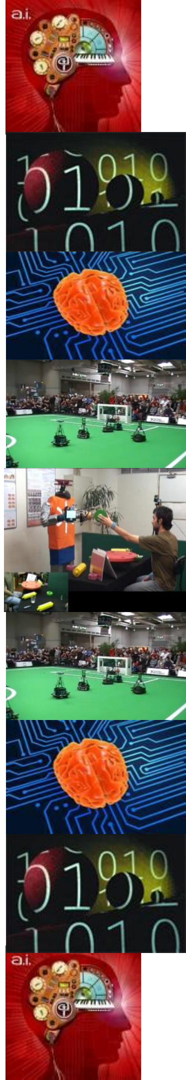
Swarm Intelligence

Ant Colony Optimisation

Introduction

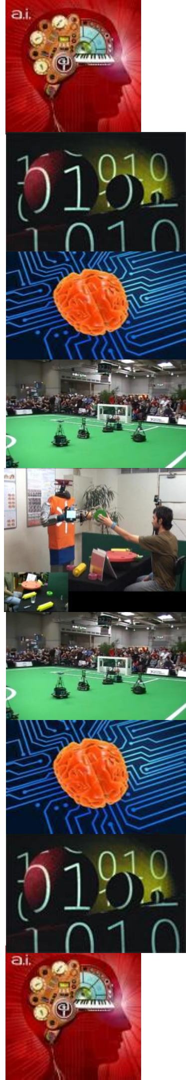
- Evolutionary computation.
- Evolutionary algorithms (EAs)
- Is a metaheuristic.



Introduction

- Swarm Intelligence (SI) emulates the collective behavior of social animals and insects.

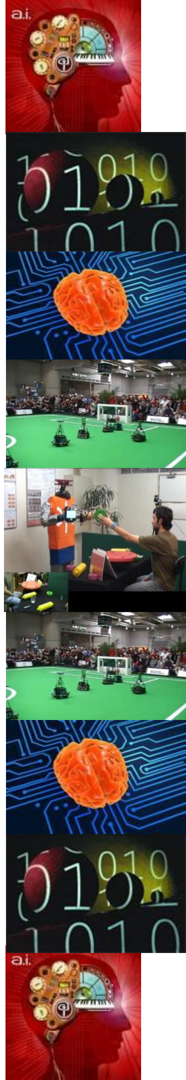
- Swarm intelligence is inspired by the behavior of social animals, such as ants, bees, and termites



Introduction

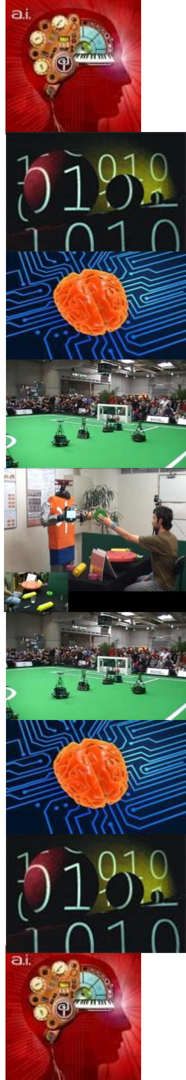
- Swarm intelligence algorithms use simple rules to guide the behavior of a group of agents.

- Optimization problems, such as finding the shortest path between two points, or optimizing a complex function can be used.



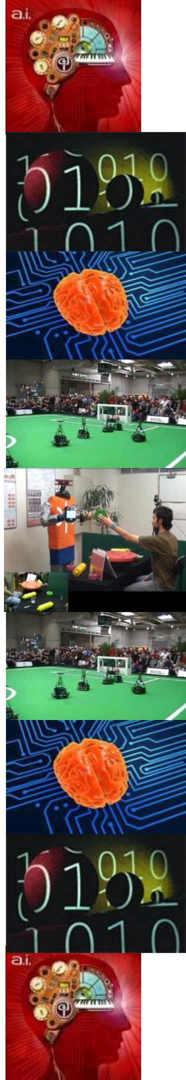
Introduction

- Ant Colony Optimization,
- Particle Swarm Optimization,
- and Artificial Bee Colony Optimization



Ant Colony Optimisation

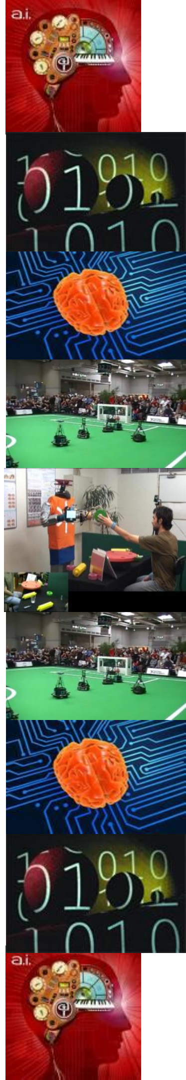
- Population-based method
- Multi-point search
- Colony of ants is the population



Ant Colony Optimisation

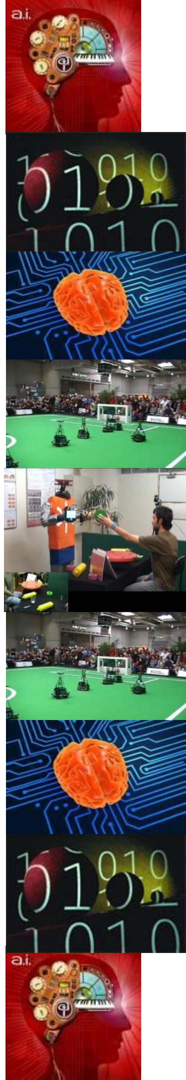
- Used to solve combinatorial optimization problems such as the travelling salesman problem.

- Problem variables have to be discrete.



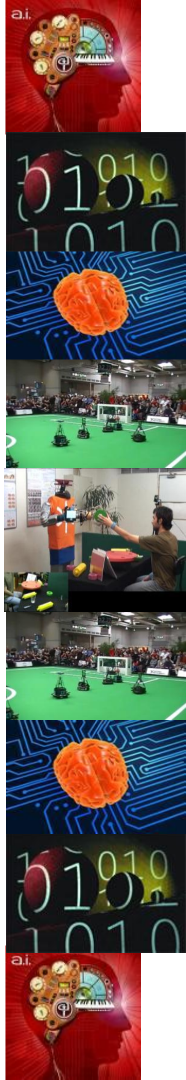
Real-World Ants

- Ants travel from nest to food sources
- Minimize distance travelled instinctively
- Pheromone deposits



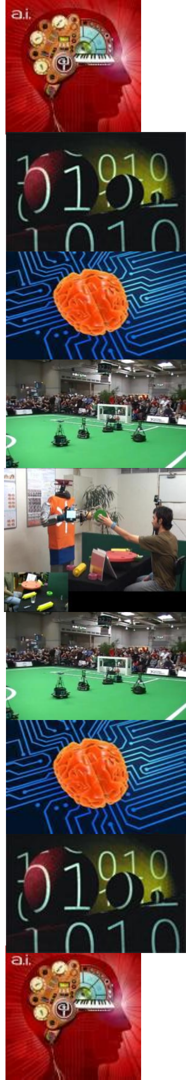
Real-World Ants

- Pheromone trails
- Strong pheromone paths
- Pheromone evaporation



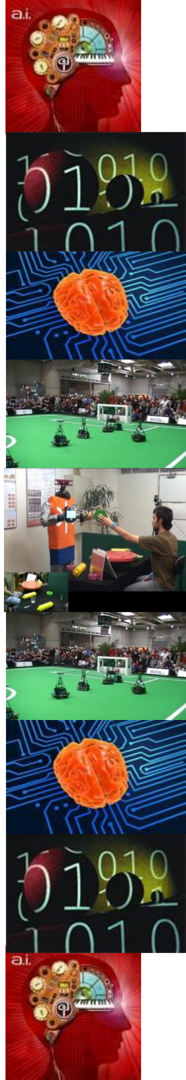
Artificial Ants

- Food collection vs. finding a solution
- Trails vs. solution components
- Pheromone deposits



Artificial Ants

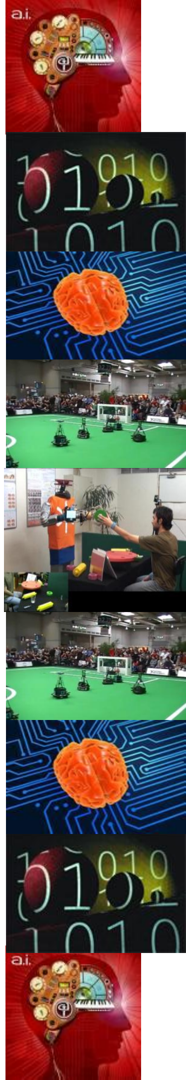
- Ants traverse a construction graph
- Solution components – nodes or edges
- Pheromone and heuristic value



Artificial Colony Optimisation

- Generates candidate solutions by repeated applications of a probabilistic solution construction procedure.

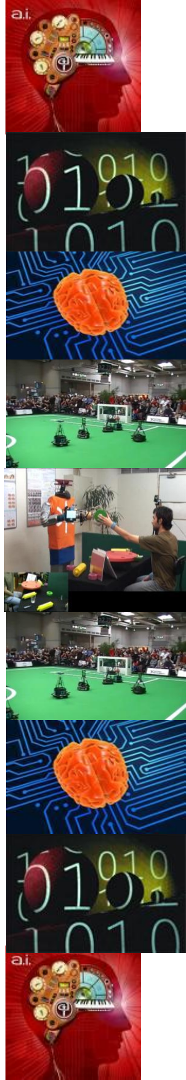
- The ACO algorithm is composed of three parts: **ant-based solution construction, pheromone update**, and **iteration**



Artificial Ants Algorithm

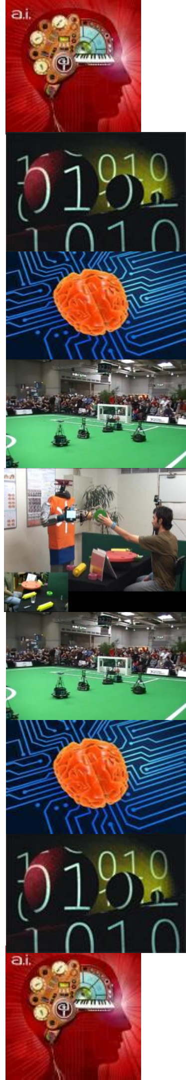
Algorithm 1 ACO Metaheuristic Algorithm

```
Set parameters and initialize pheromone values
while termination criterion is not met do
    Construction ant solutions
    Perform local search
    Update pheromones
end while
```



Artificial Ant Algorithms

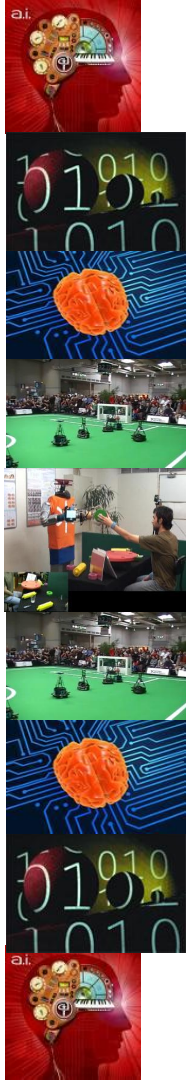
1. Initialize pheromone trails and parameters
2. while stopping criteria not met do
3. for each ant do
4. Construct a solution using pheromone trails and heuristic information
5. Evaluate the solution
6. Update the pheromone trails based on the quality of the solution
7. end for
8. Update the pheromone trails globally
9. end while
10. Return the best solution found



ACO Probability & Update

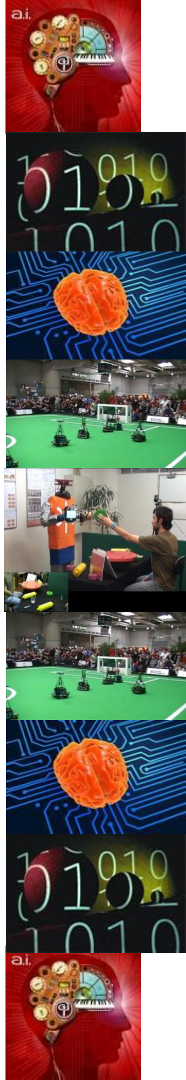
$$p(c_{ij}|s^p) = \frac{\tau_{ij}^{\alpha} * \eta_{ij}^{\beta}}{\sum_{c_{il} \in N(s^p)} \tau_{il}^{\alpha} * \eta_{il}^{\beta}}, \forall c_{il} \in N(s^p) \qquad (1)$$

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho \sum_{s|c_{ij}} F(s) \qquad (2)$$



ACO Parameters

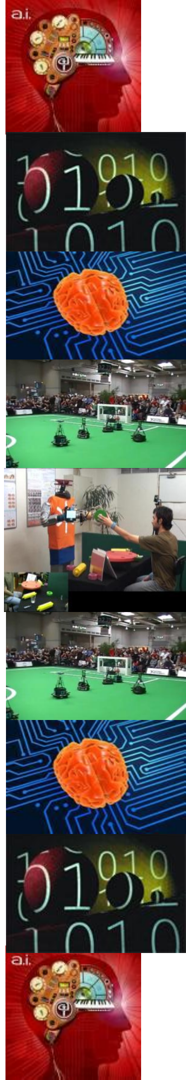
- **Number of Ants**
- **Pheromone Evaporation Rate:**
- **Pheromone Intensity:**
- **Heuristic Information:**
- **Ant Decision Rule:**
- **Local Search Strategy:**
- **Termination Criteria:**



Variations ACO

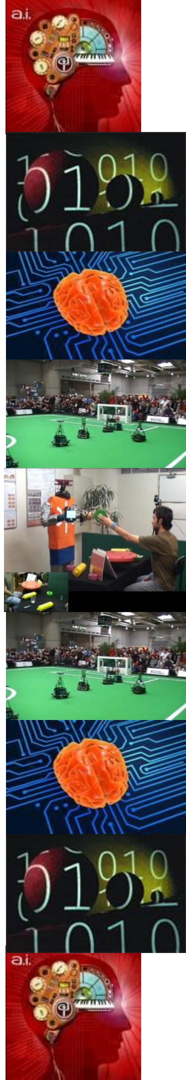
Variants differ on how they manipulate the parameters.

- Ant System
- MAX-MIN Colony
- Ant Colony System



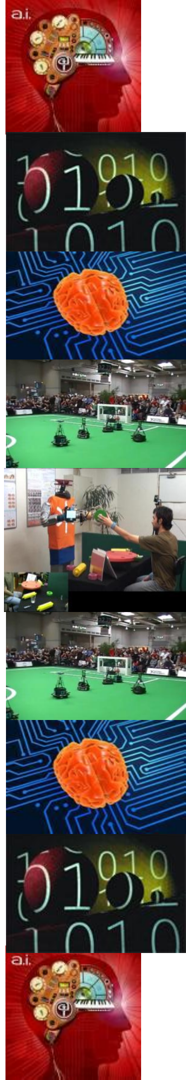
Applications of ACO

- Routing and Transportation
- Telecommunications
- Manufacturing and Production



- QUESTIONS ????

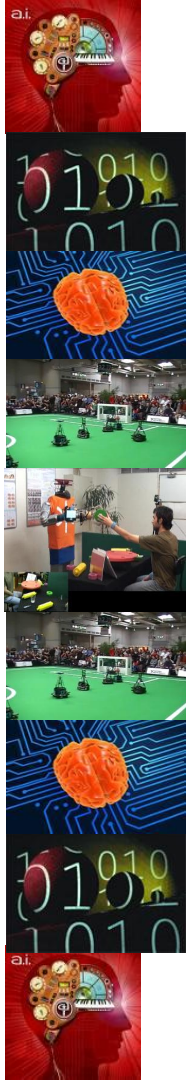
- My ???? — What controls exploration and exploitation.



Particle Swarm Optimisation

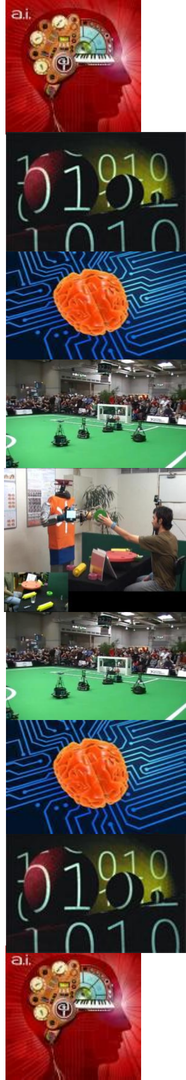
Introduction

- Based on the behaviour of flocks or swarms, e.g. fish, birds, bees.
- Perform optimization.
- Population based – multipoint search.
- Is a metaheuristic.
- Each candidate solution is a particle.



Introduction

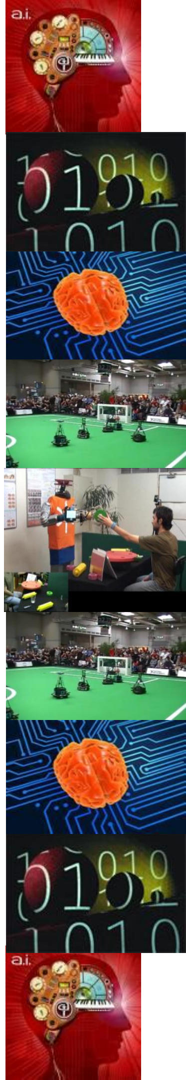
- Each particle has a:
  - Position
  - Velocity
  - Fitness value
- Particles move through the search space
- Movement based on:
  - Particle position and velocity
  - Local and global best positions



PSO Algorithm

Algorithm 1 PSO Algorithm

Initialize all x_i, v_i and $pbest_i$ values
while termination criterion is not met **do**
    for $i \leftarrow 1, N$ **do**
        Calculate $F(x_i)$
        if $F(x_i) < F(pbest_i)$ **then**
            $pbest_i = x_i$
        end if
        if $F(x_i) < F(g_{best})$ **then**
            $g_{best} = x_i$
        end if
        Update all v_i and x_i values using equation 1 and 2
    end for
end while



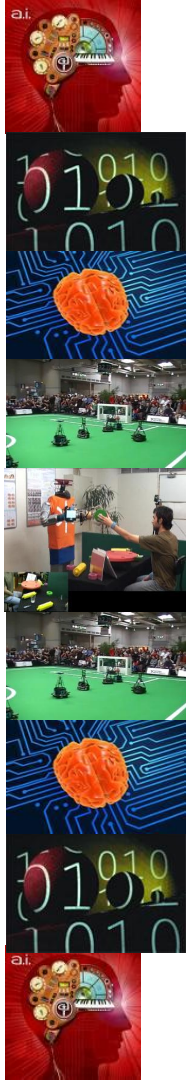
Update Equations

Equation (1) - Velocity update equation

$$v_i(t+1) = w * v_i(t) + l_1 * r_1[pbest_i(t) - x_i(t)] + l_2 * r_2[g_{best} - x_i(t)]$$

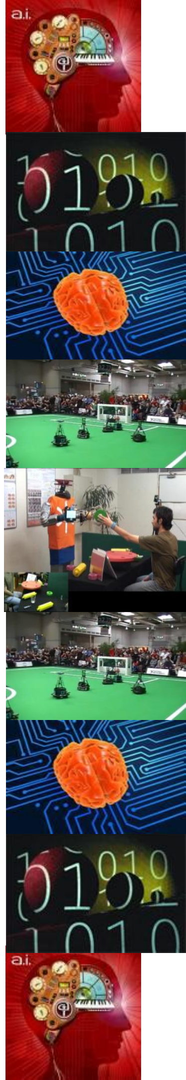
Equation (2) - Position update equation

$$x_i(t+1) = x_i(t) + v_i(t+1)$$



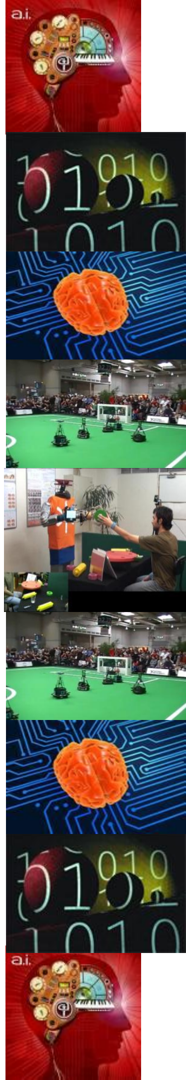
PSO-Parameters

- Swarm size - number of particles
- Vmax - the maximum velocity permitted (explore and exploit)
- Inertia weight w
- Learning factors l_1 and l_2
- Termination criterion, e.g. maximum number of iterations, minimum error.



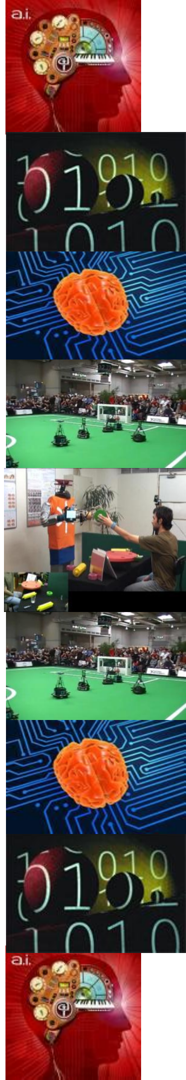
PSO Variations

- Initial values of positions and velocities
- Update rules, e.g. update the local and global position only after the swarm is updated.
- Velocity update equation



PSO Applications

- Engineering:
- Finance:
- Machine Learning:



QUESTIONS