Artificial Life & Complex Systems

Lecture 15

ABM – Applications

June 9, 2007

Max Lungarella

Contents

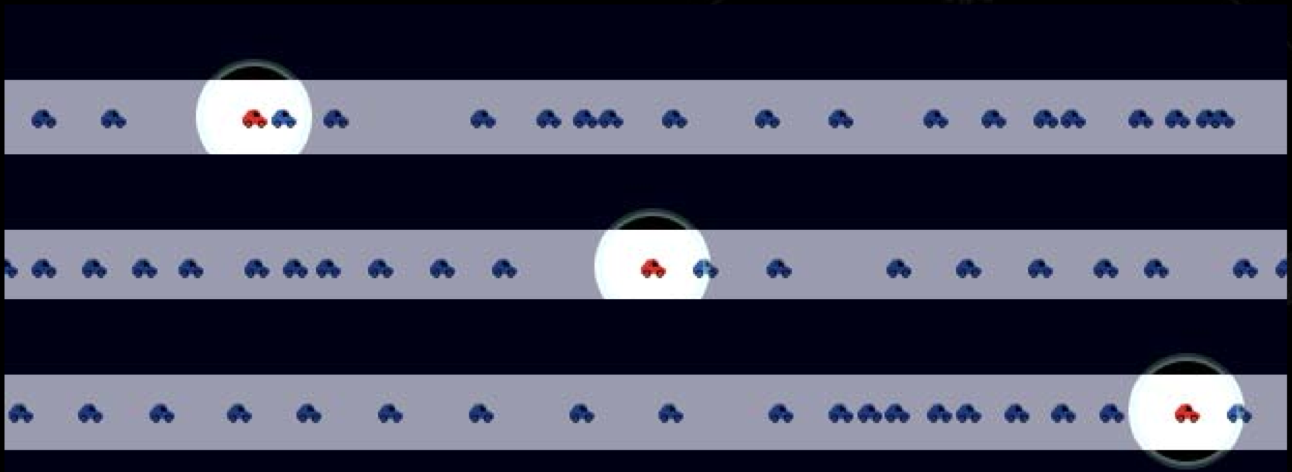
- Traffic jams
- Swarm-based AI
  - Ant colony optimization
  - Ant-based control
- (Semiotic dynamics)
- (Talking heads experiment)
- (Collaborative tagging)
- (Social segmentation)

Traffic Jams

Phenomenon

- Stream of cars breaks down into dense clumps and empty stretches
- Spontaneous symmetry-breaking of initially uniform density and speed
- No need for a central cause (such as slow vehicle, stop light, or accident)

Traffic Jams



NetLogo traffic simulation

Car's behavior

- Each car:
  - Slows down if there is another car close ahead
  - Speeds up if there is no car close ahead

Observations

- Traffic nodes move in the direction opposite to cars (traffic jams move backwards)
- Emergence of group behavior qualitatively different from individual behavior

Traffic Jams

- Add “radar trap”: what happens?
- Accident on the side of the road → “rubbernecking” effect

Traffic Jams

Concepts learned from this example:

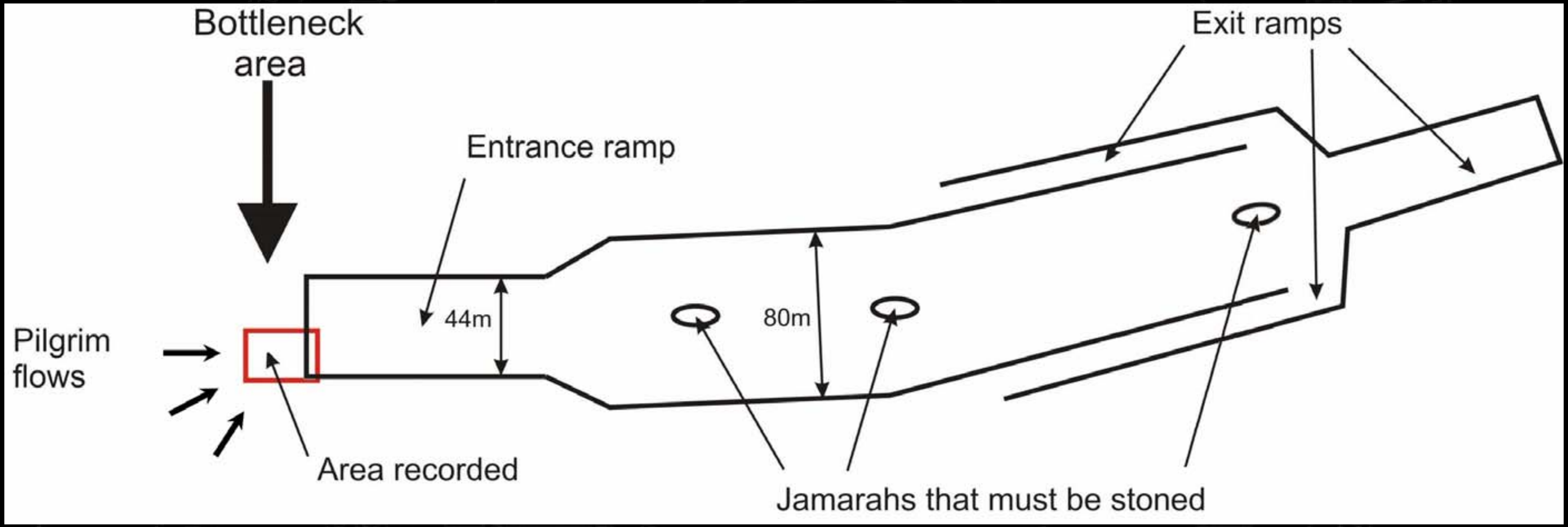
- Simple individual reactions
- Emergence of moving superstructures
- No accident, no light, no police radar (decentralized control)
- Amplification of small fluctuations (positive feedback)
- Local interactions (car vs. car)

Crowds

Haji – The religious rituals during the annual muslim pilgrimage

http://www.trafficforum.org/crowdturbulence

Crowds



http://www.trafficforum.org/crowdturbulence

Crowds

Movie:

pressure, pressure_video

Crowd Control

The Dynamics of Crowd Disasters: An Empirical Study

Dirk Helbing, Anders Johansson
Dresden University of Technology, Andreas-Schubert-Str. 23, 01062 Dresden, Germany

Habib Zein Al-Abideen
Central Directorate of Development of Projects, Minstry of Municipal and Rural Affairs, Riyadh, Kingdom of Saudi Arabia

Many observations in the dynamics of pedestrian crowds, including various self-organization phenomena, have been successfully described by simple many-particle models. For ethical reasons, however, there is a serious lack of experimental data regarding crowd panic. Therefore, we have analyzed video recordings of the crowd disaster in Mina/Makkah during the Hajj in 1426H on January 12, 2006. They reveal two subsequent, sudden transitions from laminar to stop-and-go and "turbulent" flows, which question many previous simulation models. While the transition from laminar to stop-and-go flows supports a recent model of bottleneck flows [D. Helbing et al., Phys. Rev. Lett. 97, 168001 (2006)], the subsequent transition to turbulent flow is not yet well understood. It is responsible for sudden eruptions of pressure release comparable to earthquakes, which cause sudden displacements and the falling and trampling of people. The insights of this study into the reasons for critical crowd conditions are important for the organization of safer mass events. In particularly, they allow one to understand where and when crowd accidents tend to occur. They have also led to organizational changes, which have ensured a safe Hajj in 1427H.

Crowd Control

http://www.trafficforum.org/crowdturbulence

Check Out ...

http://www.crowddynamics.com/Menu.htm

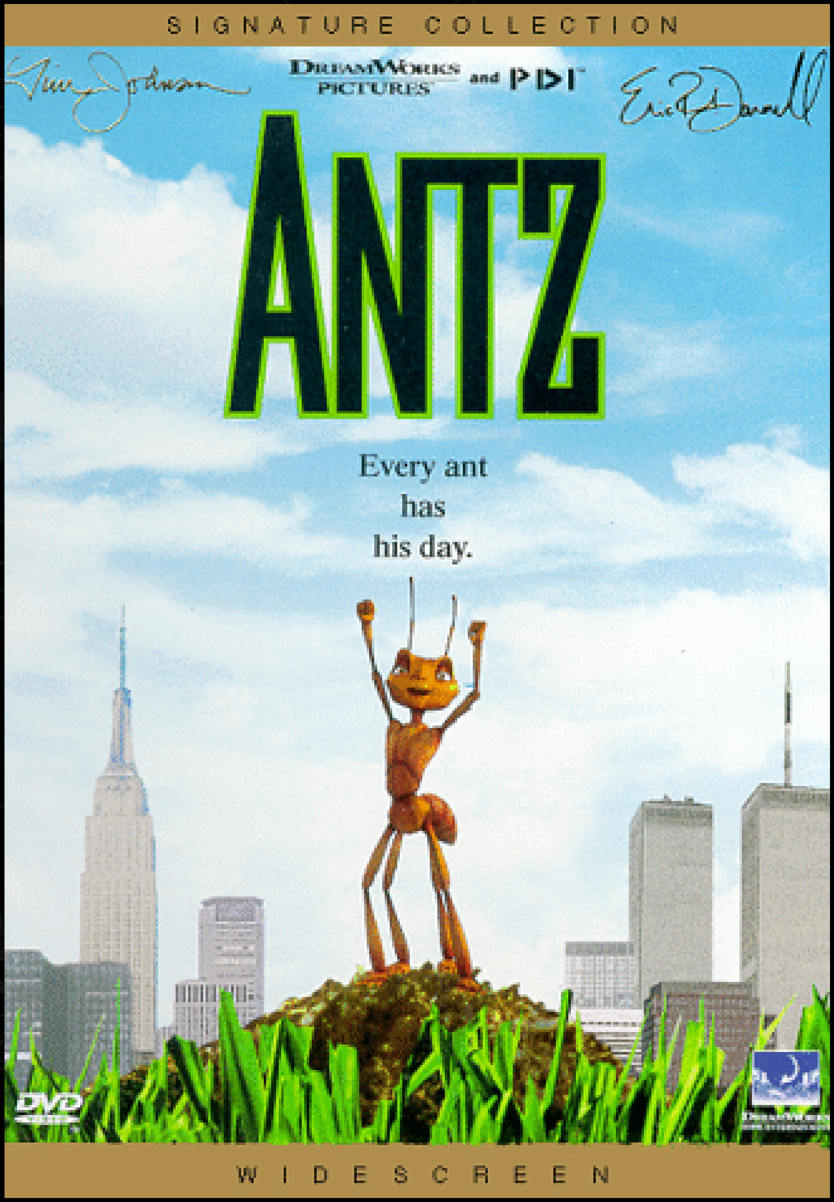
Ant Algorithms (Swarm-Based AI)

Ant Colony Optimization

- Extremely successful at a variety of very difficult combinatorial optimization problems
- Some of the best solutions known to some problems

Ant-based control of communication networks

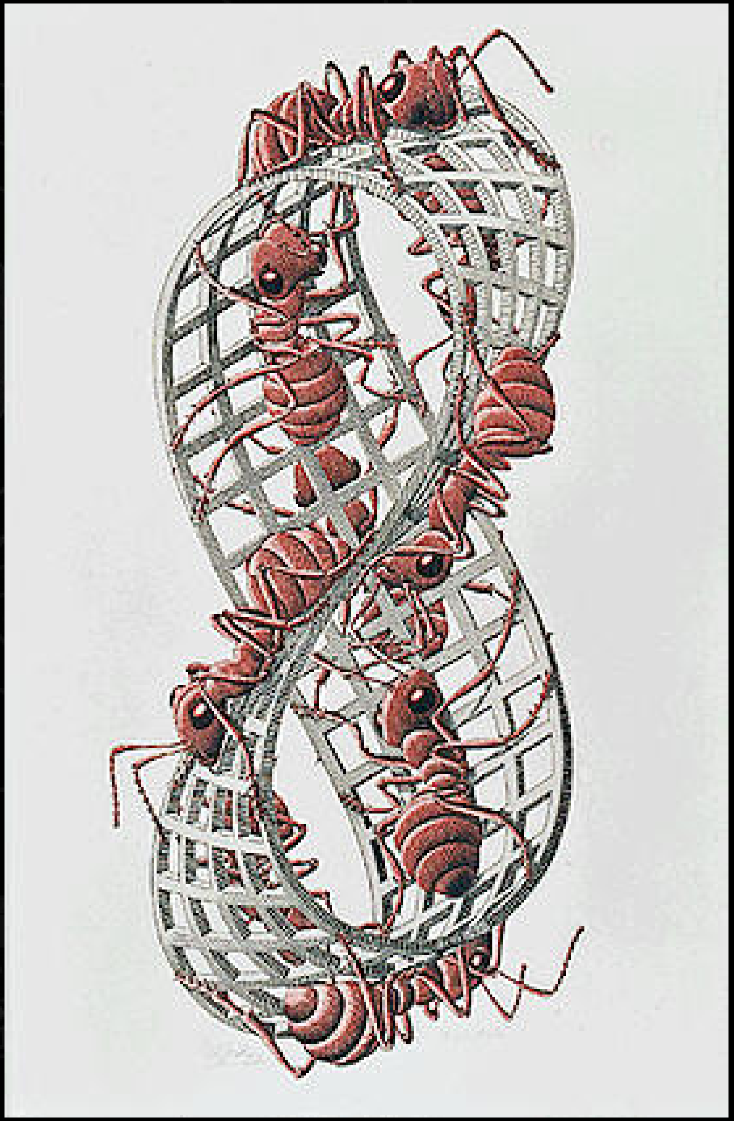
- Exceptionally flexible solutions
- Best solutions for some problems



Ant Colony Optimization (ACO)

Characteristics:

- multi-agent system consisting of simple agents which solve difficult combinatorial optimization problems through collective behavior and stigmergy
- meta-heuristic (higher-level problem solving strategy) that can be applied to all sorts of problems, which can be reformulated to a shortest path problem
- local search strategy that provides a balance between exploration (random) and exploitation (greedy)



Very Brief History – Timeline

Eugène Marais
(1937)
The Soul of the White Ant

Resemblance between the processes at work within termite society and the workings of the human body. He regarded red and white soldiers as analogous to blood cells

Pierre-Paul Grassé
(1959)
Stigmergy i.e. 'Incite to work'
Collective intelligence of social insects

How is it that a group of tiny, short-sighted, simple individuals are able to create the grand termite mounds, sometimes as high as 6 metres ?

Termites' actions are not coordinated from start to finish by any kind of purposive plan, but rather rely on how the termite's world appears at any given moment to invoke the correspondent simple behavior. No need for global knowledge or any more memory than necessary to complete the sub-task in hand.

Jean-Louis Deneubourg
(1989)
Stigmergy through pheromonal communication

Ants engaging in certain activities (food collection) leave a chemical trail which is then followed by their colleagues. A dynamic self-organising (SO) system in which order emerges entirely as a result of the properties of individual elements in the system, and not from external pressures.

Very Brief History - Timeline

- Goss et al. 1989, Deneuborg et al. 1990, experiments with Argentine ants
- Dorigo et al. 1991, applications to shortest path problems
- Now: established method for various optimization problems

Biological Inspiration

Behavior of Real Ants

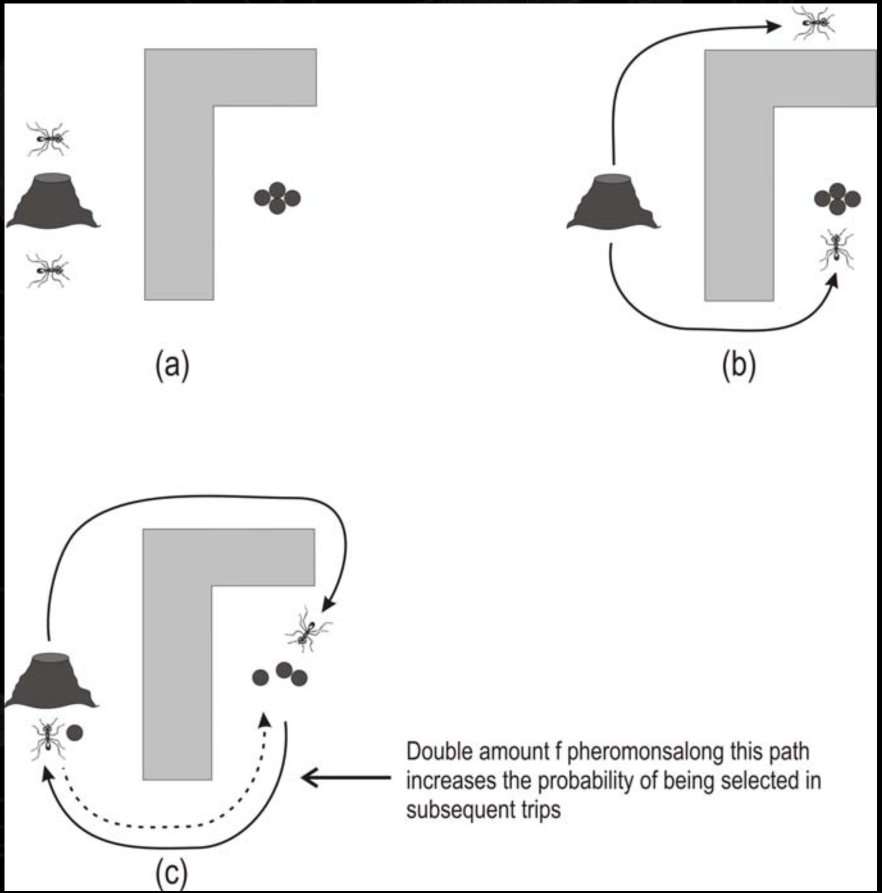
Behavior of Real Ants

Phenomenon: ant find shortest path around an obstacle separating nest and piece of candy

Experiment by Deneubourg: Initially, there is a 50% chance of an ant choosing either branch, but gradually more and more journeys are completed on the shorter branch than the longer one, causing a denser pheromone trail to be laid

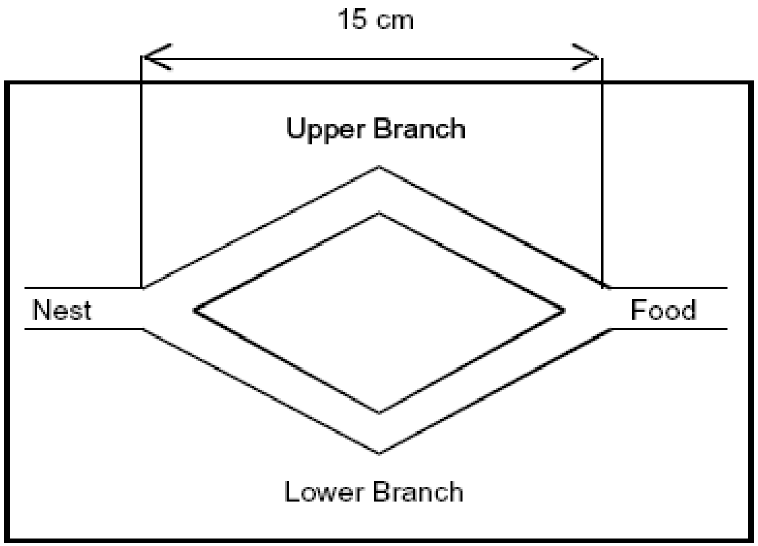
Observations:

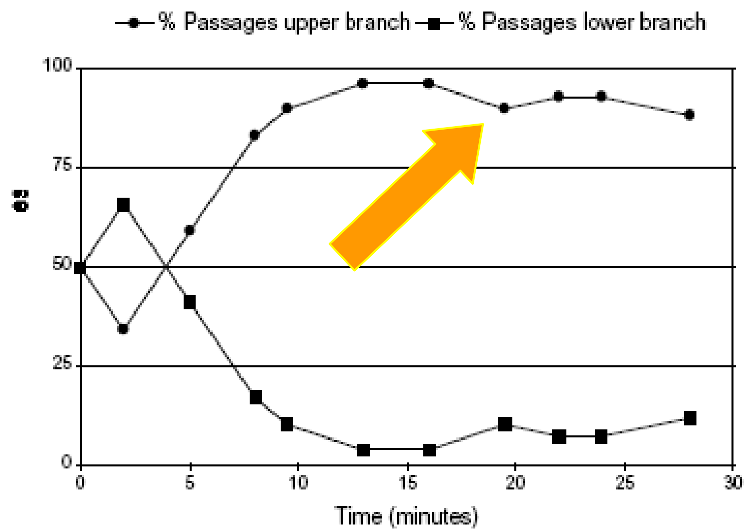
1) Ants choose paths depending on pheromone
2) After collecting food, path is marked
3) After some time, the shortest path has the highest probability



After Deneubourg et al. (1990)

The Double Bridge Experiment





After Deneubourg et al. (1990)

Note 1: ants do not directly communicate. They use a particular kind of indirect communication based on environmental modification (stigmery)

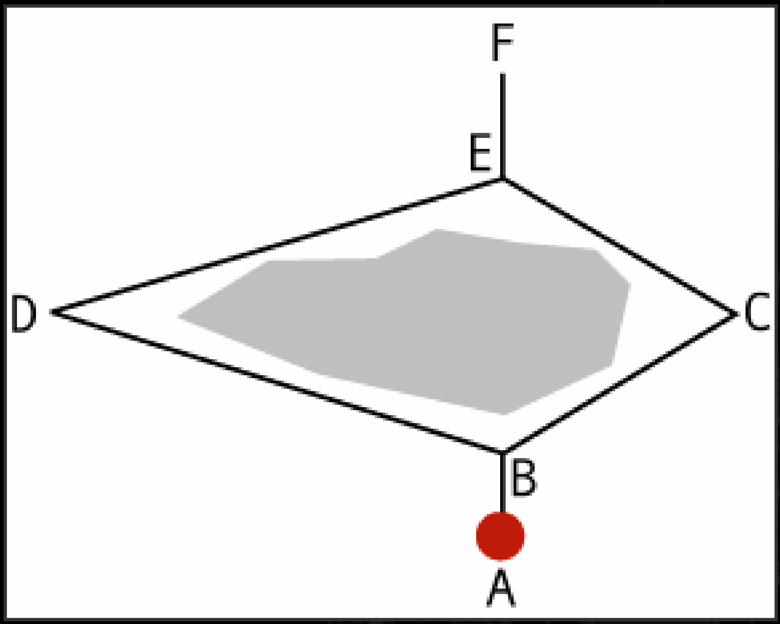
Note 2: Stigmergy is (a) indirect and non-symbolic, (b) information is local

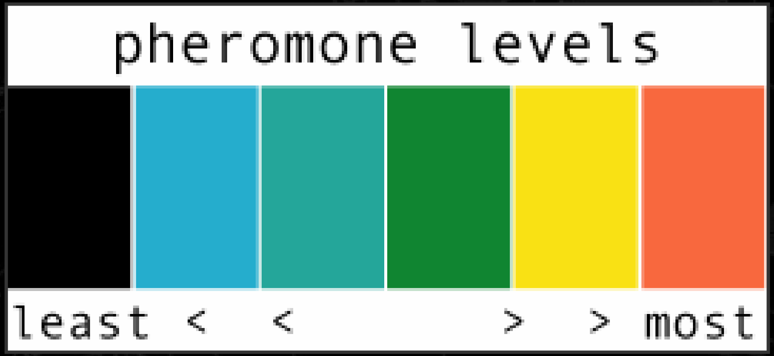
Shortest Path: Emerges from Stigmergy

Foraging ant colonies can synergistically find shortest path in distributed/dynamic environments:

- While moving back and forth between nest and food ants mark their path by pheromone laying
- Step-by-step routing decisions are biased by local intensity of pheromone field (stigmergy)
- Pheromone is the colony's collective and distributed memory: it encodes the collectively learned quality of local routing choices toward destination target

Behavior of Real Ants

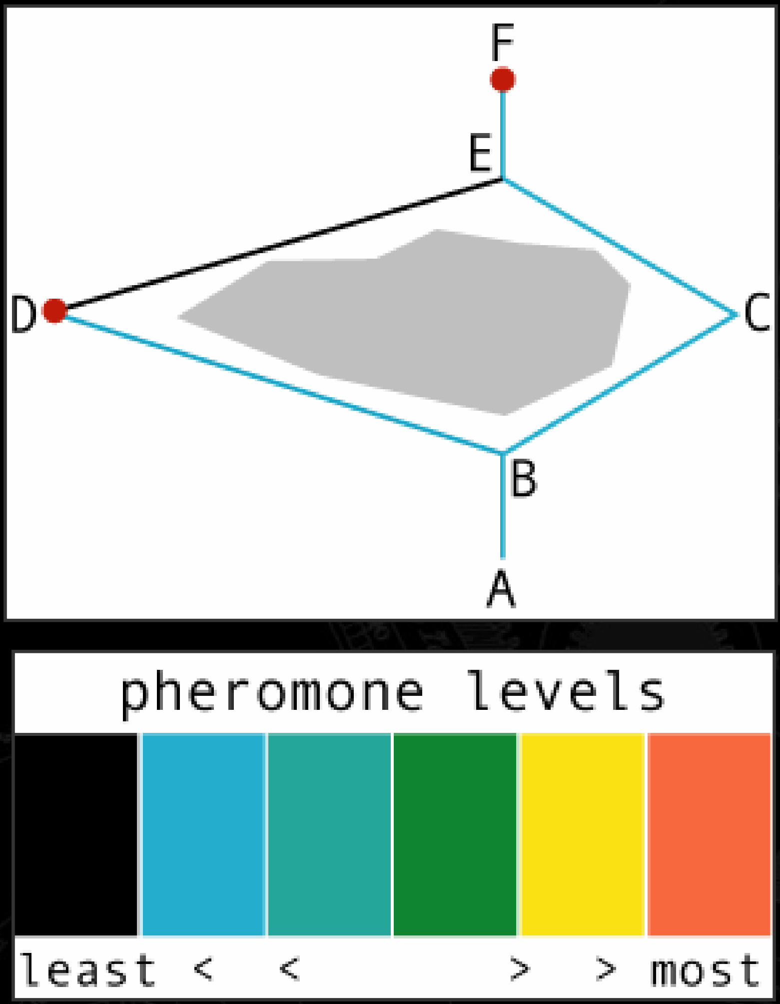




N = 20 ants

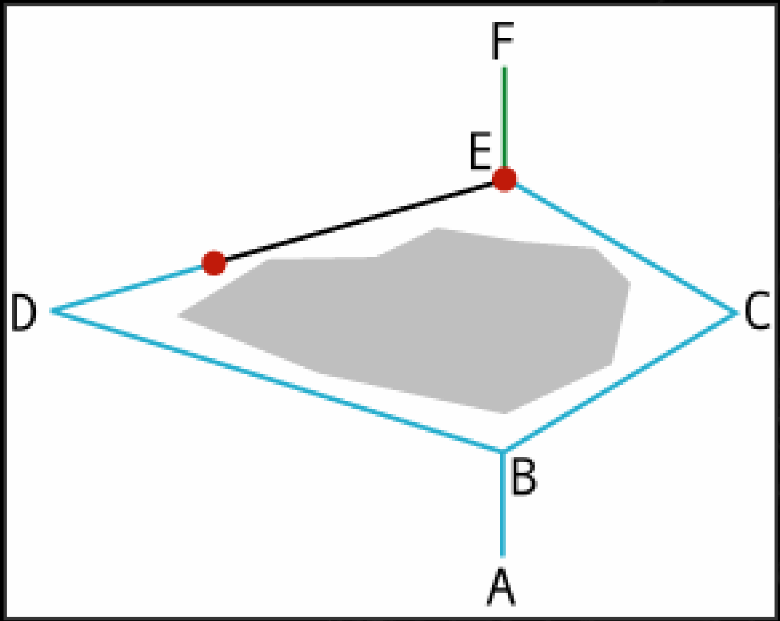
t = 0: no pheromone on edges

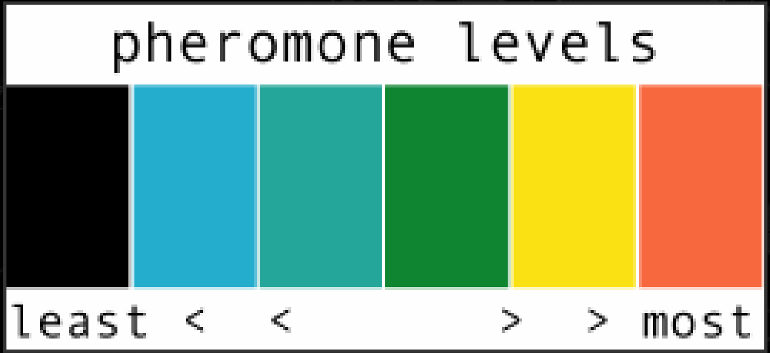
Behavior of Real Ants



- Obstacle causes split
- No guiding factor, probable equal split
- Shorter path leads to faster travel

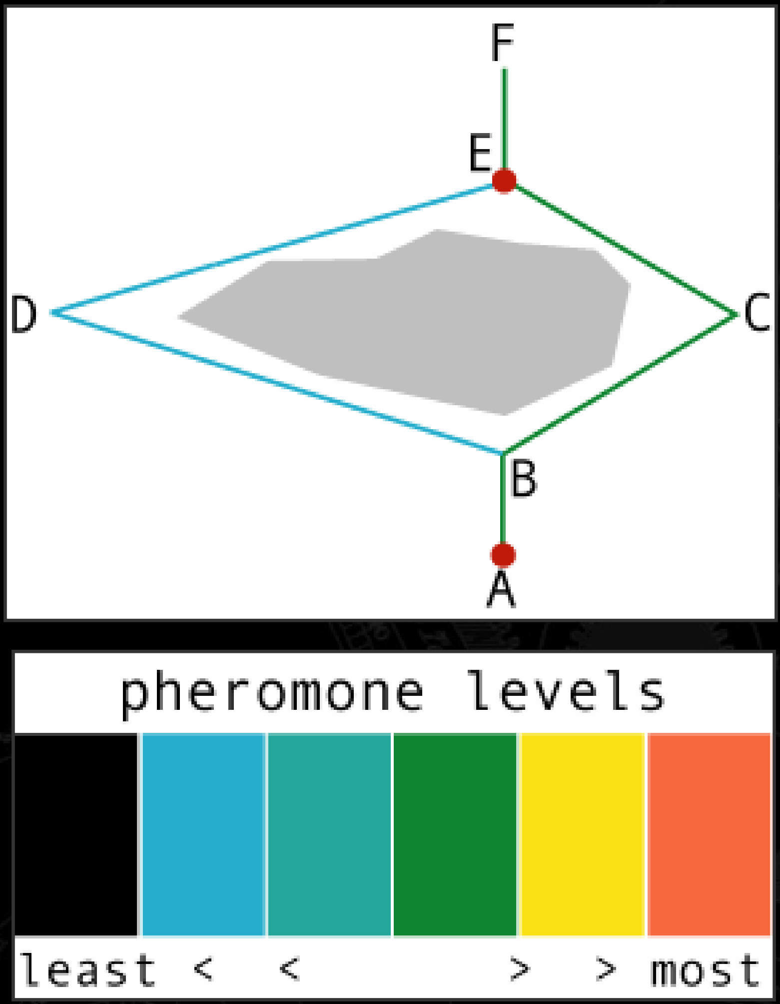
Behavior of Real Ants





The group that took the long path has yet to reach point E, while first group gets back to the path split at E

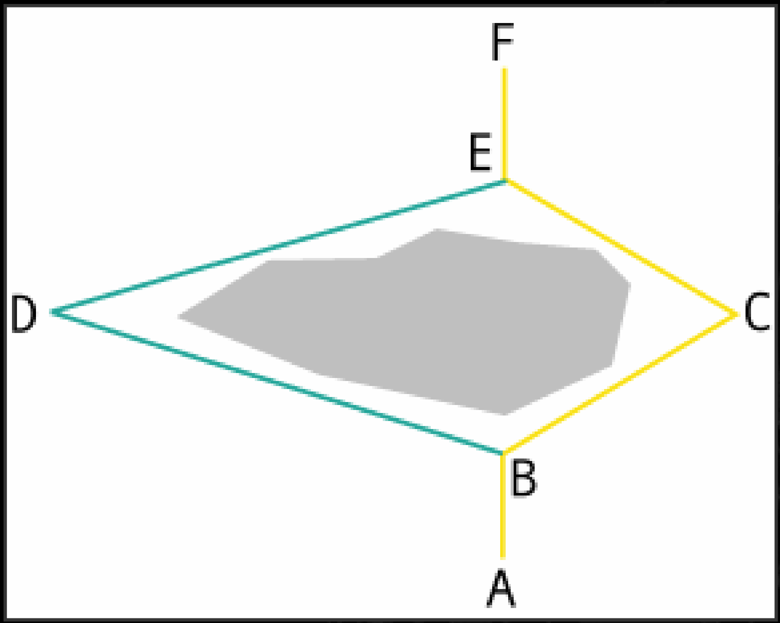
Behavior of Real Ants

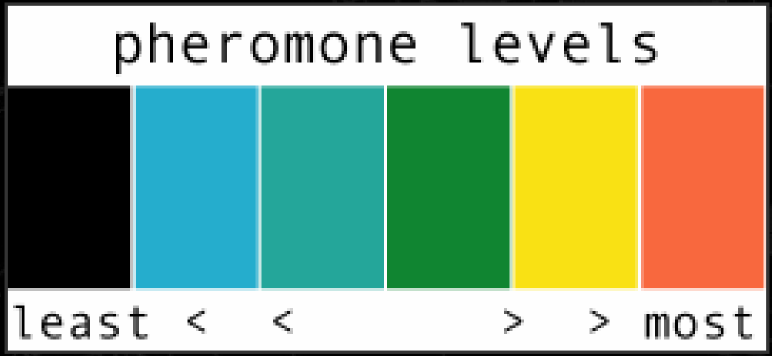


Pheromone exists on both trails, but amount differs

Edge choice is still random, but is weighted towards ECB

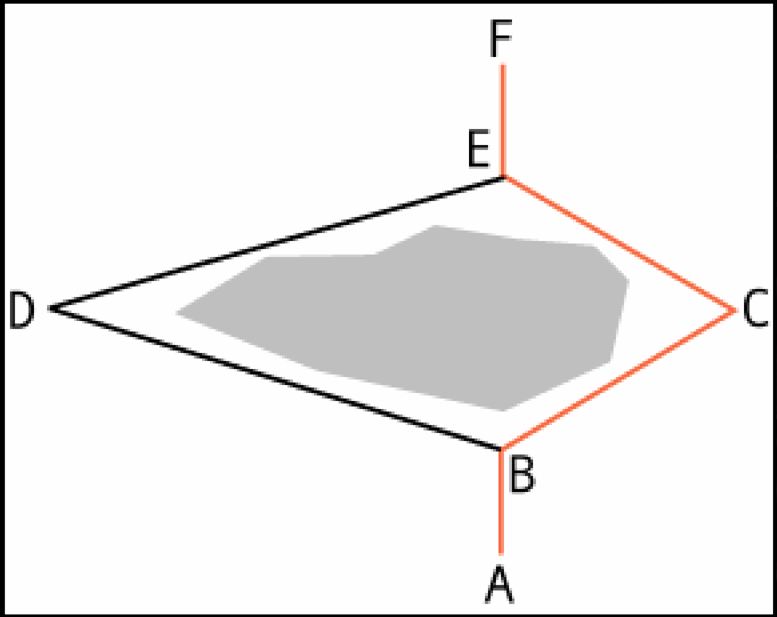
Behavior of Real Ants

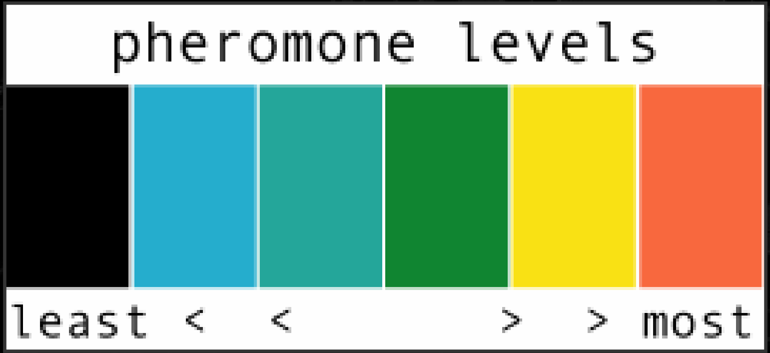




- A route begins to emerge: Most ants will begin to choose the shorter path because of the disparity between pheromone levels

Behavior of Real Ants: Lessons Learned





- Path is established: Nearly all of the ants are now taking the shortest path, which emerges because of the <u>autocatalytic behavior</u> of the ants (positive feedback)
- Self-organization through autocatalytic behavior: Good information tends to be reused over and over again, leading to more reinforcement, leading to more use
- Importance of stigmergy: information released by insects is *indirect*, *physical* and *local*
- <u>Negative feedback</u> through evaporation
- Kind of a self-correcting system with no one in charge

Ant Colony Optimization: Concept

Ants (blind) navigate from nest to food source

Shortest path is discovered via pheromone trails

- each ant moves at <u>random</u>
- <u>pheromone</u> is deposited on path
- ants detect lead ant's path, inclined to follow
- more pheromone on path increases probability of path being followed

From Ants to Agents: Ant Colony Optimization

Ideas of Ant Colony Optimization (ACO):

- Colony of cooperating (relatively simple) individuals (artificial ants); high quality solutions are the results of the cooperation among individuals of the whole colony
- Pheromone trail for stigmergic (indirect) communication (find the way to food source or nest); artificial ants change numeric information locally stored in visited problem state ("artificial pheromone trail"); "pheromone evaporation" allows ant colony to forget past history
- Sequence of local moves to find shortest paths; as real ants, artificial ants do not jump
- Stochastic decision using local information (in space an time); probabilistic decision policy is applied to move through adjacent states – no lookahead
- No explicit solution representations: The collectively learned knowledge is distributed in the pheromone

Ant Colony Optimization: Differences

Differences between real and artificial ants:

- Artificial ants live in discrete world (moves are from discrete states to discrete states)
- Artificial ants have internal states
- Artificial ants deposit an amount of pheromone which is a function of the quality of the solution found
- In many cases artificial ants update pheromone trails only after having generated a solution
- To improve overall efficiency artificial ants can be equipped with extra capabilities:
  - Look ahead
  - Local optimization
  - Backtracking

Computational Model – Meta-Heuristic

- Stigmergy can be extended to artificial agents by
  - associating appropriate variables to problem states
  - giving artificial agents only local access to variables
- Single ants find (probably) low quality solutions
- Global cooperation leads to high quality solutions
- The ants evaluate the cost of the paths they have traversed
- The shorter paths will receive a greater deposit of pheromones
- An evaporation rule will be tied with the pheromones, which will reduce the chance for poor quality solutions

ACO Meta-Heuristics

Set parameters, initialize pheromone trails

WHILE (termination condition not met) DO

ConstructAntSolutions

ApplyLocalSearch (optional)

UpdatePheromones

END WHILE

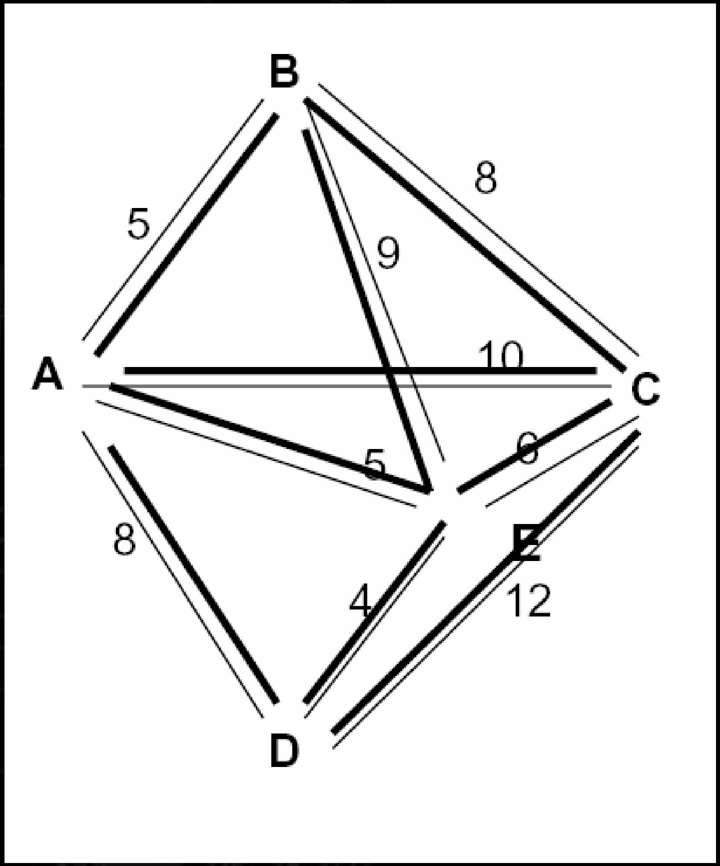
Search Space

Discrete graph

To each edge is associated a static value returned by an <u>heuristic function</u> $\eta(i,j)$ based on the edge-cost

Each edge of the graph is augmented with a <u>pheromone trail</u> $\tau(i,j)$ deposited by ants

Pheromone is dynamic and is learned at run-time



Construct_Ant_Solutions

The process of constructing solutions can be regarded as a walk on a discrete (construction) graph

The choice of a solution component is guided by a stochastic mechanism which, in turn, is biased by the pheromone $\tau(i,j)$ associated with the solution component

Apply_Local_Search

Once solutions are constructed (and before updating the pheromone table) it is common to improve the solutions obtained by the ants by local search

Note: This phase is optional

Update_Pheromones

- Increase pheromone values associated with good or promising solutions, and decrease those that are associated with bad ones
- Usually the decrement is achieved through pheromone evaporation

Main ACO Algorithms

A NON-EXHAUSTIVE LIST OF SUCCESSFUL ANT COLONY OPTIMIZATION ALGORITHMS (IN CHRONOLOGICAL ORDER).

Algorithm	*Authors*	*Year*	*References*
Ant System (AS)	Dorigo et al.	1991	[6]–[8]
Elitist AS	Dorigo et al.	1992	[7], [8]
Ant-Q	Gambardella & Dorigo	1995	[9]
Ant Colony System	Dorigo & Gambardella	1996	[10]–[12]
$\mathcal{MAX}$–$\mathcal{MIN}$ AS	Stützle & Hoos	1996	[13]–[15]
Rank-based AS	Bullnheimer et al.	1997	[16], [17]
ANTS	Maniezzo	1999	[18]
BWAS	Cordón et al.	2000	[19]
Hyper-cube AS	Blum et al.	2001	[20], [21]

Dorigo et al. (2006)

Steps for Solving a Problem by ACO

1. Represent the problem in the form of sets of (solution) components and transitions, or by a set of weighted graphs, on which ants can build solutions
2. Define the meaning of the pheromone trails
3. Define the heuristic preference for the ant while constructing a solution
4. If possible implement a efficient local search algorithm for the problem to be solved
5. Choose a specific ACO algorithm and apply to problem being solved (e.g. TSP)
6. Tune the parameters of the ACO algorithm (e.g. α and β)

Ant Colony Optimization: Applications

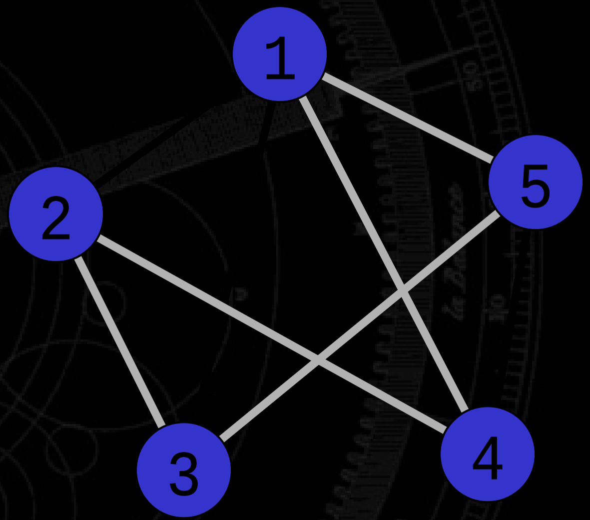
Efficiently Solves NP hard Problems

Routing

- TSP (Traveling Salesman Problem)
- Vehicle Routing
- Sequential Ordering

Assignment

- QAP (Quadratic Assignment Problem)
- Graph Coloring
- Generalized Assignment
- Frequency Assignment
- University Course Time Scheduling



Ant Colony Optimization: Applications

Table 1. List of applications of ACO algorithms to static combinatorial optimization problems. Classification by application and chronologically ordered.

Problem name	Authors	Year	Main references	Algorithm name
Traveling salesman	Dorigo, Maniezzo & Colorni	1991	[33, 39, 40]	AS
	Gambardella & Dorigo	1995	[48]	Ant-Q
	Dorigo & Gambardella	1996	[36, 37, 49]	ACS & ACS-3-opt
	Stützle & Hoos	1997	[92, 93]	$\mathcal{MM}$AS
	Bullnheimer, Hartl & Strauss	1997	[12]	AS_{rank}
Quadratic assignment	Maniezzo, Colorni & Dorigo	1994	[75]	AS-QAP
	Gambardella, Taillard & Dorigo	1997	[52]	HAS-QAP[a]
	Stützle & Hoos	1998	[94]	$\mathcal{MM}$AS-QAP
	Maniezzo & Colorni	1998	[74]	AS-QAP[b]
	Maniezzo	1998	[73]	ANTS-QAP
Job-shop scheduling	Colorni, Dorigo & Maniezzo	1994	[20]	AS-JSP
Vehicle routing	Bullnheimer, Hartl & Strauss	1996	[15, 11, 13]	AS-VRP
	Gambardella, Taillard & Agazzi	1999	[51]	HAS-VRP
Sequential ordering	Gambardella & Dorigo	1997	[50]	HAS-SOP
Graph coloring	Costa & Hertz	1997	[22]	ANTCOL
Shortest common supersequence	Michel & Middendorf	1998	[76]	AS-SCS

[a] HAS-QAP is an ant algorithm which does not follow all the aspects of the ACO meta-heuristic.
[b] This is a variant of the original AS-QAP.

From Dorigo et al. (1999)

Ant Colony Optimization: Applications

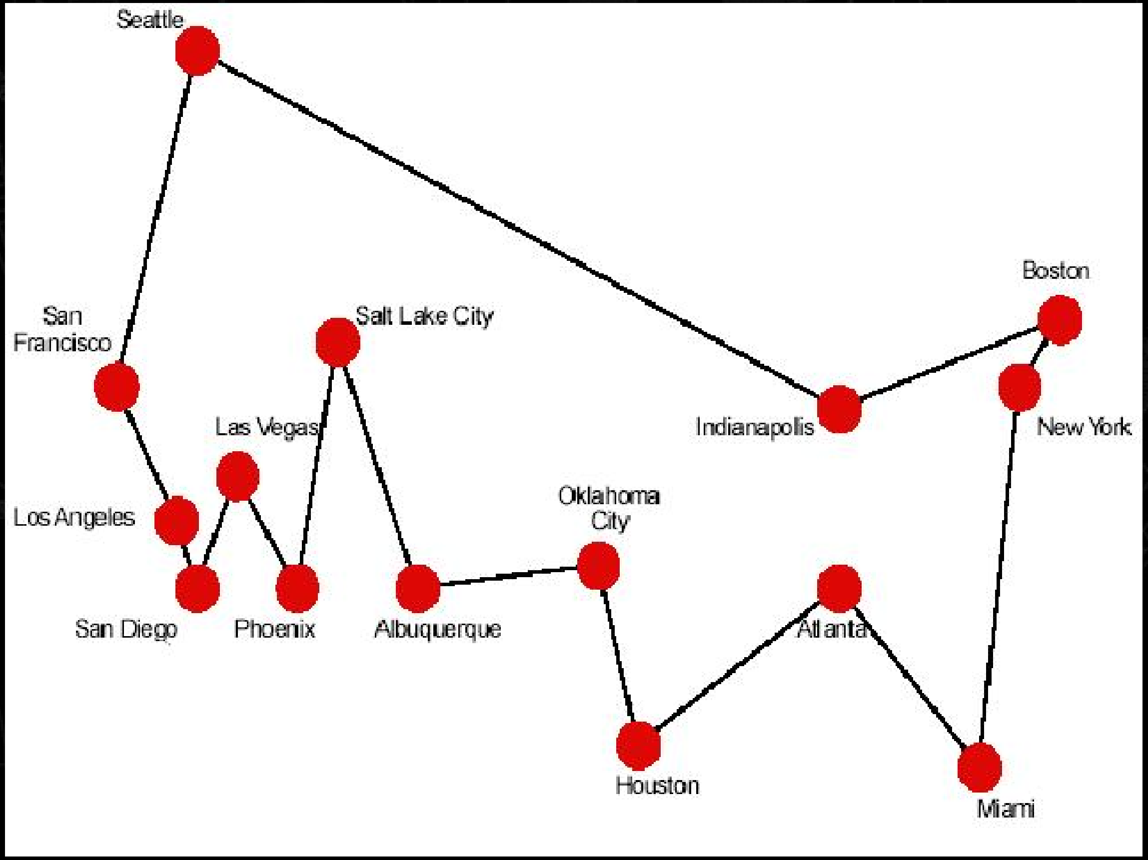
Table 2. List of applications of ACO algorithms to dynamic combinatorial optimization problems. Classification by application and chronologically ordered.

Problem name	Authors	Year	Main references	Algorithm name
Connection-oriented network routing	Schoonderwoerd, Holland, Bruten & Rothkrantz	1996	[87, 86]	ABC
	White, Pagurek & Oppacher	1998	[100]	ASGA
	Di Caro & Dorigo	1998	[31]	AntNet-FS
	Bonabeau, Henaux, Guérin, Snyers, Kuntz & Théraulaz	1998	[6]	ABC-smart ants
Connection-less network routing	Di Caro & Dorigo	1997	[26, 28, 32]	AntNet & AntNet-FA
	Subramanian, Druschel & Chen	1997	[95]	Regular ants
	Heusse, Guérin, Snyers & Kuntz	1998	[62]	CAF
	van der Put & Rothkrantz	1998	[97, 98]	ABC-backward

From Dorigo et al. (1999)

The Traveling Salesman Problem

Given a set of cities. A salesman needs to travel through all the cities following an optimal route (Hamiltonian path: goes through cities once and only once) that minimises the distance travelled.



TSP – History

- First description in 1800 by Irish mathematician Sir William R. Hamilton
- The general form is presented for first time in 1930

TSP – An Easy Problem?

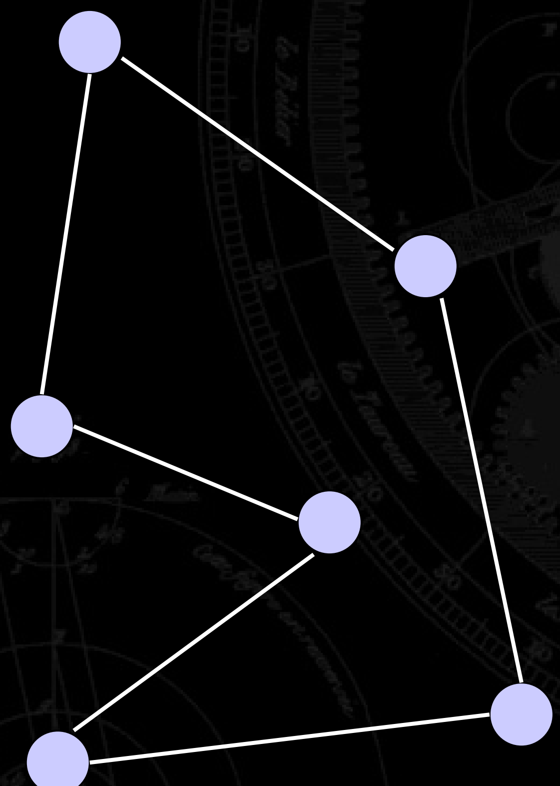
Direct solution:

Try all the possible permutations (ordered combinations) and see which one is the cheapest (using brute force)

How many different paths are where for N cities?

The Traveling Salesman Problem

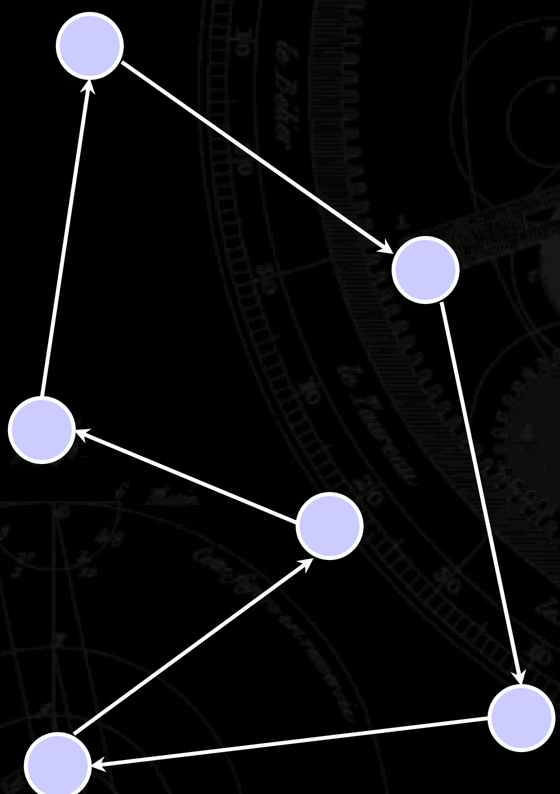
Hint:



There is a symmetric case, that is, for any two cities A and B, the distance from A to B is the same as that from B to A

The Traveling Salesman Problem

Hint:



There is a asymmetric case, that is, for any two cities A and B, the distance from A to B is the not the same as that from B to A

The Traveling Salesman Problem

Answer:

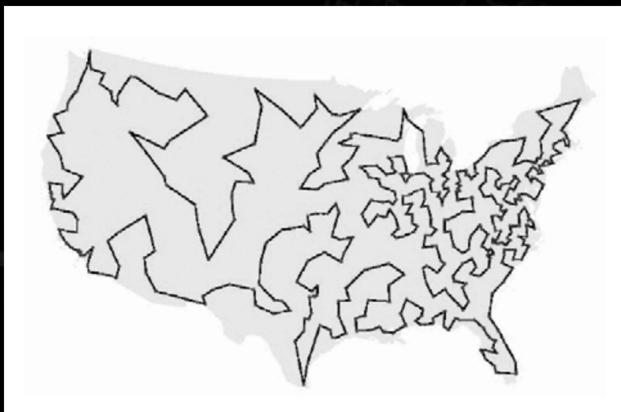
1) For the symmetric case (for any two cities A and B, the distance from A to B is the same as that from B to A) the solution is (N-1)!/2
2) For the asymmetric case: (N-1)!

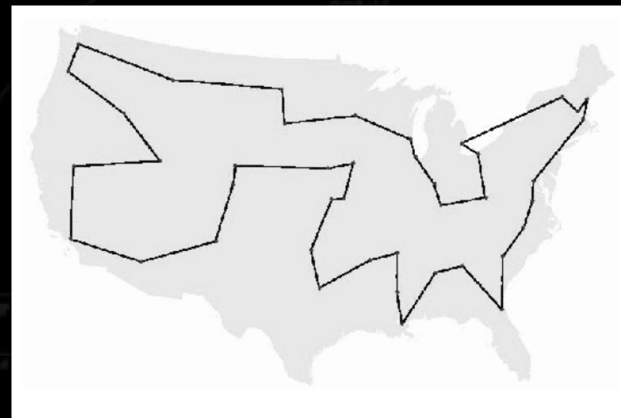
Problem is NP-hard: Not been found an algorithm that solves the general problem (for any number of cities at any arrangement), in polynomial time

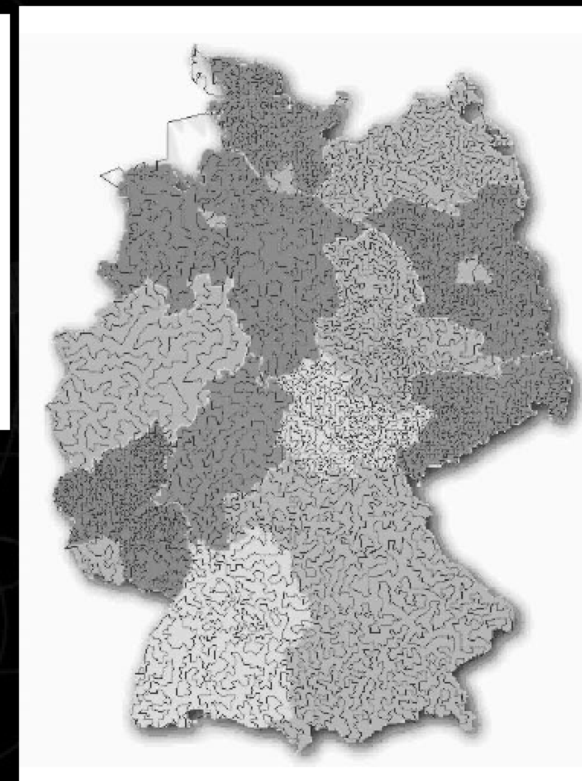
TSP Instances

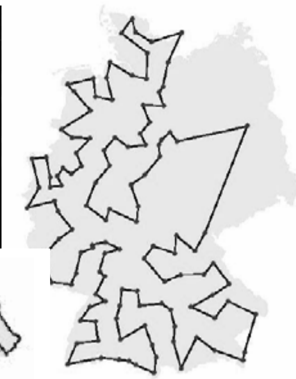
years	Research team	Problem size
1954	G.Dantzig, R. Fulkerson, and S. Johnson	49 cities
1971	M. Held and R.M.Karp	64 cities
1975	P.M.Camerini, L. Fratta, and F. Maffioli	100 cities
1977	M.Grötschel	120 cities
1980	H.Crowder and M.W.Padberg	318 cities
1987	M.Padberg and G.Rinaldi	532 cities
1987	M. Grötschel and O.Holland	666 cities
1987	M. Padberg and G.Rinaldi	2.392 cities
1994	D.Applegate, R.Bixby, V.Chvàtal, e W.Cook	7.397 cities
1998	D.Applegate, R.Bixby, V.Chvàtal, e W.Cook	13.509 cities
2001	D.Applegate, R.Bixby, V.Chvàtal, e W.Cook	15.112 cities
2004	D.Applegate, R.Bixby, V.Chvàtal, e W.Cook	24.978 cities

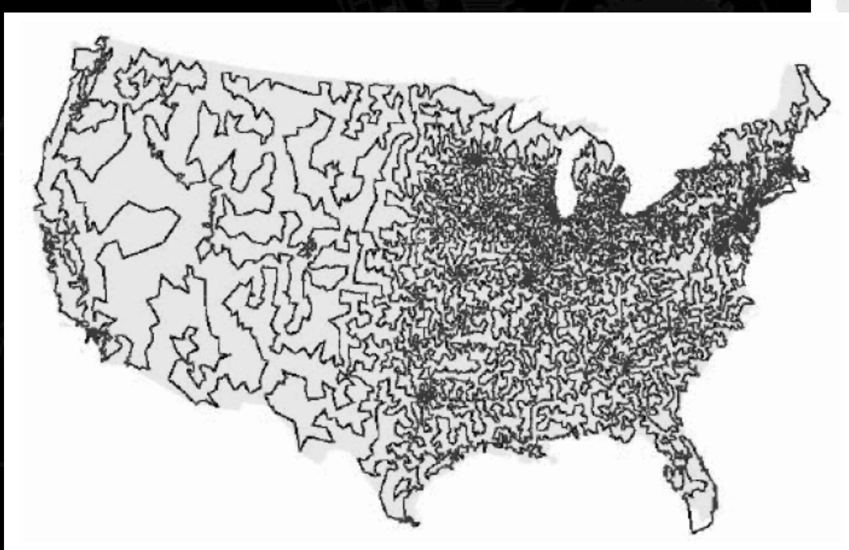
Example of Solutions

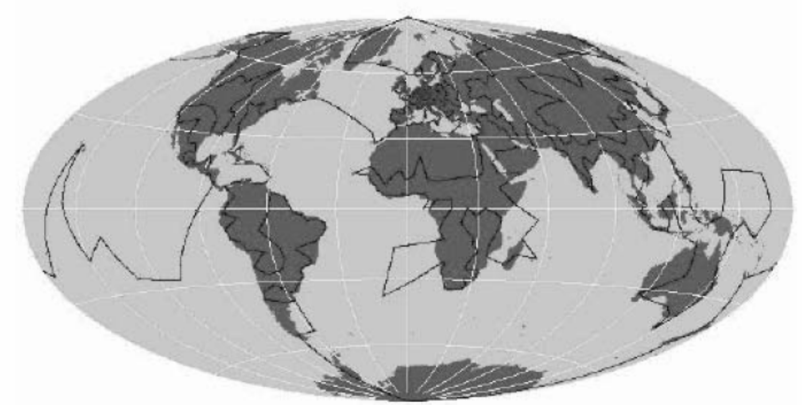




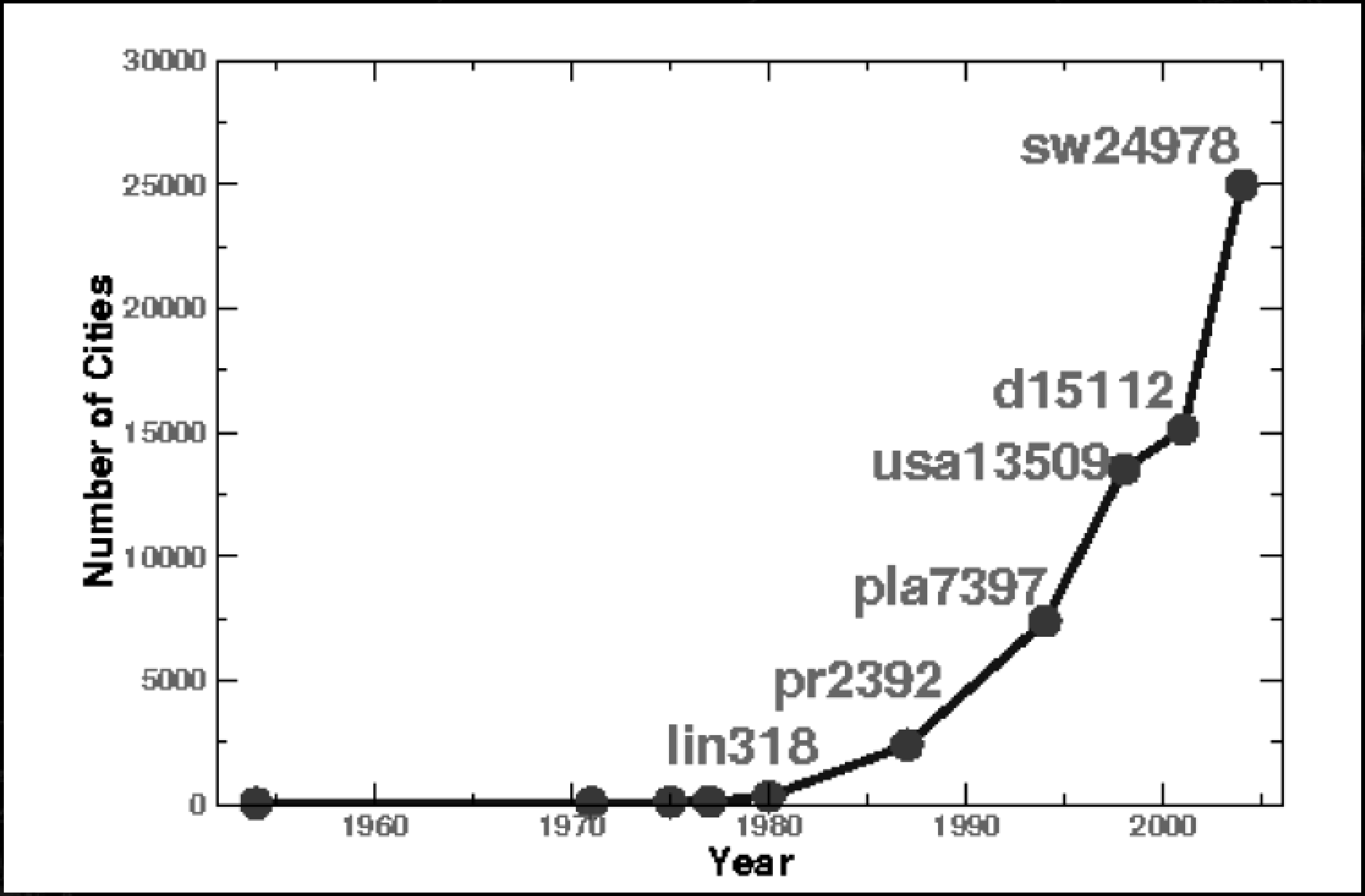








TSP as a Benchmark Problem



Computational Model – Some Terminology

Node: a vertex

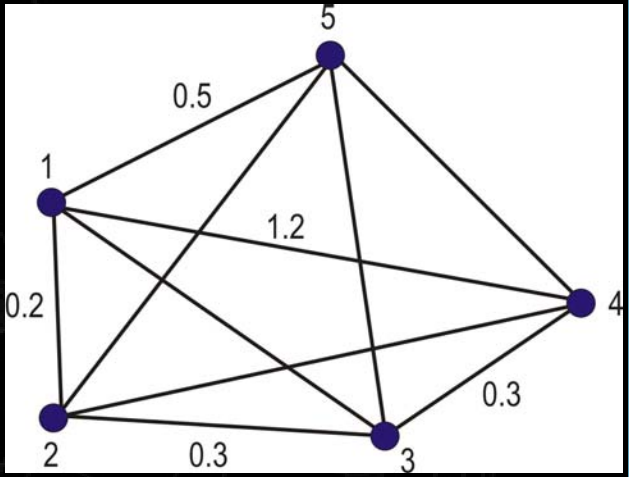
Edge: a line connecting 2 vertices

Graph: Diagram connecting nodes with edges

Weighted graph: Edges have weights

Path: Sequence of nodes-edges from between two nodes

Hamiltonian path: A path without any node revisited

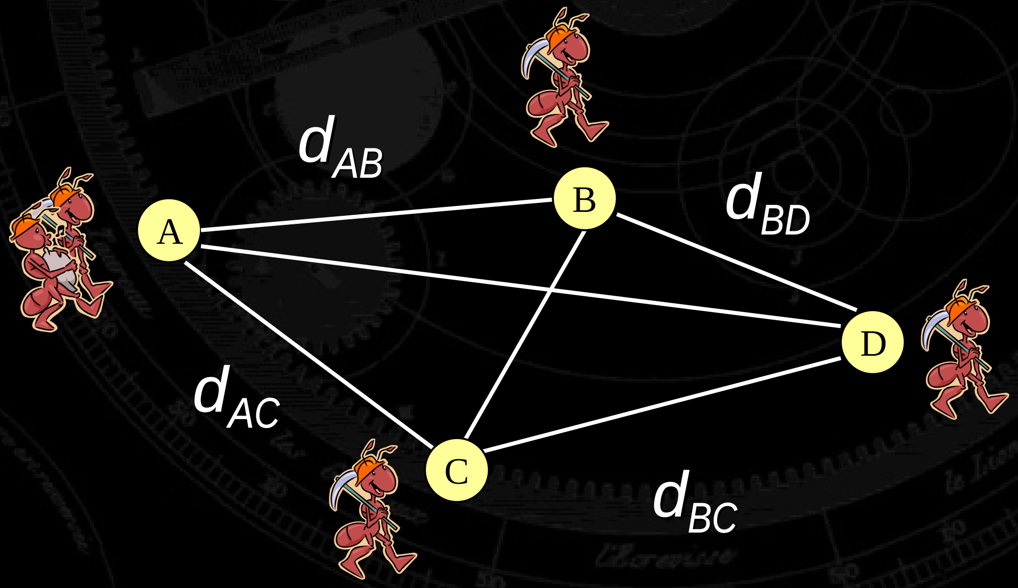


TSP – Modeling

Graph (N,E): where N = cities/nodes, E = edges

d_{ij} = the tour cost from city i to city j (edge weight)

$\eta_{ij} = 1/d_{ij}$



TSP – Modeling

We will look at two ACO algorithms

- Ant System (AS): first ACO algorithm!
- Ant Colony System (ACS)

Properties of system

1. State transition rule
2. Pheromone trail
3. Memory
4. Awareness of environment

Ant System - Initialization

At the beginning of the search process, a constant amount of pheromone $\tau(i,j)$ is assigned to all arcs

The resulting pheromone table is public information

Ant System

1. Transition rule (random-proportional rule)

= probability with which ant *k* chooses to move from city *i* to city *j*

Depends on:

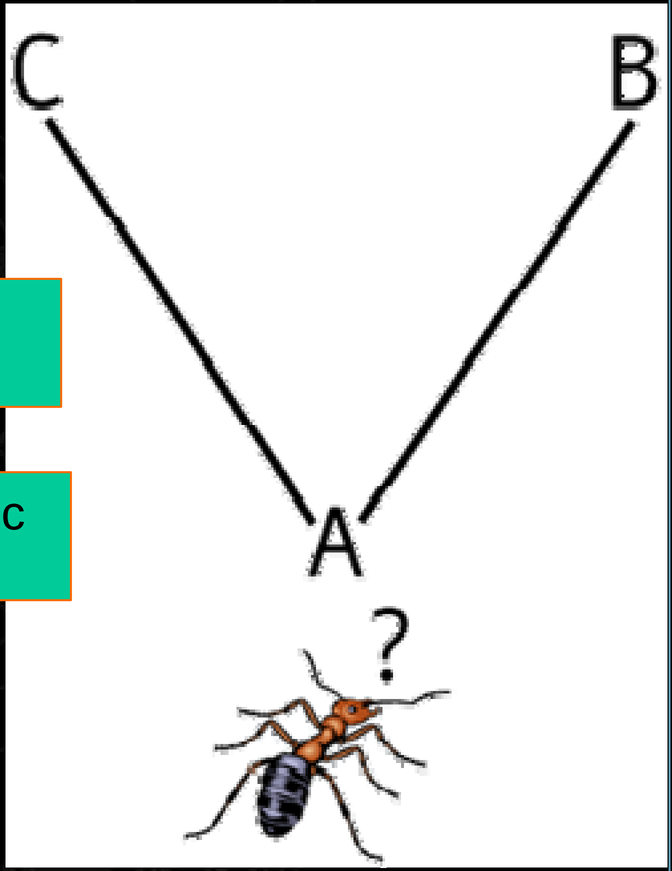
a) city distance (visibility)

b) amount of pheromone

quantity of pheromone

inv. of heuristic distance

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{j \in \text{allowed}} [\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}} & \text{if } j \in \text{allowed} \\ 0 & \text{otherwise} \end{cases}$$



where α, β control relative importance of pheromone vs. heuristic information

allowed are elements that are not in the tabu list

Ant System

1. Transition rule

$$p_{ij}^{\;k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{j \in \text{allowed}} [\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}} & \text{if } j \in \text{allowed} \\ 0 & \text{otherwise} \end{cases}$$

α, β control relative importance of pheromone vs. heuristic information

α = *0:* represents a greedy approach (closest cities are more likely to be selected; classic stochastic greedy algorithm; may lead to sub-optimal solutions)

β = *0:* represents a rapid selection of tours that may not be optimal (only pheromone amplification is at works; may lead to sub-optimal solutions)

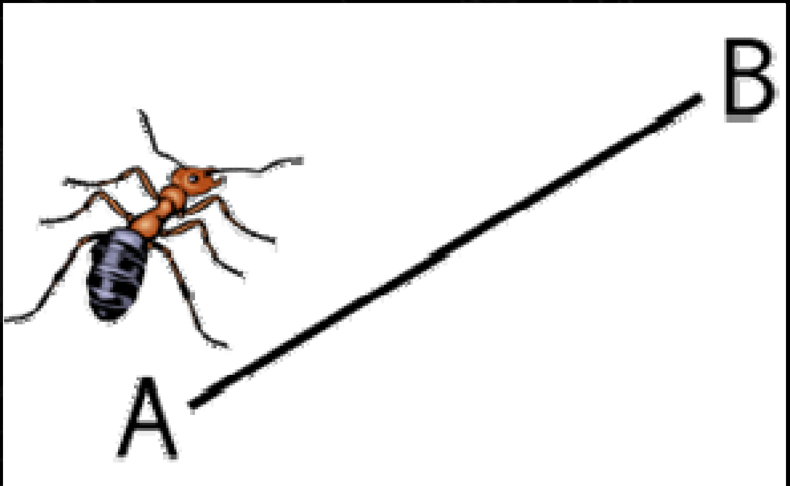
Thus, a trade-off is necessary

Ant System

2. Pheromone trail (deposit + evaporation)

Intensity of trail:

Trail update function: $\tau_{ij}(t)$



$$\tau_{ij}(t+1) = (1-\rho)\cdot\tau_{ij}(t) + \sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t)$$

ρ = coefficient of evaporation, m = number of ants

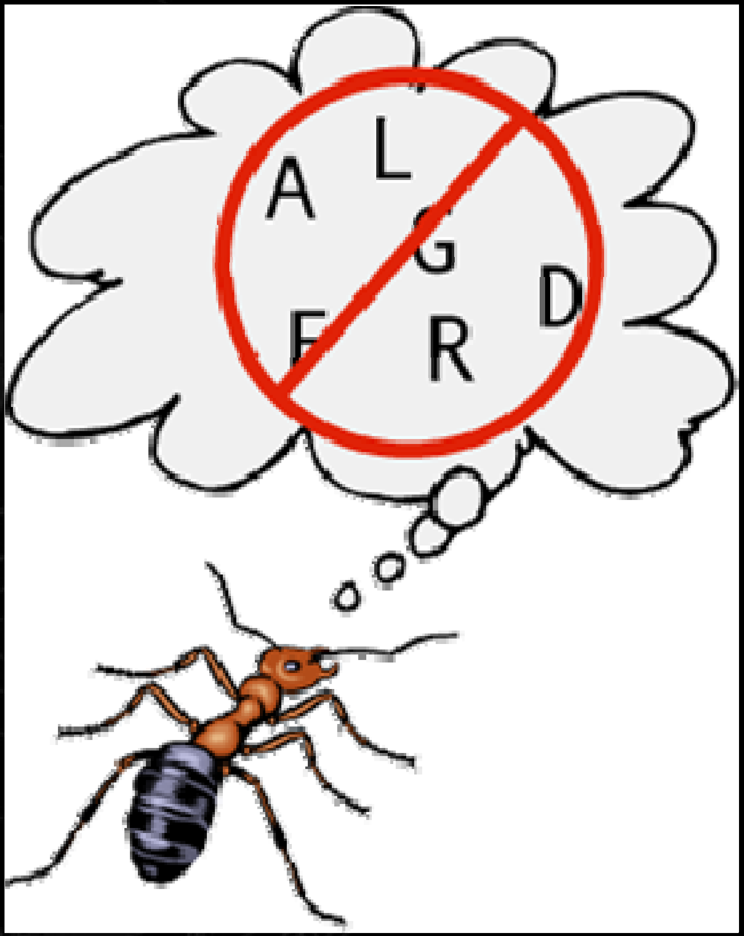
$\Delta\tau_{ij,k}(t)$ is the new quantity of pheromone laid on edge (i,j) by ant k (value depends on type of algorithm, e.g. ant-density, ant-quantity, ant-cycle)

For ant-cycle: $\Delta\tau_{ij,k}(t) = Q/L_k$ if ant k used edge (i,j), where L_k is length of tour constructed by ant k, and Q is a constant

Ant System

3. Memory

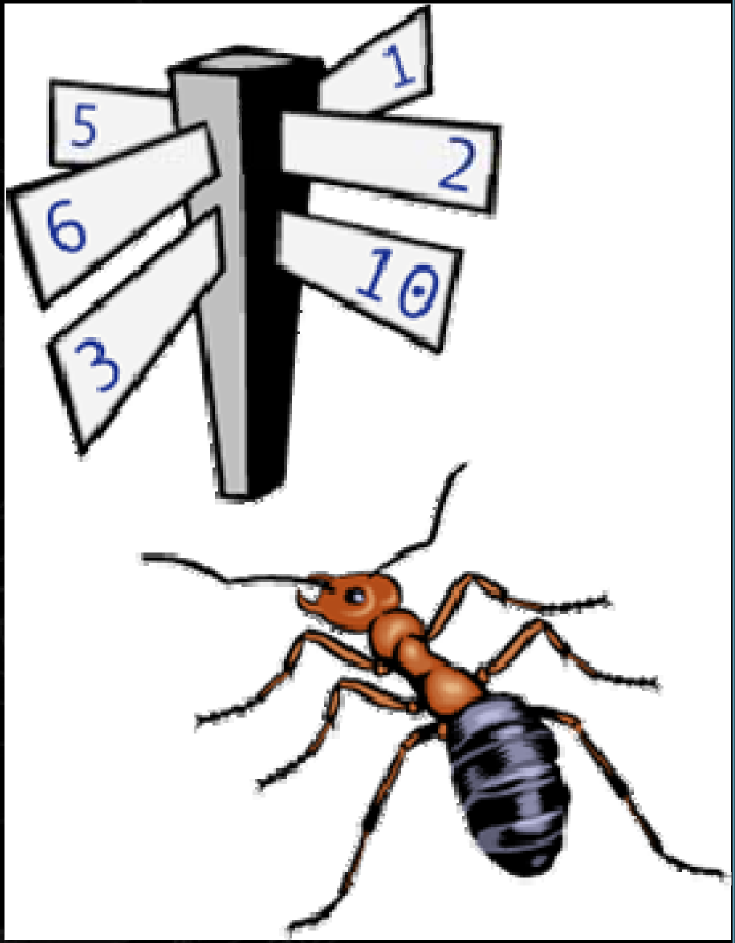
- Prevents town repeats
- Implemented as a tabu list: *$tabu_k$* is the list of the k-th ant



Ant System

4. Awareness of environment

- City distance: d_{ij}
- Visibility: $\eta_{ij} = 1/d_{ij}$



Ant System

Three AS algorithms can be defined (function of pheromone table update rule):

- ant-density: amount of deposited pheromone is constant
- ant-quantity: amount of deposited pheromone is inversely proportional to arc length
- ant-cycle: pheromone is deposited after a complete tour has been built (even if not optimal)

Ant-Cycle Algorithm

1. **Initialize all data structures:**

    Set an initial pheromone trail value $\tau(i,j)$ on edges

    Place the ants on the nodes of the graph

2. For each ant, do:

    Until every city has been visited, do:

    Using the *probability function*, choose the city to move to

    Insert the chosen town into the ant's *tabu list*

3. Update the trail values using the *global updating rule* and the best tour found thus far

4. Memorize the shortest path thus far and empty all *tabu lists*

5. If the end condition (usually defined as a number of cycles) is not met, then:

    Set t:= t + n

    Go to step 2

    else:

    Print the shortest path and stop

Ant-Cycle Algorithm

1. **Initialize all data structur**

   Set an initial pheromo

   Place the ants on the r

$$p_{ij}^{\ k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{j \in \text{allowed}} [\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}} & \text{if } j \in \text{allowed} \\ 0 & \text{otherwise} \end{cases}$$

2. For each ant, do:

   Until every city has been visited, do:

   Using the *probability function*, choose the city to move to

   Insert the chosen town into the ant's *tabu list*

3. Update the trail values using the *global updating rule* and the best tour found thus far

4. Memorize the shortest path thus far and empty all *tabu lists*

5. If the end condition (usually defined as a number of cycles) is not met, then:

   Set t:= t + n

   Go to step 2

   else:

   Print the shortest path and stop

Ant-Cycle Algorithm

1. **Initialize all data structures:**

   Set an initial pheromone trail value $\tau(i,j)$ on edges

   Place the ants on the nodes of the graph

2. For each ant, do:

   Until every city has been visited, do:

   Using the *probability function*, choose the city to move to

   Insert the chosen town into the ant's *tabu list*

3. Update the trail values using the *global updating rule* and the best tour found thus far

4. Memorize the shortest path thus far and empty all *tabu lists*

5. If the end condition (usually defined as a number of cycles) is not met, then:

   Set t:= t + n

   Go to step 2

   else:

   Print the shortest path and stop

Ant-Cycle Algorithm

1. **Initialize all data structures:**

    Set an initial pheromone trail value $\tau(i,j)$ on edges

    Place the ants on the nodes of the graph

2. For each ant, do:

    Until every city has been visited, do:

    Using the *probability function*, choose the city to move to

    Insert the chosen town into the ant's *tabu list*

3. Update the trail values using the *global updating rule* and the best tour found thus far

4. Memorize the shortest path thus far and empty all *tabu lists*

5. If the end condition (usually defined as a number of cycles) is not met, then:

    Set t:= t + n

    Go to step 2

    else:

    Print the shortest path and stop

Ant-Cycle Algorithm

1. **Initialize all data structures:**

   Set an initial pheromone trail value $\tau(i,j)$ on edges

   Place the ants on the nodes of the graph

2. For each ant, do:

   Until every city has been visited, do:

   Using the *probability function*, choose the city to move to

   Insert the chosen town into the ant's *tabu list*

3. Update the trail values using the *global updating rule* and the best tour found thus far

4. Memorize the shortest path thus far and empty all *tabu lists*

5. If the end condition (usually defined as a number of cycles) is not met, then:

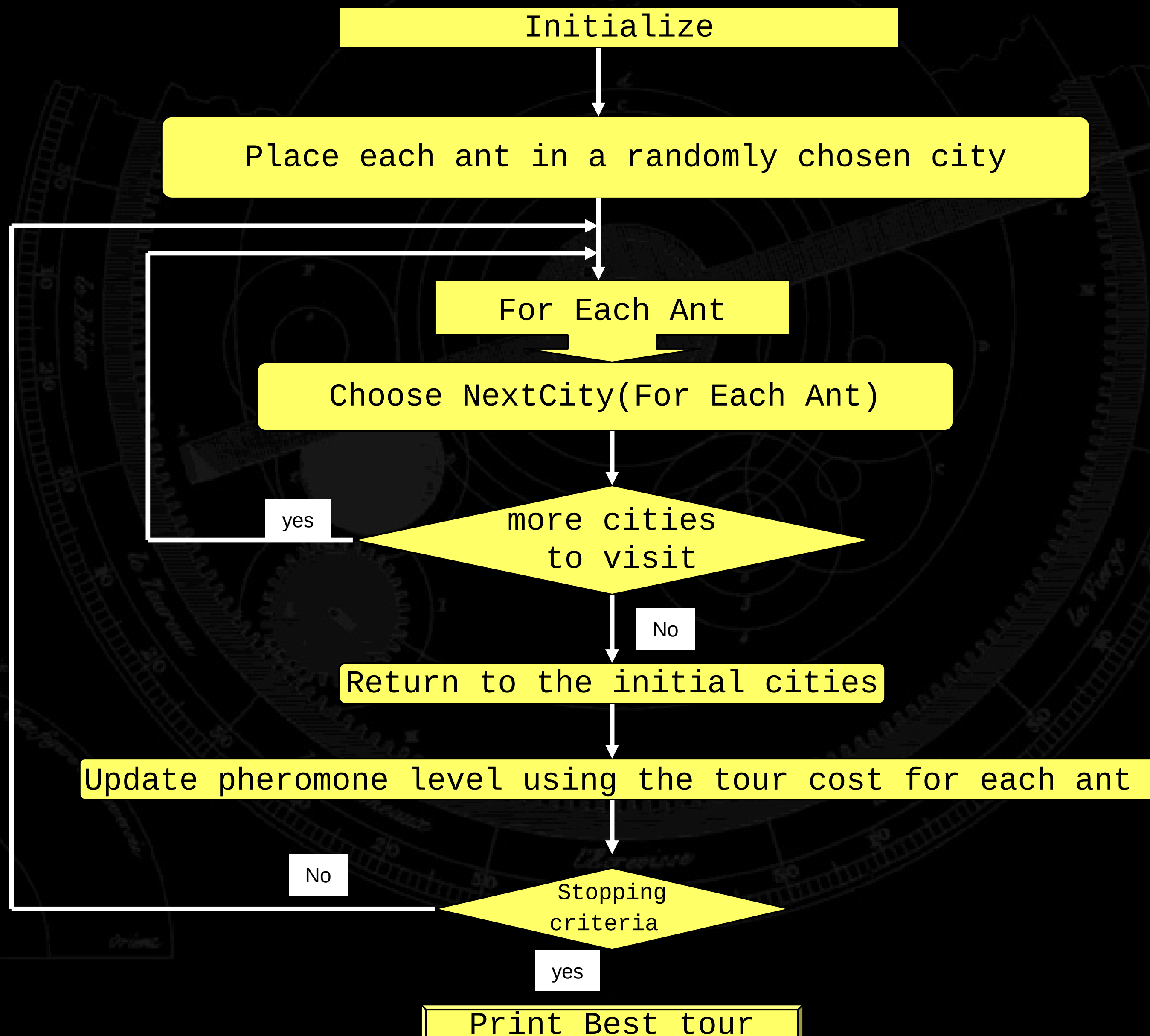
   Set t:= t + n

   Go to step 2

   else:

   Print the shortest path and stop

Ant-Cycle Algorithm



AS: Performance

- Ant System was used to show that an ant based approach was possible
- The limitation is its performance
- AS was modified and extended with ACS: Ant Colony System (Dorigo and Gambardella, 1994)

Ant Colony System (ACS)

- A new next city selection mechanism is introduced
- Pheromone is modified (decreased) locally as soon as edge are selected (local pheromone update); local update is performed by all ants after each construction step
- Pheromone concentration decreases and leads subsequent ants to choose other edges: search is diversified
- Pheromone is updated globally (at the end of each iteration) only by best ant (best-so-far or iteration-best)

ACS: State Transition Rule (Decision Rule)

Yet another important difference between AS and ACS

q0 is random variable in [0,1]

$$s = \begin{cases} \underset{u \in J_k(r)}{arg\ max} \left\{ [\tau(r,u)] \cdot [\eta(r,u)]^{\beta} \right\} & \text{if } q \le q_0 \quad \text{(Exploitation)} \\ S & \text{otherwise} \quad \text{(Exploration)} \end{cases}$$

where

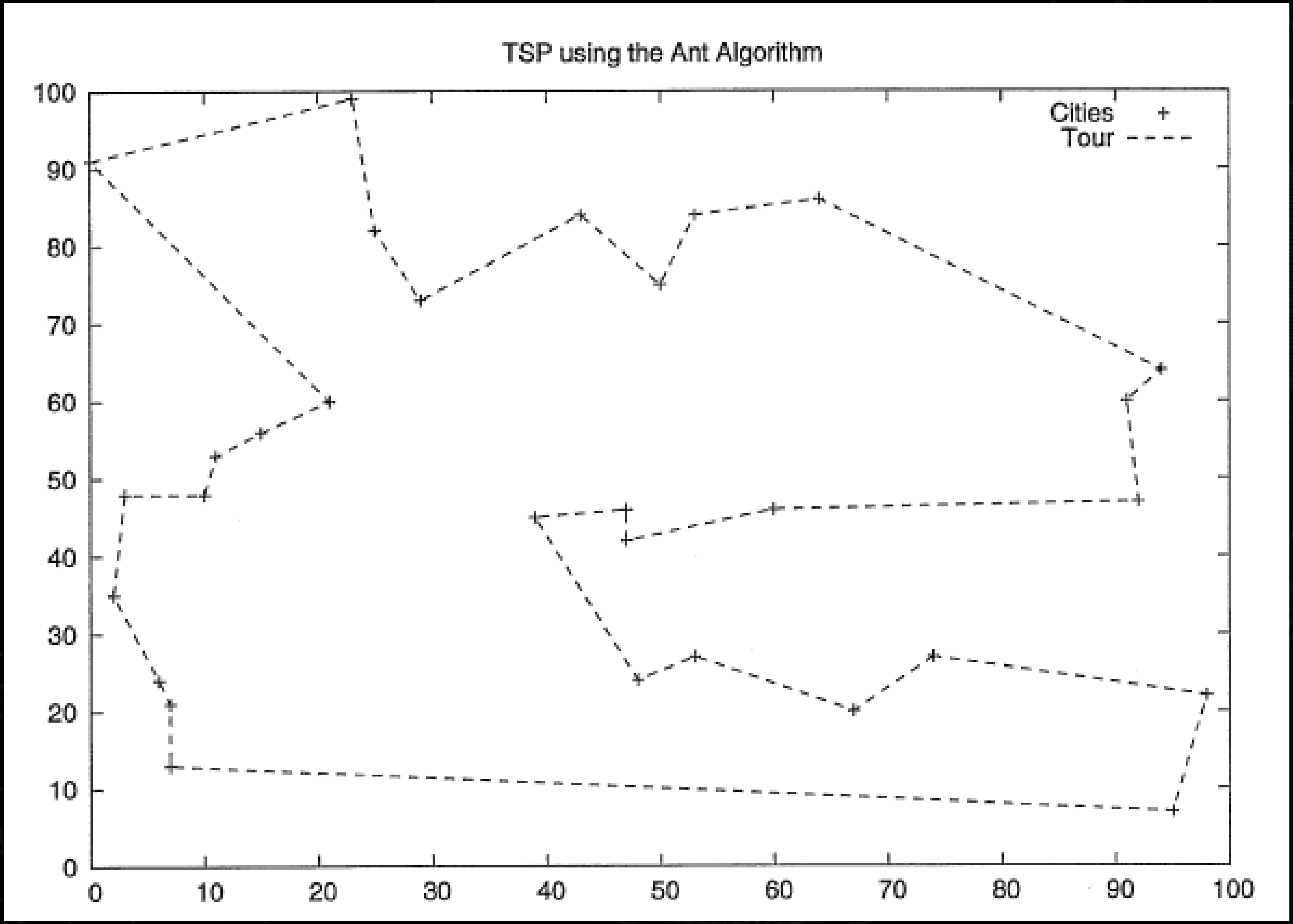
- S is a stochastic variable distributed as follows:

$$p_k(r,s) = \begin{cases} \dfrac{[\tau(r,s)] \cdot [\eta(r,s)]^{\beta}}{\sum\limits_{u \in J_k(r)} [\tau(r,u)] \cdot [\eta(r,u)]^{\beta}} & \text{if } s \in J_k(r) \\ 0 & \text{otherwise} \end{cases}$$

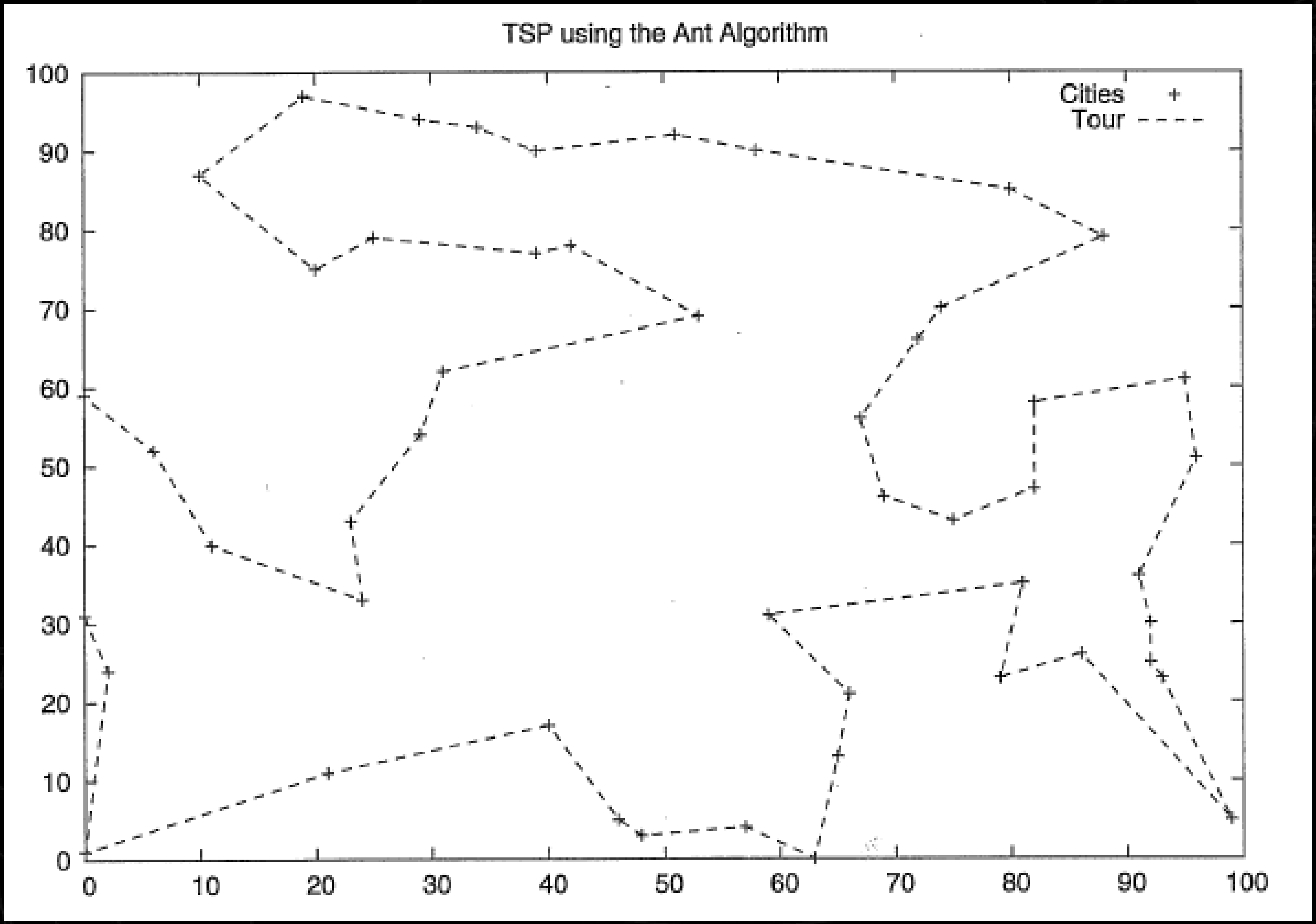
- τ is the trail
- η is the inverse of the distance
- $J_k(r)$ is the set of cities still to be visited by ant k positioned on city r
- β and q_0 are parameters

TSP: Sample Runs

TSP: Sample Run For 30 Cities



TSP: Sample Run For 50 Cities



Results (Average Tour Length)

Table 1. Comparison of ACS with other nature-inspired algorithms on random instances of the symmetric TSP. Comparisons on average tour length obtained on five 50-city problems. SA = simulated annealing, EN = elastic net, SOM = self organizing map, FI = farthest insertion. Results on SA, EN, and SOM are from (Durbin and Willshaw,1987; Potvin, 1993). FI results are averaged over 15 trials starting from different initial cities. ACS was run for 1,250 iterations using m=20 ants and the results are averaged over 15 trials. The best average tour length for each problem is in boldface.

Problem name	ACS	SA	EN	SOM	FI
City set 1	**5.86**	5.88	5.98	6.06	6.03
City set 2	6.05	**6.01**	6.03	6.25	6.28
City set 3	**5.57**	5.65	5.70	5.83	5.85
City set 4	**5.70**	5.81	5.86	5.87	5.96
City set 5	**6.17**	6.33	6.49	6.70	6.71

Advantages and Disadvantages of ACO

For TSPs (Traveling Salesman Problem), relatively efficient

- Small number of nodes: TSPs solved by exhaustive search
- Large number of nodes: TSPs are computationally difficult to solve (NP-hard), i.e. exponential time to convergence

ACS performs better against other global optimization techniques for TSP (neural net, genetic algorithms, simulated annealing)

In particular, compared to genetic algorithms:

  - ACO is affected less by poor initial solutions (due to combination of random path selection and colony memory)

Advantages and Disadvantages (Continued)

- Can be used in dynamic applications (adapts to changes such as new distances, etc.)
- As with GAs, good choice for constrained discrete problems (not a gradient-based algorithm)

A NON-EXHAUSTIVE LIST OF APPLICATIONS OF ACO ALGORITHMS GROUPED BY PROBLEM TYPE.

Problem type	*Problem name*	*Authors*	*Year*	*References*
Routing	Traveling salesman	Dorigo et al.	1991, 1996	[6], [8]
		Dorigo & Gambardella	1997	[11]
		Stützle & Hoos	1997, 2000	[15], [47]
	Vehicle routing	Gambardella et al.	1999	[48]
		Reimann et al.	2004	[49]
	Sequential ordering	Gambardella & Dorigo	2000	[50]
Assignment	Quadratic assignment	Stützle & Hoos	2000	[15]
		Maniezzo	1999	[18]
	Course timetabling	Socha et al.	2002, 2003	[35], [51]
	Graph coloring	Costa & Hertz	1997	[52]
Scheduling	Project scheduling	Merkle et al.	2002	[53]
	Total weighted tardiness	den Besten et al.	2000	[54]
	Total weighted tardiness	Merkle & Middendorf	2000	[55]
	Open shop	Blum	2005	[56]
Subset	Set covering	Lessing et al.	2004	[57]
	l-cardinality trees	Blum & Blesa	2005	[58]
	Multiple knapsack	Leguizamón & Michalewicz	1999	[59]
	Maximum clique	Fenet & Solnon	2003	[60]
Other	Constraint satisfaction	Solnon	2000, 2002	[61], [62]
	Classification rules	Parpinelli et al.	2002	[63]
		Martens et al.	2006	[64]
	Bayesian networks	Campos, Fernández-Luna,	2002	[65], [66]
	Protein folding	Shmygelska & Hoos	2005	[67]
	Docking	Korb et al.	2006	[68]

Advantages and Disadvantages (Continued)

Theoretical analysis is difficult:

- Due to sequences of random decisions (not independent)
- Probability distribution changes by iteration
- Research is experimental rather than theoretical

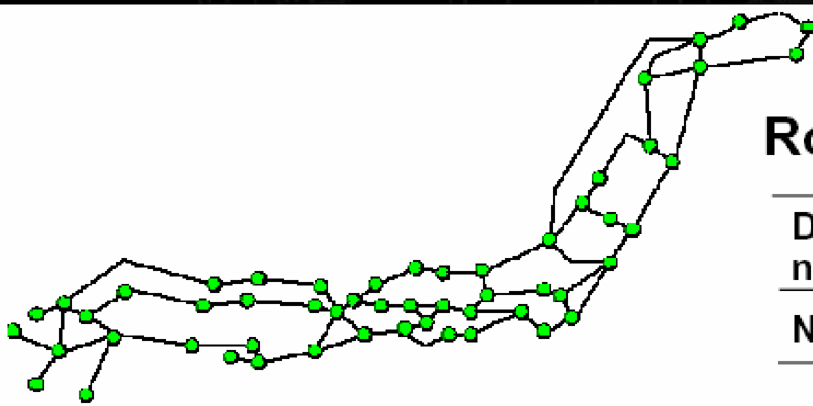
Convergence is guaranteed, <u>but</u> time to convergence uncertain

Coding is somewhat complicated, not straightforward

- Pheromone “trail” additions/deletions, global updates and local updates
- Large number of different ACO algorithms to exploit different problem characteristics

The Routing (or Load Balancing) Problem

Load balancing is the construction of call-routing schemes which distribute the changing load over the system to minimize lost calls. The practical goal of routing algorithms is to build routing tables.



Routing table of node k (N-nodes net)

Destination node	1	...	j	...	k-1	k+1	...	N
Next node	i_1	...	i_j	...	i_{k-1}	i_{k+1}	...	i_N

Routing is difficult because costs are dynamic

Adaptive routing is difficult because changes in the control policy determine changes in the costs and vice versa

Conventional Solution

Conventional routing often relies on:

- Global state available at all nodes
- Centralized control (scales badly!)
- Fixed "shortest path" (Dijkstra) algorithms
- Limited ability to deal with congestion or failure

Ideally, would like to have network adapt routing patterns to take advantage of free resources and move existing traffic if possible

Swarm Intelligent Networking

Ant Based Control (ABC; Shoonderwoerd et al., 1996)

- ABC is the first SI routing (load balancing) algorithm for telecommunications networks
- Introduced to route calls on a *circuit-switched* telephone network
- Test: UK telephone network

AntNet (DiCaro and Dorigo, 1998)

- Target: *packet-switching* network style (e.g. Internet)
- Test: more exhaustive on several networks

Swarm Intelligent Networking

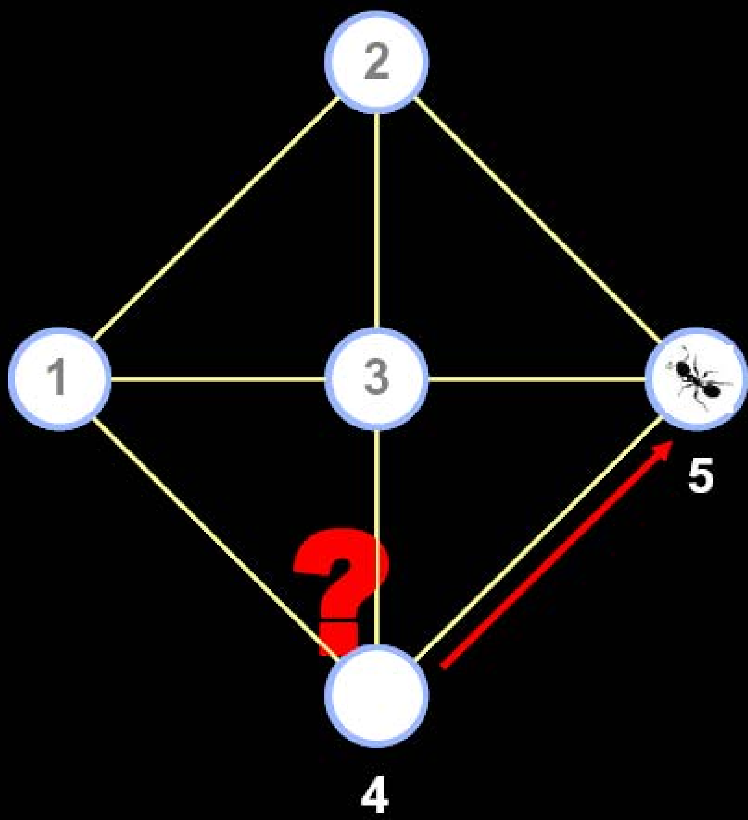
Main idea:

Populate network with artificial ants which leave a certain amount of simulated pheromone at each node they encounter. The amount of pheromone is a function of:

a) the congestion of the node
b) the distance the ant has traveled from the source

Ants select the next node on their journey on the basis of the local pheromone distribution

A Simple Network Example with 5 Nodes



Routing table of node 4

4

Destination Node

Neigboring Nodes	1	2	3	5
1	0,8	0,2	0,1	0,1
3	0,1	0,3	0,8	0,1
5	0,1	0,5	0,1	0,8

Ant destination
Updated table
Original table

Routing (pheromone) table is used by the ants to randomly explore the network; measures desirability

ABC Algorithm – Definitions

- *d* = destination; *s* = source; *n* = neighbor node
- Assumption: same level of traffic congestion *s->d* and *d->s* (OK for telephone networks)
- *N* nodes total; k_i = neighboring nodes of node *i*
- Routing table node *i* (time-variant matrix with k_i rows and *N-1* columns):

$$R_i = [p^i_{n,d}(t)]_{k_i,N-1}$$

$$p^i_{n,d}(t)$$

For ants: probability that an ant with destination *d* will be routed from *i* to *n*

For calls: deterministic path (pick up the higher value for choosing the route; operate independently from ants)

ABC – Routing Table Update

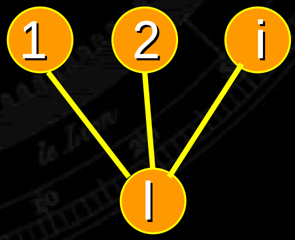
- Ants launched from any node continuously (at an optimal rate); travel from *s* to *d;* destination node *d* is random
- Ants die when they reach *d*
- For routing table updating: ant has only information about traffic at visited nodes, arriving at node *i* ant updates pheromone table entry of node *i* corresponding to source node *s*
- Update of routing table (=pheromone table) according to:

$$p_{i,l} = \frac{p_{i,l} + \Delta p}{1 + \Delta p}$$

Reinforce, *l*=node from which ant has just come

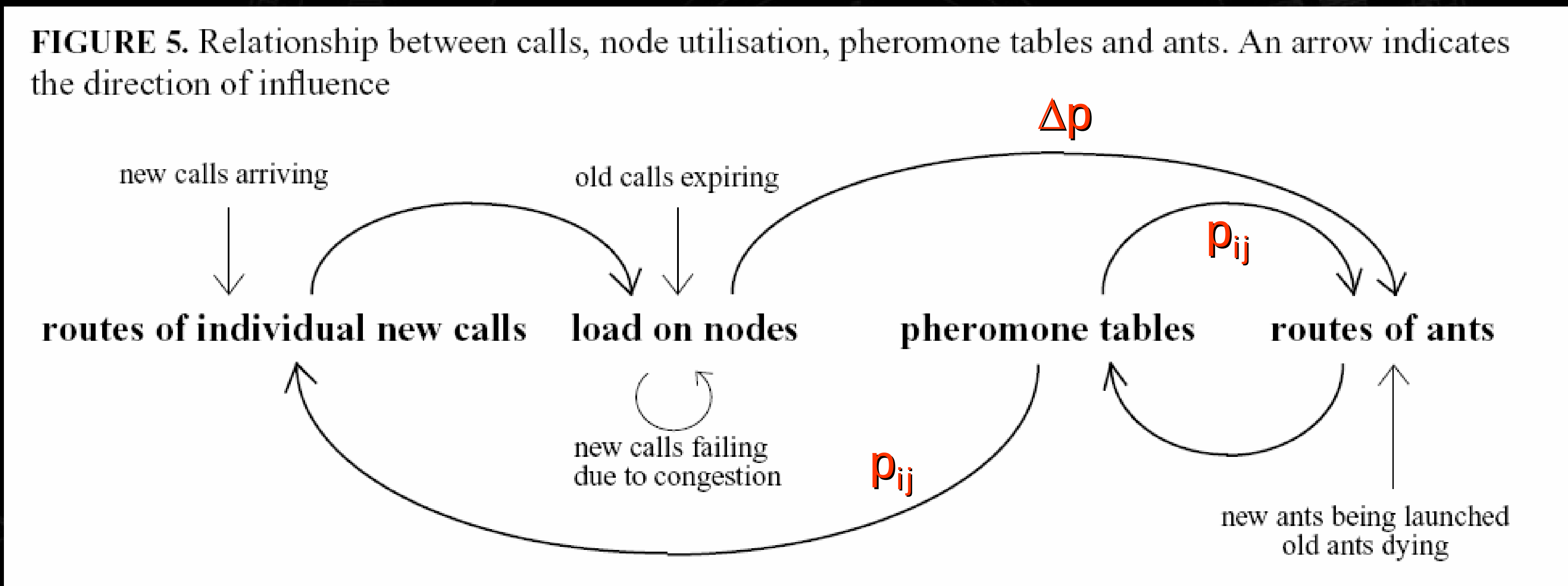
$$p_{n \neq i,l} = \frac{p_{n \neq i,l}}{1 + \Delta p}$$

Decay (evaporation)



Δp determined by age *T* of packet ($\Delta p = a/T + b$); ants have age: older ones influence routing less)

ABC – Interaction Between Calls and Ants



ABC – General Framework

1. Ants are regularly launched with random destinations on every part of the system
2. Ants walk randomly according to probabilities in pheromone tables for their destinations
3. Ants update probabilities in pheromone table for location they were launched from, by increasing probability of selection of previous location by subsequent ants
4. Increase of probabilities is a decreasing function of age of ant
5. Ant get delayed on parts of the system that are heavily used
6. Ants can be penalized or rewarded as a function of local system utilization
7. To avoid overtraining through freezing of pheromone trails, some noise can be added to behavior of ants

ABC – Sample Results

Call failure percentage with different algorithms – static call probabilities

- 30-nodes BT network
- 10 runs
- 15’000 time steps total

	Mean	Standard dev.
Without load balancing (fixed, shortest routes)	12.57%	2.16%
Original mobile agents	9.19%	0.78%
Improved mobile agents	4.22%	0.77%
Ants (0% noise)	1.79%	0.54%
Ants (5% noise)	1.99%	0.54%

Ant-Based Control

Other methods that build upon ABC:

- ANTNET
- Ant Routing based on the Ant System (AS)

Swarm Intelligent Networks: AntNet

AntNet = Net Routing

Communications networks are unpredictable

- Sudden interest in a particular web site or a local crisis will lead to surges of network activity
- Efficient traffic re-routing needed, minimising delays and congestion through quieter network sections

Congestion resembles food source depletion near an ant colony

- Ants must search for new routes, dynamically updating the virtual pheromone trail between nodes

[Di Caro, Dorigo] developed AntNet, an ant-based routing algorithm

- Outperformed all other routing methods

Applications to Industrial Problems

www.eurobios.com

- Number of different scheduling problems

www.antoptima.com

- Vehicle routing problems
  - Optimization of the distribution of heating oil with a nonhomogeneous fleet of trucks (Pina Petroli in CH)
  - Routing of hundreds of vehicles of Migros, Barilla, …

That's All Folks!

Artificial Life: Epilogue

"Artificial Life is the study of man-made systems that exhibit behaviors characteristic of natural living system. It complements the traditional biological sciences concerned with the analysis of living organisms by attempting to synthesize life-like behavior within computers and other artificial media. By extending the empirical foundation upon which biology is based beyond the carbon-chain life that has evolved on Earth, Artificial Life can contribute to theoretical biology by locating life-as-we-know-it within the larger picture of life-as-it-could-be." *(Langton, 1989)*

Artificial Life 2: Epilogue

Artificial Life is about the <u>synthetic study</u> of complex adaptive (life-like) systems (natural and artificial ones). The goal is not only to provide biological models but also to investigate general principles of life.

Artificial Life: Epilogue

Traditional biology (Science, in general) is analytic and top-down: starts from the top (e.g. organism level) and seeks for explanations in terms of lower level entities

Artificial life is synthetic and bottom up: starts at the bottom (e.g. molecular level) and works its way by synthesizing complex systems from many simple interacting entities

Artificial Life: Epilogue

Artificial life studies life-as-it-is and as-it-might-be

Artificial Life is:

- Artifactual (man-made), but *not* unreal
- Bottom-up, not top-down
- Synthetic, not analytical
  - Top-down analysis informs and tests Alife research. But bottom-up synthesis plays a larger role than in more traditional sciences (synthetic methodology)
- Leverages *emergence*

Goals:

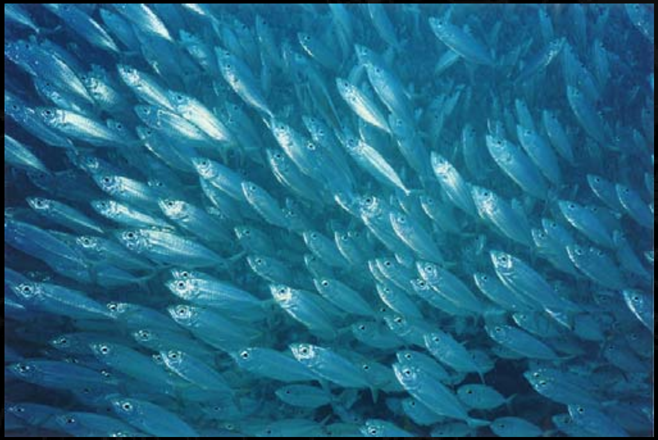
1) **understand complex adaptive (living) systems** – e.g., brains, organisms, insect colonies, ecosystems, economies, etc.
2) **use principles derived from this study to design new, successful complex adaptive systems** – e.g., self-configuring robots, smart structures, swarms of unmanned vehicles, self-organized networks, etc.

Epilogue: Pattern Formation

Pattern = organized arrangement of objects in space and time

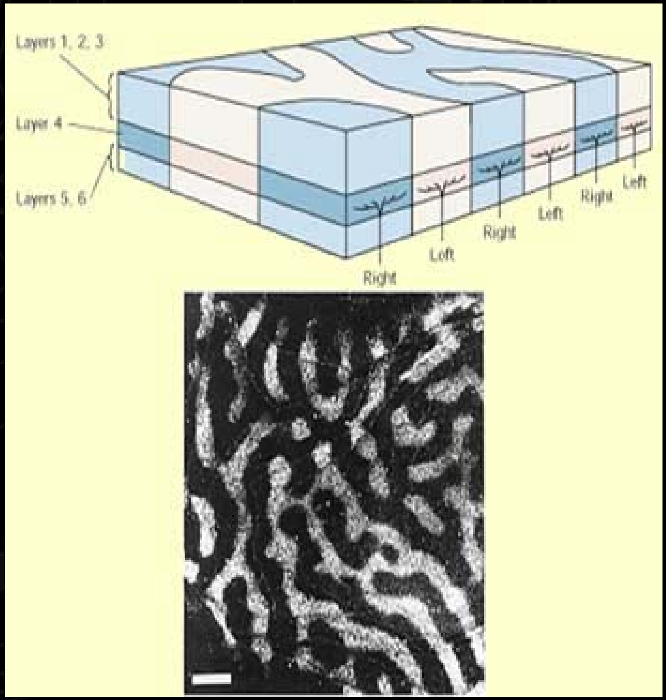
Examples of biological patterns:

- Schools of fish
- Raiding column of army ants
- Ocular dominance stripes in visual cortex
- Synchronous flashing of fireflies
- Pigmentation patterns on shells



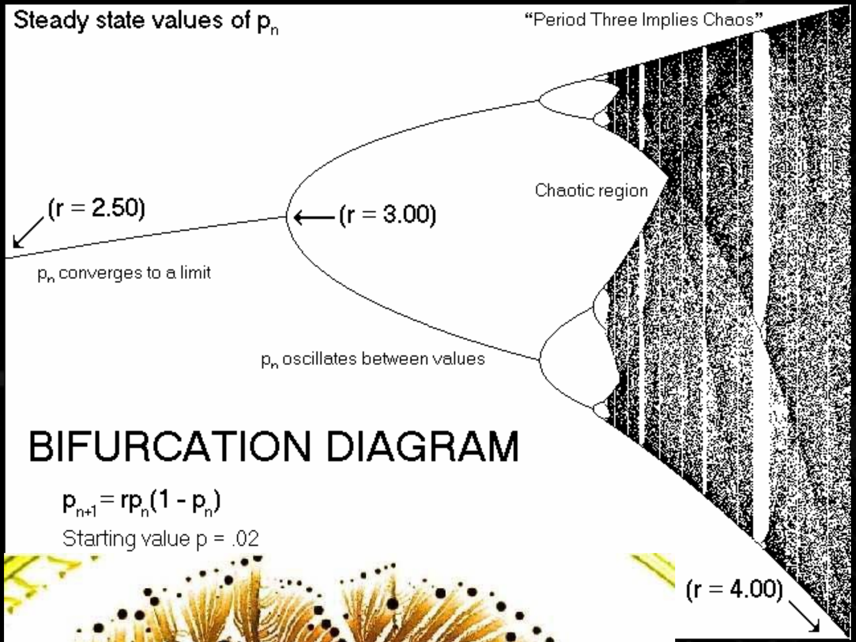
school of fish

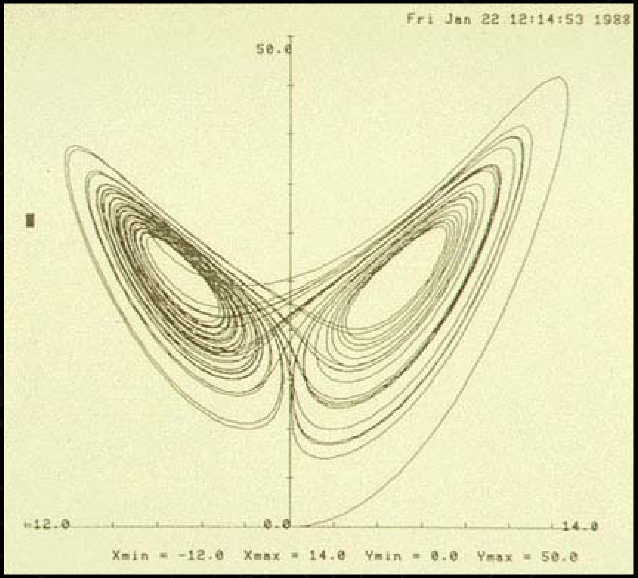
fireflies

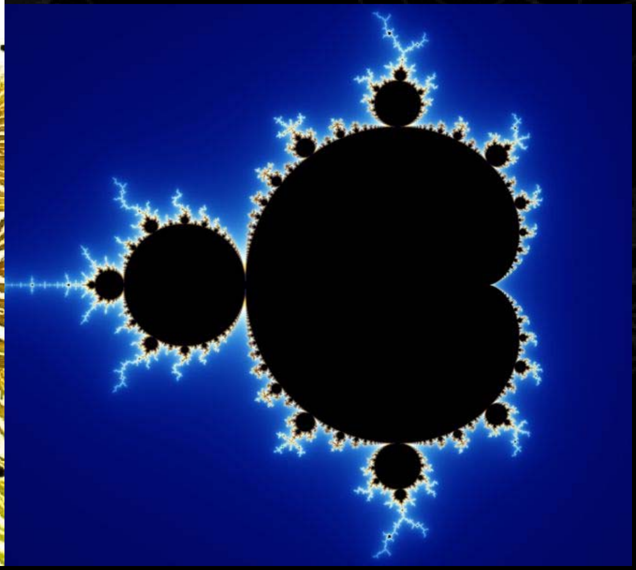


inputs from left eye (black) and right eye (white)

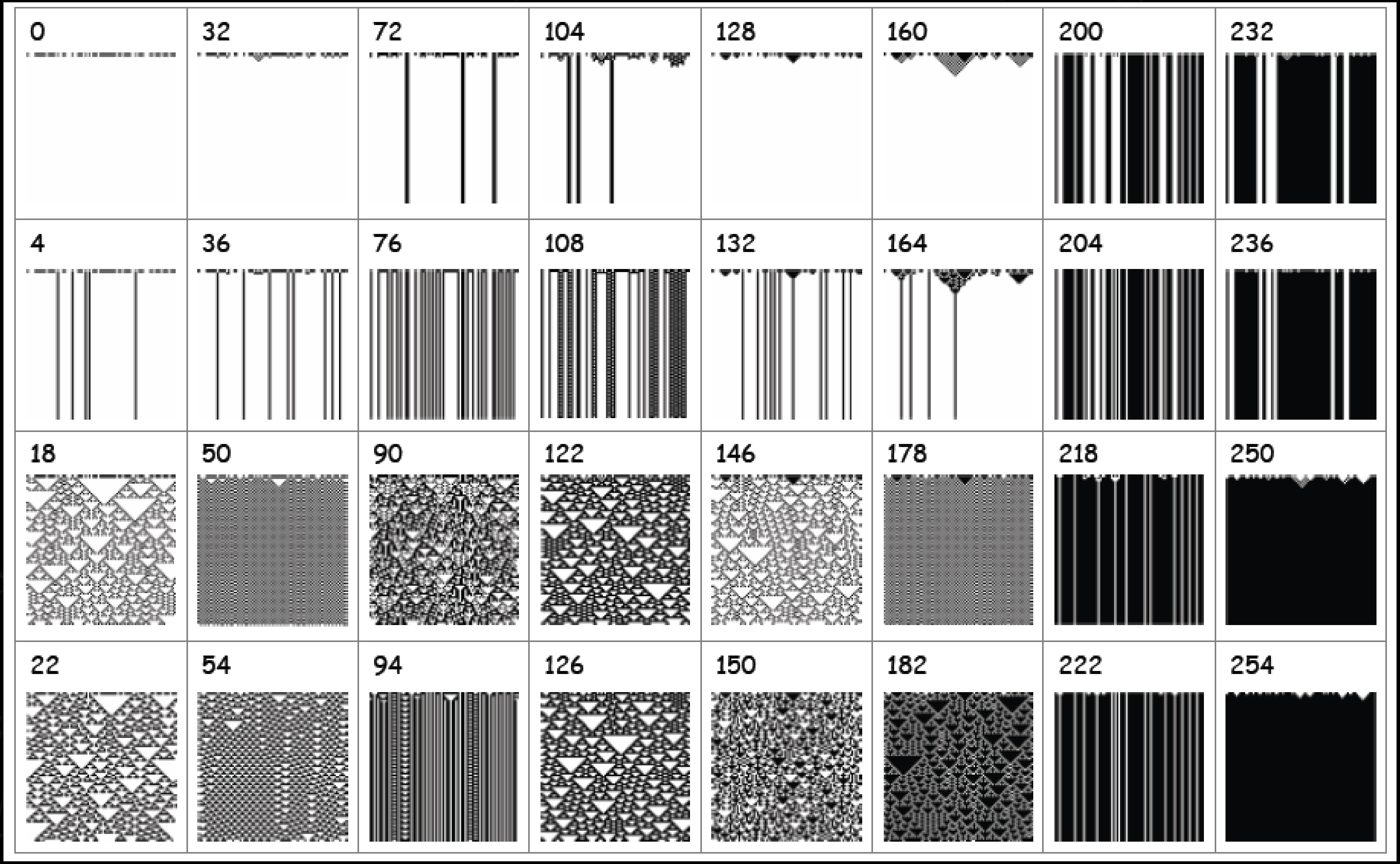
Epilogue: Dynamical Systems



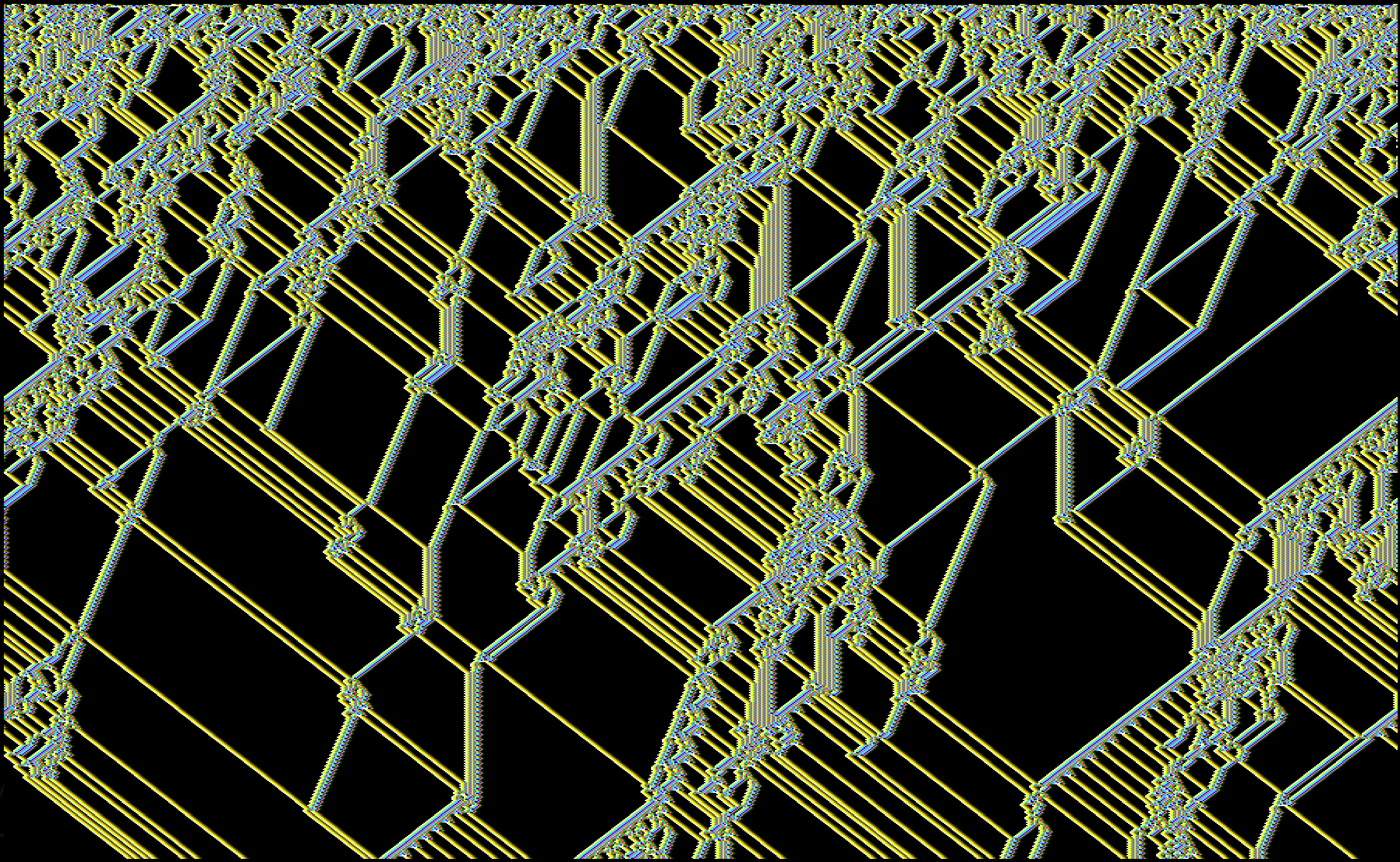




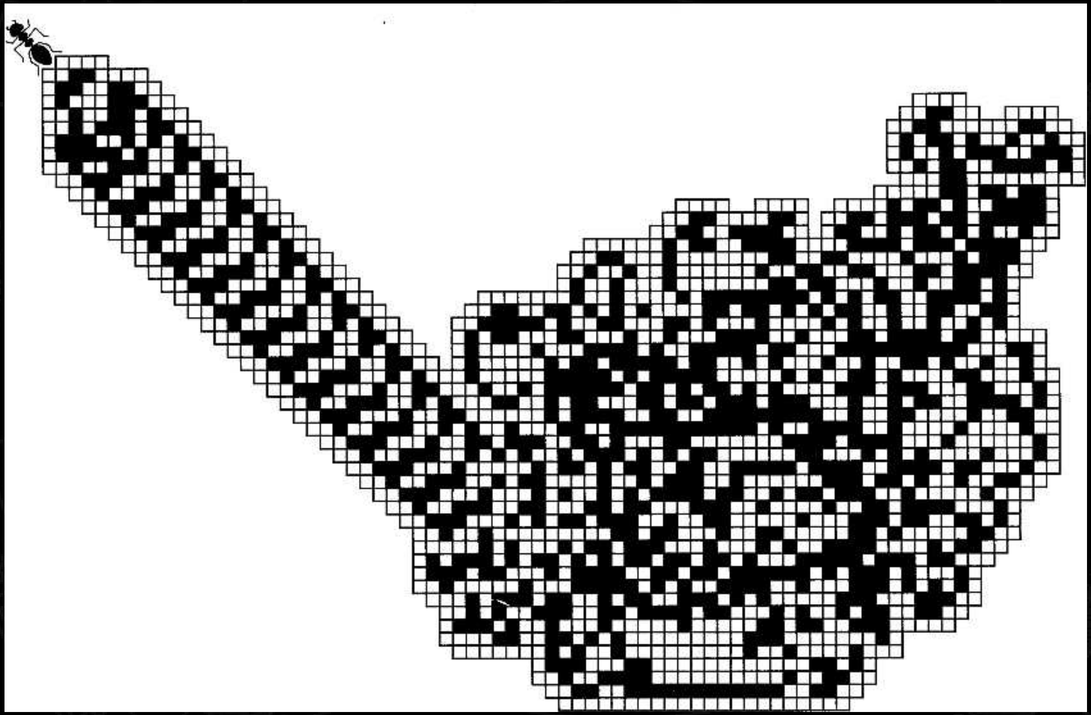
Epilogue: Cellular Automata



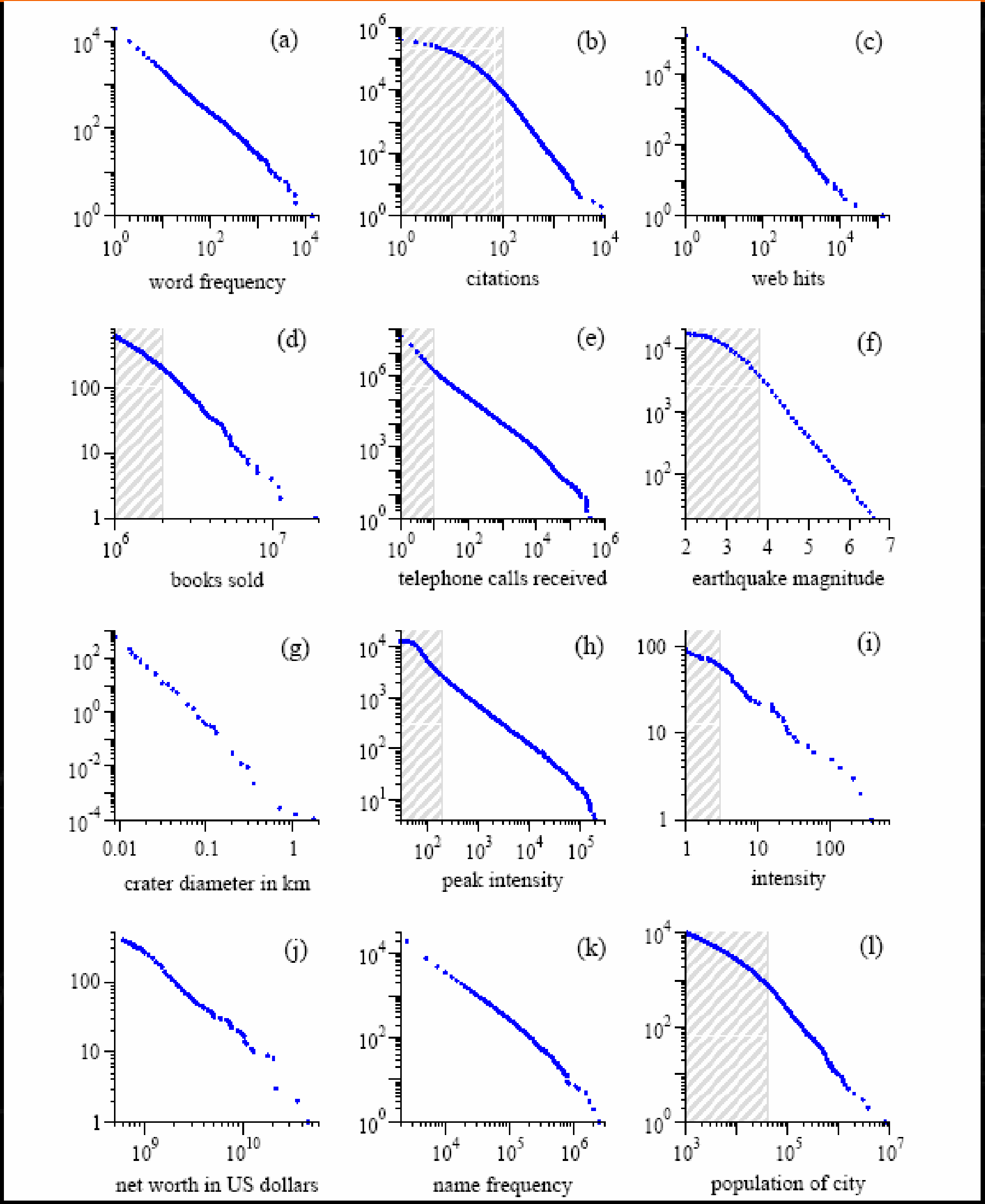
Epilogue: Cellular Automata

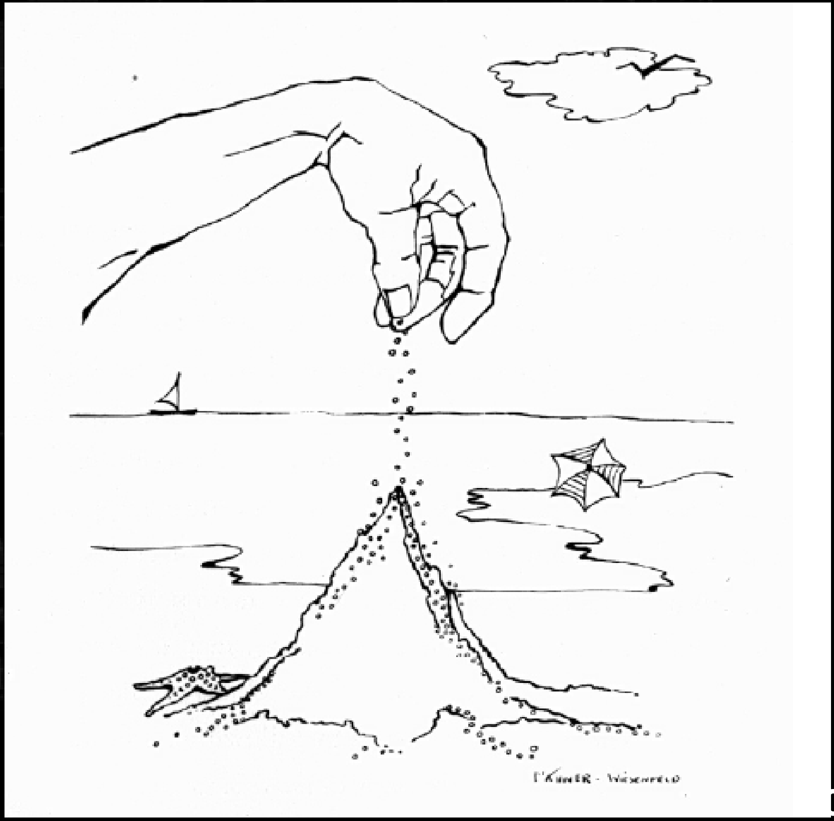


Epilogue: Cellular Automata

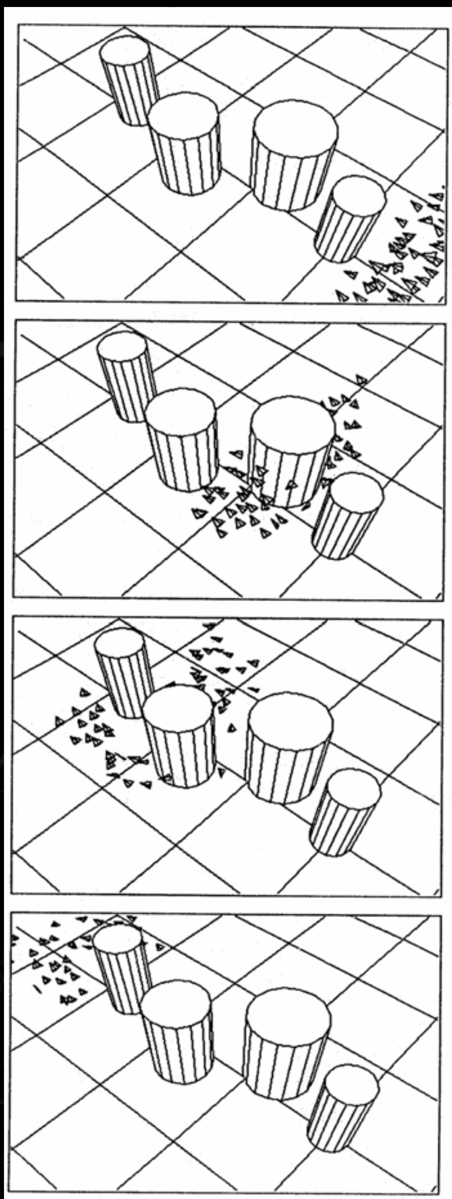


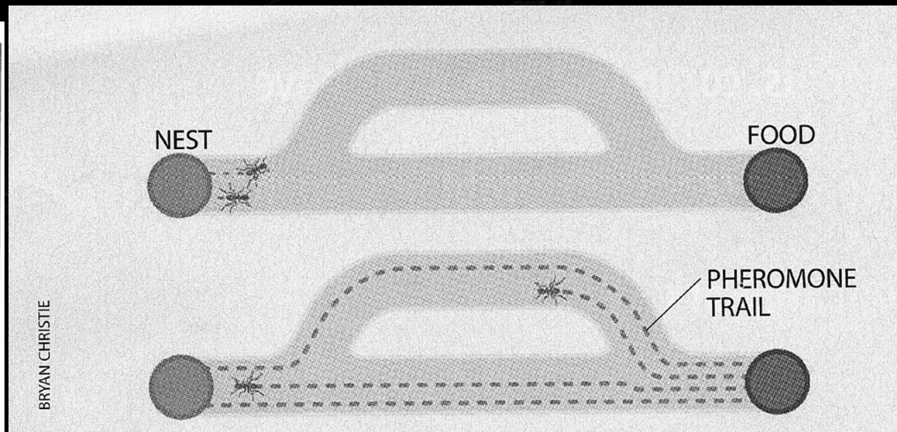
Epilogue: Power Laws and Sanpiles

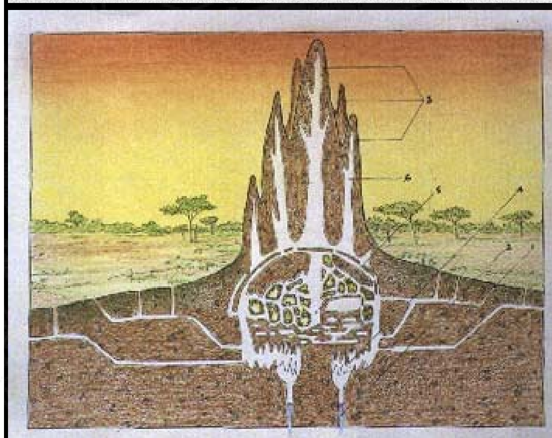


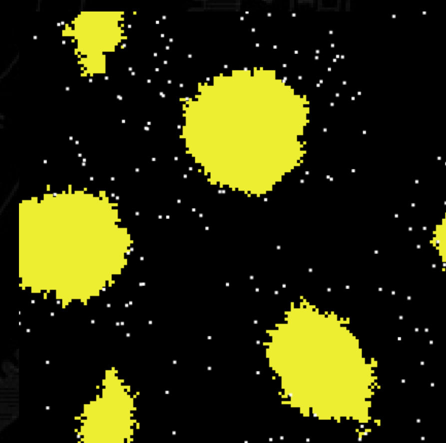


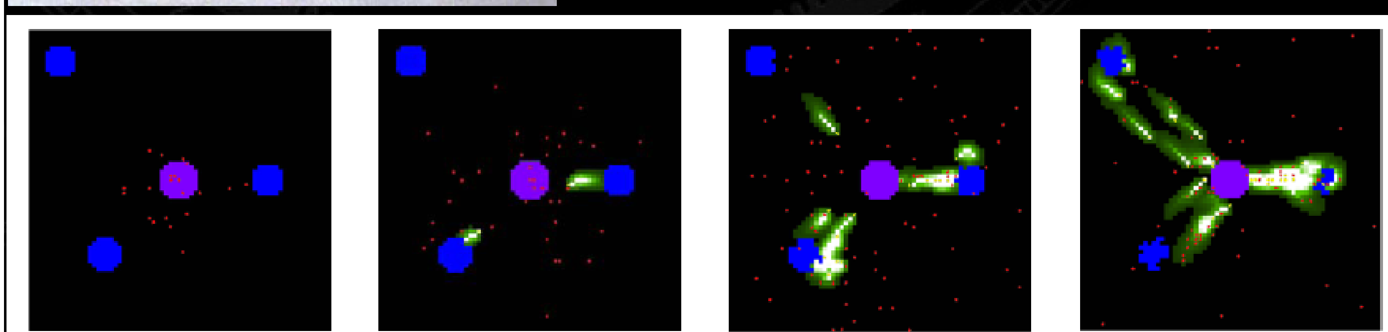
Epilogue: Swarms and Collective Intelligence



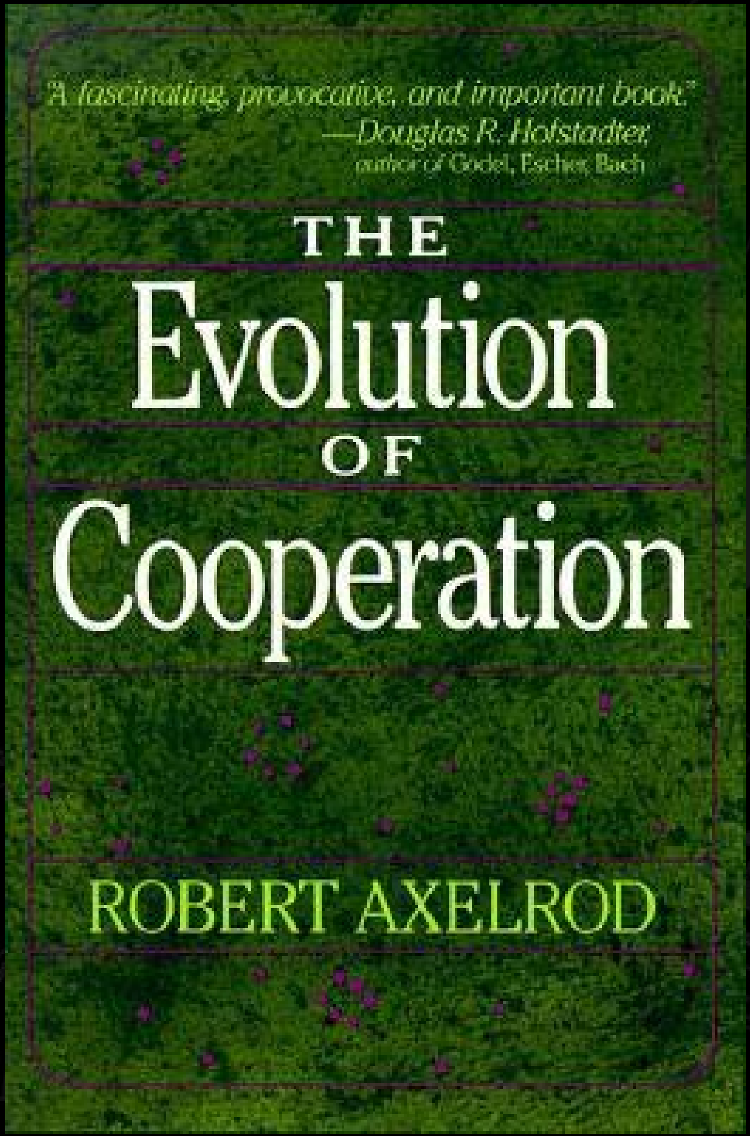








Epilogue: Evolution of Cooperation



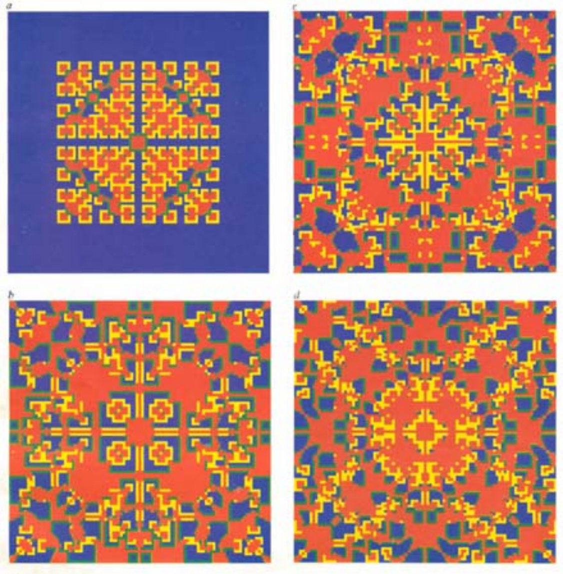
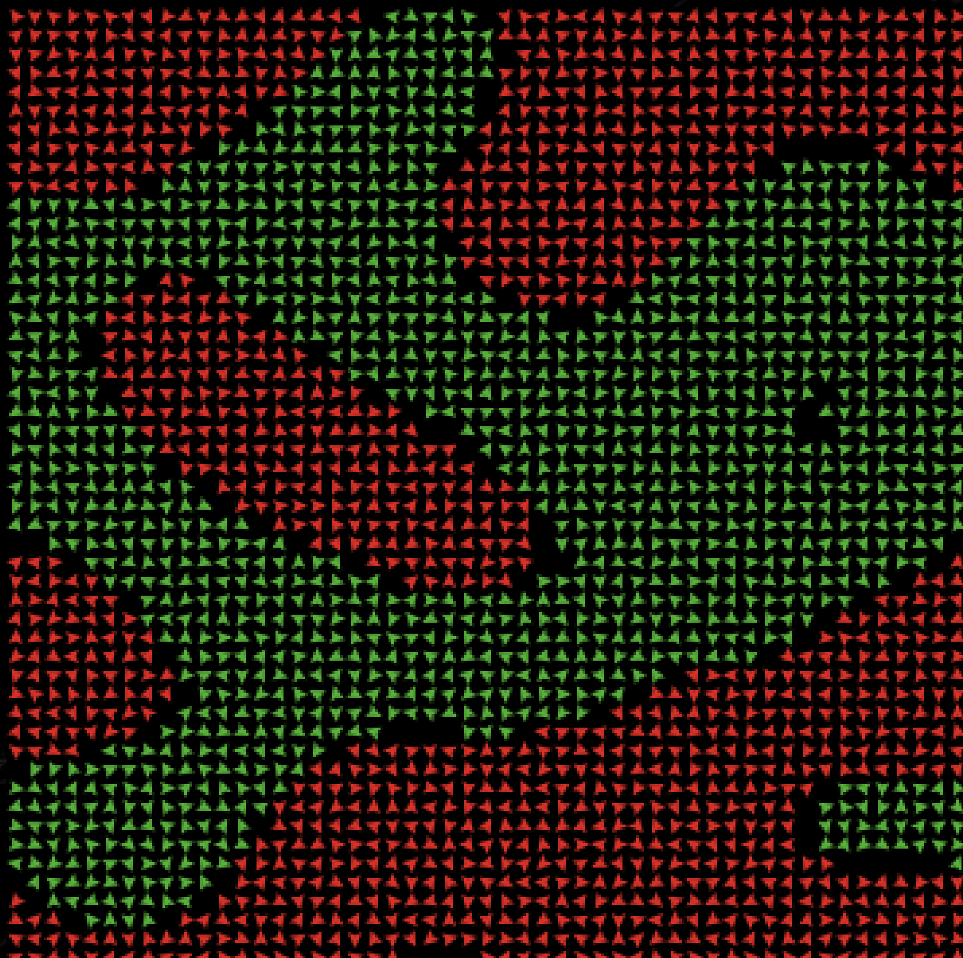


FIG. 3 Spatial games can generate an 'evolutionary kaleidoscope'. This simulation is started with a single D at the centre of a 99 × 99 square-lattice world of C with fixed boundary conditions. Again $1.8 < b < 2$. This generates an (almost) infinite sequence of different patterns. The initial symmetry is always maintained, because the rules of the game are symmetrical. The frequency of C oscillates (chaotically) around a time average of 12 log 2-8 (of course). a, Generation $t = 30$; b, $t = 217$; c, $t = 219$; d, $t = 221$.

Epilogue: Artificial Societies



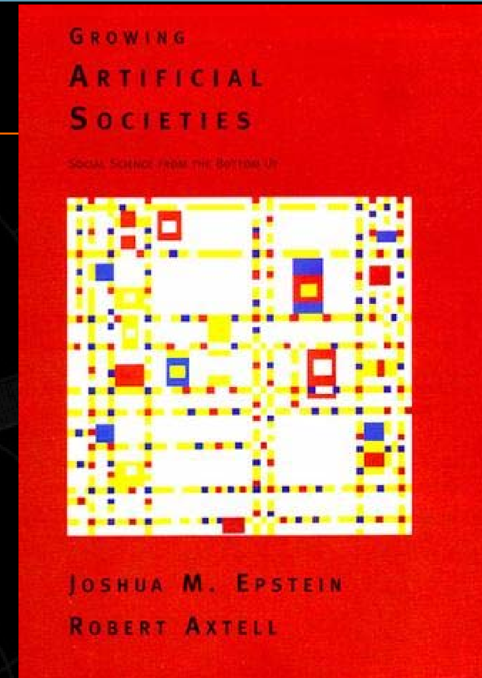
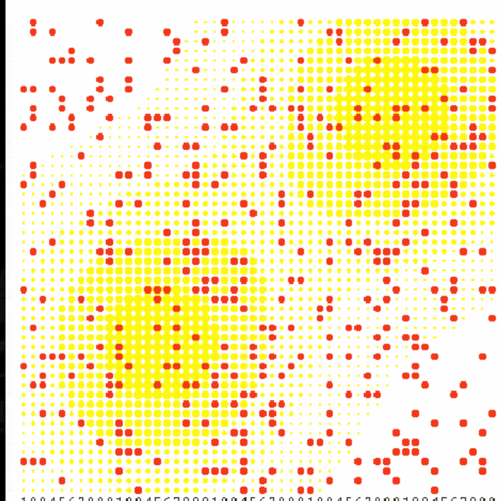
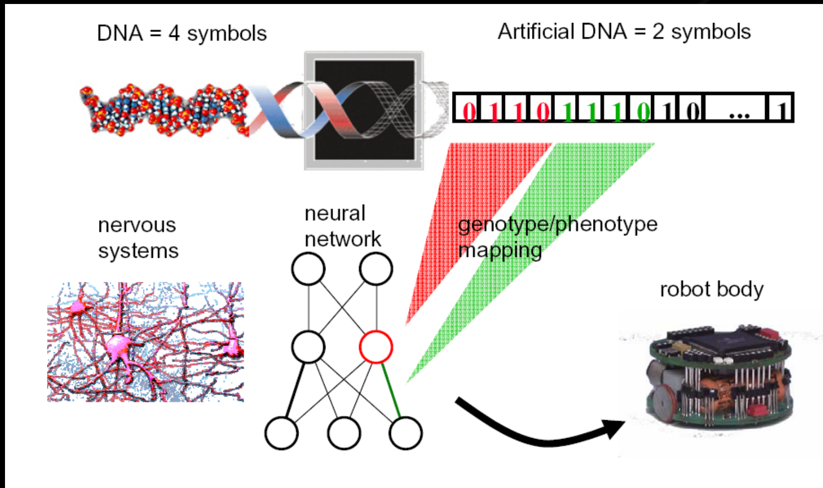
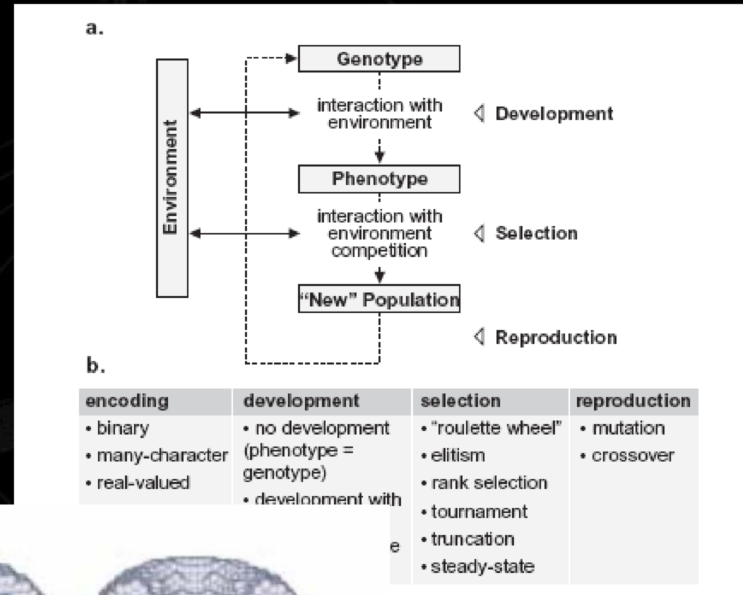


Figure II-2. Sugarscape with Agents



Epilogue: Artificial Evolution





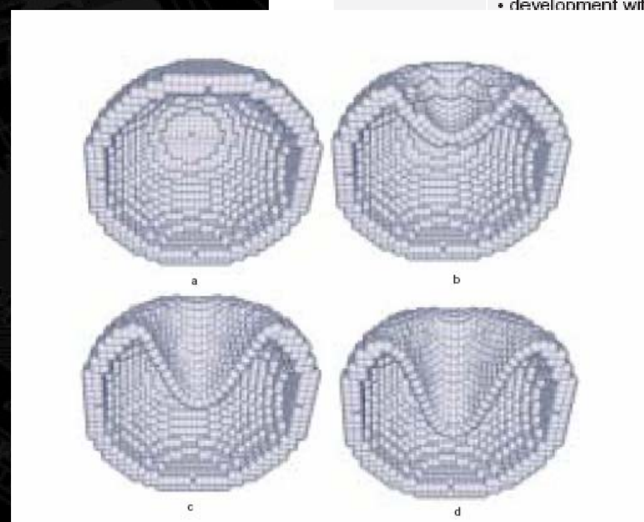
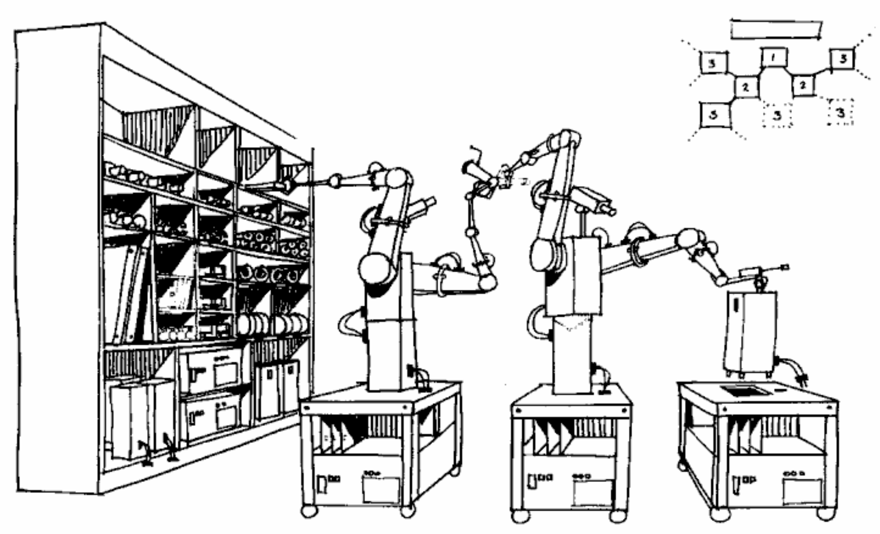


Figure 7: Four stages of simulated invagination. Cells were positioned spherically and the task was to evolve a mechanism able to invaginate a part of the sphere as illustrated in the above sequence of the dynamics of this developmental process.

Epilogue: Self-Replication



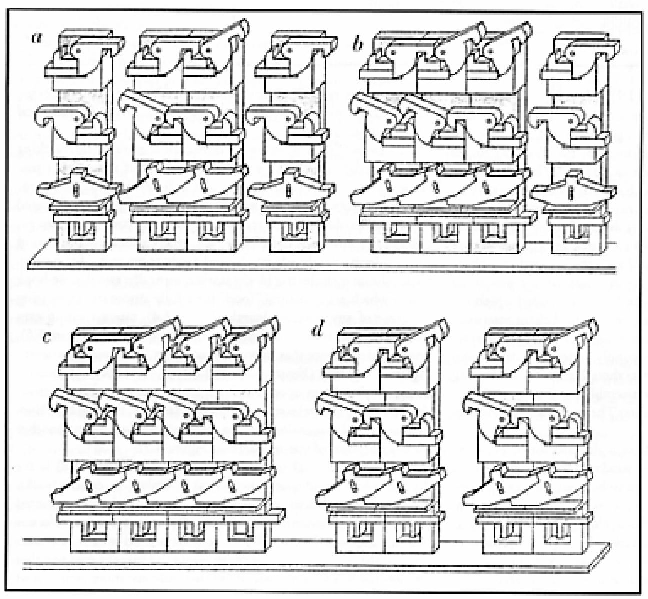


Figure 3.8. One replication cycle of the Penrose block replicator, from Penrose.[483]

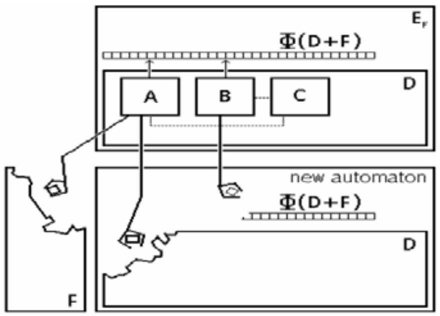
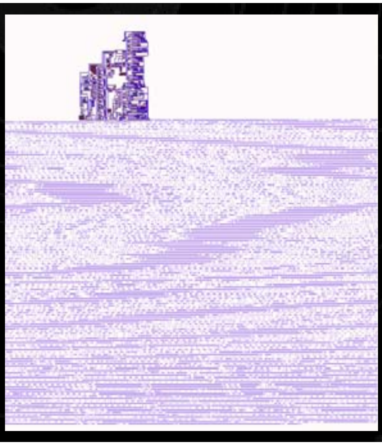
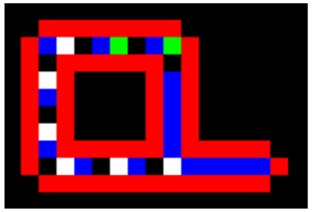


Figure 2: *Schematic view of von Neumann's theoretical self-replicating machine. A: general construction machine; B: general copying machine; C: control machine.*



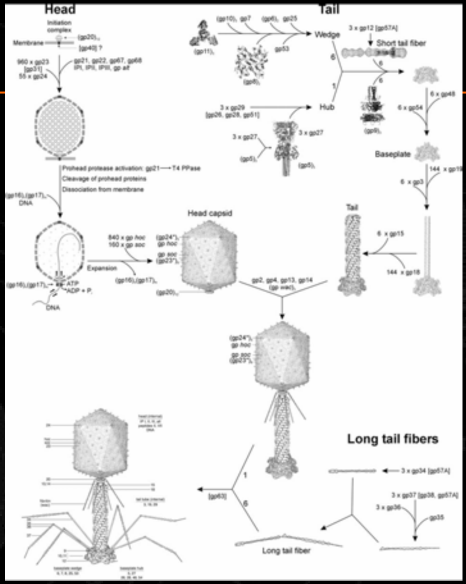


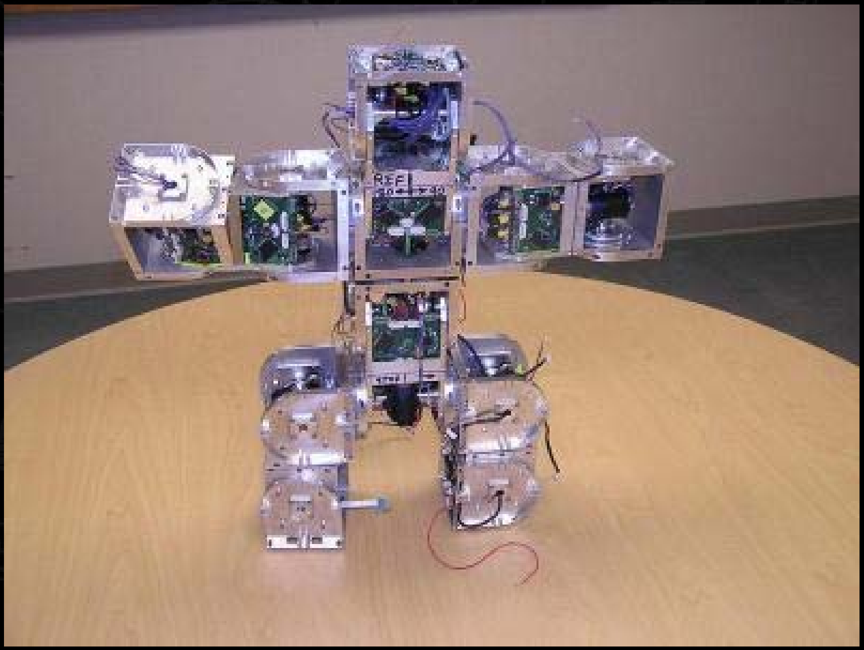
```
  2 2 2 2 2 2 2 2
2 1 7 0 1 4 0 1 4 2
2 0 2 2 2 2 2 2 0 2
2 7 2         2 1 2
2 1 2         2 1 2
2 0 2         2 1 2
2 7 2         2 1 2
2 1 2 2 2 2 2 2 1 2 2 2 2 2
2 0 7 1 0 7 1 0 7 1 1 1 1 1 2
  2 2 2 2 2 2 2 2 2 2 2 2 2
```

Figure 4: *Initial configuration of a Langton Loop. Note the red "sheath" (state 2) that protects the flowing information. The data flows round the loop anti-clockwise. It is copied at the junction of the arm and one copy is sent round the loop, to keep the structure "alive", and the other copy is sent down the arm to be interpreted into extending and bending it.*

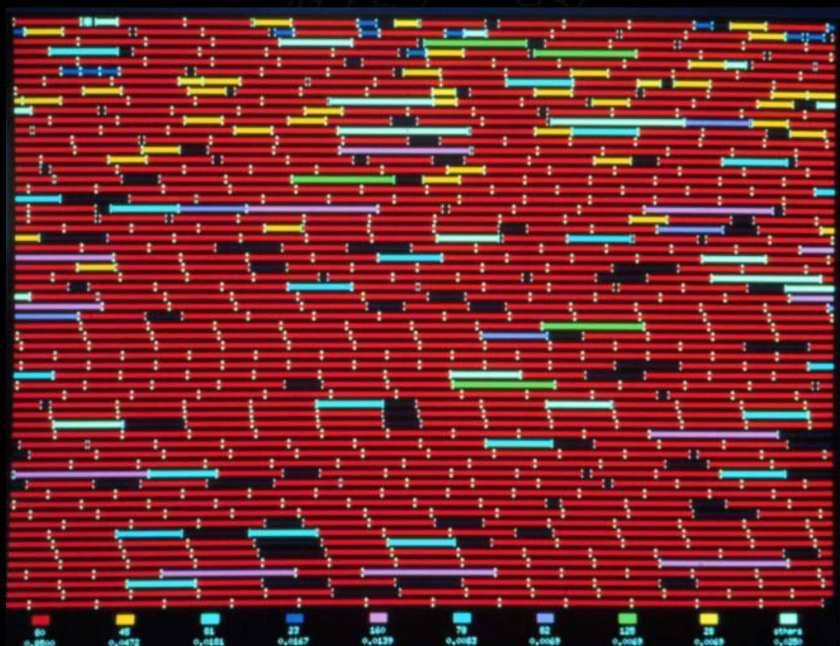
Epilogue: Self-Assembly

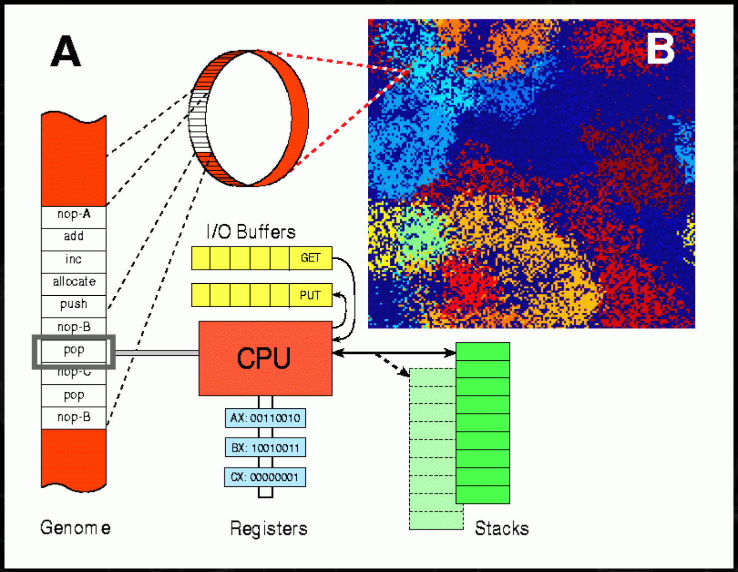
Fig. 1. Four programmable parts partially assembled into a triangle. The parts bind upon random collisions and communicate via IR, deciding whether to remain bound or to detach. A graph grammar stored on the microcontroller of each part determines the ultimate global structure that will emerge. The parts are not self-motive but instead are "mixed" on an air table by overhead oscillating fans.



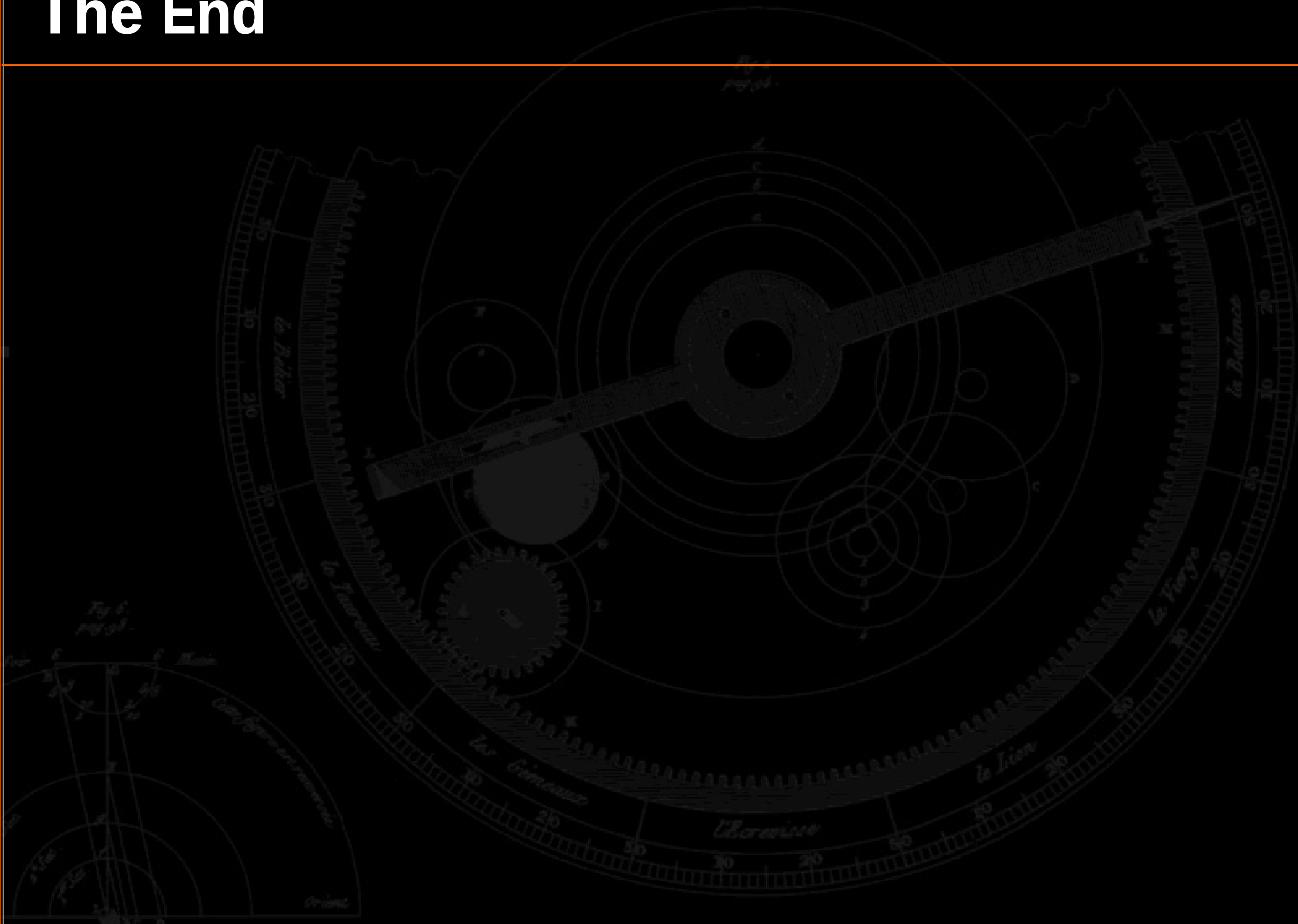


Epilogue: Computational Ecologies





The End


le Belier
le Taureau
les Gemeaux
l'Ecrevisse
le Lion
la Vierge
la Balance