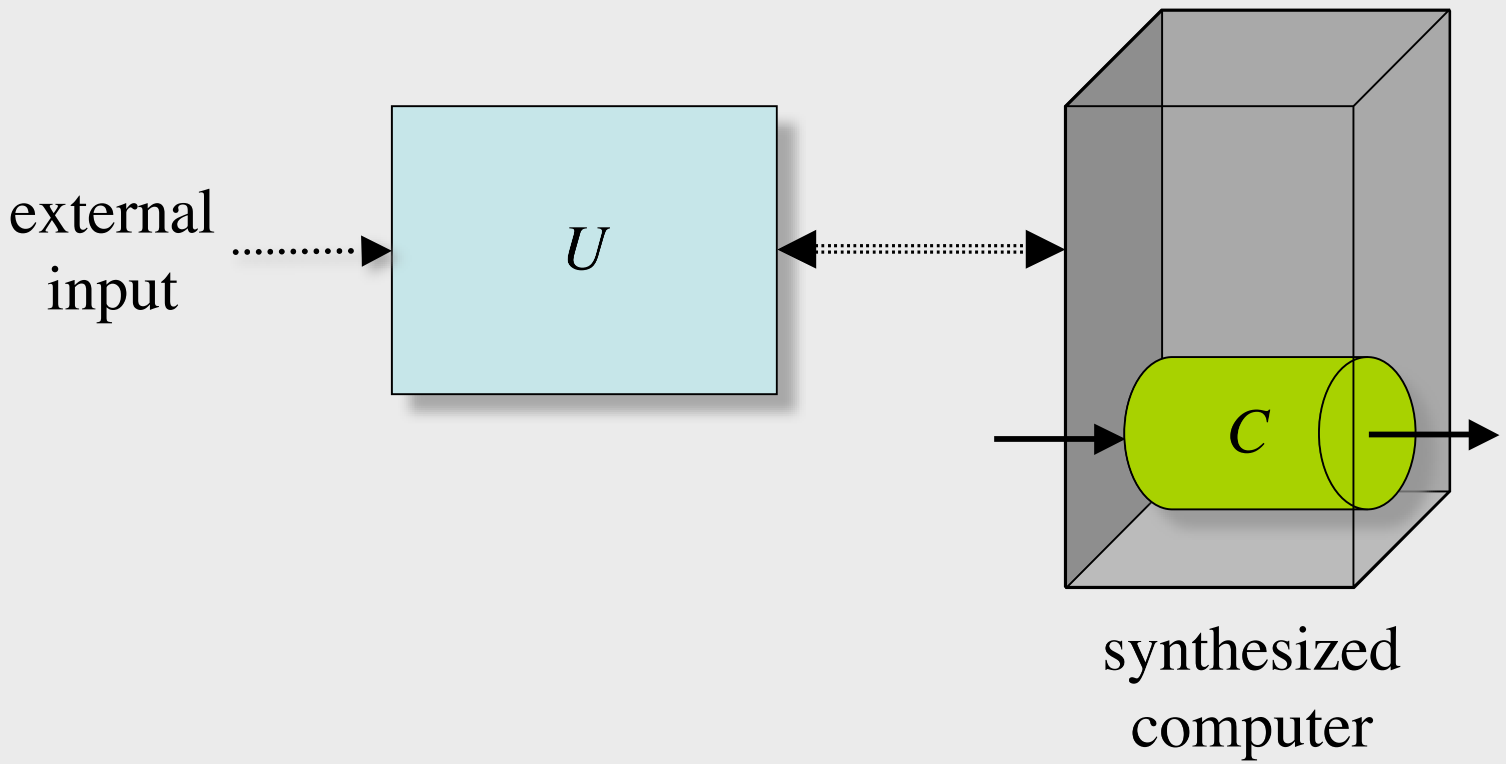
Initialization



Computation

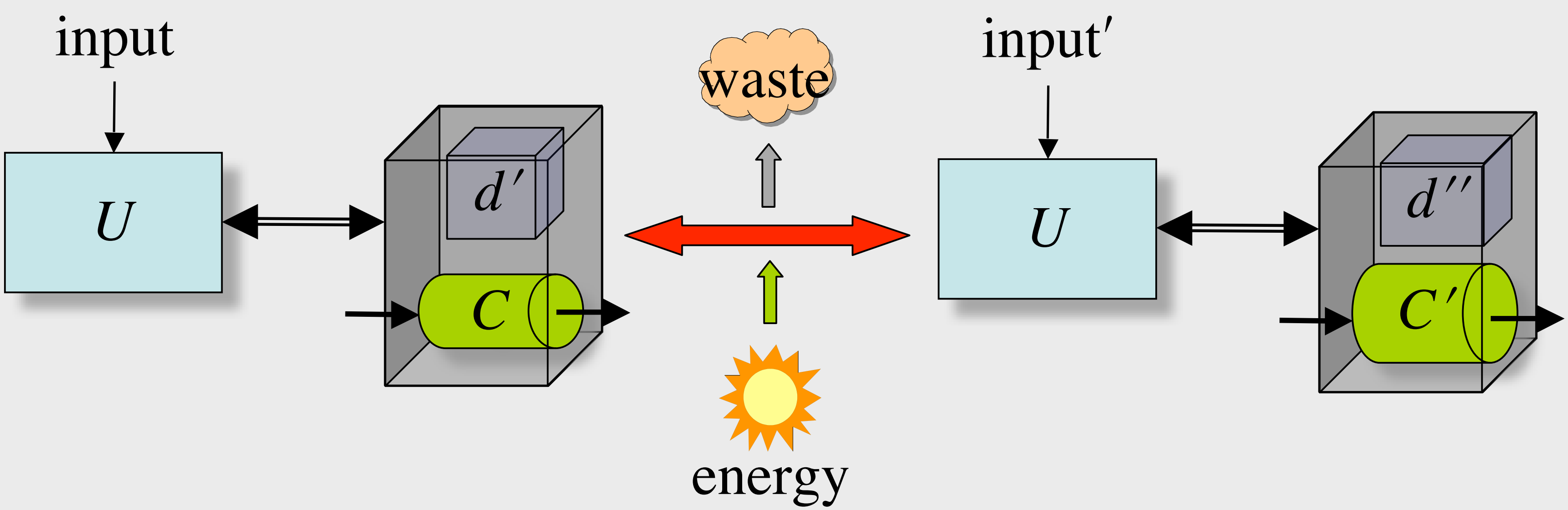


Completion



Equilibrium vs. Stationary Configurations

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



Thermodynamics of a Configuration

- Either, configuration is a stable state
  - damage may shift to undesirable equilibrium
- Or, configuration is a stationary state 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
  $\Rightarrow$ 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?

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

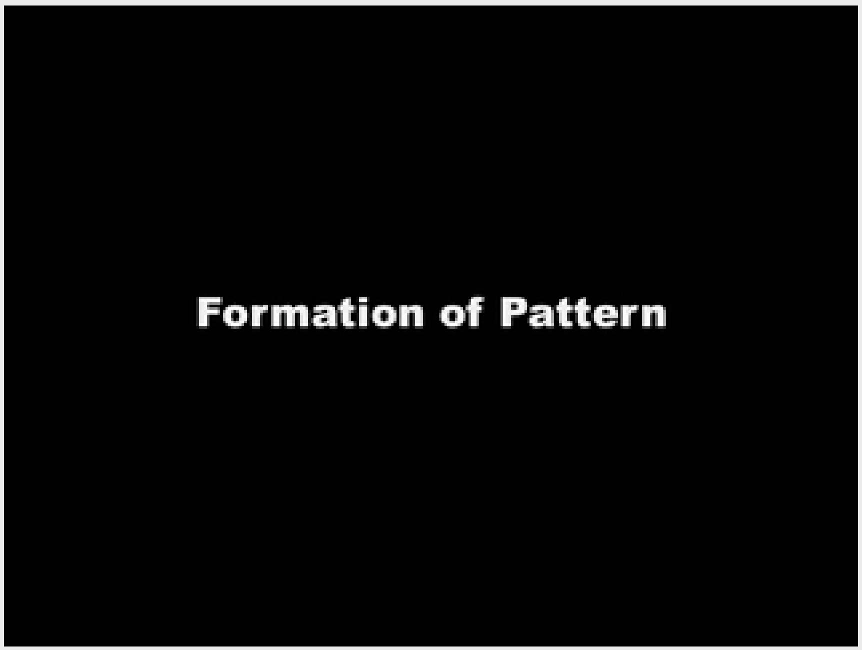
$$\frac{\partial A}{\partial t} = d_{\mathrm{Ax}} \frac{\partial^2 A}{\partial x^2} + d_{\mathrm{Ay}} \frac{\partial^2 A}{\partial y^2} + k_{\mathrm{A}} \sigma\left[m_{\mathrm{A}}\left(A + B - I\right)\right]$$

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

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

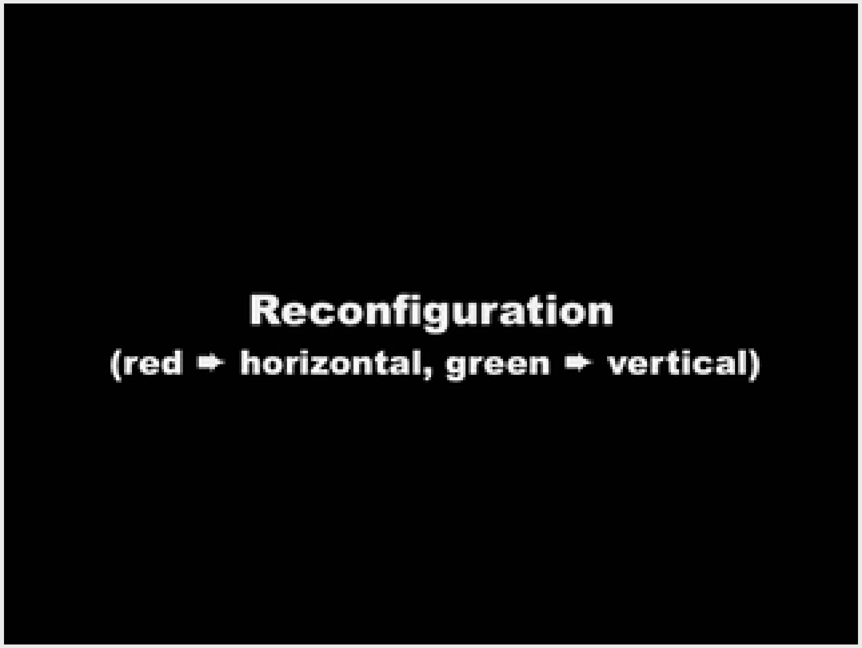
Stationary State

- 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

Conclusions

- Radical reconfiguration can be accomplished by using computation to change matter
  - external control of macrostructure
  - self-organization of microstructure
- A simple, flexible architecture can compute over a variety of abstract spaces (including analog & digital)