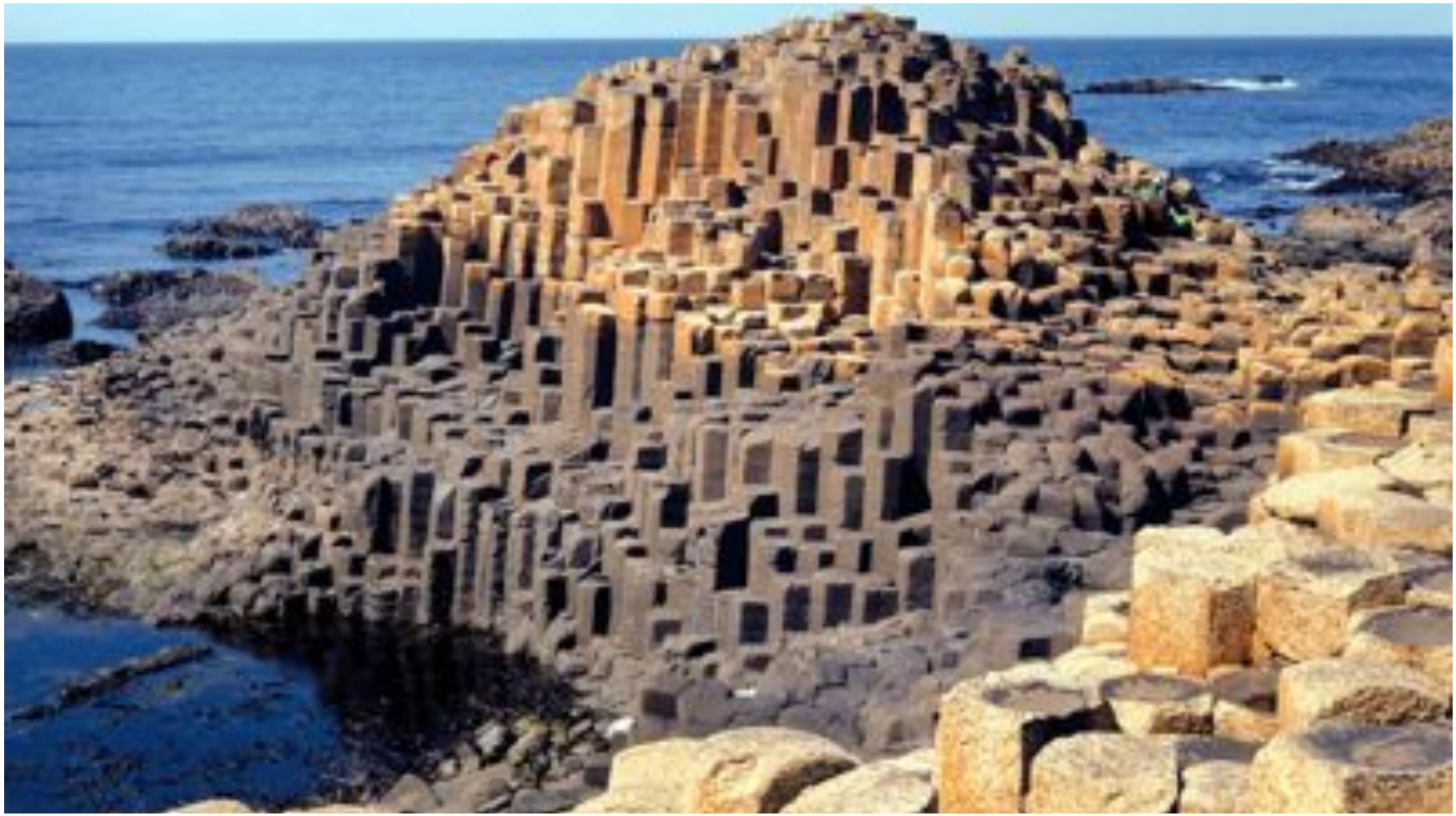
Parallel Computation in Nature

Constructed by aliens, or naturally “computed”?



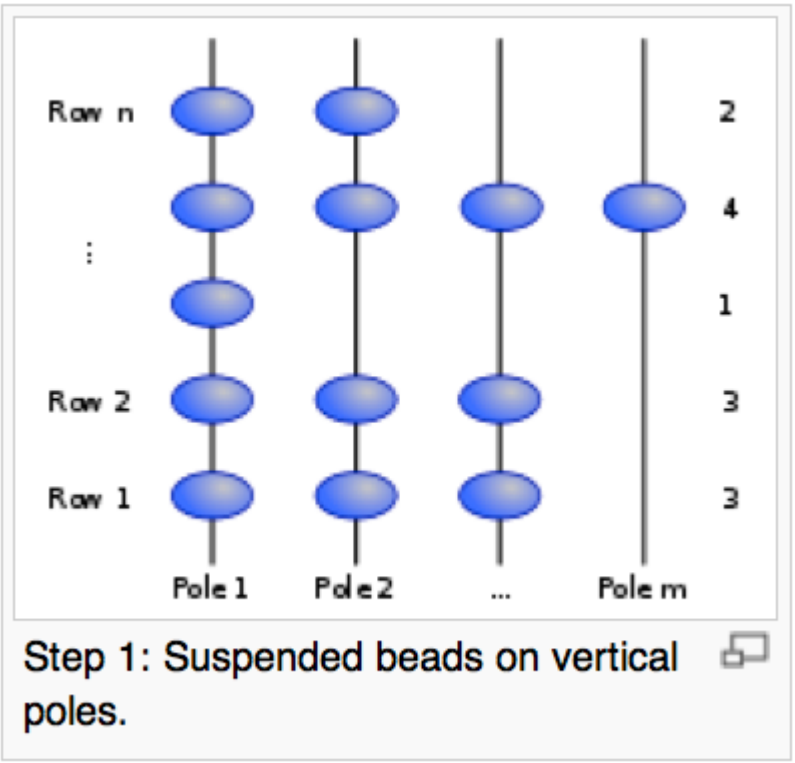
http://en.wikipedia.org/wiki/Giant's_Causeway

Evidence of a Designer?

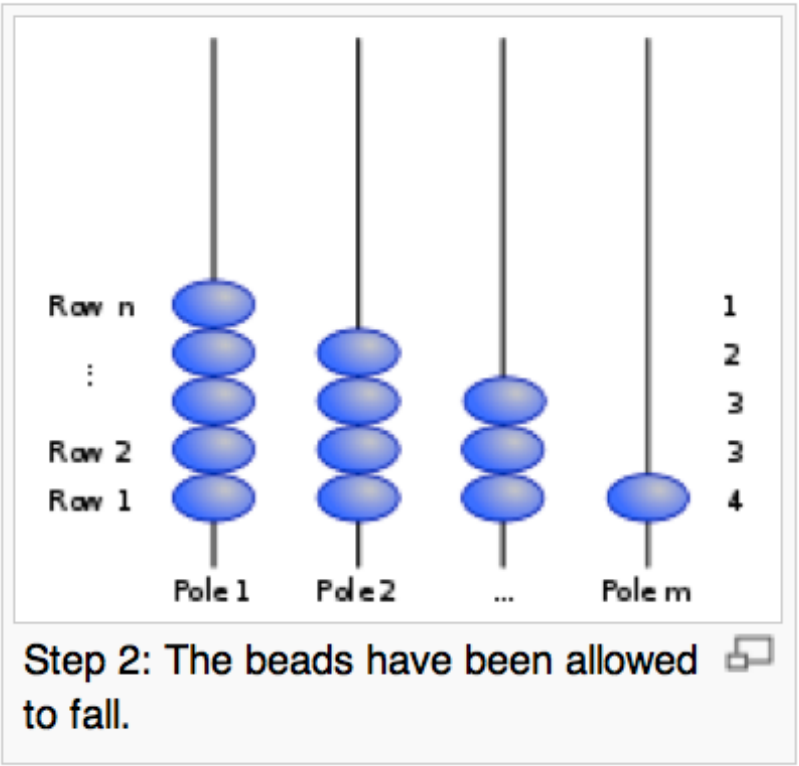
http://en.wikipedia.org/wiki/Crystallization

“Natural” Sorting Algorithms

Bead Sort



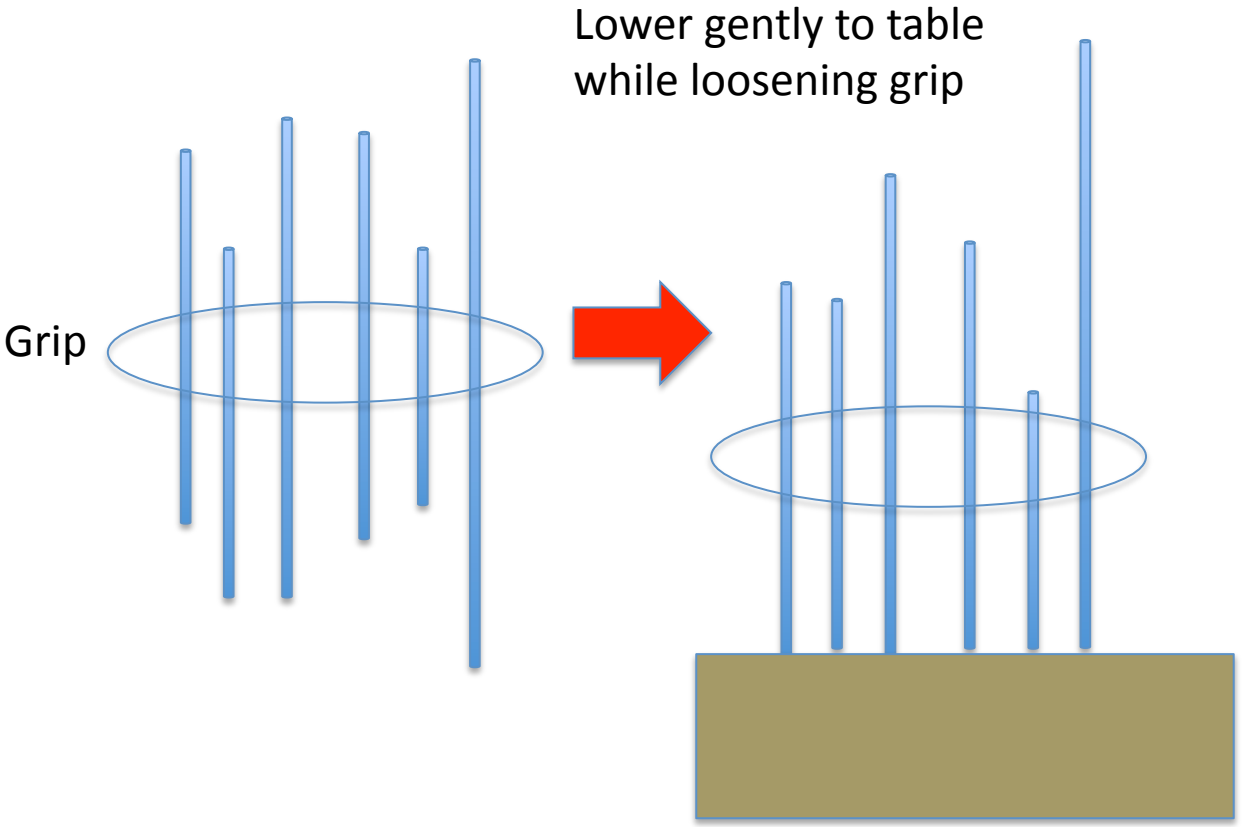
Step 1: Suspended beads on vertical poles.



Step 2: The beads have been allowed to fall.

“Natural” Sorting Algorithms

Spaghetti Sort



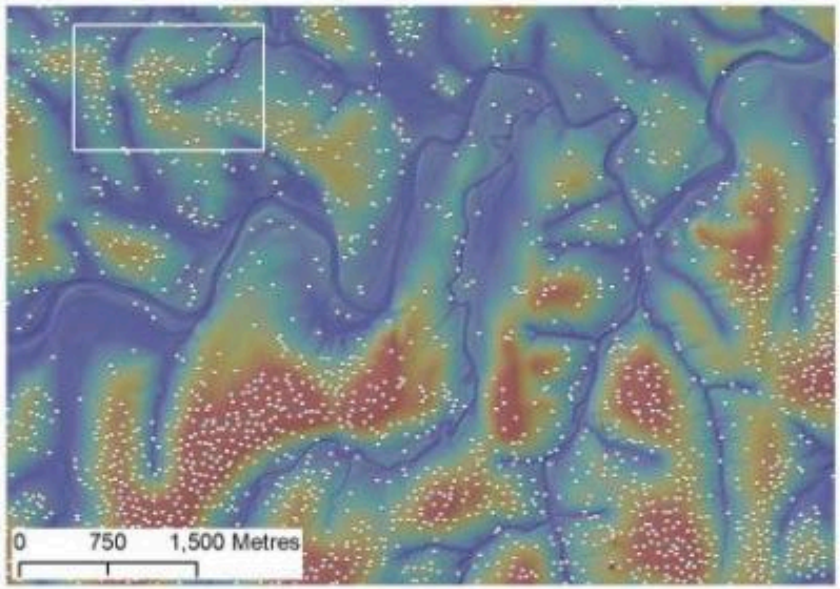
(only works with **uncooked** spaghetti)

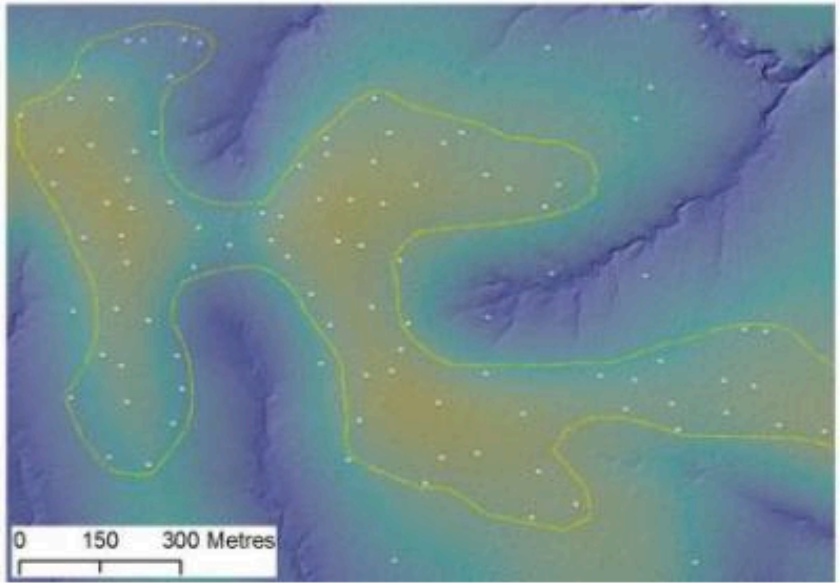
Characteristics of Computation in Nature

- Results “emergent” or “self-organizing”, rather than pre-programmed
- A collection of agents or cells operate in parallel.
- Each agent depends only on a finite set of other agents; no central coordinator.
- Agents influence each other mutually.
- The other agents are in proximity to the agent being influenced.
- May exhibit layering, hierarchy.
- Agents could be physical, chemical, biological.

Related Terms

- “Complexity”
- “Chaos” (“Butterfly effect”, arose out of attempts to predict the weather mathematically)
  video: http://www.youtube.com/watch?v=Gtn4mZtiQHA&NR=1
  Self Organization of Life – Part 3



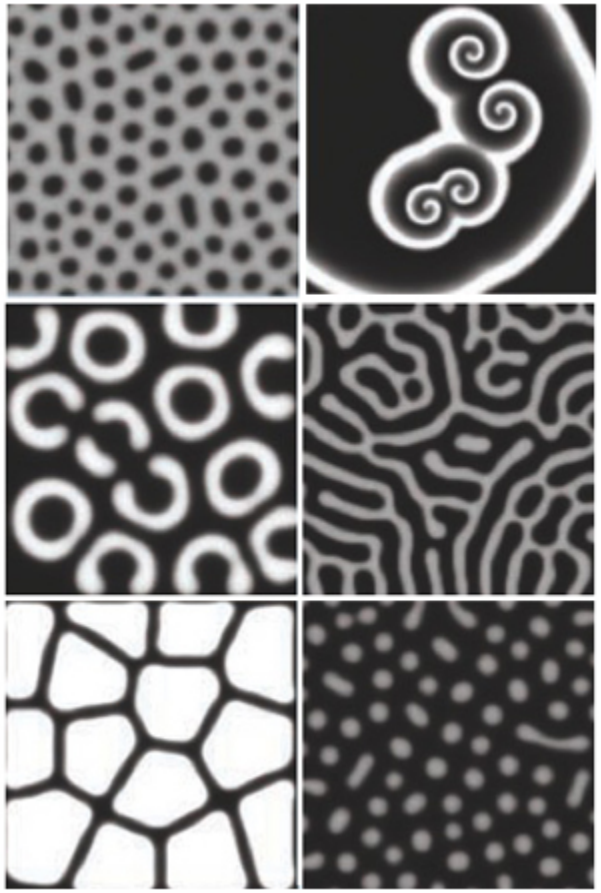


This mapping shows the distribution of termite mounds relative to seeplines, parts of savanna slopes where water has flowed below ground through sandy, porous soil and backs up at areas rich in clay. Warm colors indicated higher elevations above river channels and the dots indicate termite mounds.

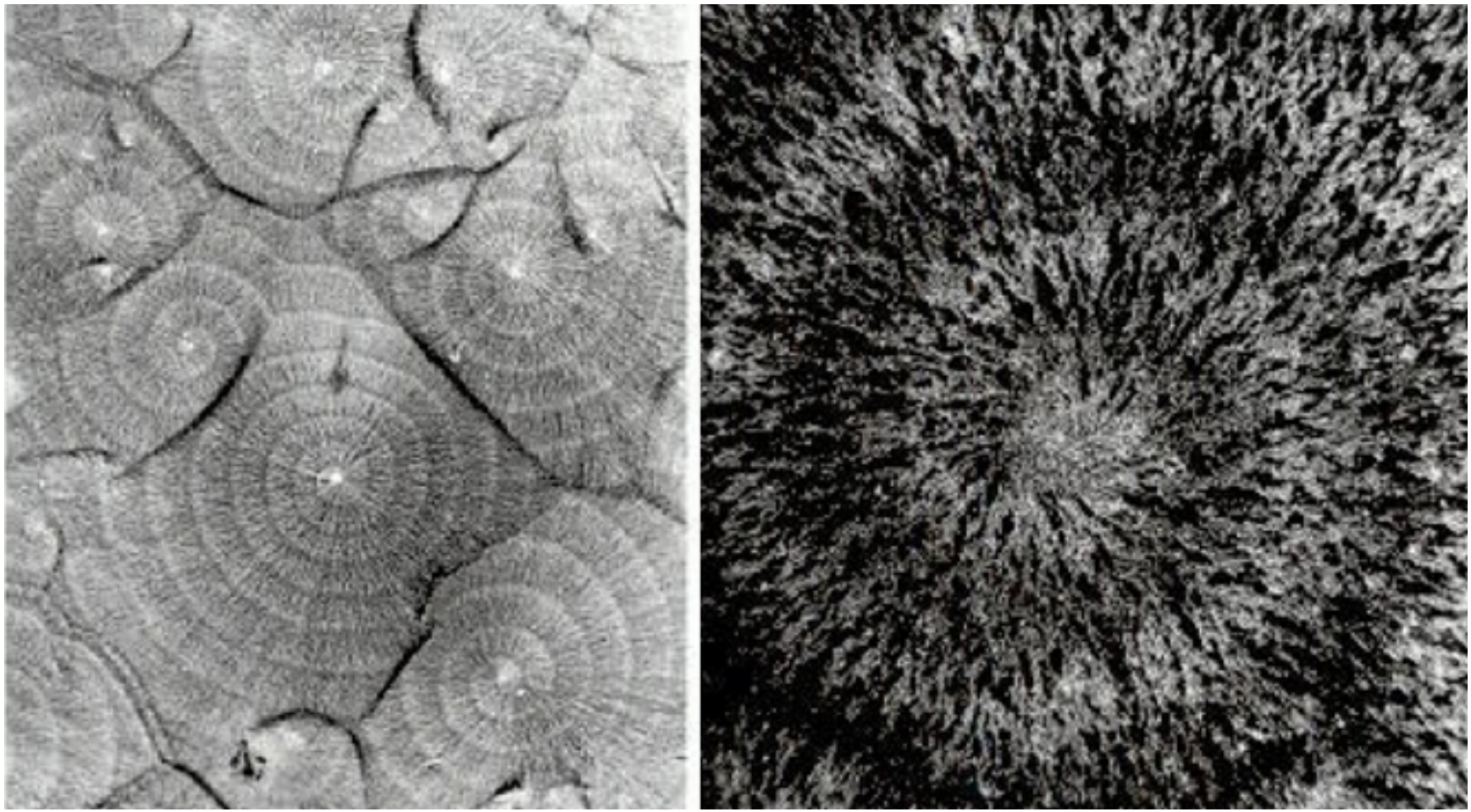
CLOSE ✕

Turing: Emergence of Patterns

- "The Chemical Basis of Morphogenesis", 1952
- Differential equation model
- Able to explain patterns such as those on the right
- Reference for this and following slides:
  http://www.wired.com/wiredscience/2011/02/turing-patterns/

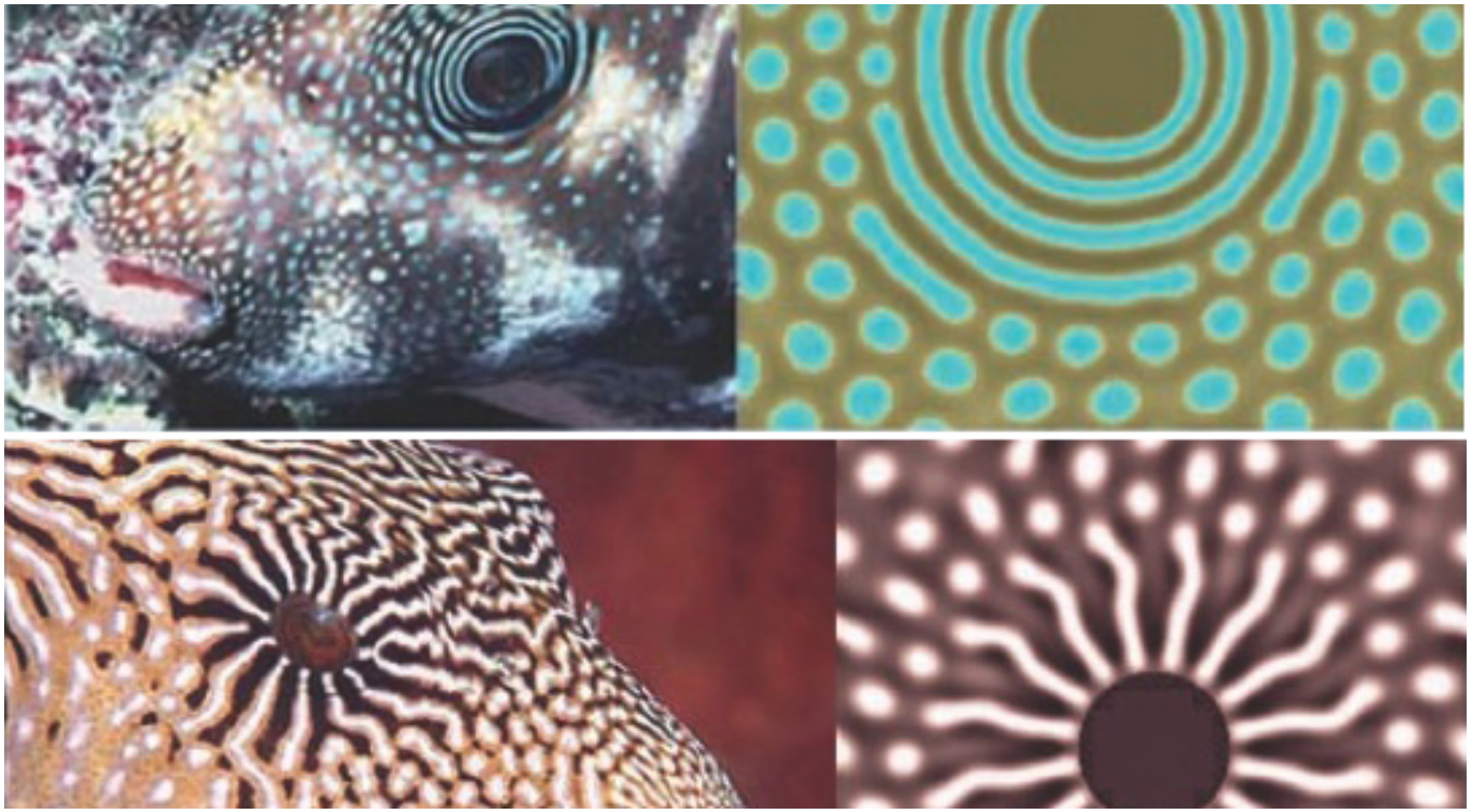


Turing Patterns in Slime Mold

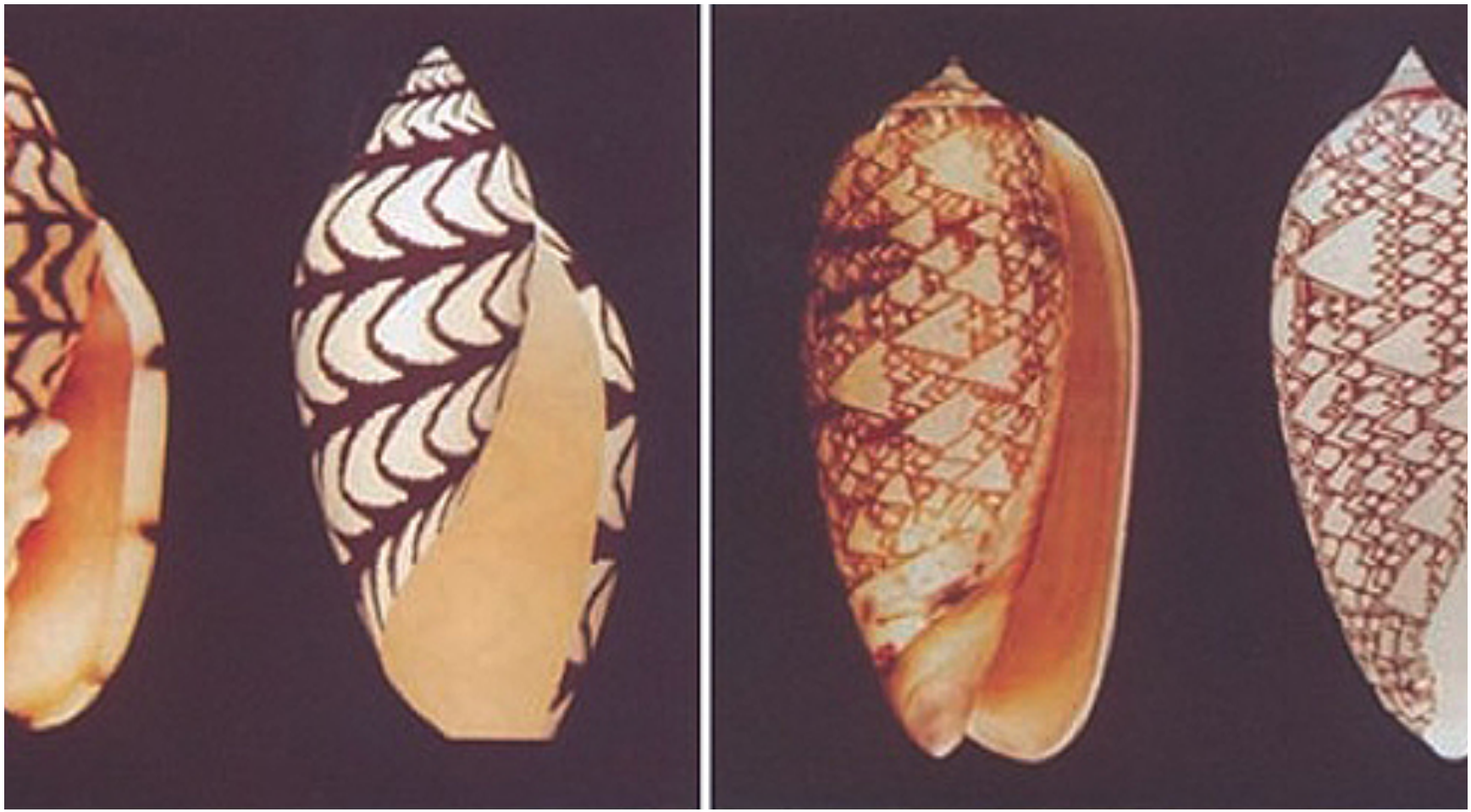


Turing Pattern in the Cosmos

Turing Patterns in Animals

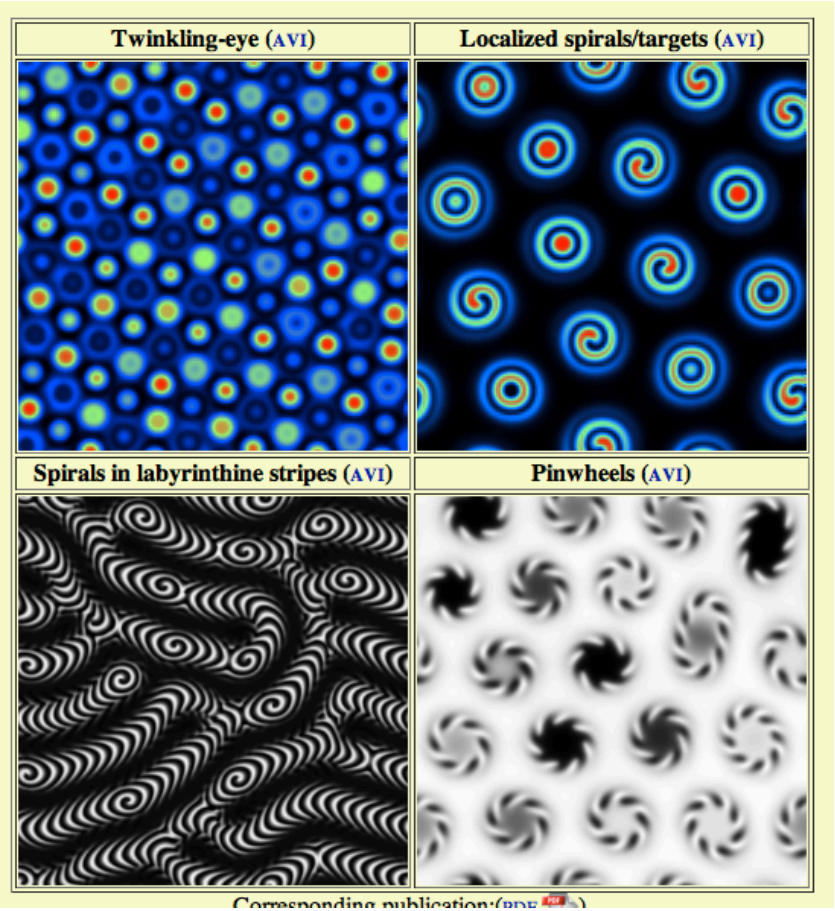


Turing Patterns in Animals



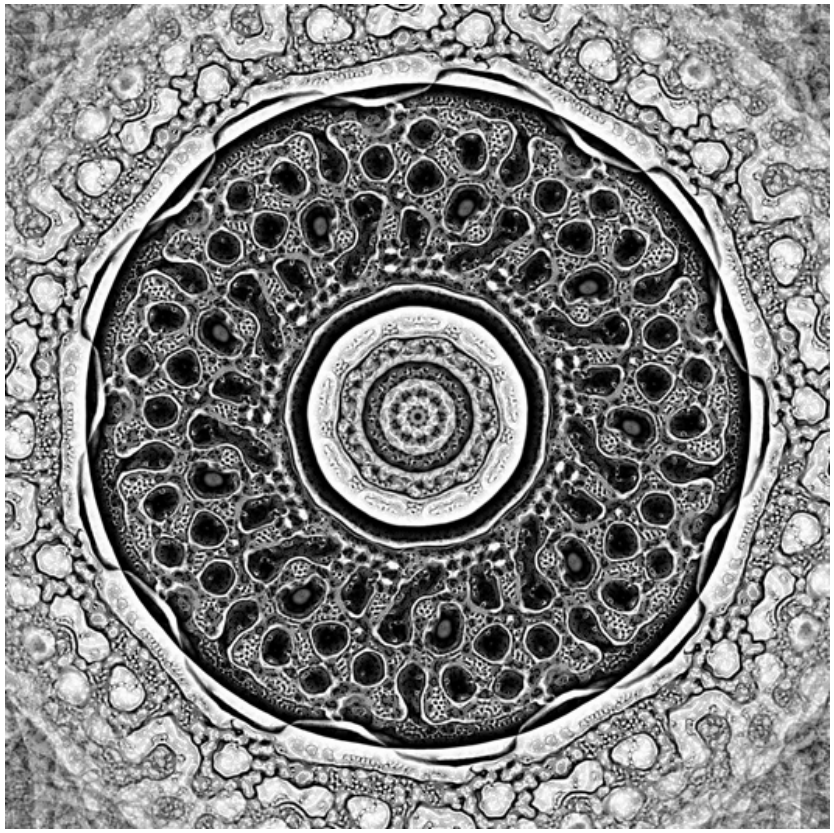
Oscillatory Turing Patterns

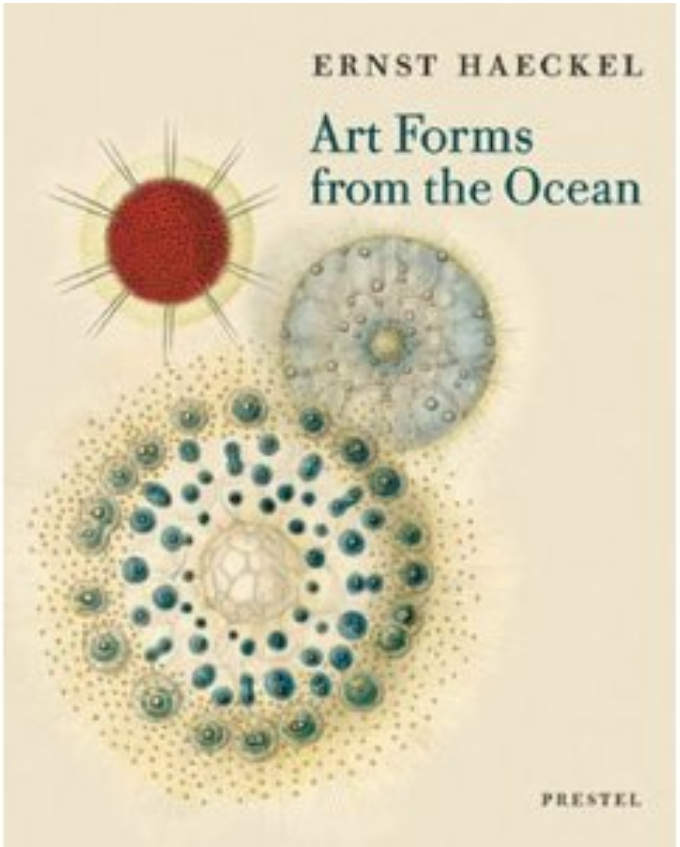
- See:
  http://hopf.chem.brandeis.edu/yanglingfa/pattern/oscTu/index.html



Radial Symmetry

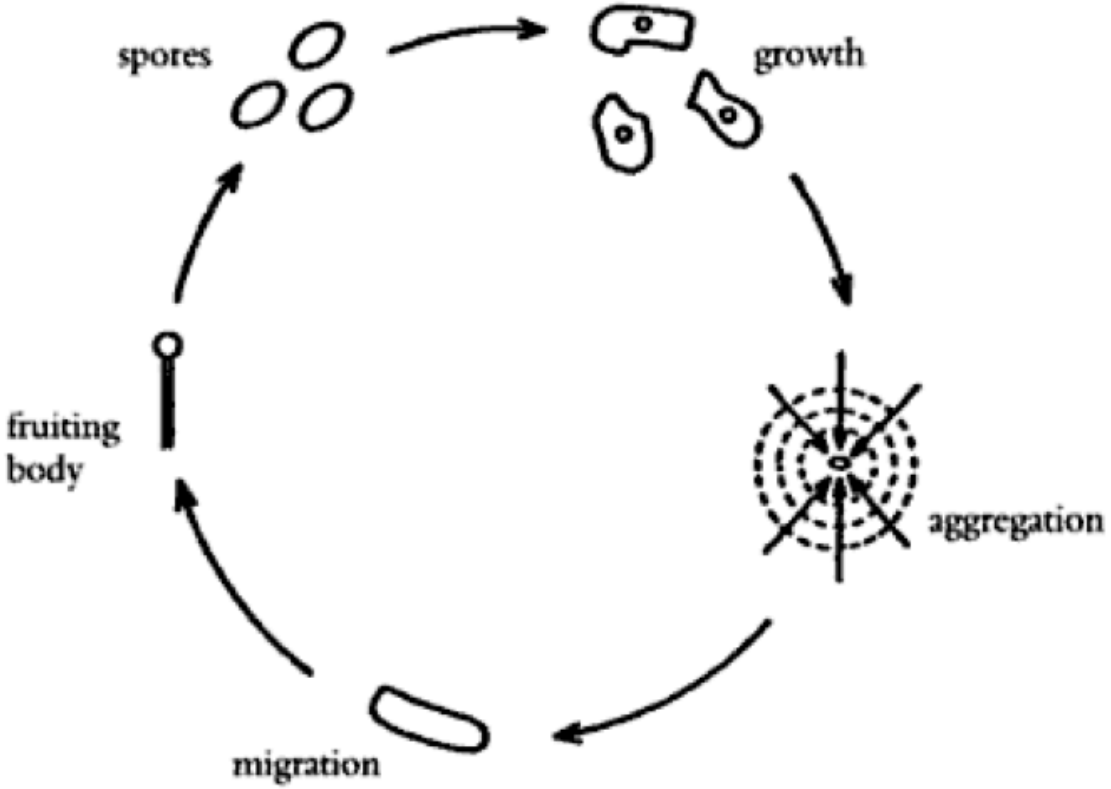
- See:
  http://vagueterrain.net/journal14/jonathan-mccabe/01

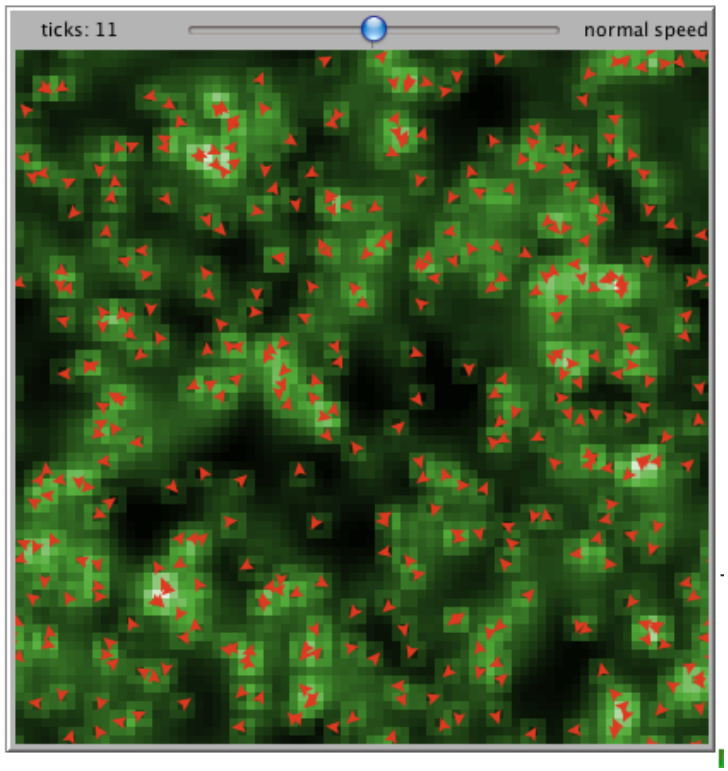




Slime Mold Patterns studied with Star Logo (Resnick, 1997)

Demo: http://ccl.northwestern.edu/netlogo/models/run.cgi?Slime.651.477





See also: http://www.math.umn.edu/~othmer/papers/imart.pdf

Star Logo

- A parallel programming language kids can use (maybe)

```
;; create-agents procedure
;; called by the setup procedure
;; Creates 100 turtles and gives them a random age between 0 and 99, scales
;; their color to represent their age and places them randomly on the canvas.

to create-agents
  create-and-do 100
      [ setage random 100
       setxy random screen-width random screen-height
       scale-color white age 100 0]
end
```

Firefly Clock Synchronization

Buck, John. (1988). Synchronous Rhythmic Flashing of Fireflies. The Quarterly Review of Biology, September 1988, 265 - 286.

Carlson, A.D. & Copeland, J. (1985). Flash Communication in Fireflies. The Quarterly Review of Biology, December 1985, 415 – 433.

Wilensky, U. (1997). NetLogo Fireflies model. http://ccl.northwestern.edu/netlogo/models/Fireflies. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Video

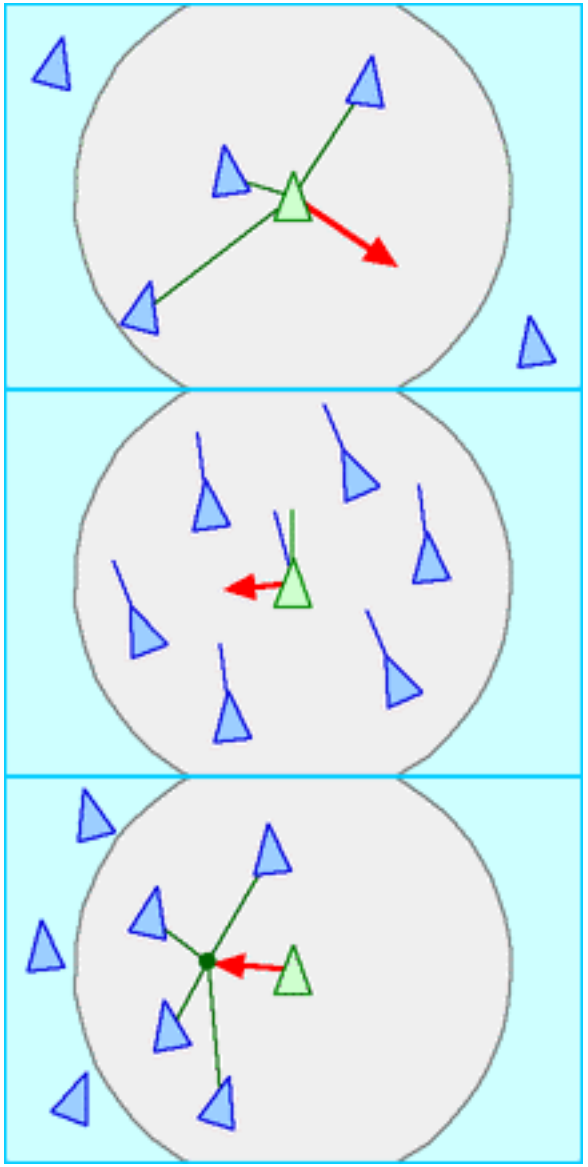
- http://www.youtube.com/watch?v=TTaulpClHco

Swarm Intelligence

Swarm Intelligence

- Local interaction paradigms give rise to global phenomena.
- Boids
  http://www.youtube.com/watch?v=GUkjC-69vaw&feature=fvs
  http://www.red3d.com/cwr/boids/ (Craig Reynolds)
- Fish:
  http://www.youtube.com/watch?v=rLBrK0K-Ny8&feature=related
- Video: Swarm-bots pulling a child:
  http://www.youtube.com/watch?v=CJOubyiITsE&feature=related
- Cuda Flocking Simulation:
  http://www.youtube.com/watch?v=7cjorOe810o&NR=1

Example: Boids Rules



Separation: steer to avoid crowding local flockmate

Alignment: steer towards the average **heading** of local flockmates

Cohesion: steer to move toward the average **position** of local flockmates

http://www.red3d.com/cwr/boids/

Ant-Colony Optimization (ACO), 1992

- Pheromone trail idea:
  - Ants initially wander randomly at first, leaving a pheromone trail.
  - Ants tend to follow stronger trail, which strengthens the trail.
  - Ants also tend to prefer shortest route.
  - Pheromones evaporate after a time, so less-used trails dissipate.

http://en.wikipedia.org/wiki/Ant_colony_optimization

Ant-Colony Optimization

- Parallel Computation:
  - Ant = Solution under construction
  - Move = Extension of partial solution
  - Probability of move from x to y:

$$p_{xy}^{k} = \frac{(\tau_{xy}^{\alpha})(\eta_{xy}^{\beta})}{\sum (\tau_{xy}^{\alpha})(\eta_{xy}^{\beta})}$$

τ_{xy} is the amount of pheromone deposited for transition from state x to y

$0 \leq \alpha$ is a parameter to control the influence of τ_{xy}

η_{xy} is the desirability of state transition xy (a priori knowledge, typically $1 / d_{xy}$, where d is the distance)

$\beta \leq 1$ is a parameter to control the influence of η_{xy}

Example: TSP

- Each ant must visit each city exactly once.
- A distant city has less chance of being chosen (the visibility).
- The more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen.
- Having completed its journey, the ant deposits more pheromones on all edges it traversed, **if** the journey is short.
- After each iteration, trails of pheromones evaporate.

Updating Pheromones

$$\tau_{xy}^{k} = (1-\rho)\tau_{xy}^{k} + \Delta\tau_{xy}^{k}$$

Example for TSP:

$$\Delta\tau_{xy}^{k} = \begin{cases} Q/L_k & \text{if ant } k \text{ uses curve } xy \text{ in its tour} \\ 0 & \text{otherwise} \end{cases}$$

where L_k is the cost of the kth ant's tour (typically length) and Q is a constant.

Stigmergy Principle, 1959

- ACO is a special case
- Stigmergy = indirect coordination between agents or actions, based on traces left by them in the environment.
- Introduced by Pierre-Paul Grassé to describe termite behavior.
- Another example: Wiki

http://en.wikipedia.org/wiki/Stigmergy

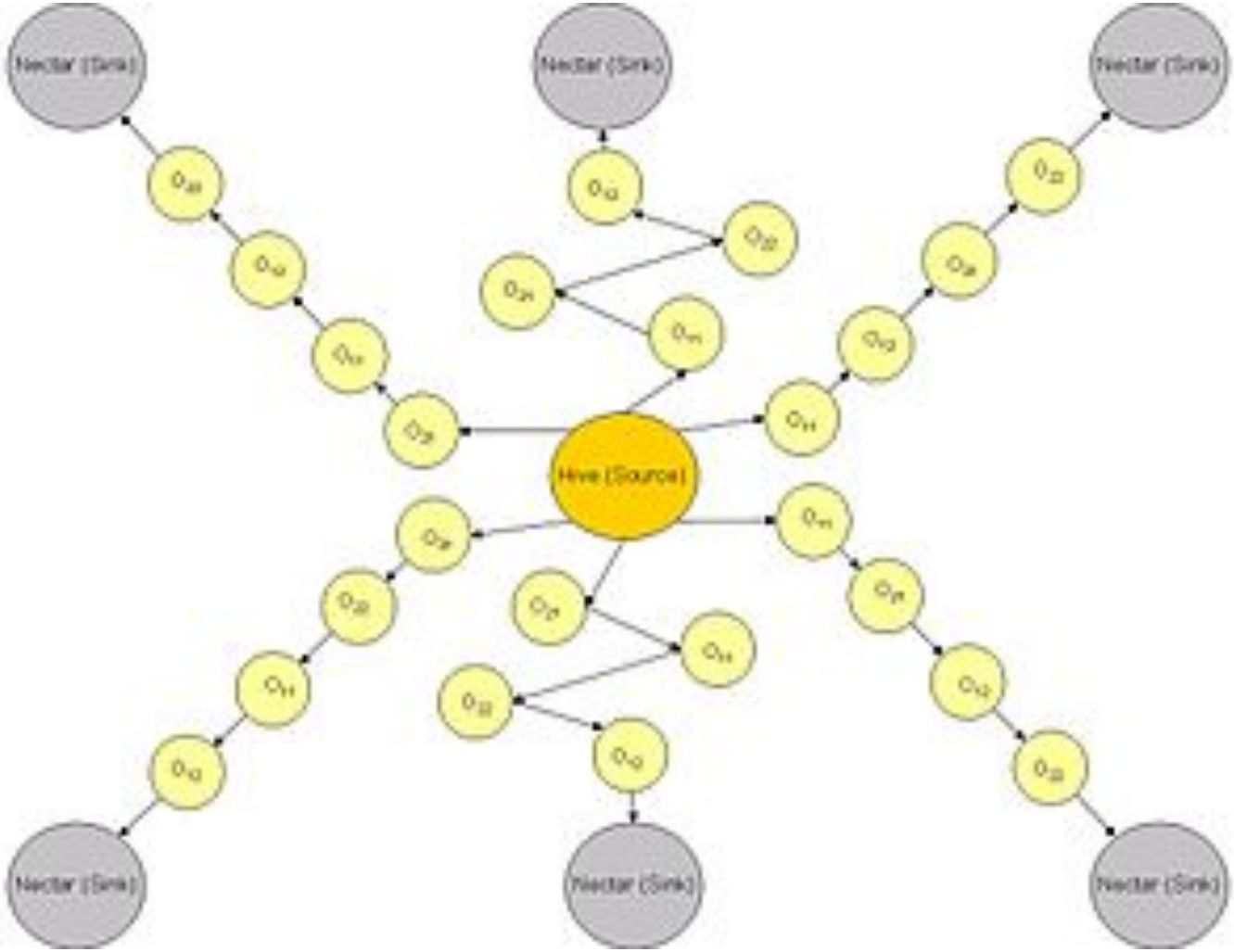
Bees Algorithm, 2005

The pseudo code for the bees algorithm in its simplest form:

1. Initialise population with random solutions.
2. Evaluate fitness of the population.
3. While (stopping criterion not met) //Forming new population.
4. Select sites for neighbourhood search.
5. Recruit bees for selected sites (more bees for best e sites) and evaluate fitnesses.
6. Select the fittest bee from each patch.
7. Assign remaining bees to search randomly and evaluate their fitnesses.
8. End While.

http://en.wikipedia.org/wiki/Bees_algorithm

Bees Algorithm Applied to Scheduling

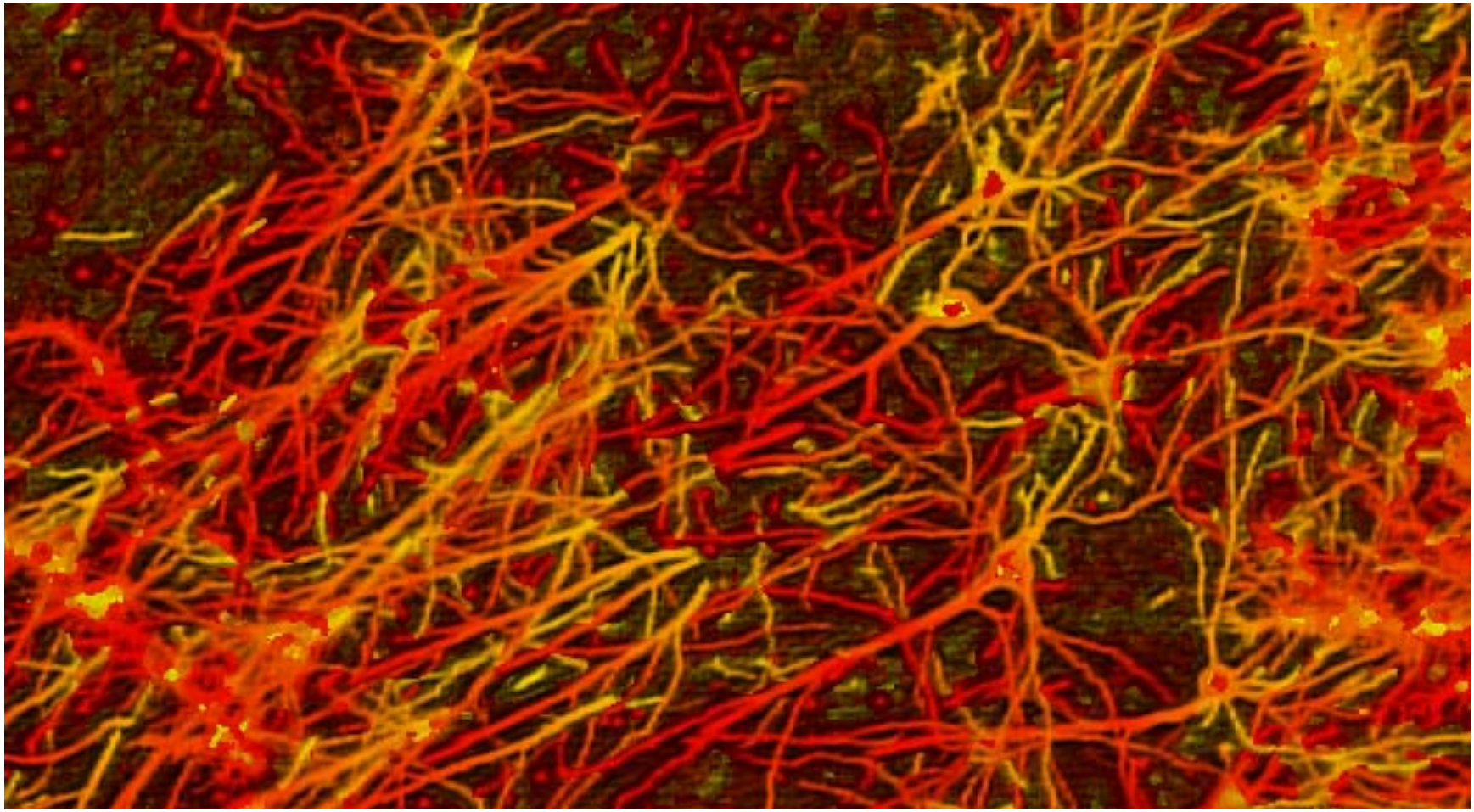


http://en.wikipedia.org/wiki/Bee_Colony_Optimization

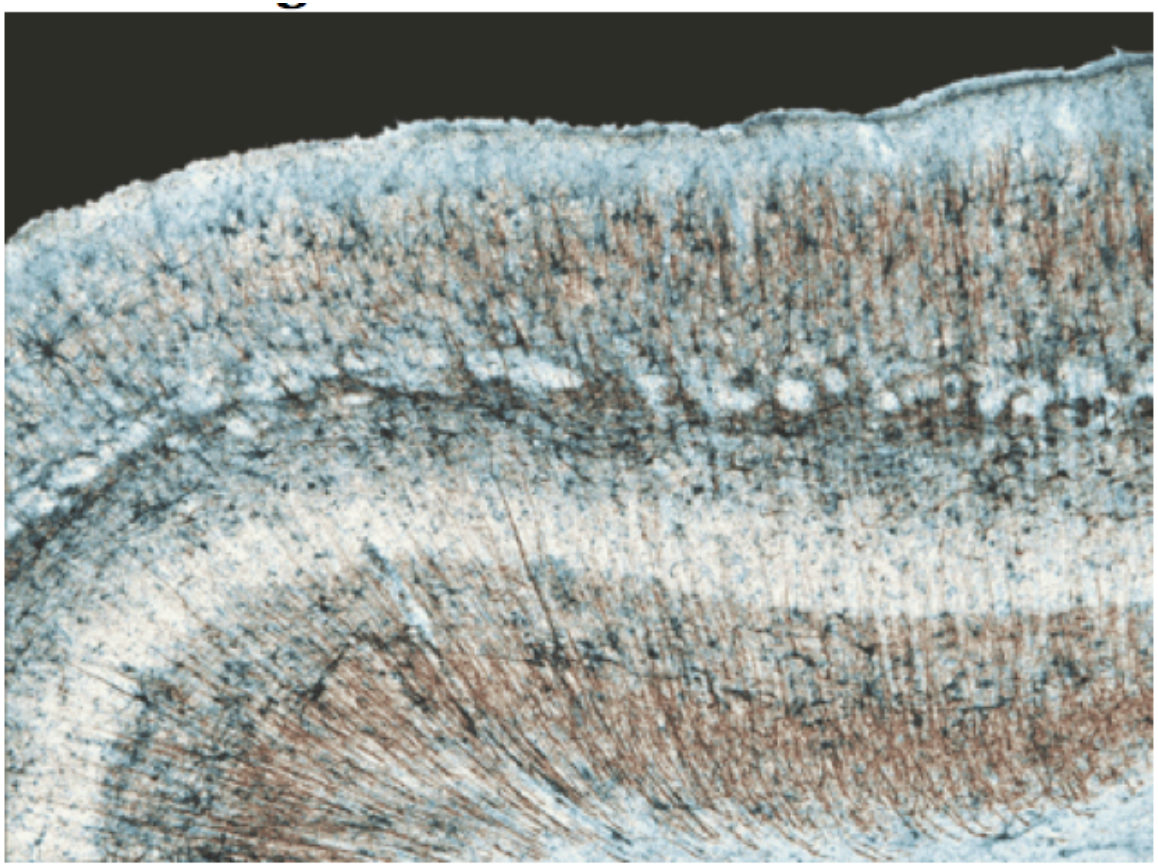
From Nature to Models

- Various computing models have been inspired by nature:
  - Neural networks
  - Cellular automata
  - Developmental systems (“L-systems”)
  - Membrane computing (“P-systems”)
  - DNA computing
  - Immuno-computing

Neural Networks



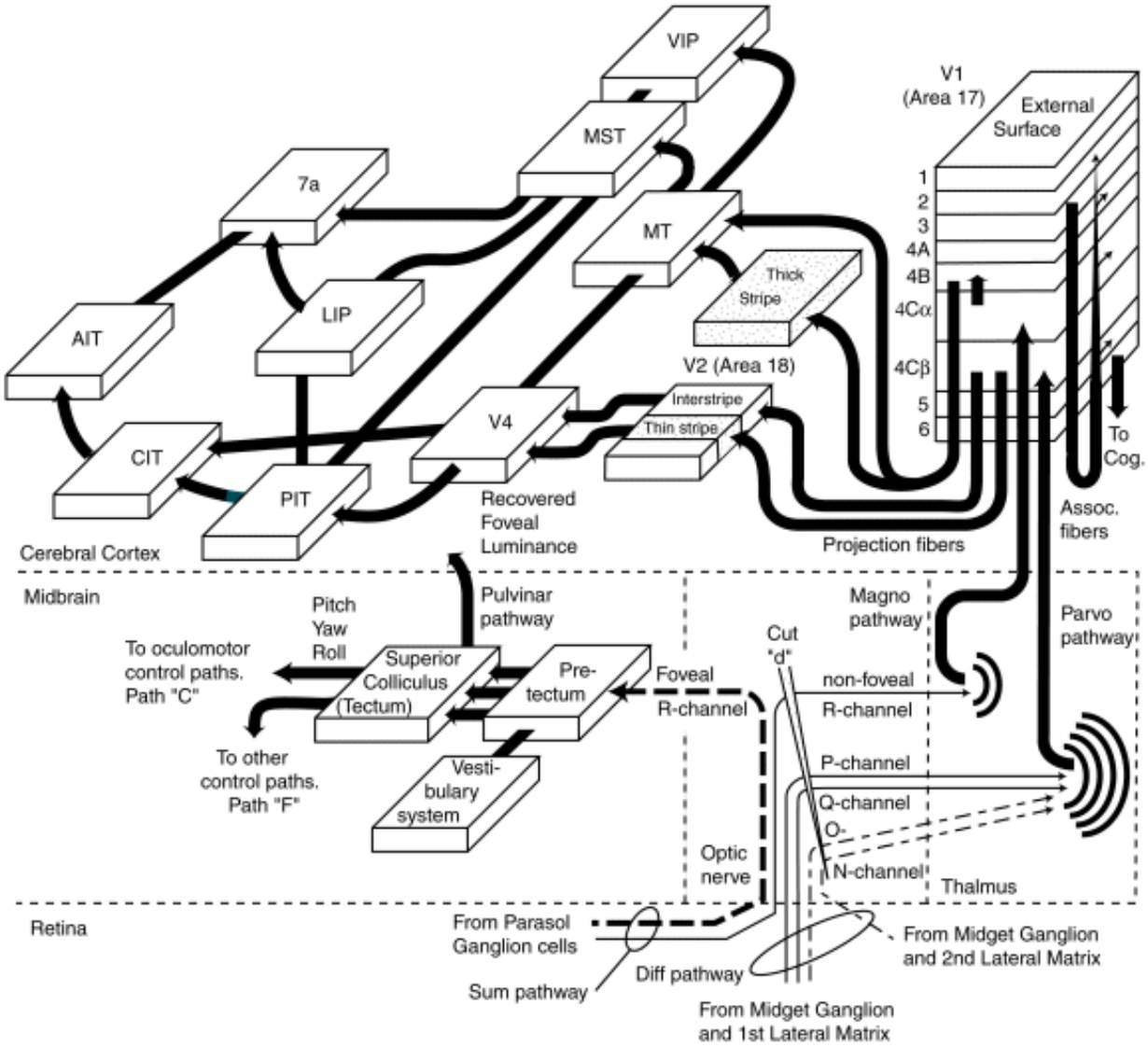
http://www.willamette.edu/~gorr/classes/cs449/brain.html



Cerebral Cortex J.
July 2002 12 (7)

Cover Picture: Human-specific compartmental organization of human primary visual cortex (area V1; Brodmann's area 17). Photomicrograph showing the upper bank of the calcarine fissure; the image was rotated to show the cortical layers in conventional orientation. This coronal section was double-immunostained for nonphosphorylated neurofilaments with the SMI-32 antibody (reddish-brown reaction product) and for perineuronal nets with the Cat-301 antibody (blue reaction product). Staining with these antibodies reveals a striking pattern of alternating light- and dark-staining compartments in layer 4A. The dark territories are packed with cells bodies and neuropil that stain with SMI-32 or Cat-301. The light-staining compartments contain numerous small somas that are immunoreactive for calbindin, as demonstrated by other double- immunostaining experiments in this series. Comparative studies of humans, apes and monkeys indicate that this pattern of compartmentation is unique to humans. See Preuss and Coleman, pp. 671–691.

Neural Schematic of Human Visual Cortex



Caution

- Real neurons respond to trains of analog spikes.
- They are not truly digital.
- Most CS and AI studies ANN's (Artificial Neural Networks)

Artificial Neural Network Varieties

Supervised

- Perceptron
- Multi-Layer Perceptron
- Radial Basis Function Network
- CMAC (Cerebral Model-Articulated Control)
- Real-Time Recurrent Network
- Counterpropagation network
- Hopfield Network
- Bi-Directional Associative Memory
- Boltzmann Machine
- Deep-Belief Network
- Neocognitron
- Support Vector Machine

. . .

Unsupervised

- Adaptive Resonance Theory Network
- Self-Organizing Map
- LVQ (Learning Vector Quantizer)
- Growing Neural Gas
- PCA (Principal Components Analysis) Network
- ICA (Independent Components Analysis) Network

Brain Simulator

This brain simulator was developed by IBM's Almaden research center. It consists of 1.6 billion virtual neurons connected by 9 trillion synapses, which is almost a cell-by-cell simulation of the human visual cortex.

With all the massive components, over 150,000 gigabytes of memory and power consumption of over million watts, the simulator is only as good as a cat's brain. But the achievement is much better than the smaller rat brain simulator that they made two years ago.

To simulate the entire human cortex, they need computing power that is 1,000 times faster than what they currently have managed, but this could only be achieved in about a decade from now.

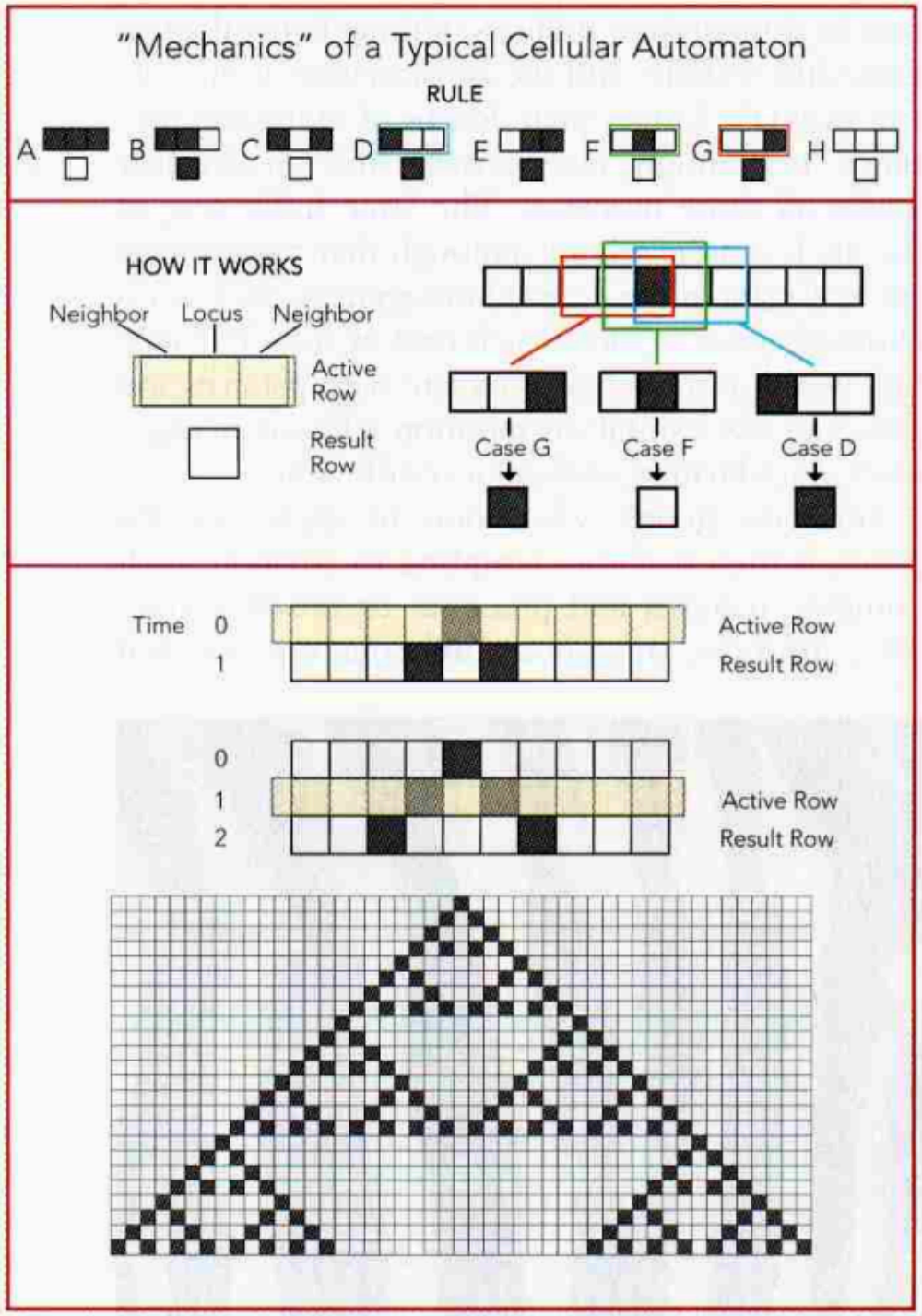
http://www.techchee.com/2009/11/20/ibms-biggest-brain-simulator-works-as-good-as-a-cats-brain-only/

Computer vs. Neurons (Brain)

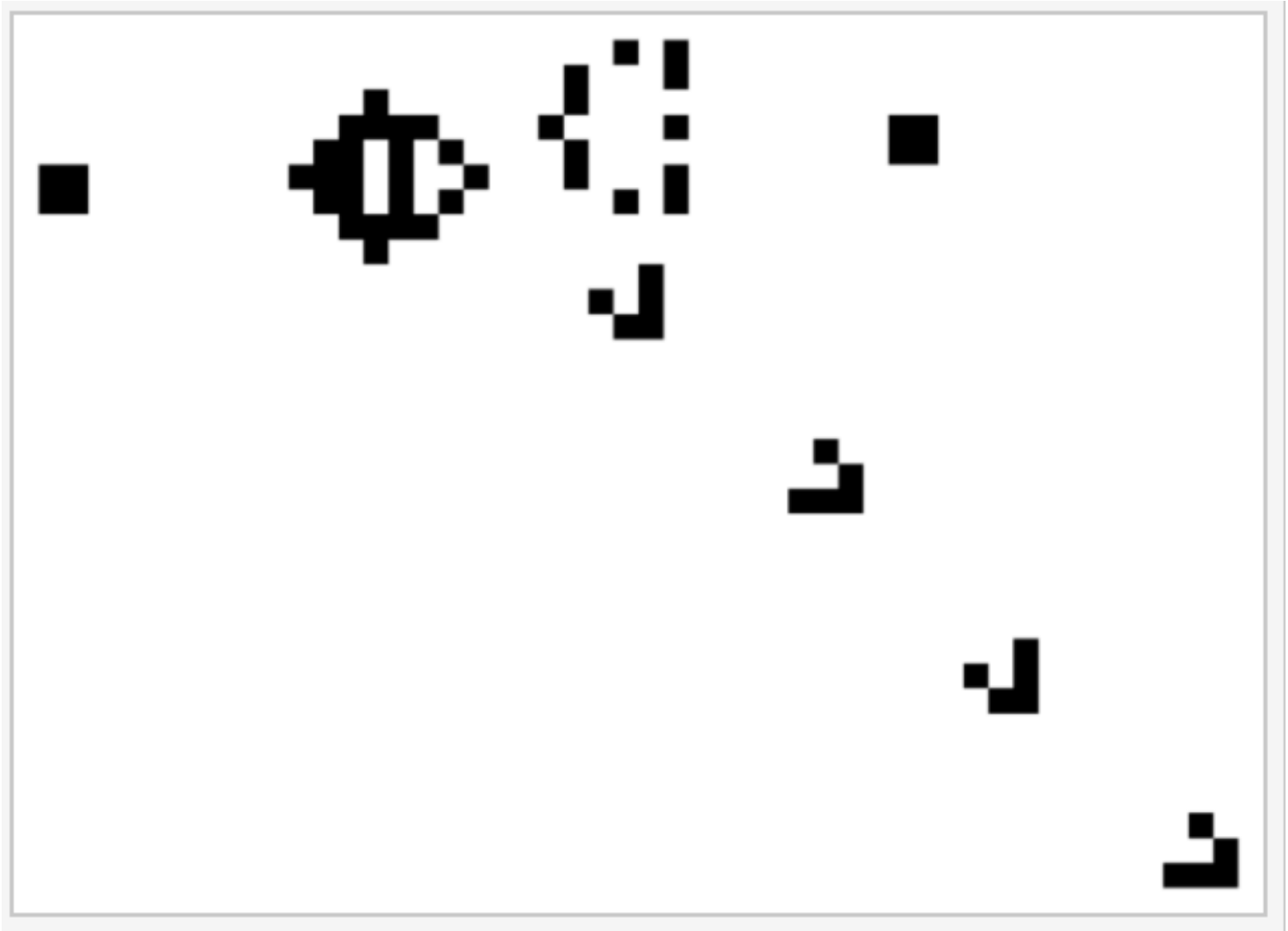
Computer	Human Brain
Electronic transitions	Chemical transitions
a billion or so transistors	100 billion neurons, each with about 1000 connections to others
0.01 μm transistor size	about 10 μm neuron size
usually synchronous	somewhat asynchronous
100 GHz switching rate	1 kHz switching rate
10^{-16} Joule switching energy	10^{-18} Joule switching energy
separate memory and processor	undifferentiated memory vs processor
direct-access memory	associative memory

Cellular Automata

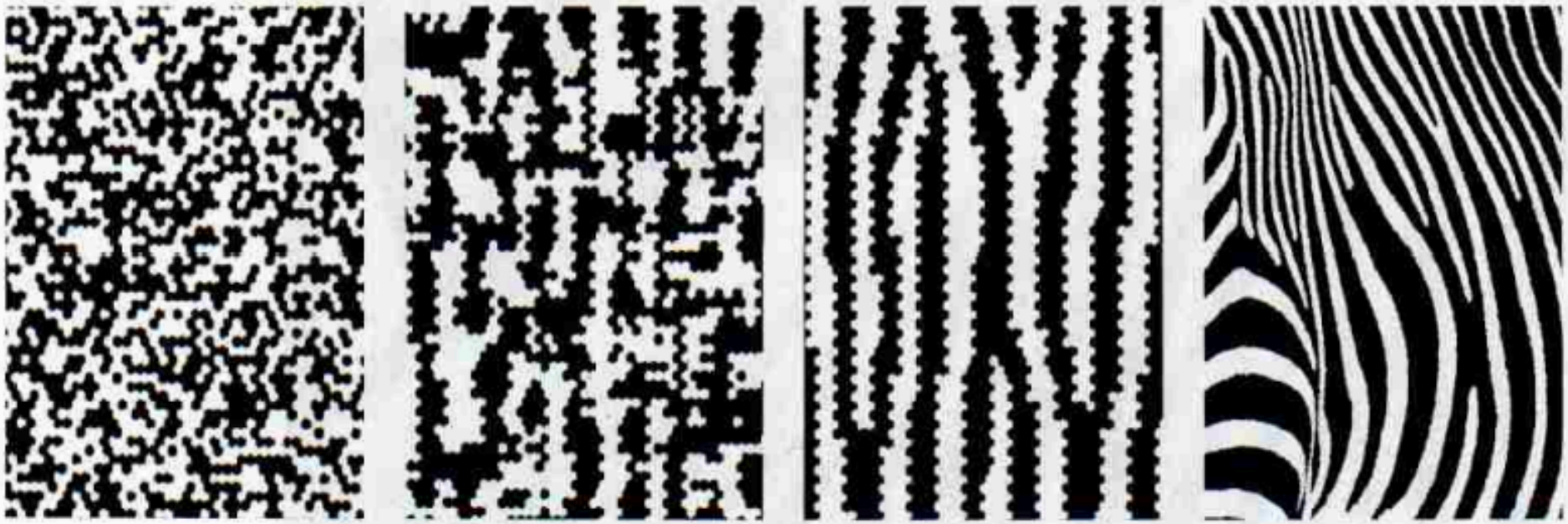
- First studied by von Neumann and Ulam
- An alternative to describing systems with differential equations.
- The same concept is known as “lattice gases” in relation to statistical mechanics.


"Mechanics" of a Typical Cellular Automaton
RULE
A
B
C
D
E
F
G
H
HOW IT WORKS
Neighbor
Locus
Neighbor
Active Row
Result Row
Case G
Case F
Case D
Time
0
1
Active Row
Result Row
0
1
2
Active Row
Result Row

Best Known 2D CA: Conway's "Game of Life"



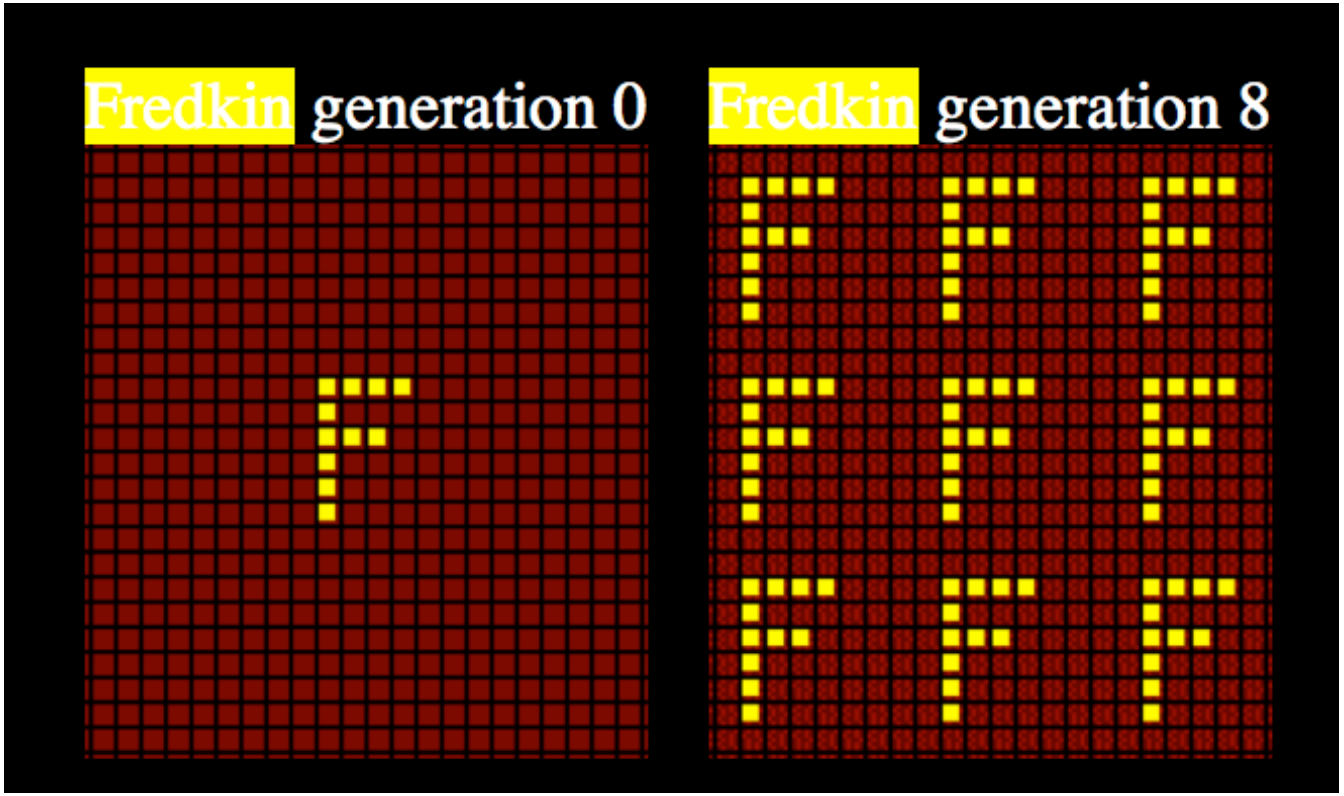
Gosper's Glider Gun Example



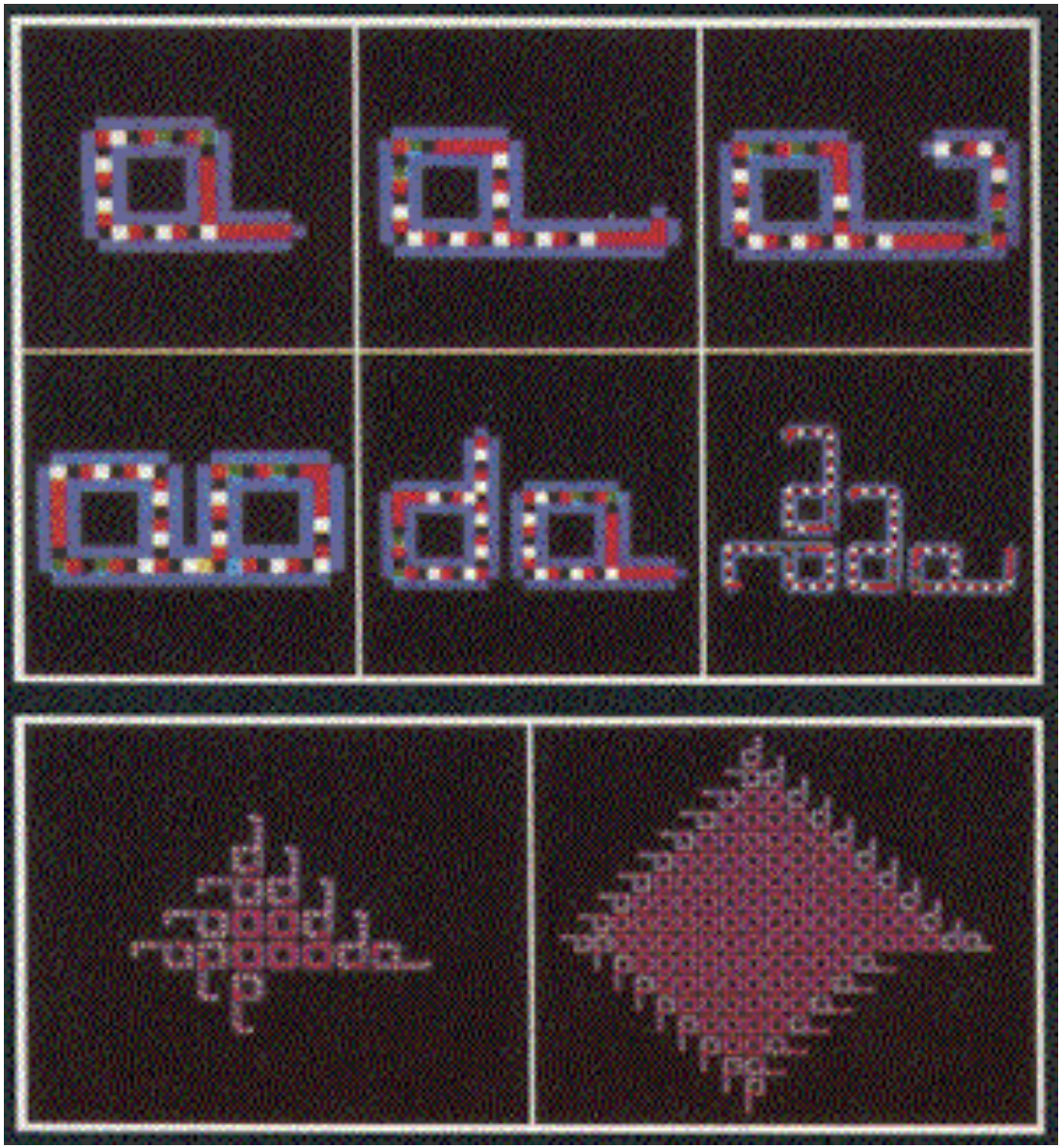
Zebra skin markings are simulated here by a two-dimensional cellular automaton, from an initial, random distribution of black and white cells (far left). Even with the first tick of the clock a pattern begins to appear (second from left). At the tenth tick of the clock, a stable pattern emerges (third from left), which is quite similar to an actual zebra coat (far right).

Cellular Automata

- Simple self-reproduction: Fredkin's Automaton



Langton's Self-Reproducing Loops



Universal Computation in Cellular Automata

- von Neumann showed how to embed a Universal Turing Machine + Self Reproduction
- Game-of-Life also can simulate any Turing machine.

L-Systems

- Lindenmayer-Systems
- Similar to grammar rewriting rules, except:
  - All non-terminal rewritten simultaneously

L-System Example: Algae Growth

Lindenmayer's original L-system for modelling the growth of algae.

variables : A B

constants : none

start : A

rules : (A → AB), (B → A)

which produces:

$n = 0$: A

$n = 1$: AB

$n = 2$: ABA

$n = 3$: ABAAB

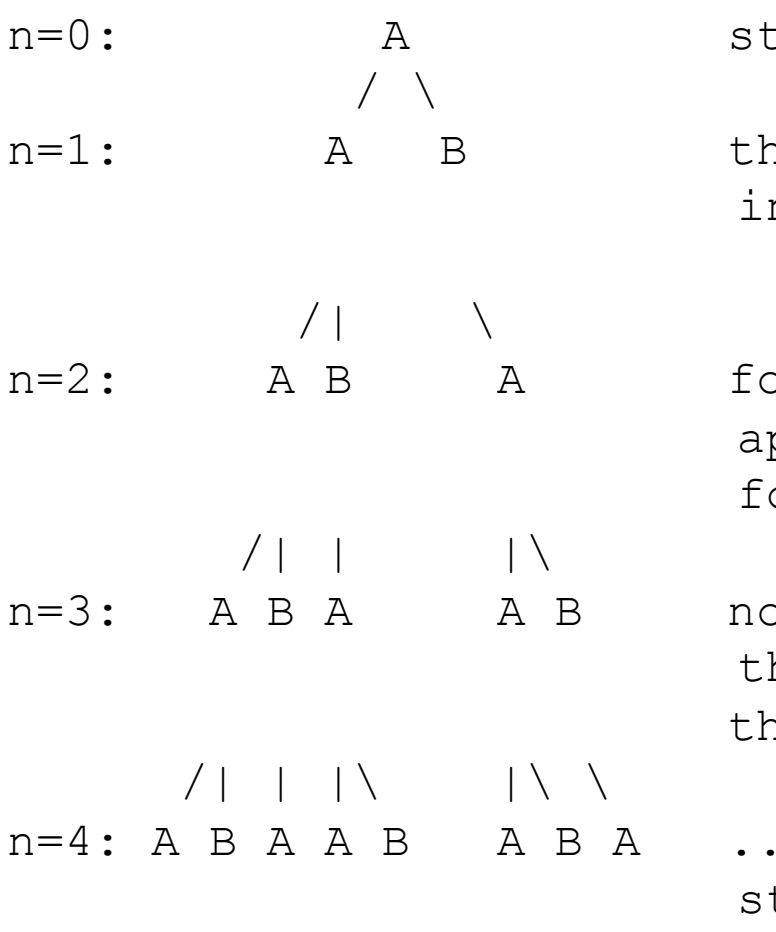
$n = 4$: ABAABABA

$n = 5$: ABAABABAABAAB

$n = 6$: ABAABABAABAABABAABABA

$n = 7$: ABAABABAABAABABAABABAABAABABAABAAB

L-System Example: Algae Growth



start (axiom/initiator)

the initial single A spawned into AB by rule (A → AB), rule (B → A) couldn't be applied

former string AB with all rules applied, A spawned into AB again, former B turned into A

note all A's producing a copy of themselves in the first place, then a B, which turns ...

... into an A one generation later, starting to spawn/repeat/recurse then

L-System: Fractal Plant Drawing

variables : X F
constants : + −
start : X
rules : (X → F-[[X]+X]+F[+FX]-X), (F → FF)
angle : 25°
Here, F means "draw forward", - means "turn left 25°", and + means "turn right 25°". X does not correspond to any drawing action and is used to control the evolution of the curve.
[corresponds to saving the current values for position and angle, which are restored when the corresponding] is executed.

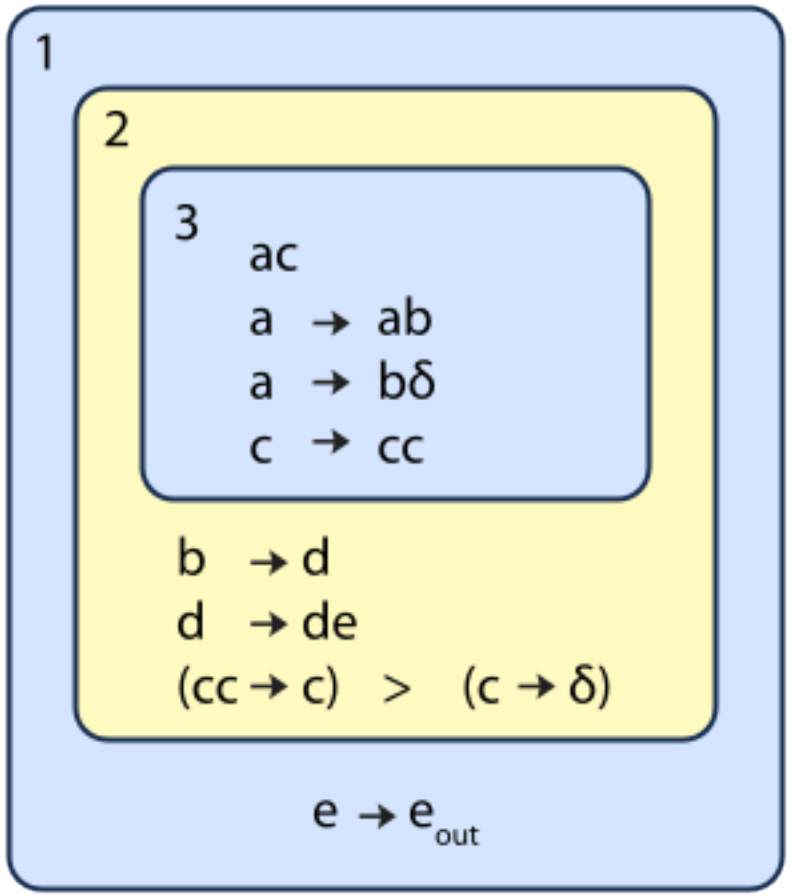
http://algorithmicbotany.org/papers/#abop "Algorithmic Botany"

P-Systems, 1998

- “P” is from Gheorghe Păun
- Membrane Computing
- Model for chemicals moving through cellular membranes

P-System Example

The outermost membrane, 1, is the container membrane for this P system and contains a single out rule. Membrane 2 contains four here rules, with two in a priority relationship: cc → c will always be applied in preference to c → cδ. The delta symbol represents the special "dissolve" symbol. The innermost membrane, 3, contains a set of symbols ("ac") and three rules, of type here. In this initial state no rules outside of membrane 3 are applicable: there are no symbols outside of that membrane. However, during evolution of the system, as objects are passed between membranes, the rules in other membranes will become active.



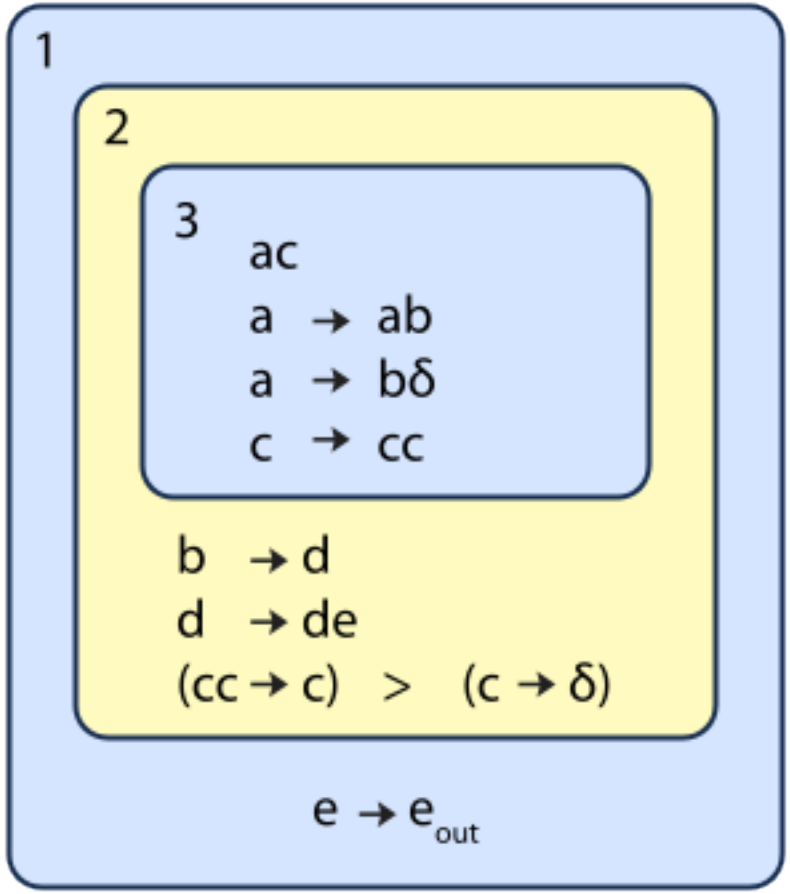
P-System Example

Step 1
From the initial configuration only membrane 3 has any object content: "ac"

"c" is assigned to c → cc
"a" is assigned to a → ab

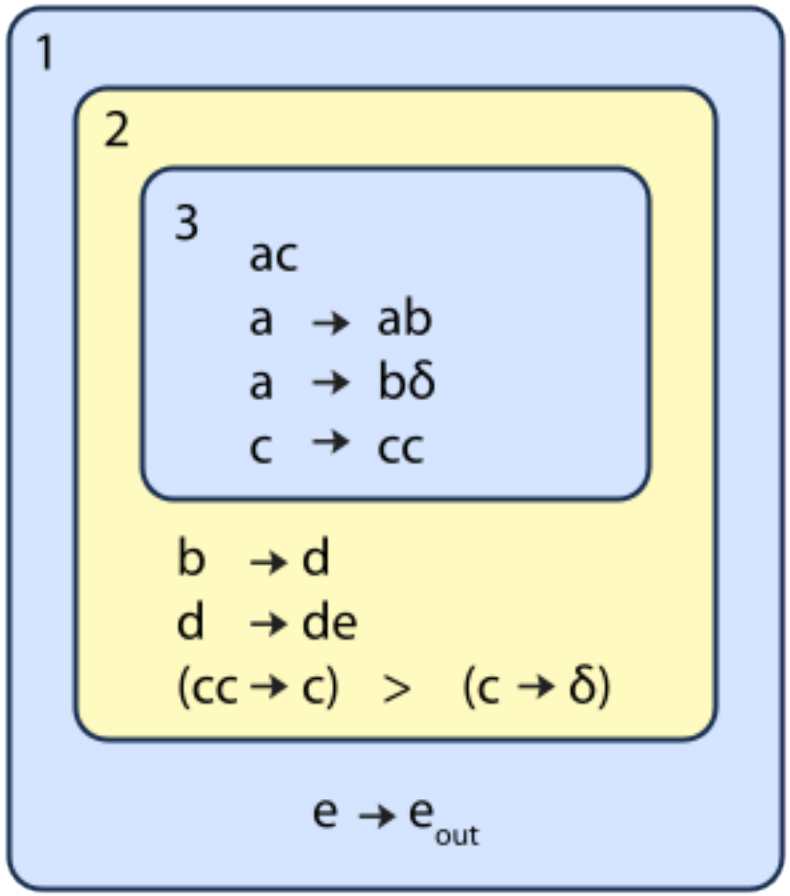
Step 2
Membrane 3 now contains: "abcc"

"a" is assigned to a → bδ
"c" is assigned to c → cc
"c" is assigned to c → cc



P-System Example

Membrane 3 now dissolves, as the dissolve symbol (δ) has been encountered and all object content from this membrane passes into membrane 2.



Step 3
Membrane 2 now contains: “bbcccc”

"b" is assigned to b → d
"b" is assigned to b → d
"cc" is assigned to cc → c
"cc" is assigned to cc → c

Step 4
Membrane 2 now contains: "ddcc"

"d" is assigned to d → de
"d" is assigned to d → de
"cc" is assigned to cc → c

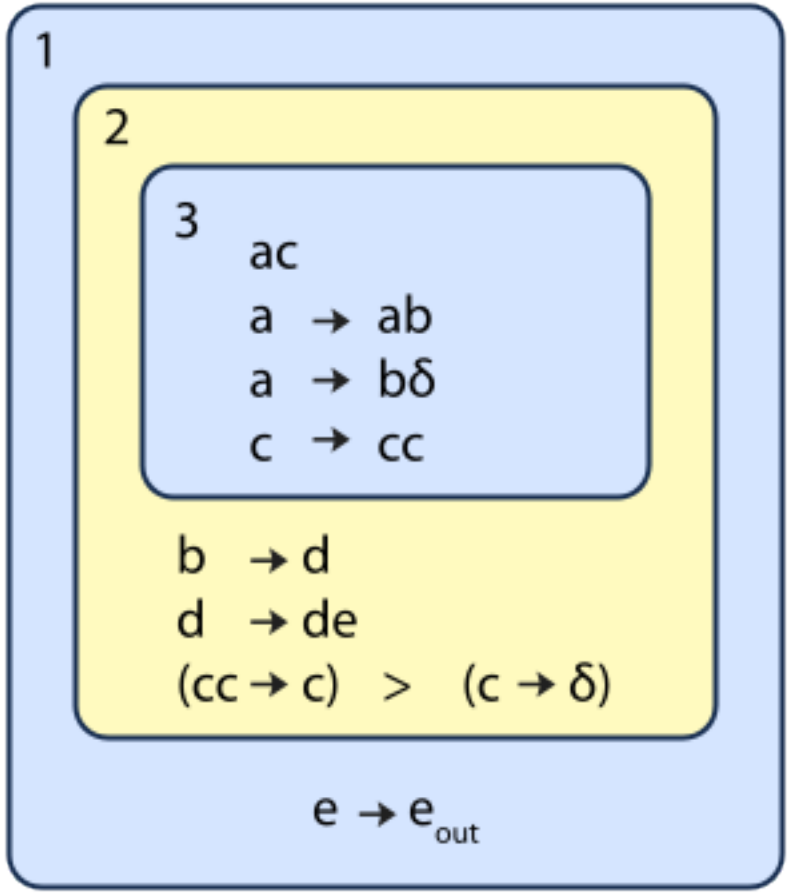
P-System Example

Step 5
Membrane 2 now contains: “dedec”

"d" is assigned to d → de
"d" is assigned to d → de
"c" is assigned to c → δ
Notice that the priority over c → δ has been lifted now the required inputs for cc→ c no longer exist. Membrane 2 now dissolves, and all object content passes to membrane 1.

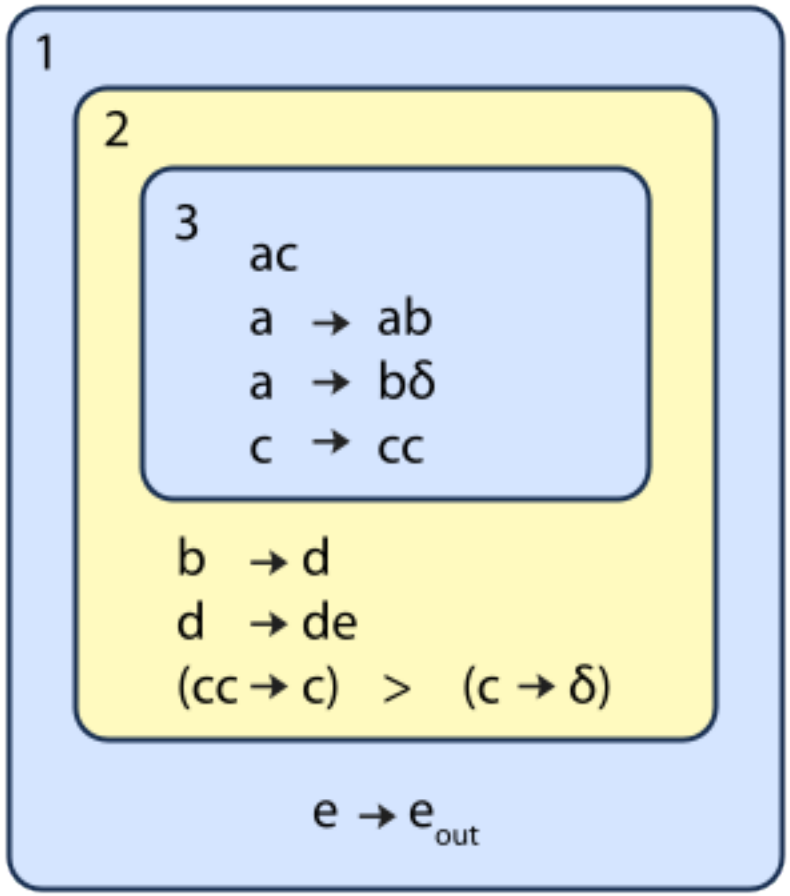
Step 6
Membrane 1 now contains: “deedee”

"e” is assigned to e → eout
"e” is assigned to e → eout
"e” is assigned to e → eout
"e” is assigned to e → eout



P-System Example

Membrane 1 now contains: "dd" and, due to the out rule e → eout, the environment contains: "eeee." At this point the computation halts as no further assignments of objects to rules is possible. The result of the computation is four "e" symbols.



DNA Computing (Adleman, 1994)

- Highly-Parallel Solution to the Directed Hamiltonian Path problem (7-vertex instance)

Science, New Series, Vol. 266, No. 5187 (Nov. 11, 1994), pp. 1021-1024

Molecular Computation of Solutions to Combinatorial Problems

Leonard M. Adleman

The tools of molecular biology were used to solve an instance of the directed Hamiltonian path problem. A small graph was encoded in molecules of DNA, and the ''operations'' of the computation were performed with standard protocols and enzymes. This experiment demonstrates the feasibility of carrying out computations at the molecular level.

http://www.jstor.org/stable/2885489

Adleman Example

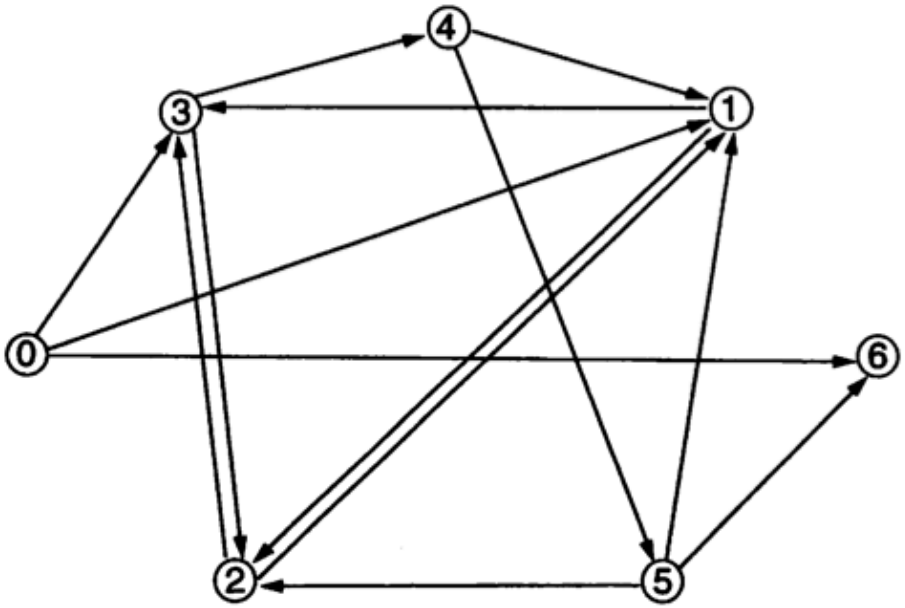


Fig. 1. Directed graph. When $v_{in} = 0$ and $v_{out} = 6$, a unique Hamiltonian path exists: 0→1, 1→2, 2→3, 3→4, 4→5, 5→6.

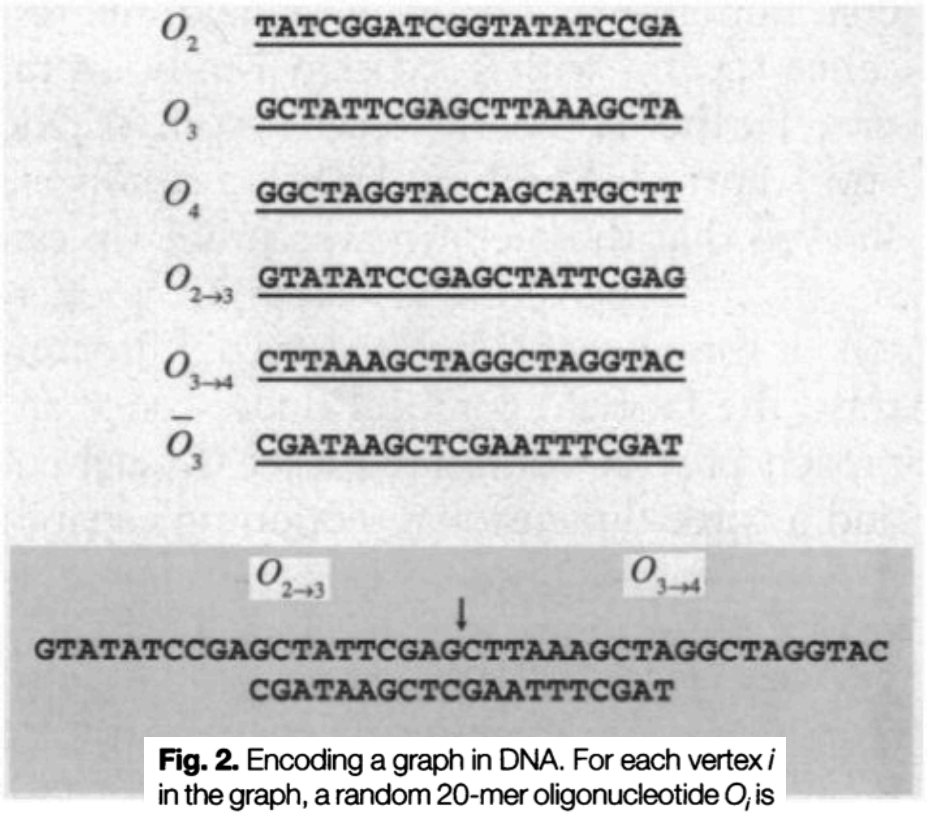


Fig. 2. Encoding a graph in DNA. For each vertex i in the graph, a random 20-mer oligonucleotide O_i is generated (shown are O_2, O_3, and O_4, for vertices 2, 3, and 4, respectively). For edge $i \to j$ in the graph, an oligonucleotide $O_{i\to j}$ is derived from the 3′ 10-mer of O_i and from the 5′ 10-mer of O_j (shown are $O_{2\to3}$ for edge 2→3 and $O_{3\to4}$ for edge 3→4). For each vertex i in the graph, $\overline{O}_i$ is the Watson-Crick complement of O_i (shown is $\overline{O}_3$, the complement of O_3). $\overline{O}_3$ serves as a splint to bind $O_{2\to3}$ and $O_{3\to4}$ in preparation for ligation. All oligonucleotides are written 5′ to 3′, except $\overline{O}_3$.

Speedup?

What is the power of this method of computation? It is premature to give definitive answers; however, some remarks seem in order. A typical desktop computer can execute approximately 10^6 operations per second. The fastest supercomputers currently available can execute approximately 10^{12} operations per second. If the ligation (concatenation) of two DNA molecules is considered as a single operation and if it is assumed that about half of the approximately 4×10^{14} edge oligonucleotides in Step 1 were ligated, then during Step 1 approximately 10^{14} operations were executed. Clearly, this step could be scaled-up considerably, and 10^{20} or more operations seems entirely plausible (for example, by using micromole rather than picomole quantities). At this scale, the number of operations per second during the ligation step would exceed that of current supercomputers by more than a thousandfold. Furthermore, hydrolysis of a single molecule of adenosine triphosphate to adenosine monophosphate plus pyrophosphate provides the Gibbs free energy ($\Delta G = -8$ kcal mol^{-1}) for one ligation operation (*10*, *11*); hence in principle 1 J is sufficient for approximately 2×10^{19} such operations.

This is remarkable energy efficiency, considering that the second law of thermodynamics dictates a theoretical maximum of 34×10^{19} (irreversible) operations per joule (at 300 K) (*12*, *13*). Existing supercomputers are far less energy-efficient, executing at most 10^9 operations per joule. The energy con-

Immunocomputing

Evolutionary Computing

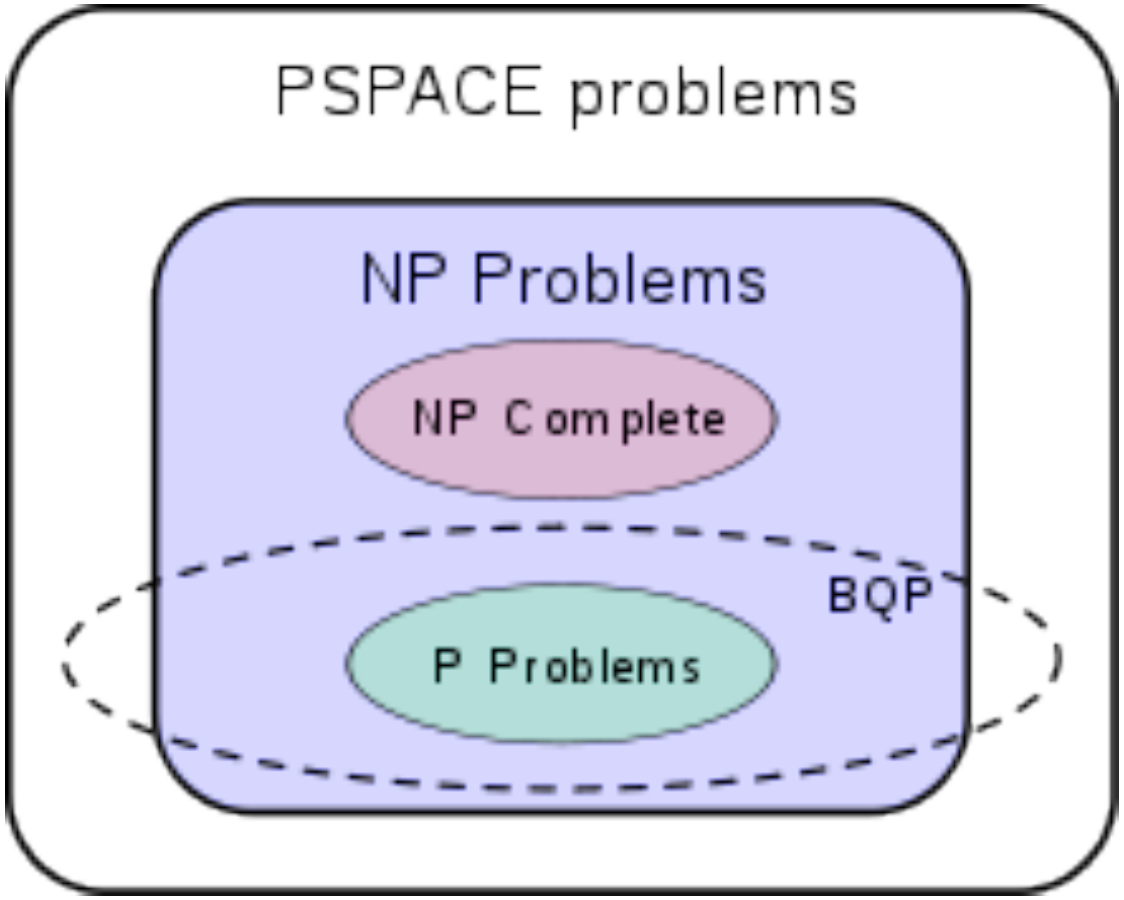
- Learn by populations, rather than by individuals
- (Lamarckian evolution can also learn by individuals)
- Parallelism in evaluating fitness of individuals

Quantum Computing

- Forward-Looking as far as practicality
- Probabilistic
- Qubits (complex numbers) rather than bits
- Mathematics:
  http://www.youtube.com/watch?v=BUfdJR61Fvw&NR=1
- Physical Implementation Aspects:
  http://www.youtube.com/watch?v=BUfdJR61Fvw&NR=1

BQP: Bounded-Error Quantum Polynomial

Conjectured placement



http://en.wikipedia.org/wiki/Quantum_computer