Swarm Intelligence

13 May 2017 16:33

Key idea: Simple units work together to achi perform optimisation problems.

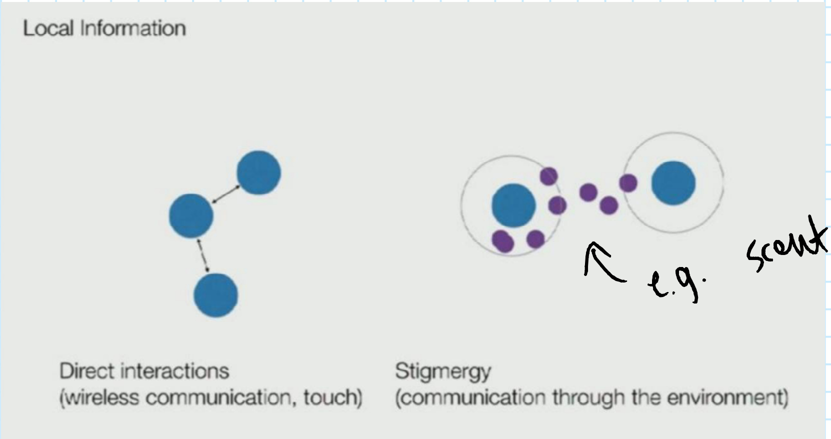
Technological Motivations

- Robust
- Scalable
- Versatile / flexible
- More capable than the sum of their parts

Agents

- Large numbers
- Relatively homogeneous
- Relatively incapable
- Interactions based on simple behavioural rules that exploit only local information

A supposed to heterogenous, e.g. ants with specialists



e.g. scent, visual

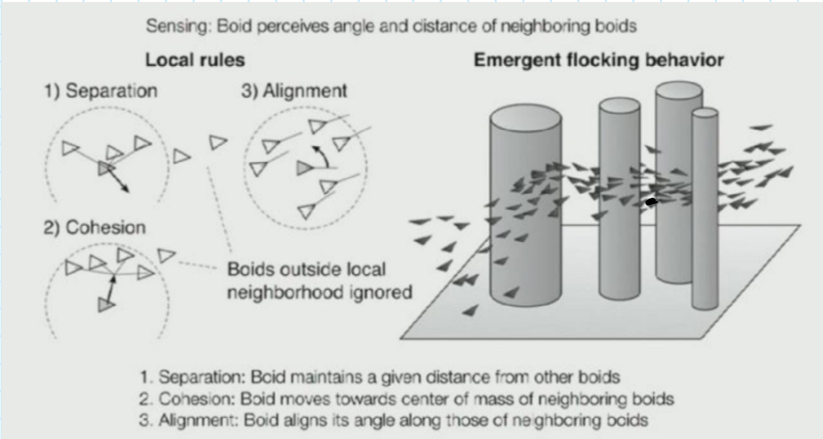
Steps to engineering swarms:

- Determine communication paradigm
- Establish communication symbols
- Design decision rules for agent actions
- "Prove" the system exhibits global behavior.
  - Trial by error
  - Bio-inspired
  - ML / GE

Reynolds Flocking (1987)

- Flocking-like behavior

- Flocking-like behavior
- Agents are call boids. Rules each give a vector that is added to the timestep's velocity vector. This is used to compute the next position.



Additional rules include; replusion from obstacles, adjusted communication distance.

Particle Swarm Optimisation:

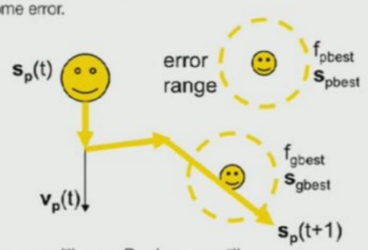
- Inspired by birds flocking to areas of the best food area.
- Rules to produce $\underset{\sim}{v}$:

The flock is most likely to succeed when birds combine three strategies:
Brave: keep flying in the same direction
Conservative: fly back towards its own best previous position
Swarm: move towards its best neighbor

- The idea is agents keep track of local optima, however the overall tendency is towards global optima

Particle's actions

A particle computes the next position by taking into account a fraction of its current velocity v, the direction to its previous best location pbest, and the direction to the location of the best neighbor gbest. The movement towards other particles has some error.

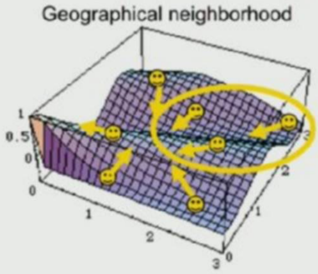


$v_p(t+1) = a \times v_p(t) + b \times R \times (s_{pbest} - s_p(t)) + c \times R \times (s_{gbest} - s_p(t))$
$s_p(t+1) = s_p(t) + v_p(t+1)$
where a, b, c are learning constants between 0 and 1
R is a random number between 0 and 1

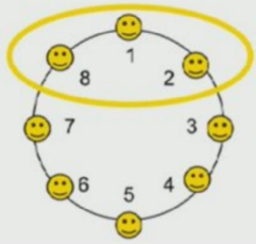
Initialization

Swarm size: Typically 20 particles for problems with dimensionality 2 - 200
Initial position of each particle: Random
Neighborhood topology: Global, geographical or social (list based)
Neighborhood size: Typically 3 to 5
Set max velocity to v_{max}; if $v(t+1)$ is larger, clip it to v_{max}
Iterate until best solution is found or no further improvement

Geographical neighborhood



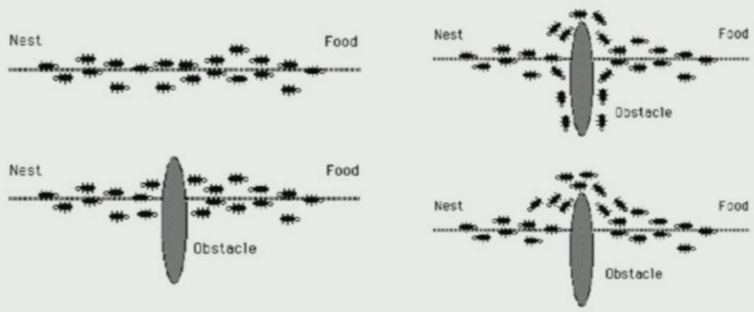
Social neighborhood



Solving Shortest Paths

Ant trails

They use stigmergy.



Finding the shortest path

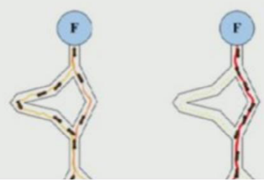
As they move, ants deposit pheromone

Pheromone decays in time

allows forgetting of bad paths

Ants follow path with highest pheromone concentration

Without pheromone, equal probability of choosing short or long



Finding the shortest path

allows forgetting of bad paths

As they move, ants deposit pheromone

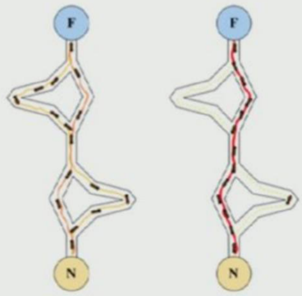
Pheromone decays in time

Ants follow path with highest pheromone concentration

Without pheromone, equal probability of choosing short or long path

Shorter path allows higher number of passages and therefore pheromone level will be higher on shorter path.

Ants will increasingly tend to choose shorter path.



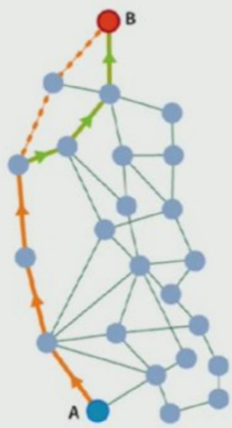
Ant colony optimization

The system is dynamic and robust

Ant Colony Optimization is an algorithm developed by Dorigo et al. in 1994 inspired upon stigmergic communication to find the shortest path in a network.

Typical examples are telephone, internet, and any problem that can be described as Travel Salesman Problem. Used/adopted by British Telecom, MCI Worldcom, Barilla, etc.

Advantage of algorithm is that, as ants do, it allows dynamic rerouting through shortest path if one node is broken. Most other algorithms instead assume that the network is static.



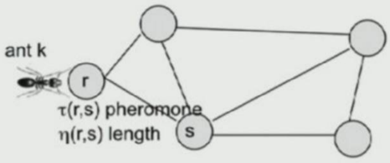
Ant Colony Optimization

Each ant generates a complete tour of nodes using probabilistic transition rule encouraging choice of edge with high pheromone and short distance

Pheromone level on each edge is updated by considering evaporation and deposit by each ant

Pheromone levels only of edges traveled by best ant are increased in inverse proportion to length of path.

Result is that edges that belong to short tours receive greater amount of pheromone



Transition Rule

$J_k(r)$ is the set of nodes agent k can visit from node r.

Find a random number q between 0 and 1. If q is smaller than q0, then choose edge with largest amount of pheromone τ and shortest length η, otherwise use probabilistic rule:

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

Ant k sitting on city r moves to city s with probability proportional to amount of pheromone τ and length η of edge relative to all other cities connected to r that remain to be visited.

Choice of exponent β determines importance of edge length with respect to pheromone.

P_k is a probability distribution over the edges of a directed graph.
Pheromone levels decrease on each timestep,

$$\tau(r,s) \leftarrow (1-\rho)\tau(r,s) + \rho\tau_0$$

Initialization

Use approximately 100 ants

Distribute them on random nodes

Initial pheromone level is equal for all edges and inversely proportional to number of nodes n times estimated length of optimal path Lnn

Initial pheromone level	$\tau 0 = (n \cdot L_{nn})^{-1}$
Importance of length over pheromone	$\beta = 2$
Exploration threshold	$q0 = 0.9$
Pheromone update rate	$\rho = 0.1$

Modular Robots

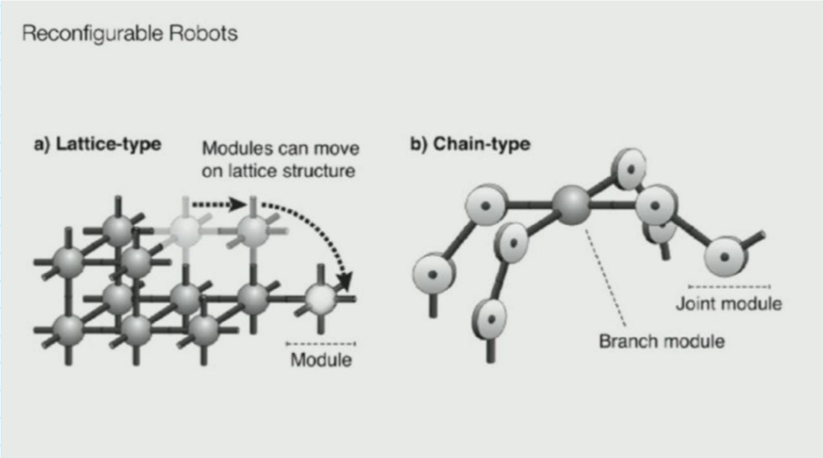
Reconfigurable Robots

A modular robot, usually composed of several identical components, which can be re-organized to create morphologies suitable for different tasks.

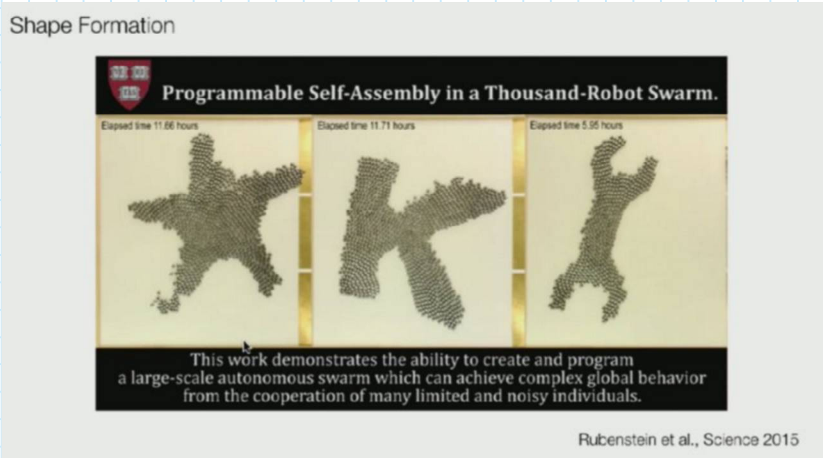
Inspiration:
cells (cellular automata)
individuals (swarm intelligence)

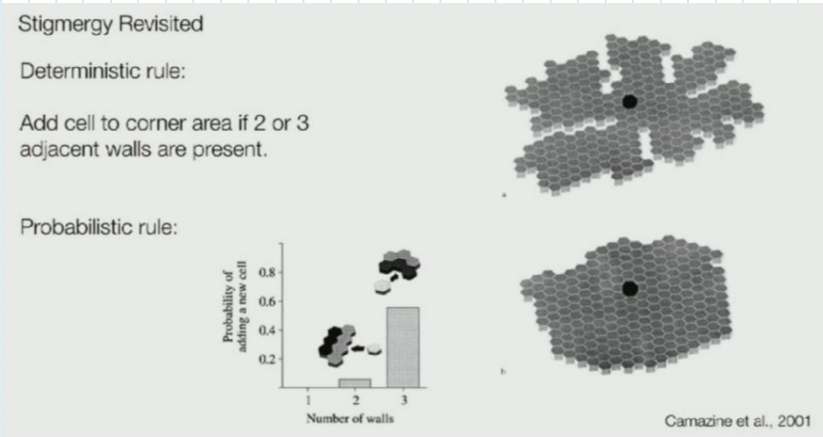
Chain-type reconfigurable robots
Lattice-type reconfigurable robots
Mobile reconfigurable robots
Further types of reconfigurable robots

Related to morphological computation, swarms adjust their morphology for different functions.

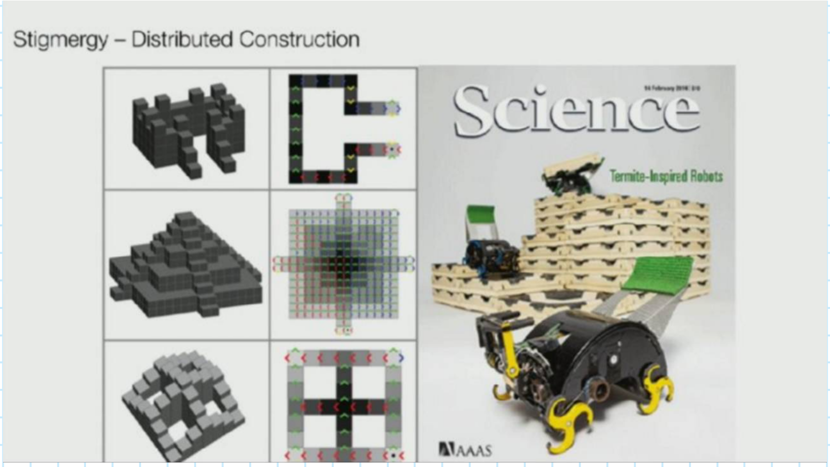


We want to design agent rules to creating a desired shape.





This ties into distrubuted construction, whereby agents deposit building units



Agents maintain a plan for the final structure and decide where to place their next block. Important considerations need to be made so as to not obstruct one another - bots must "understand" their actions.