Introduction to Multi-Agent Programming

8. Swarm Intelligence

Flocking, Foraging, Ant Systems, TSP solving

Alexander Kleiner, Bernhard Nebel

Contents

- Introduction
- Swarming & Flocking
- Foraging strategies in ants
- Ant Colony Optimization (ACO)
  - Solving TSPs
- *Case-study*: Team coordination of virtual robots
- Summary

Introduction

- What is swarm intelligence ?
- Swarm intelligence is motivated from insects
  - Colonies of social insects can achieve flexible, intelligent, and complex system level performance from stereotyped, unreliable, unintelligent, and simple elements
  - Insects follow simple rules, use simple local communication (scent trails, sound, touch) with low computational demands
  - Global structure (e.g. nest) reliably *emerges* from the unreliable actions of many
- The modeling of social insects by means of self-Organization can be utilized to motivate the design of methods for distributed problem solving, known as Swarm Intelligent Systems

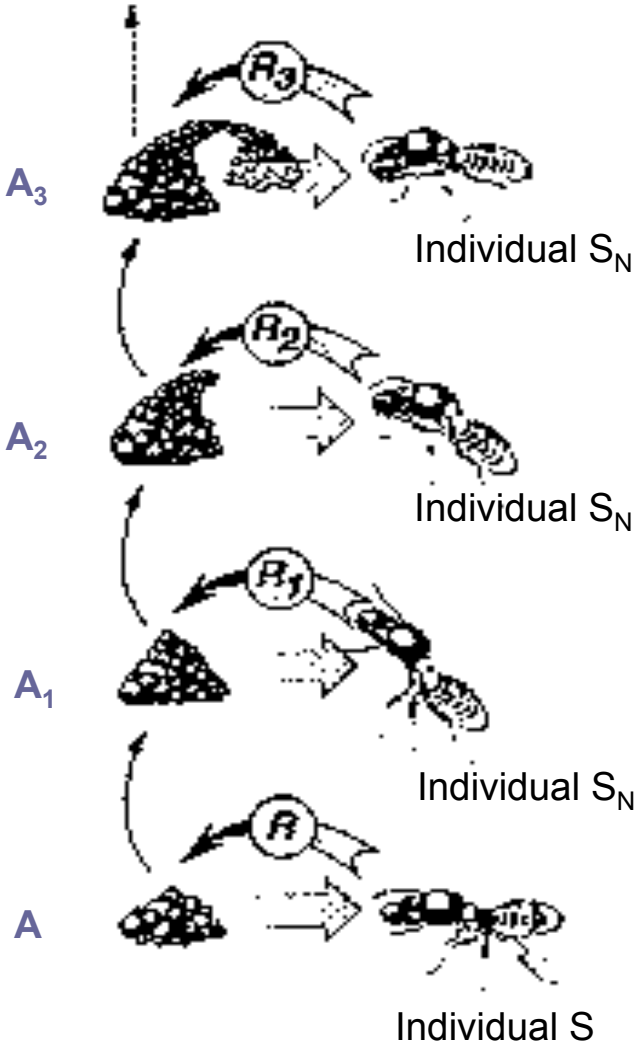
Introduction
Biological Inspiration

- Bees:
  - Communicate the distance and bearing of food sources by dancing
  - Food sources are exploited according to quality and distance from the hive
- Termites
  - Build large cone-shaped outer walls with ventilation ducts
- Ants
  - Leafcutter ants (*Atta*) cut leaves from plants to grow fungi
  - Weaver ant (*Oecophylla*) workers form chains of their own bodies, allowing them to cross wide gaps and to generate enough force to join leaves together. When the leaves are in place, the ants connect both edges with a continuous thread of silk emitted by a mature larva held by a worker

Introduction
Self-organization in social insects

- Relies on four basic ingredients:
  - Positive feedback (amplification)
    - Recruitment to a food source by laying or following a trail (e.g. ant pheromones or bee dance)
  - Negative feedback
    - Counterbalances the positive feedback
    - In form of saturation (limited number of workers), exhaustion (of the food source), or competition (crowding at the food source)
  - Fluctuation
    - Random walks, errors, random task switching
    - Can be seen as "exploration" for finding unexploited food sources
  - Multiple interactions / Stigmergy
    - Direct: antennation, food or liquid exchange, visual contact, chemical contact (the odor of nestmates), …
    - Indirect: Two individuals interact indirectly if one modifies the environment and the other one responds to this modification later in time (Stigmergy)

Stigmergy Example



Pillar construction by termites:

1) Assume the architecture reaches state A that triggers action R from worker S (i.e. drop a soil pellet) transforming the architecture into A_1

2) A_1 stimulates another response R_1 from S or any other worker S_N and so forth

Swarming & Flocking

Real-world example

Anchovies

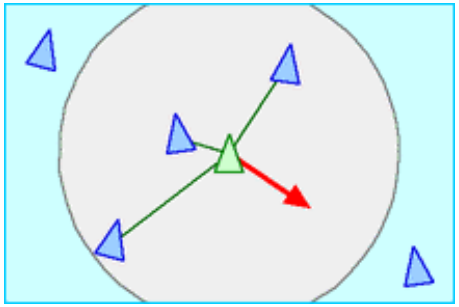
Swarming & Flocking

- Aggregation of similar animals that travel into the same direction
- Applications: Movie effects (Lord of the rings, Lion King), Network routing, swarm robotics, computer games
- In the late 80's Craig Reynolds created a simple model of animal motion that he called Boids
  - **Flock** is a group of objects that exhibit the general class of polarized (aligned), non-colliding, aggregate motion
  - **Boid** is a simulated bird-like object, i.e., it exhibits this type of behavior. It can be a fish, bee, dinosaur, etc.
- The boids model can be implemented by only 3 rules defining a boid's steering behavior

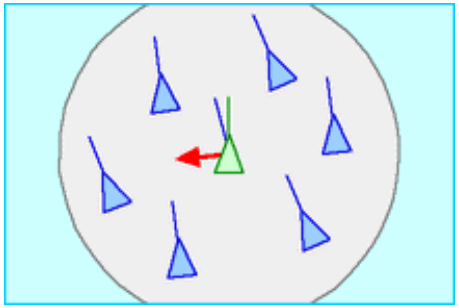
Boids model

Only 3 simple rules needed

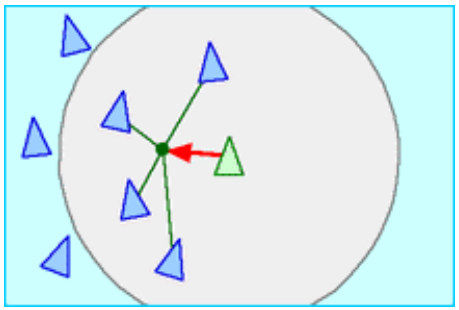
Separation: steer to avoid crowding local mates



Alignment: steer towards the average heading and speed of local mates

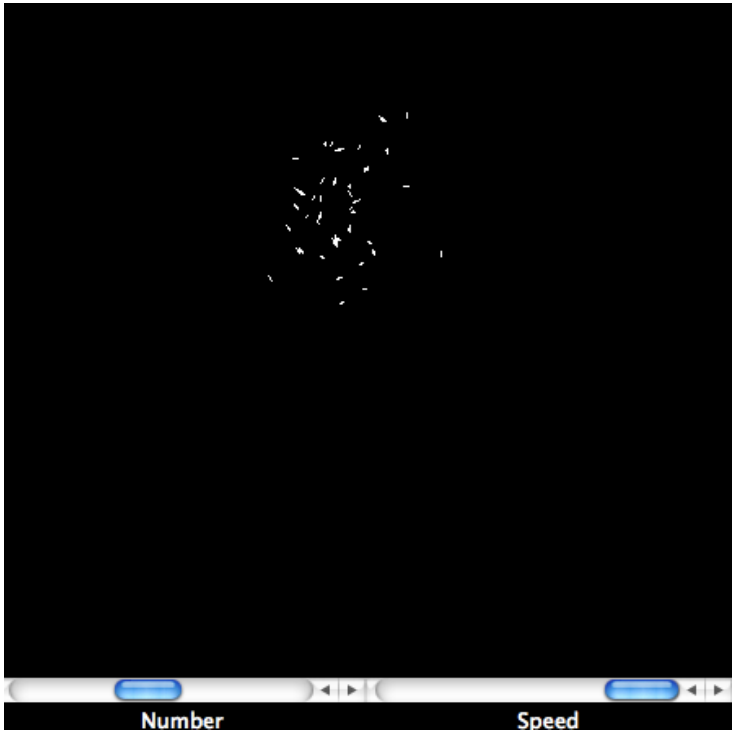


Cohesion: steer to move toward the average position of local mates



Boids model

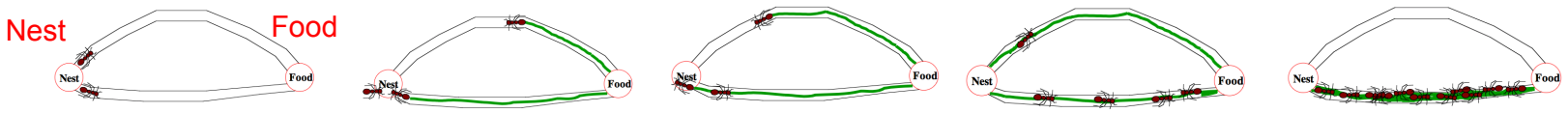
Java Demo



Taken from http://www.alxvy.org/

Foraging Strategies in Ants

- *Some* ants establish indirect communication based on the deposition of pheromone over the path they follow
  - A single ant moves at random, but when it finds a pheromone trail, there is a high probability to follow the trail
  - Ants foraging for food deposit pheromones over their routes. When finding a food source, they return to the nest reinforcing their trails
  - By this, other ants have greater probability to start following such trails and thereby reinforcing it by more pheromones
  - This process works as a positive feedback loop system because the higher the intensity of the pheromone over a trail, the higher the probability that ants start traveling through it



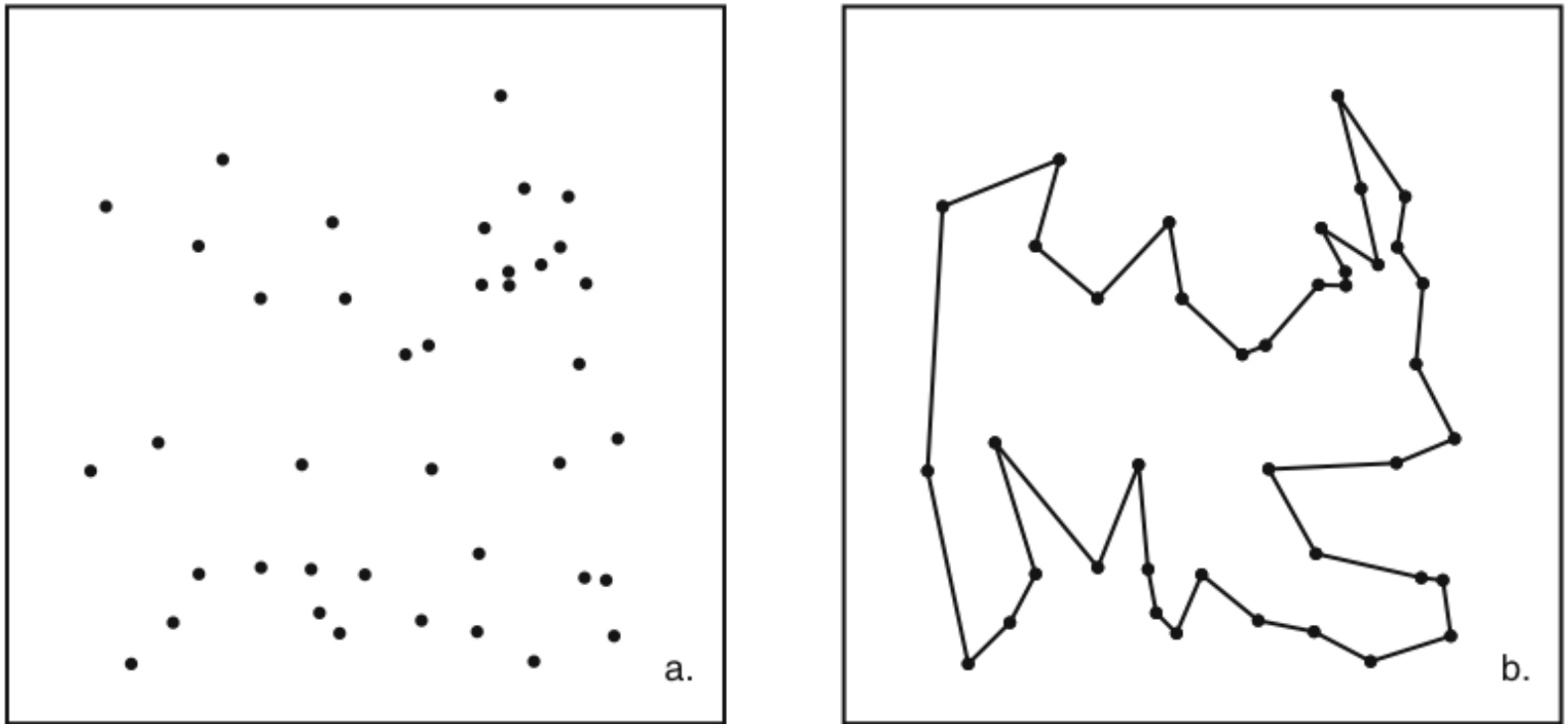
Ants exploring two paths to a food source. The shorter path finally wins due to a higher density of pheromones

Ant Colony Optimization (ACO)

Solving TSPs

- ACO can be used to solve graph problems such as the Traveling Salesman Problem (TSP)
  - For finding *good* but not necessarily *optimal* solutions!
- Goal: find a closed tour of minimal length connecting *n* given cities, while visiting every city only once
- Ant colony solution concept:
  - Using a positive feedback mechanism based on an analogy with the trail laying/following behavior, to reinforce to keep good solutions
  - Negative feedback by pheromone evaporation

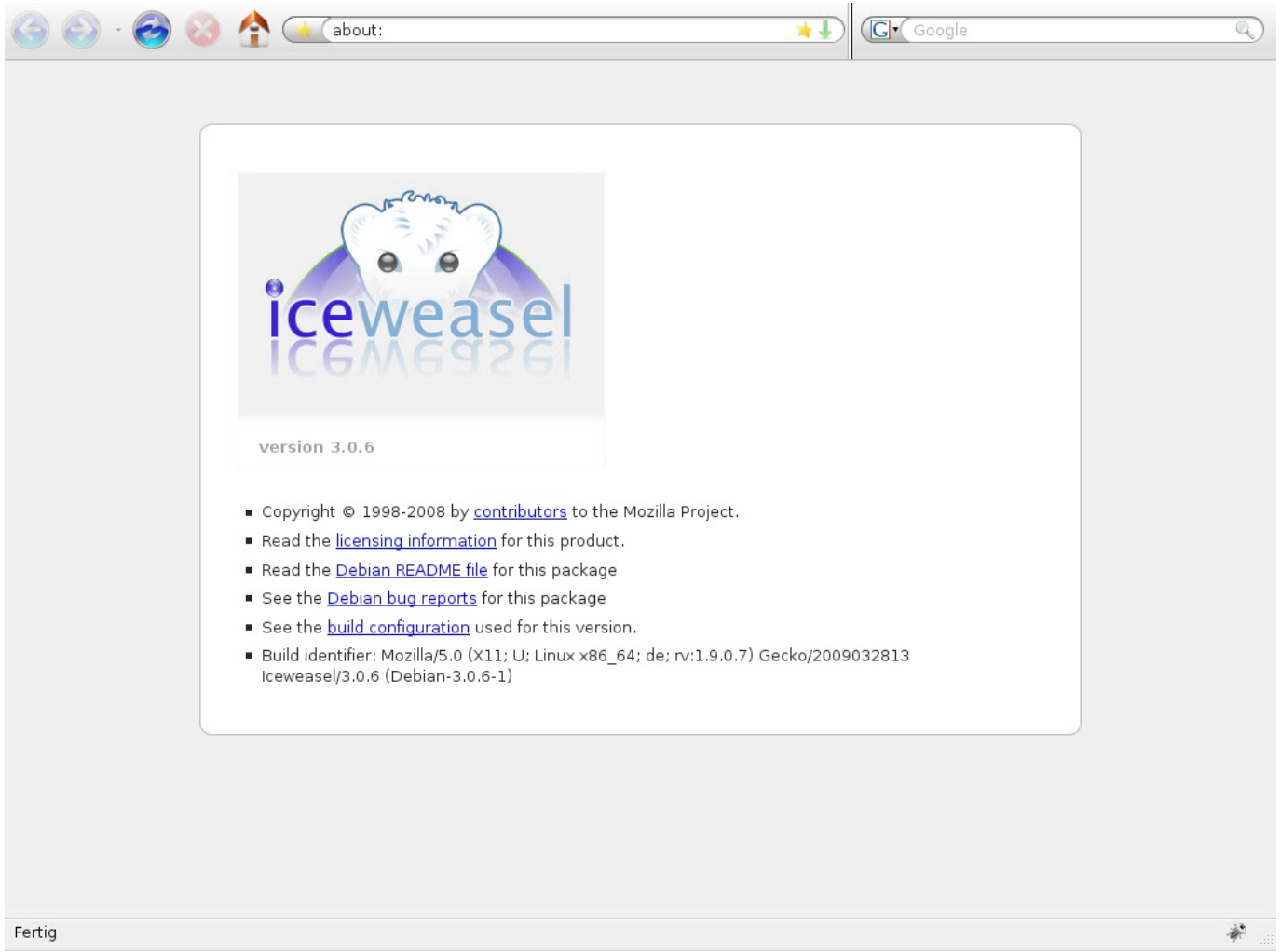
Traveling Salesman Problem (TSP)



Example 40-node TSP with solution

Note TSPs are NP-Complete problems, i.e. finding solutions with increasing number of cities becomes intractable

Traveling Salesman Problem (TSP) with GoogleMaps



Found at: http://zrp.tournament.de/

Ant Colony Optimization (1)

Solution to the TSP

- Ants move on the problem graph from one city to another until completing a tour
- Each transition depends on:
  - Whether the city has already been visited (tabu list). We denote the set of cities **not** visited by ant *k* when located at city *i* with J_i^k
  - We denote $n_{ij} = 1/d_{ij}$ the visibility, computed from the distance between two cities i and j. Can be seen as a heuristic preferring nearby cities.
  - The amount of virtual pheromone $\tau_{ij}(t)$ on the edge connecting city *i* with city *j* at time *t*

Ant Colony Optimization (2)

Random Transition Rule

- The transition rule, i.e. probability for ant *k* to go to city *j* while building its *t-th* tour is given by:

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [n_{ij}]^{\beta}}{\sum_{l \in J_i^k} [\tau_{il}(t)]^{\alpha} \cdot [n_{il}]^{\beta}} \quad if \; j \in J_i^k \;, else \; 0$$

- Where α and β are parameters controlling the trade-off between trail intensity and visibility

Ant Colony Optimization (3)

Trail update

- Pheromone increase:
  - After completing a tour (episode), each ant *k* lays a quantity of pheromone $\Delta\tau_{ij}^{k}$ on each visited edge (i,j)
  - The quantity depends on the ant's performance during tour T^k at iteration t:

$$\Delta\tau_{ij}^{k}(t) = \frac{Q}{L_k(t)} \quad if\,(i,j) \in T^k(t),\ else\ 0$$

  - Where $L_k(t)$ is the length, and Q is a parameter that should be set close to the optimal tour length

- Pheromone decrease:
  - Pheromone decay (evaporation) controlled by parameter ρ, with $0 \leq \rho < 1$

Ant Colony Optimization (4)

Trail update

- Resulting update rule:

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

with:

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

i.e. summing up the influences from all *m* ants

Ant Colony Optimization (5)

Elitist ants

- Idea borrowed from genetic algorithms: always keep the best n solutions in the genetic pool
- An elitist ant is an ant that reinforces the edge belonging to T^+ (the best tour found so far) by the quantity Q/L^+, where L^+ is the length of T^+
- During each iteration we add *e* elitist ants to the usual ants
- Hence, the edge belonging to T^+ gets an extra reinforcement of $e*Q/L^+$

Ant Colony Optimization (6)
Complete Algorithm

/* Initialization */

For every edge (i,j) do

$$\tau_{ij}(0) = \tau_0$$

End

For k = 1 to m do

Place ant k on a randomly chosen city

End

Let T^+ be the shortest found tour and L^+ its length

For t = 1 to t_{max} do *// t_{max} is the number of episodes*

For k = 1 to m do *// m is the number of ants*

Build tour $T^k(t)$ by choosing n-1 times next city j with probability:

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [n_{ij}]^\beta}{\sum_{l \in J_i^k} [\tau_{il}(t)]^\alpha \cdot [n_{il}]^\beta} \quad if\ j \in J_i^k\ , else\ 0$$

End

Ant Colony Optimization (7)

Complete Algorithm

For k = 1 to m do

Compute length $L^k(t)$ of tour $T^k(t)$ of ant k

If $L^k(t) < L^+(t)$ **then**

$T^+(t) = T^k(t)$

End

End

For every edge (i,j) **do**

$$\tau_{ij}(t) \leftarrow (1-\rho)\cdot\tau_{ij}(t) + \Delta\tau_{ij}(t) + e\cdot\Delta\tau^e_{ij}(t) \quad \textbf{with} \quad \Delta\tau_{ij}(t) = \sum_{k=1}^{m}\Delta\tau^k_{ij}(t),$$

$$\Delta\tau^k_{ij}(t) = \begin{cases} Q/L^k(t) & if(i,j)\in T^k(t) \\ 0 & ohterwise \end{cases},$$

$$\Delta\tau^e_{ij}(t) = \begin{cases} Q/L^+(t) & if(i,j)\in T^+(t) \\ 0 & ohterwise \end{cases}.$$

End

Ant Colony Optimization (8)

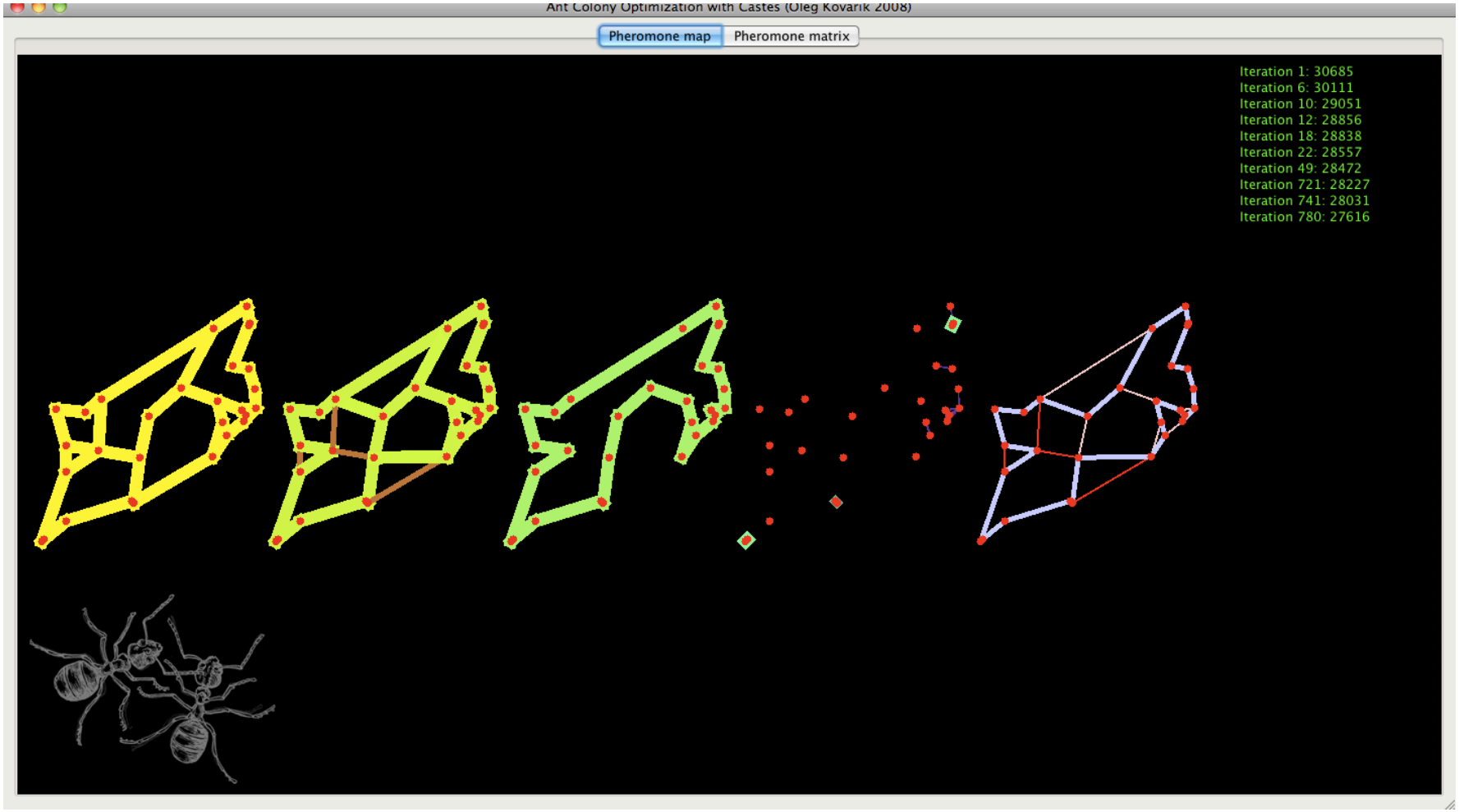
Complete Algorithm

```
   For every edge (i,j) do
      τ_ij(t+1) = τ_ij(t)
   End
End // Episodes

Print shortest tour T+ and its length L+
```

Ant Colony Optimization (9)

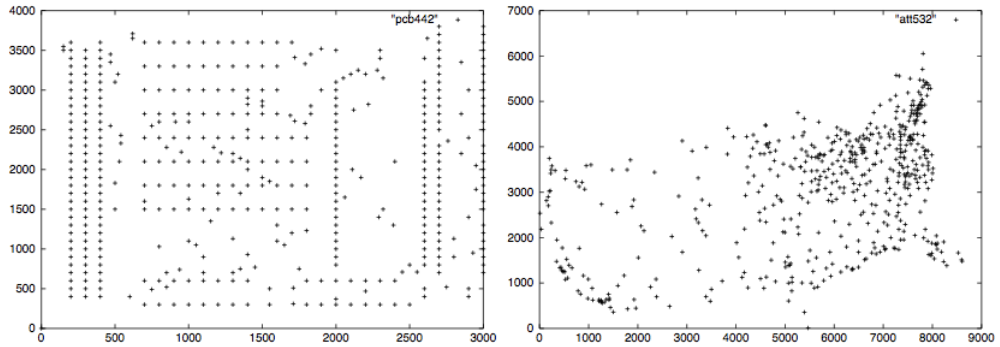
DEMO



From: http://kovarik.felk.cvut.cz/ant-algorithms/index.php

Try it out by yourself!

- TSPLIB:
  - A library of sample instances for the TSP (and related problems)
  - http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/



TSP instances pcb442 (left side) and att532 (right side).
The instance pcb442 stems from a drilling problem in a printed circuit board application, the instance att532 comprises 532 cities in the USA.

- Ant Colony Optimization implementations:
  - http://iridia.ulb.ac.be/~mdorigo/ACO/aco-code/public-software.html
  - http://kovarik.felk.cvut.cz/ant-algorithms/research-download.php#acoc

Case-study: Team coordination of virtual robots

USARSim: A simulator for emergency response

- Based on the Unreal game engine (UT2004, Epic Games)
- Realistic models for
  - USAR environments, indoor & outdoor
  - Robots, such as Pioneer2 DX, Sony AIBO, …
  - Sensors, such as Laser Range Finder, Color Camera, IMU, Wheel Odometry, RFID
- Agents connect via a TCP/IP interface
- Path loss simulation (e.g. WLAN)
- Research challenges:
  - Autonomous control of large robot teams (up to 12)
  - Multi-robot disaster area mapping
  - Coordination of heterogeneous robots with different manipulation and sensing capabilities

RFID-based Exploration
Hybrid: local exploration and global planning

- Task: Find all victims in the world with a team of robots

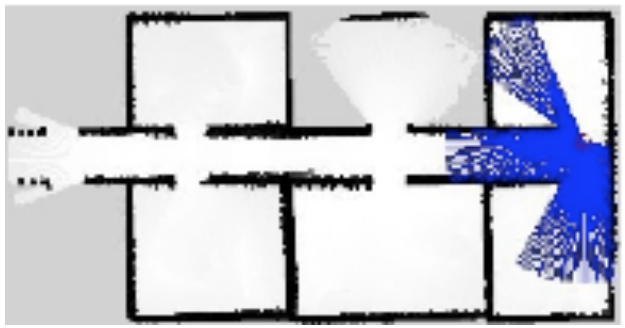
- Local exploration (LE):
  - Indirect communication vi RFID
  - Scales-up with # of robots and environment size
  - Inefficient exploration due to local minima
- Global task assignment and path planning:
  - Based on node graph abstraction of the environment
  - Monitors LE and computes new agent-node assignment If exploration overlap is high
  - Requires communication

Local Exploration
Navigation

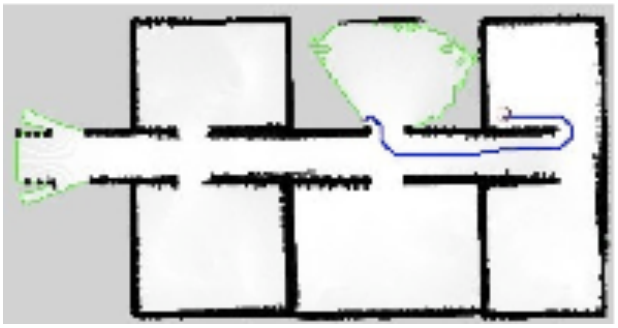
- Local trajectory planning:
  - Based on evidence grid, e.g. limited to 4X4 meters
  - Exploration targets taken from extracted frontier cells
  - Efficient A* planning towards selected FP
  - Cost function considering path length and occupancy:

$$c(s_{i+1}) = c(s_i) + d(s_{i+1}, s_i) * (1 + \alpha * occ(s_{i+1}))$$

α regulates the influence of occupied cells



Occupancy Grid generated from laser scans



Extracted frontier cells and A* plan to selected target

Local Exploration
Coordination & Frontier Cell Selection

- RFID tag distribution and detection:
  - Deployment of *new* RFIDs with respect to the detected RFID density
  - Detection of nearby RFIDs and consequent update of Local RFID Set (LRS)
  - Programming of RFID memory with visited locations (relative position)
- Coordination:
  - Discretization of node vicinity into equally sized patches
  - Node memory for counting visits of each patch [Svennebring and Koenig, 2004])
- Frontier selection by minimizing the following cost function:

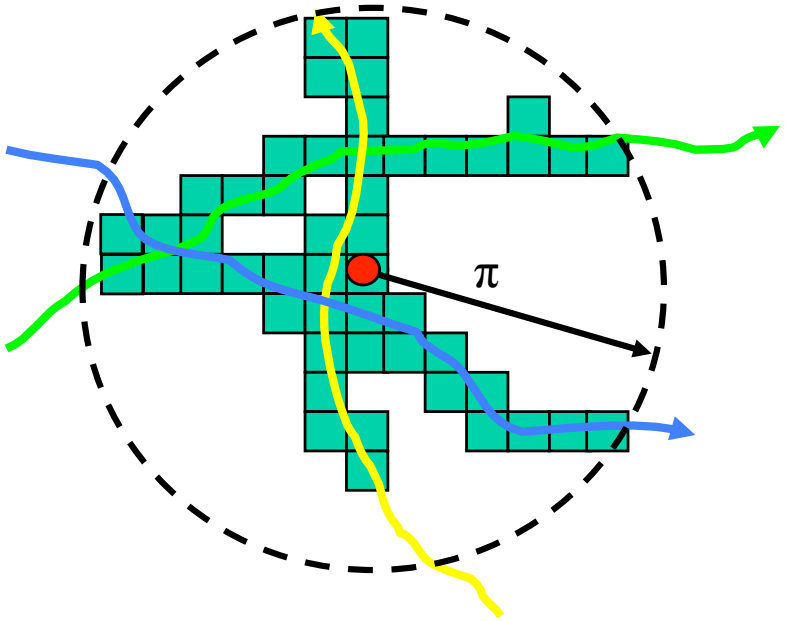
$$F_v\left(l_{f_j}\right) = \sum_{r \in LRS} \sum_{p \in P_r} \frac{count\,(p)}{d\left(l_{f_j}, p\right)}$$

l_{fi} : frontier cell location,
LRS: set of nodes within range,
Pr: set of patches around node r,
d(.): the Euclidean distance

This models ant pheromones!

Local Exploration cont.

Discretization of the node's vicinity π



RFID node

Robot trajectories

Discretized visited areas counted in memory

Relative addressing!

Results Local Team Coordination

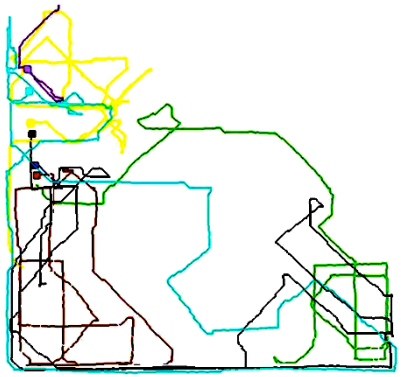
Virtual rescue scenarios from NIST (RoboCup'06)

		RRFreiburg	GROK	IUB	SPQR	STEEL	UVA
	Area [m2]	**579**	27	227	96	134	262
SemiFinal1	# Robots	8	1	6	4	6	6
	Tot. Length	2503,25	39,81	257,7	143,91	190,59	365,51
	Area [m2]	**1276**	82	139	123	139	286
Semifinal 2	# Robots	8	1	6	5	6	7
	Tot. Length	1991,86	79,64	152,91	124,226	271,69	401,45
	Area [m2]	**1203**	-	210	-	-	-
Final 1	# Robots	8	-	8	-	-	-
	Tot. Length	2536,55	-	224,667	-	-	-
	Area [m2]	**350**	-	136	-	-	-
Final 2	# Robots	8	-	6	-	-	-
	Tot. Length	1761,63	-	254,681	-	-	-

Largest explored area (by 8 robots)

Final 1 (indoor, 1276m^2)

Each color denotes the path of a single robot



Final 2 (outdoor. 1203m^2)

Global Exploration

Task assignment and planning

- Task assignment:
  - Sequential robot planning to best targets [Burgard et al., 2005]
  - Genetic algorithm (GA) for finding optimal planning sequence
    - Score computed from multi-robot plan cost
    - Initialized by greedy sequence
- Computation of multi-robot plan:
  - A* time space planning to multiple goals [Bennewitz et al., 2001]
  - *Plan costs*: joint plan length + conflict penalties (infinite if deadlock)
  - *Heuristic:* based on pre-computed shortest Dijkstra tree ignoring conflicts

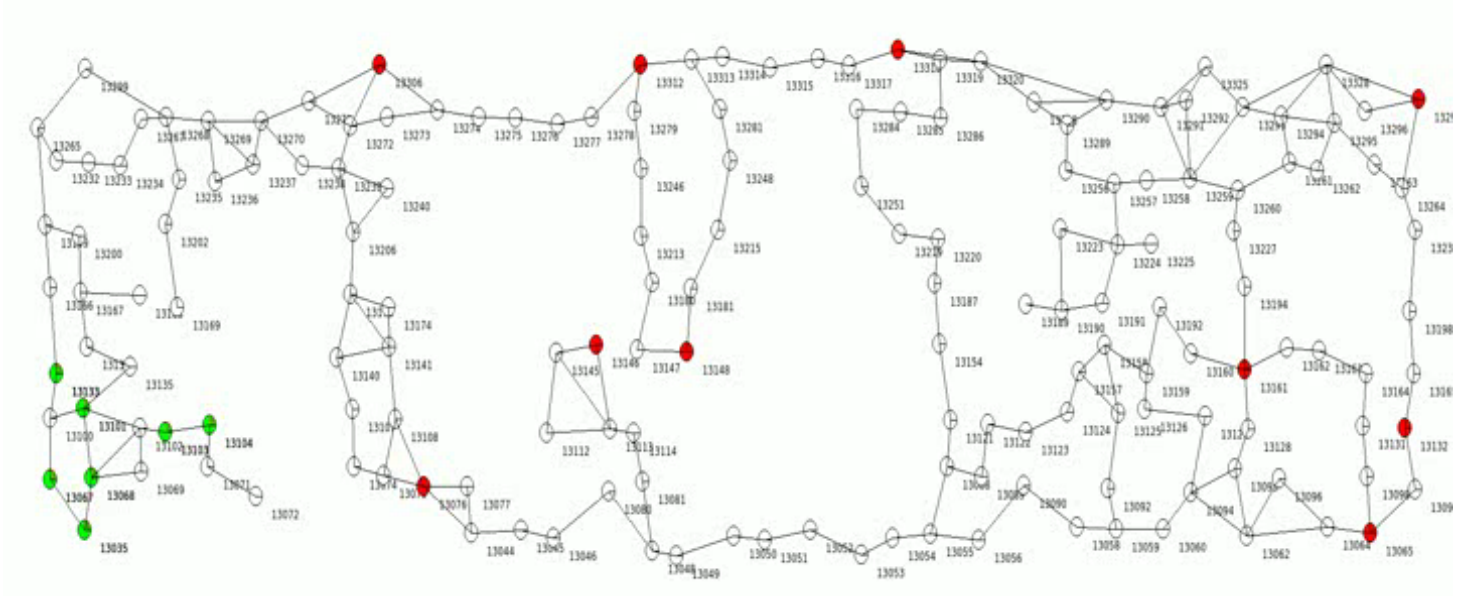
Results Global Team Coordination

Task assignment and planning on node graph (USARSim outdoor map)

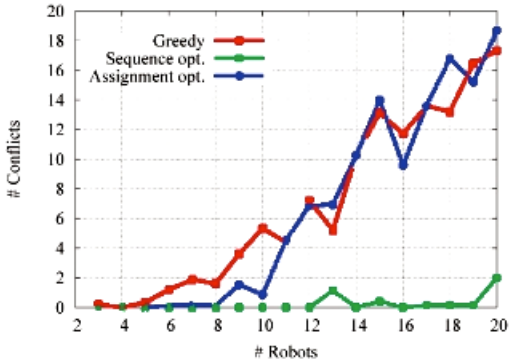
Goal nodes

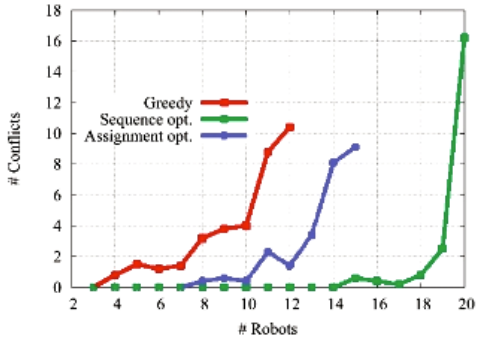
Robot start nodes

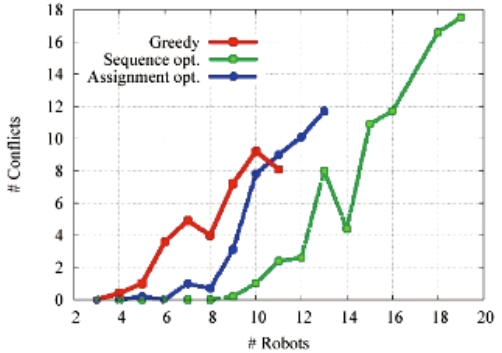
Multi-robot plan



Conflicts vs. # of robots: **Greedy** (red), **GA assignment** (blue), **GA sequence** (green)



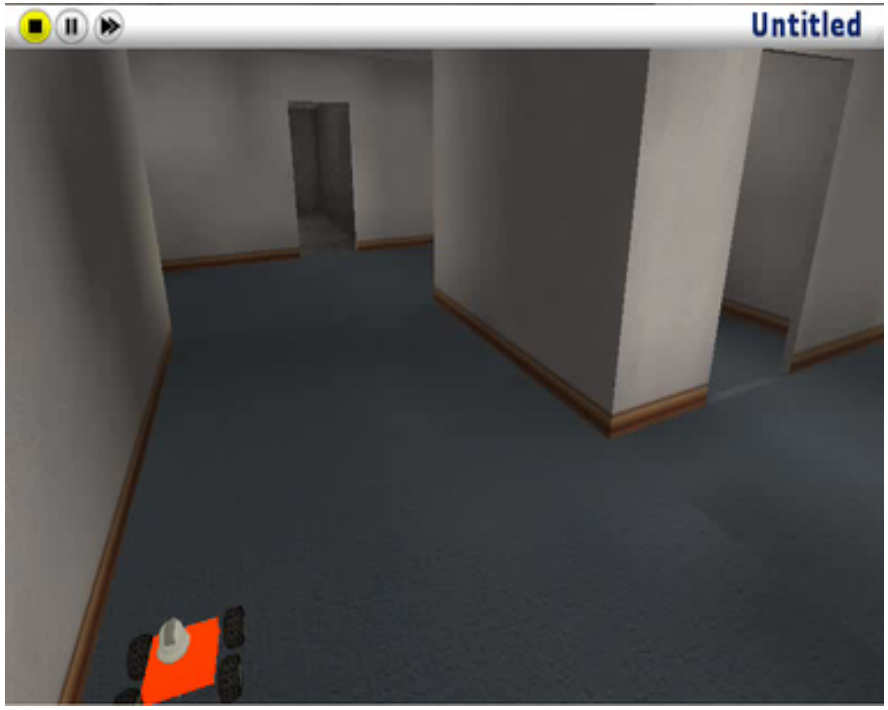




Rescue Virtual Competition

Videos from RoboCup'06

Semi-Final`06



Final`06

Summary

- Flocking is a very simple mechanism that has been uses quite successfully in many applications
  - Can be used to simplify multi-agent path planning of a group
- The foraging behavior of ants has motivated Ant Colony Optimization (ACO) algorithms
  - Although sub-optimal, they are powerful to find fast good solutions in TSPs
  - Numerous of extensions to the presented approach have been proposed
  - Other problems that have been solved: Task Allocation, Graph Partitioning, Constraint Satisfaction, Transport problems, …
- RFIDs might be a good choice for simulating pheromones (at least when they are getting cheaper)

Literature

- Bonabeau E., Dorigo M., and Theraulaz G. **Swarm Intelligence: From Natural to Artificial Systems**, *Oxford University Press*, 1999.
- Reynolds, C. W. (1987) **Flocks, Herds, and Schools: A Distributed Behavioral Model**, in *Computer Graphics*, 21 (4) (SIGGRAPH '87 Conference Proceedings) pages 25-34.
- V.A. Ziparo, A. Kleiner, B. Nebel, and D. Nardi, **RFID-Based Exploration for Large Robot Teams**, In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), 2007*
- Svennebring, J. and Koenig, S. **Building terrain-covering ant robots: A feasibility study**. *Autonomous Robots*, 16 (3):313–332, 2004.

Illustrations and Ideas presented in this lecture are mainly from the above publications.