ECM3412/ECMM409
Nature Inspired Computation
Lecture 10

Ant Colony Algorithm Applications

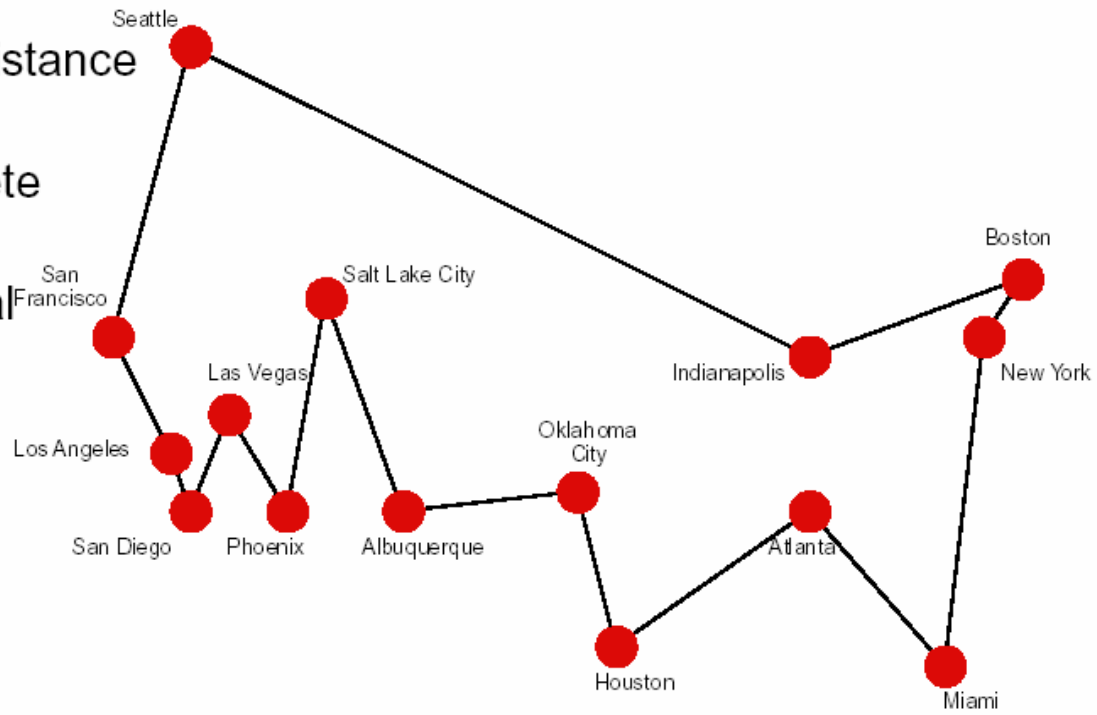
The Traveling Salesperson Problem: A reminder

Problem: given N cities, and a distance function d between cities, find a tour that:

(1) goes through every city once and only once

(2) minimizes the total distance

- Problem is NP-complete
- Classical combinatorial optimization problem to test algorithms



The *Ant System* Algorithm

Loop

Randomly position *num_ant* ants on *num_city* cities

For step := 1 to *num_city*

For k := 1 to *num_ant*

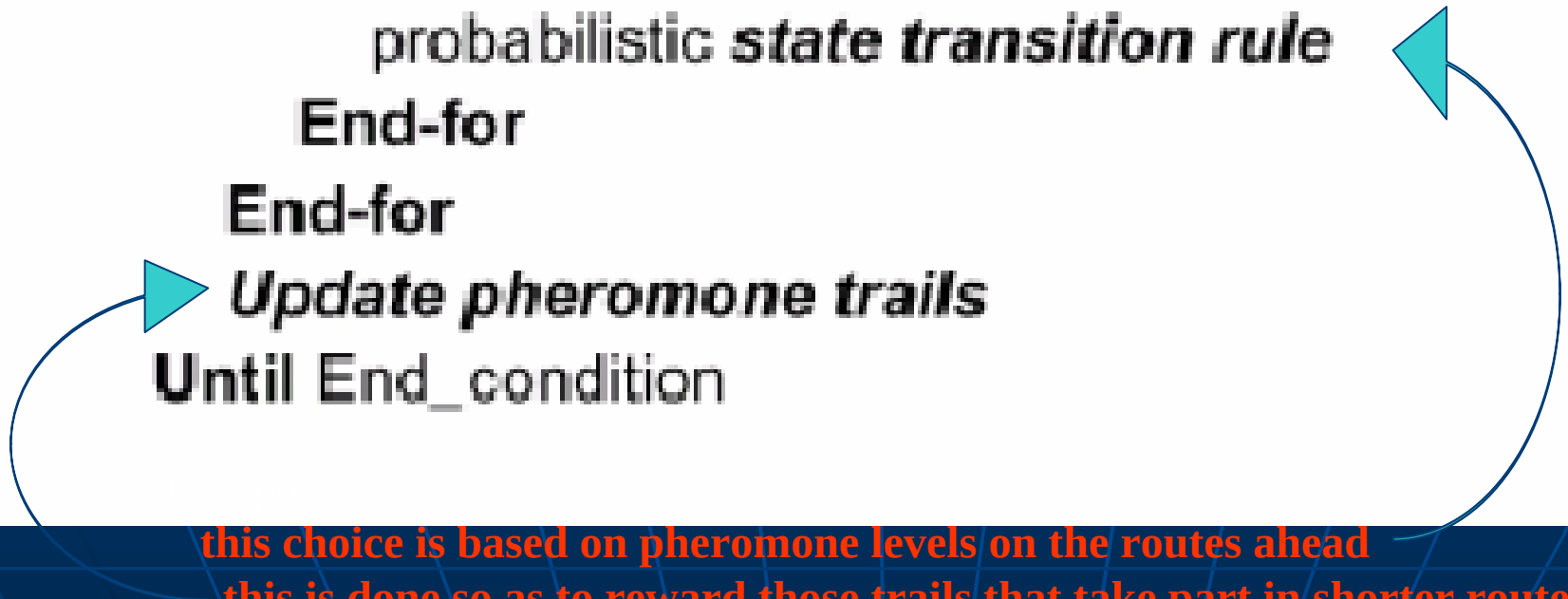
Choose the next city to move to applying a probabilistic ***state transition rule***

End-for

End-for

Update pheromone trails

Until End_condition



this choice is based on pheromone levels on the routes ahead

this is done so as to reward those trails that take part in shorter routes

About the transition rule

For each ant the transition from city i to city j at iteration t depends on:

- Has the city been visited? Each ant k maintains a tabu list in memory that defines the set J_i^k cities still to be visited when at city i.
- A (static) local heuristic $\eta_{ij} = 1/d_{ij}$ that represents the desirability of visiting city j when in city i.
- The (learned, global) amount of virtual pheromone on trail (i,j) $= \tau_{ij}(t)$.

The Transition Rule

- The probability for ant k to go from city i to city j while in its t^{th} tour is:

$$p_{ij}^{k}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}(t)\right]^{\beta}}{\sum_{h \in J_i^k} \left[\tau_{ih}(t)\right]^{\alpha} \cdot \left[\eta_{ih}(t)\right]^{\beta}}$$

- If j has been visited then p() =0
- Note α β are parameters – trades off global vs local information
- If α is small then the closest cities are favored – classic greedy algorithm
- If β is small only pheromone +ve feedback at work – may choose non-optimal paths too quickly

The Update Rule

- After the completion of a tour each ant k lays a quantity of pheromone on each edge (i,j) that it has used.
- The amount is equal to Q/L (fitness)
- Also, Pheromone evaporates: $\tau \rightarrow (1-\rho)\,\tau$
- Rule:

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\cdot\tau_{ij}(t) + \Delta\tau_{ij}(t)$$

where

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{n} Quality^{k}$$

$Quality^{k}$, in the TSP case, can be set to the inverse of the length of the solution found by ant k

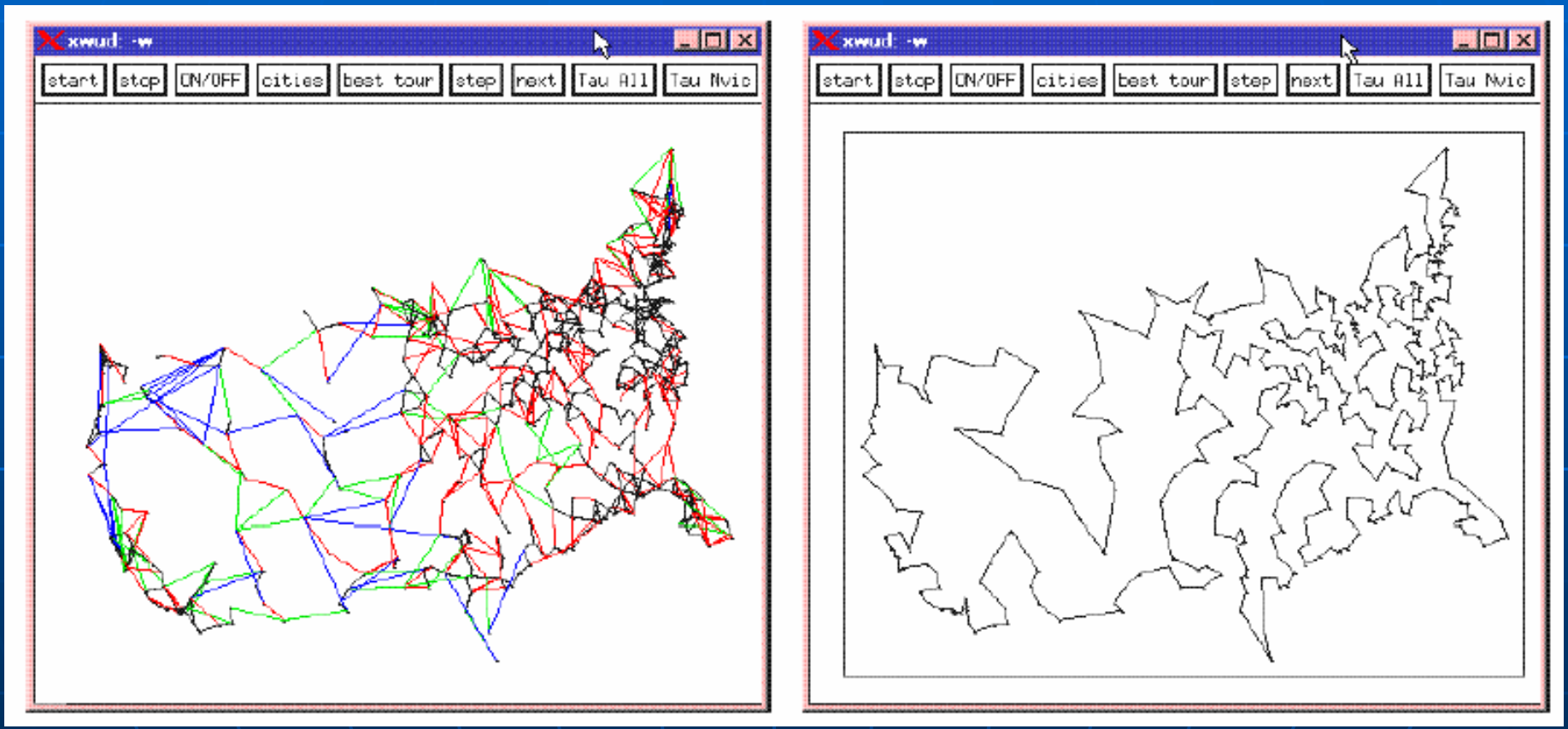
Early Ant Results

- Algorithm found best solutions on small problems (30 city)
- On larger problems converged to good solutions – but not the best
- On "static" problems like TSP hard to beat specialist algorithms
- Ants are "dynamic" optimizers – should we even expect good performance on static problems
- Coupling ant with local optimizers gave world class results….

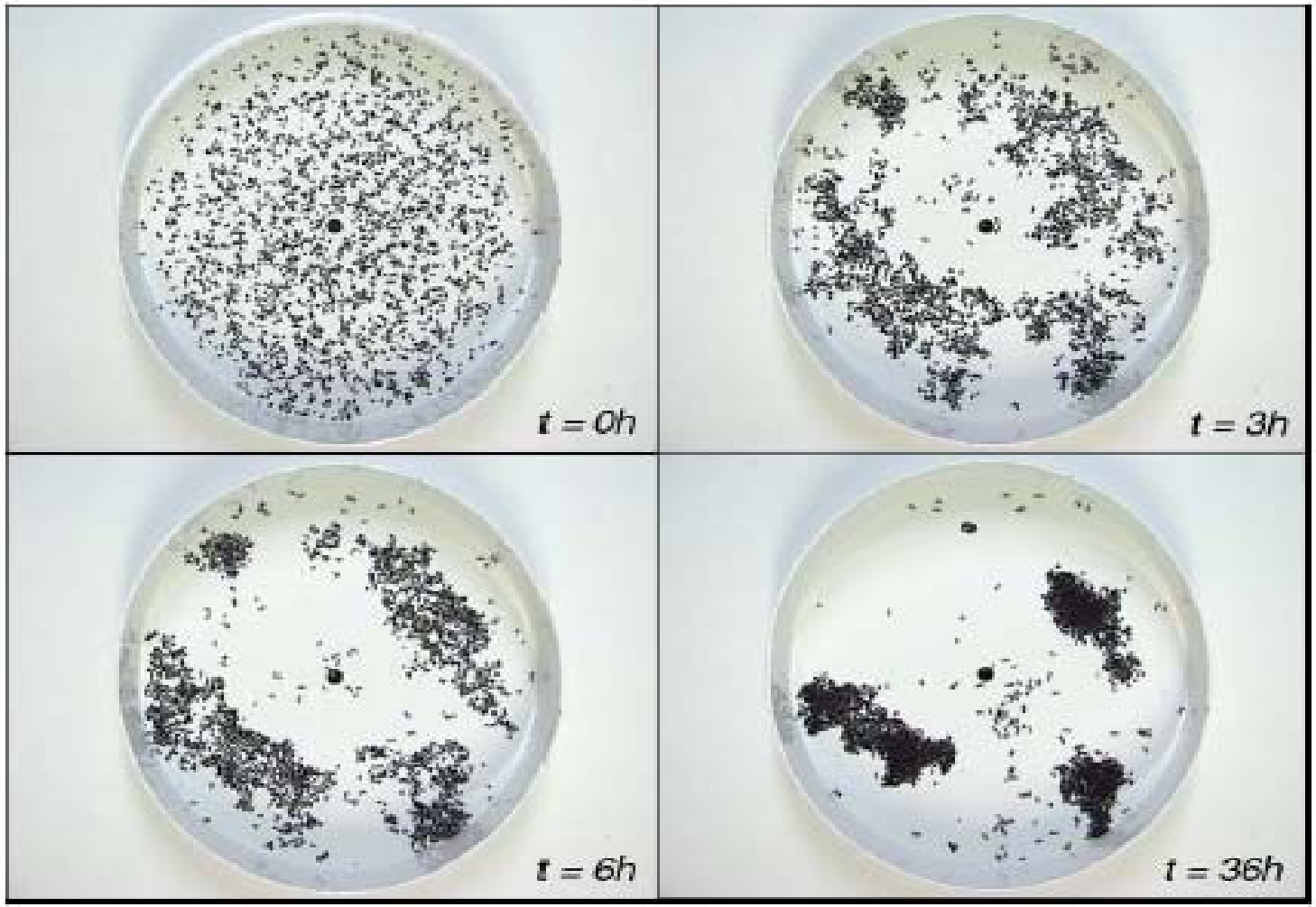
ACS ran for 1250 iterations (end criterion) using 20 ants; average over 15 runs for all implementations; see course book for more results

Problem	Ant Colony System (Dorigo & Gambardella,1997)		Genetic Algorithms (Whitley *et coll.*, 1989)		Simulated Annealing (Lin *et coll.*, 1993)	
	Shortest Tour Lenght	Number of tours	Shortest Tour Lenght	Number of tours	Shortest Tour Lenght	Number of tours
Eil50 (50 cities)	425	1830	428	25000	443	68512
Eil75 (75 cities)	535	3480	545	80000	580	173250
KroA100 (100 cities)	21282	4820	21761	103000	N/A	N/A

An ant route


start
stop
ON/OFF
cities
best tour
step
next
Tau All
Tau Nvic
start
stop
ON/OFF
cities
best tour
step
next
Tau All
Tau Nvic

Ants do Clustering!



Corpse aggregation in *Messor Sancta*

Real Ants

- Real ants peform clustering in 2 ways:
- Corpse Aggregation (previous slide)
  - This is where an ant will pick up a lone corpse and transfer it to an ant ‘cemetery’
- Brood Sorting
  - Ants sort their brood according to the level of maturity of the young.
    - Small eggs get clustered with other small eggs, larvae with other larvae etc…

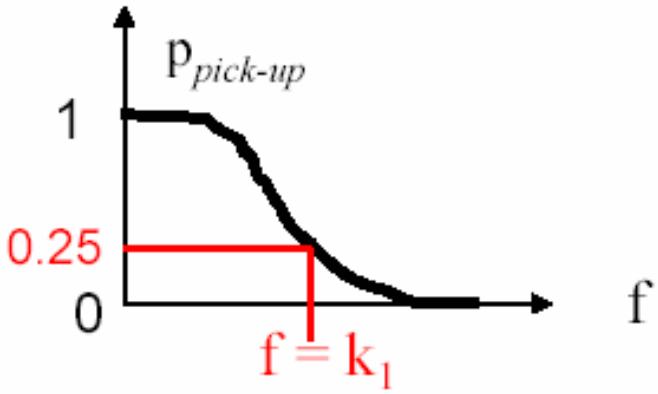
How corpse aggregation seems to work at the individual level

- When an ant encounters a corpse, it will pick it up with a probability which increases with the degree of isolation of the corpse
- When an ant is carrying a corpse, it will drop it with a probability which increases with the number of corpses in the vicinity
- Modulation of pick up/drop probabilities as a function of the pheromone clouds around the cluster -> quantitative (continuous) stigmergy

Turning that into an Algorithm

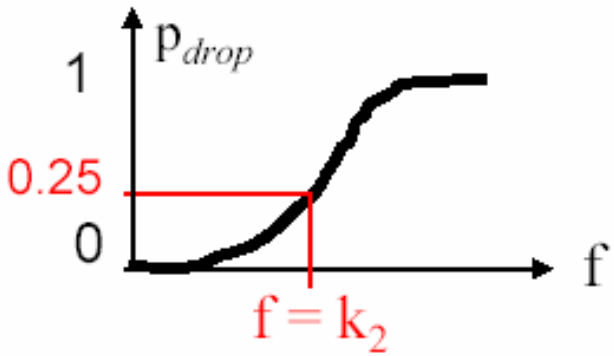
The probability that an agent which is not carrying an item will pick up an item

$$p_{pick\text{-}up} = \left(\frac{k_1}{k_1 + f}\right)^2$$



Probability that an agent carrying an item will drop the item

$$p_{drop} = \left(\frac{f}{k_2 + f}\right)^2$$



f: fraction of neighborhood sites occupied by object o (perceived stimulus)
k_1, k_2: threshold constants

From Aggregation to Sorting

- A sorting problem implies two or more objects to be manipulated
- Sorting = building clusters consisting of the different type of objects involved in the problem

Individual behavioral algorithm

Probability of picking up an object of type i:

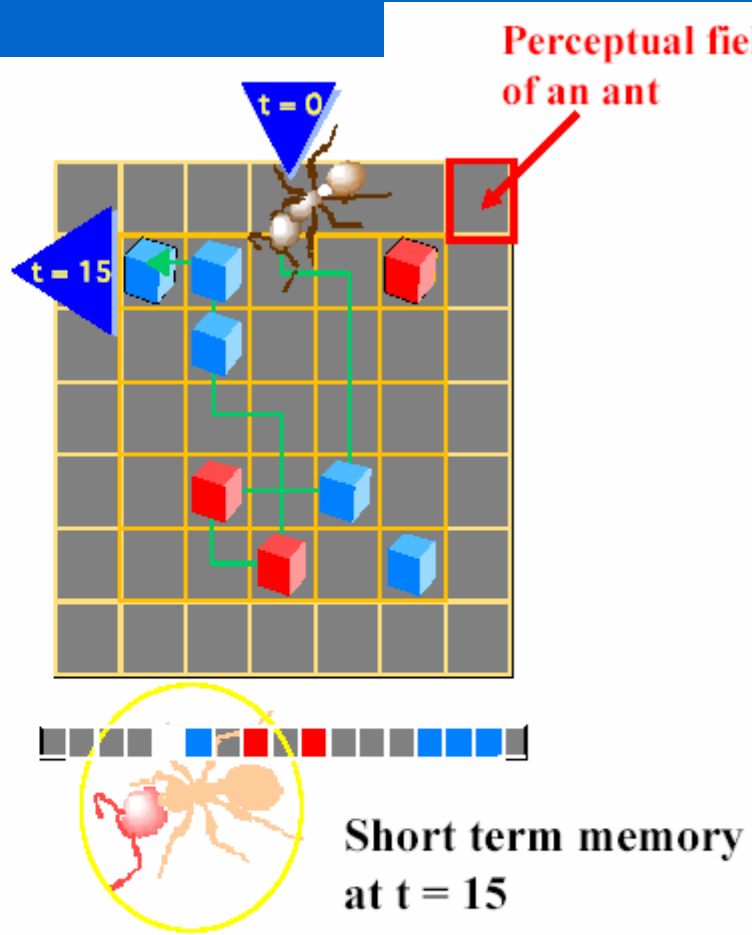
$$p_{pick\text{-}up}(o_i) = (k_1 / (k_1 + f(o_i)))^2$$

Probability of dropping an object of type i which is being carried:

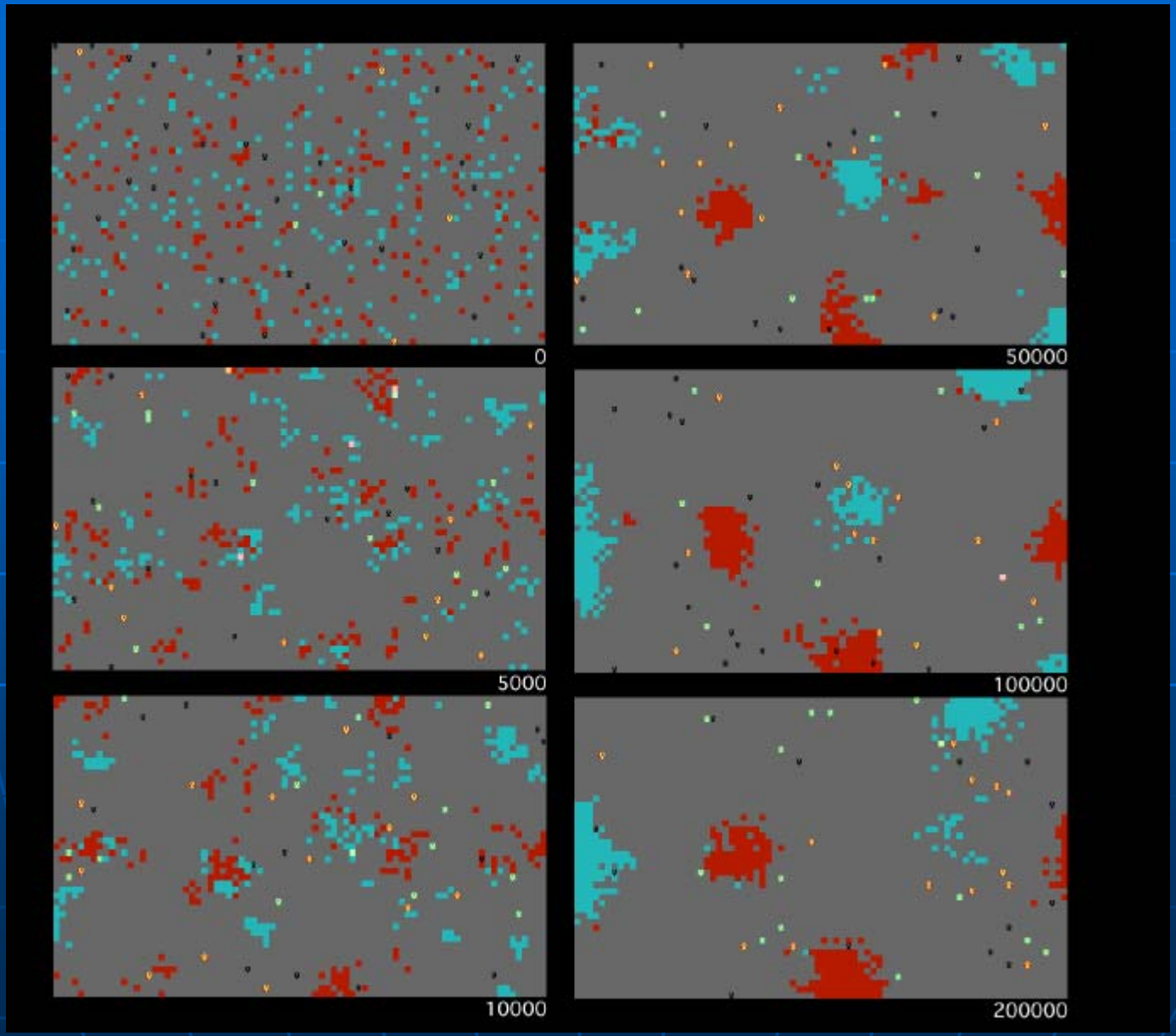
$$p_{drop}(o_i) = (f(o_i) / (k_2 + f(o_i)))^2$$

$f(o_i)$: fraction of neighboring sites occupied by objects of the same type of the object o_i

k_1, k_2 : threshold constants



In this algorithm $f(o_i)$ are estimated based on a short-term memory tuned to the underlying random walk!


0
50000
5000
100000
10000
200000

A basic ant-based clustering algorithm

- Objects to cluster are placed randomly in cells in a 2D space
- A number of ants are spread randomly in the same space

Repeat Until *a termination condition*
  For each ant *A*
    If *A* is carrying an object, it will either:
      drop the object at its current cell, with
        probability P_{drop}
      make a move to a random neighbouring cell
    If *A* is not carrying an object, it will either:
      pick up an object
      make a move to a random neighbouring cell.

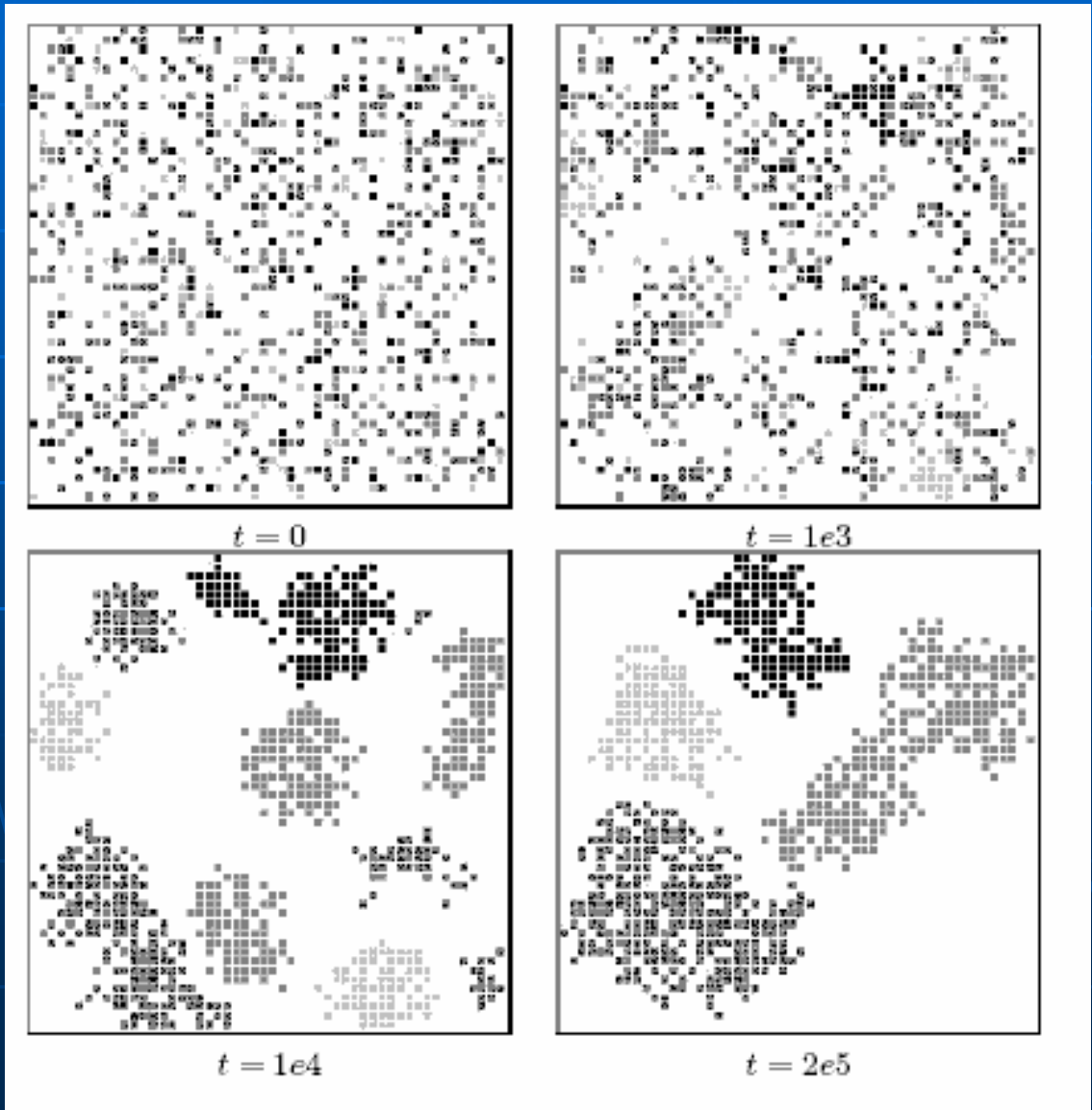
Various ways to turn this into a real algorithm by playing with the probability equations for picking up and dropping, etc.

Further Improvements

- Stagnation Count:
  - Problem: If probability of drop is too low – many ants spend a long time walking around
  - Solution: The longer the ant carries the object, the more inclined it is to release it until a point where it drops it regardless of location
- Adaptive Similarity Scaling:
  - Problem: If similarities are not strong enough, clustering can take time.
  - Solution: Introduce an adaptive measure which increases/decreases the likelihood of dropping based on similarity.

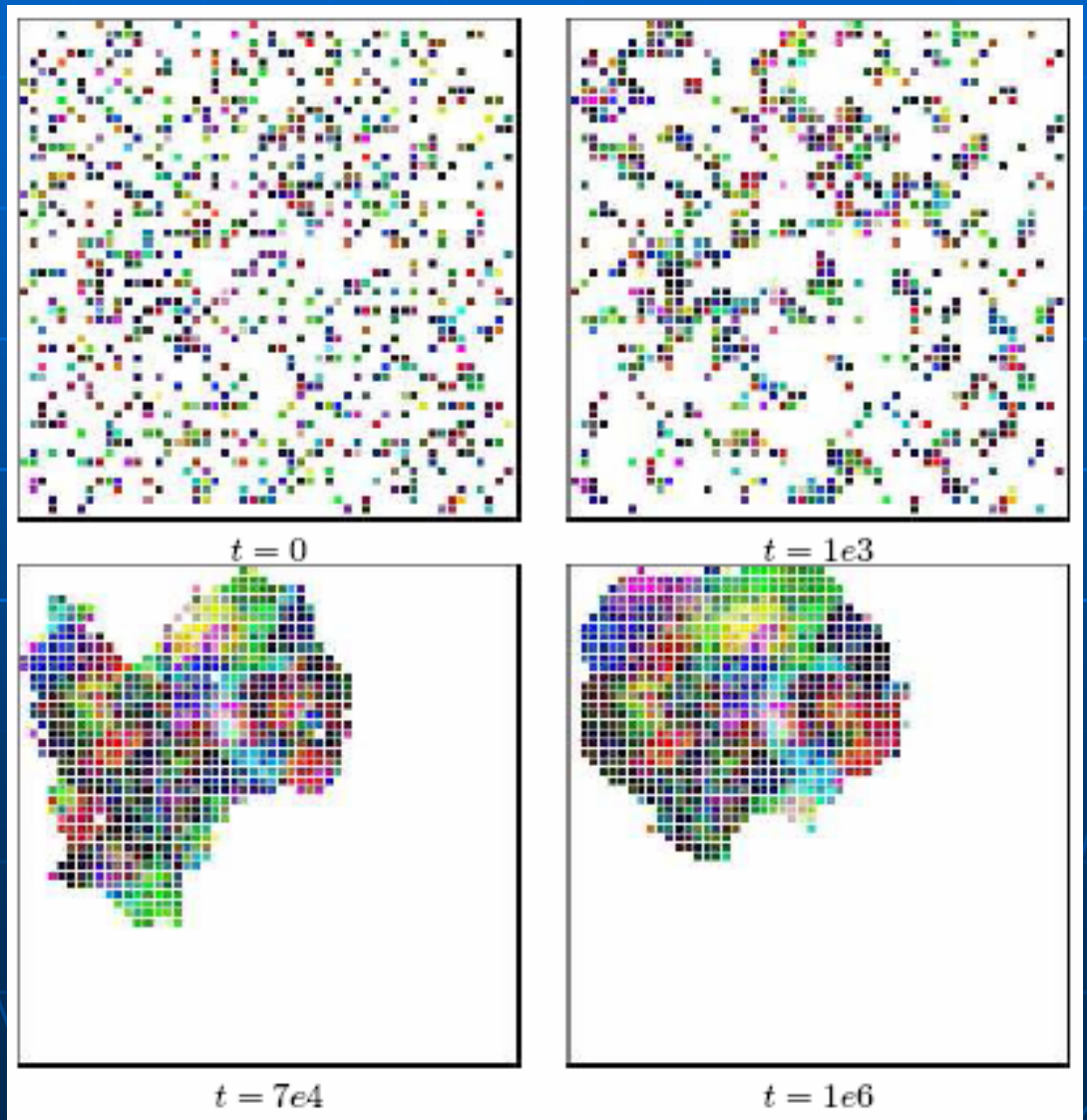
Examples

- 800 objects, 4 shades of grey, 10 ants



RGB Example

- 800 objects, all different colours, 9 ants – each ant only sensitive to R,G or B.



University Timetable Scheduling

(Socha, Sampels & Manfrin, 2003)

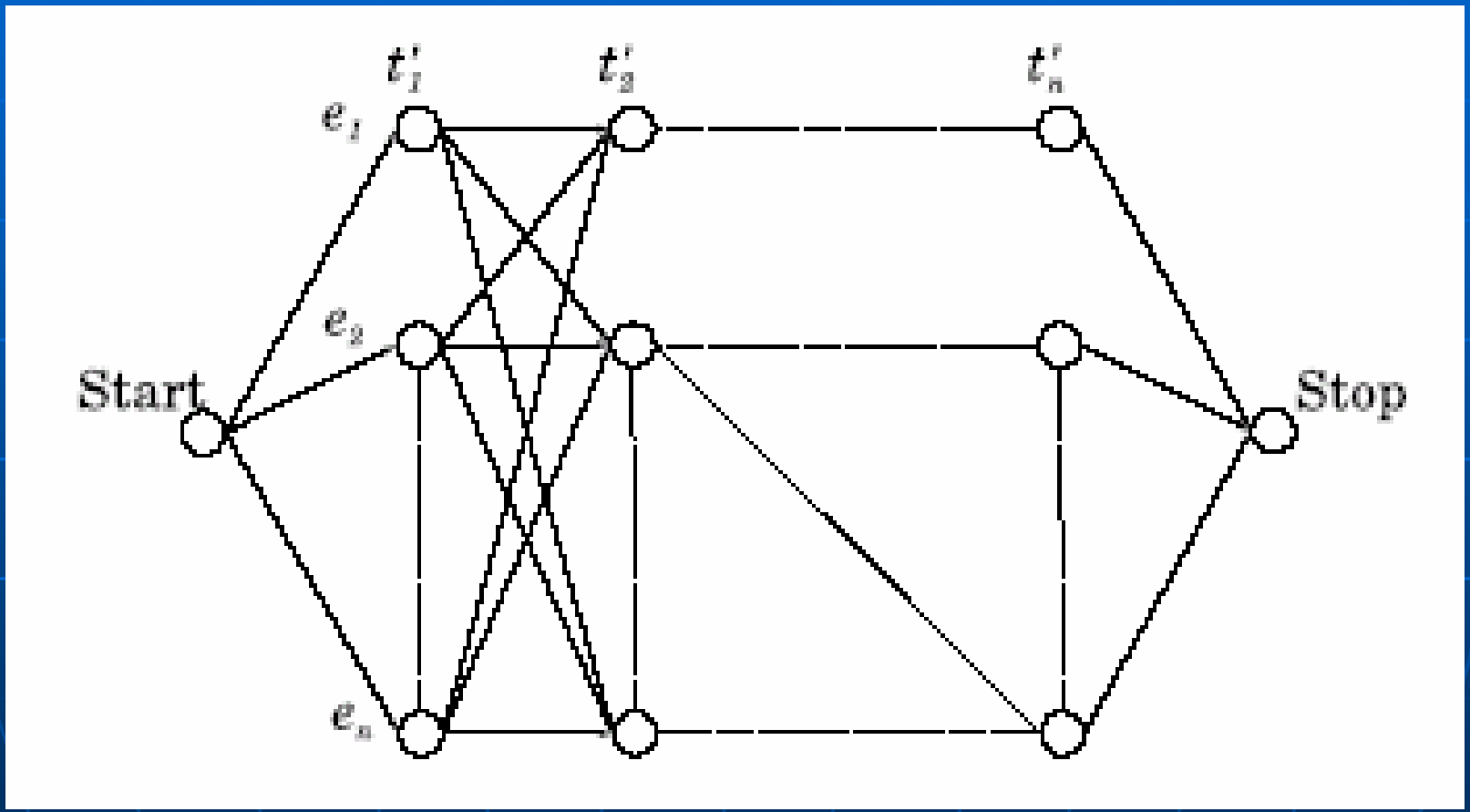
	mon	tue	wed	thur
9:00	E4 E5		E6	E3 E7
11:00		E2		
2:00	E8			
4:00	E1			

- Create a timetable where:
- Hard Constraints
  - No student attends more than one event at the same time
  - The room is big enough for all the attending students
  - Only one event is in each room at any timeslot

- Soft Constraints
  - No student has a class in the last slot of the day
  - No student has more than two classes in a row
  - No student has only one class on one day

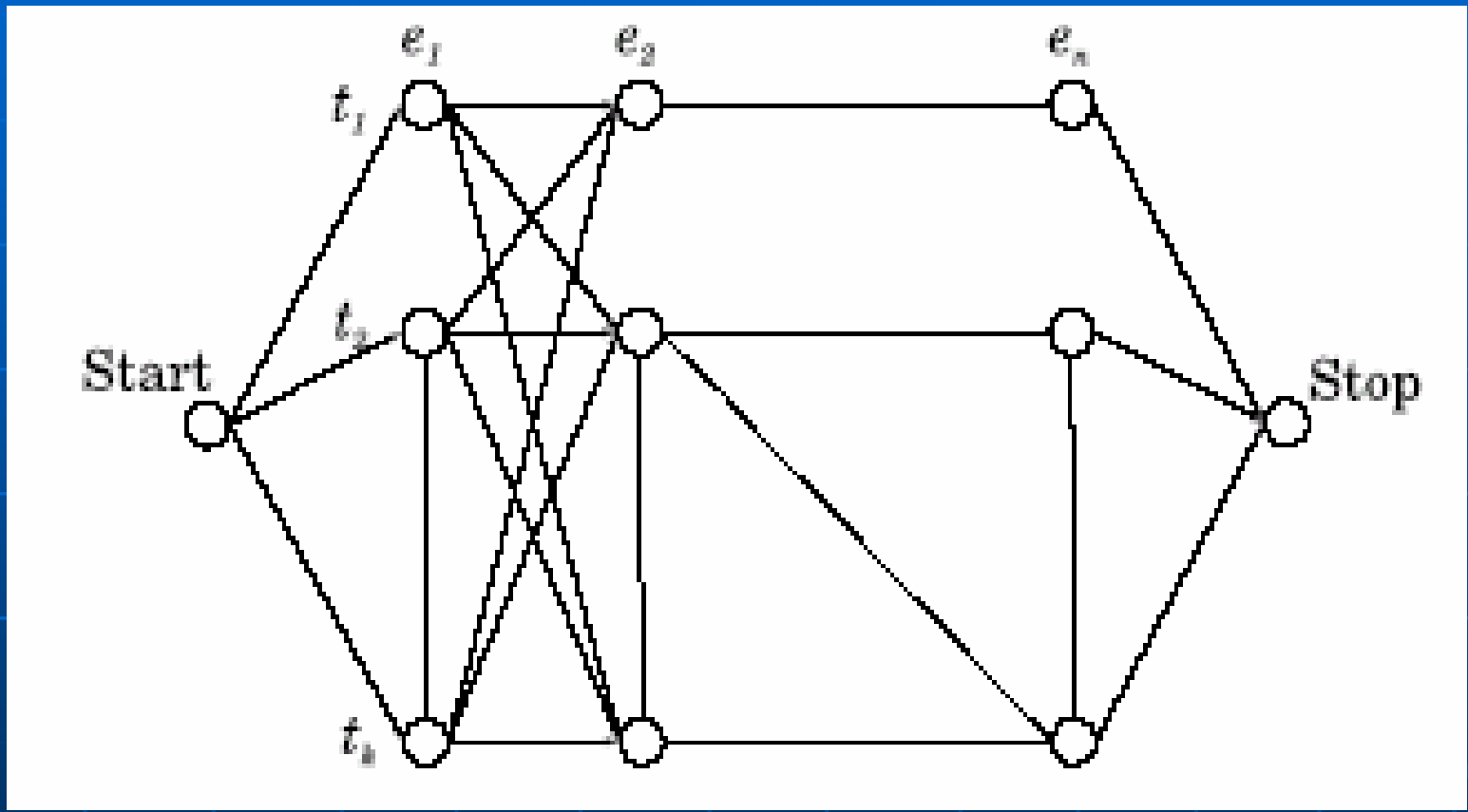
- So, how can we get ants to solve this problem?

Construction Graph 1



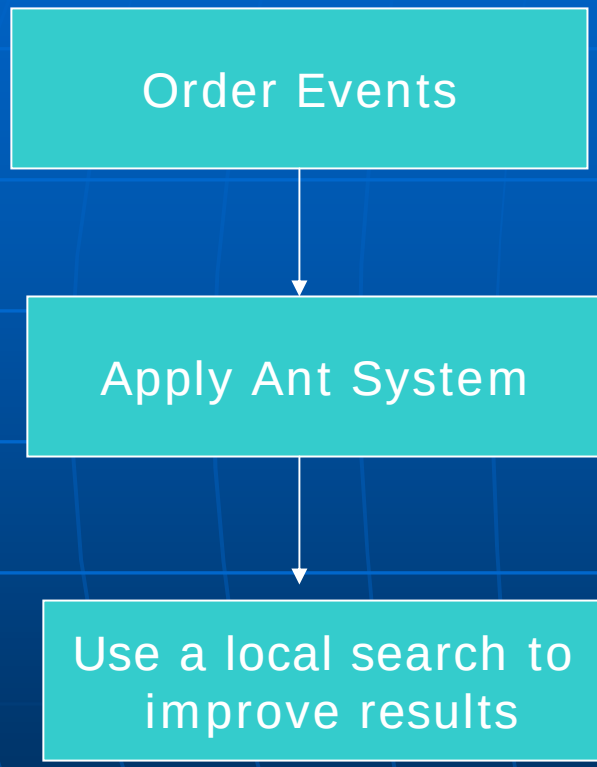
BUT! We must make sure every event is assigned – this doesn't do that – what can we do?

Construction Graph 2



- Converts the timetabling problem into a path-finding problem suitable for our ants

Algorithm Execution



Events are ordered according to whether they are ‘difficult’ or not

Ants traverse the construction graph, laying & following pheromone etc… in a similar way to the TSP

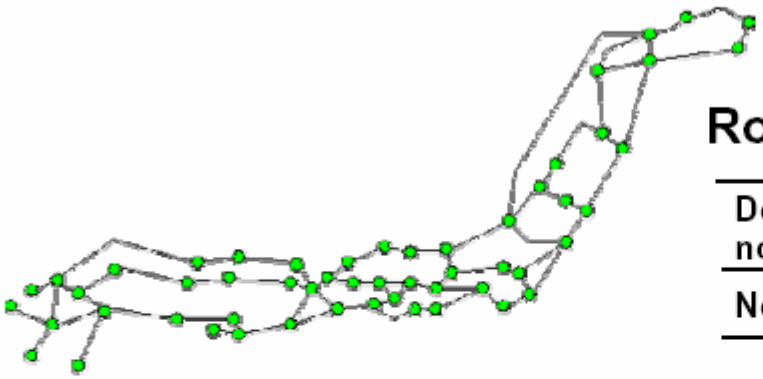
Local search procedure used to subsequently improve results

Mixed Results

- On Medium problems, simulated annealing is better for a given amount of computational time.
- On Large problems, Ant Colony optimisation is better for a given amount of computational time.
- BUT! It does show an innovative approach to solving a problem not obviously path-driven.

Packet routing with ants

- 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