Artificial Intelligence

Evolutionary Algorithms

Lesson 4: Evolutionary Algorithm Meta Heuristics

Vincenzo Piuri

Università degli Studi di Milano

Contents

- Swarm- and population-based optimization
- Population-based incremental learning
- Particle Swarm Optimization
- Ant Colony Optimization

Swarm- and Population-Based Optimization (1)

- Swarm intelligence
  - Part of developing intelligent multi-agent systems
  - Inspired by the behavior of certain species
    - Social insects (e.g. ants, termites, bees etc.)
    - Animals living in swarms (e.g. fish, birds etc.)

Swarm- and Population-Based Optimization (2)

- Main idea
  - Generally quite simple individuals with limited skills
  - Self-coordinated without central control
  - Individuals exchanging information (cooperation)

Swarm- and Population-Based Optimization (3)

- Techniques
  - Genetic/Evolutionary Algorithms
    - Biological pattern: evolution of life
    - Exchange of information by recombination of genotypes
    - Every individual serves as a candidate solution
  - Population Based Incremental Learning
    - Biological pattern: evolution of life
    - Exchange of information by prevalence in population
    - Every individual serves as a candidate solution

Swarm- and Population-Based Optimization (4)

- Particle Swarm Optimization
  - Biological pattern: foraging of fish or bird swarms for food
  - Exchange of information by aggregation of single solutions
  - Every individual serves as a candidate solution
- Ant Colony Optimization
  - Biological pattern: ants searching a route for food
  - Exchange of information by manipulating their environments
  - Individuals generate a candidate solution

Population-based Incremental Learning (1)

- Genetic algorithm without population
  - Instead: only store population statistics
- Specific individuals are generated randomly according to the statistical frequency

Population-based Incremental Learning (2)

- Recombination: uniform crossover
  - Implicitly when generating an individual
- Selection: choosing the best individuals B for updating the population statistics
- Mutation: bit-flipping
  - Slightly random changes within the population statistics

Population-based Incremental Learning (3)

- Typical parameters
  - learning rate α
    - Low: emphasizes exploration
    - High: emphasizes fine tuning

parameter	co-domain
population size λ	20–100
learning rate α	0.05–0.2
mutation rate p_m	0.001–0.02
mutation constant β	0.05

Population-based Incremental Learning (4)

- Problems
  - Algorithm might learn dependencies between certain single bits
  - Algorithm considers single bits isolated from each other
  - Same population statistics can represent different populations

population 1					population 2			
1	1	0	0	individual 1	1	0	1	0
1	1	0	0	individual 2	0	1	1	0
0	0	1	1	individual 3	0	1	0	1
0	0	1	1	individual 4	1	0	0	1
0.5	0.5	0.5	0.5	population statistics	0.5	0.5	0.5	0.5

Population-based Incremental Learning (5)

- Alternatives
  - Better techniques for estimating the distribution of beneficial candidate solutions
  - Modelling of internal dependencies

Particle Swarm Optimization (1)

- Fish or birds are searching for rich food resources in swarms
- Orientation based on individual search (cognitive part) and other individuals close to them within the swarm (social part)
- Living within a swarm reduces the risk of getting eaten by a predator

Particle Swarm Optimization (2)

- Motivation
  - Behavior of swarms when searching for food: randomly swarming out, but always returning to the swarm to exchange information with the other individuals
- Approach
  - Use a “swarm” of m candidate solutions instead of single ones

Particle Swarm Optimization (3)

- Procedure
  - Take every candidate solution as a “particle” searching for food at the position x_i with a velocity of v_i
  - Combine elements of ground-oriented search (e.g. gradient descent approach) and population-based search (e.g. evolutionary algorithms)

Particle Swarm Optimization (4)

- Procedure
  - Update for position and velocity of particle i

$$\mathbf{v}_i(t+1) = \alpha \mathbf{v}_i(t) + \beta_1 \left(\mathbf{x}_i^{(\text{local})}(t) - \mathbf{x}_i(t) \right) + \beta_2 \left(\mathbf{x}^{(\text{global})}(t) - \mathbf{x}_i(t) \right)$$
$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t)$$

  - $x_i^{(local)}$ is the local memory of an individual (particle)
    - The best coordinates being visited by this individual within the search space
  - $x_i^{(global)}$ is the global memory of the swarm
    - The best coordinates being visited by any individual of the swarm within the search space (best solution so far)

Particle Swarm Optimization (5)

- Extensions
  - Reduced search space
  - Local environment of a particle
    - Use best local memory of a single particle instead of global swarm memory
      - » e.g. particles surrounding the currently updated one
  - Automatic parameter adjustment
      - » e.g. changing the swarm size
  - Diversity control
    - Prevent early convergence to suboptimal solutions
      - » e.g. by introducing a new random number for updating the speed to increase diversity

Ant Colony Optimization (1)

- Since food has to be fetched from its source and carried to the nest, ants form transportation roads
- They label all their routes with scents (pheromones) for other ants may then trace their routes
- Routes to food sources are minimized

Ant Colony Optimization (2)

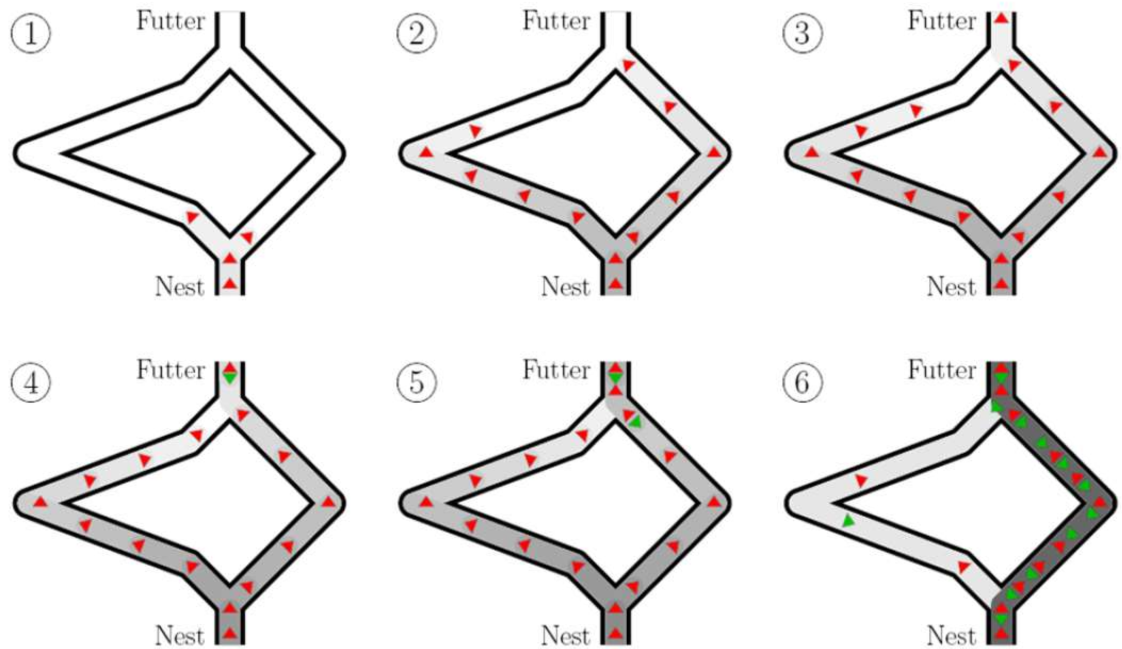
- Motivation
  - Some ant species are able to find shortest route to food sources by placing and tracing pheromones (scents)
  - Intuitively: short routes are labeled with more pheromone during the same time
  - Routes are randomly chosen according to the current pheromone
  - Distribution: the more pheromone there is, the more probable is it for ants to choose this way
  - The amount of pheromone might vary according to the quality and amount of food found

Ant Colony Optimization (3)

- Main principle: stigmergy
  - Ants are communicating implicitly by placing pheromones
  - Stigmergy (indirect communication by changing the environmental circumstances) allows for globally adapted behavior due to locally found information

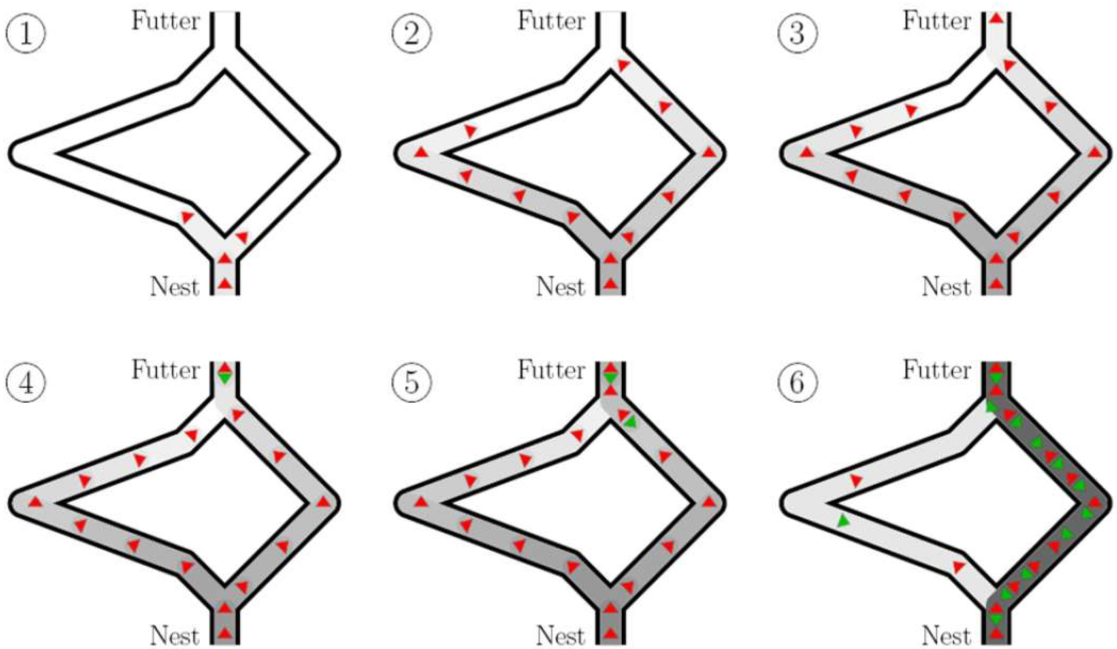
Double-Bridge Experiment (1)

- Ant nest and food source are connected by 2 bridges that differ in length



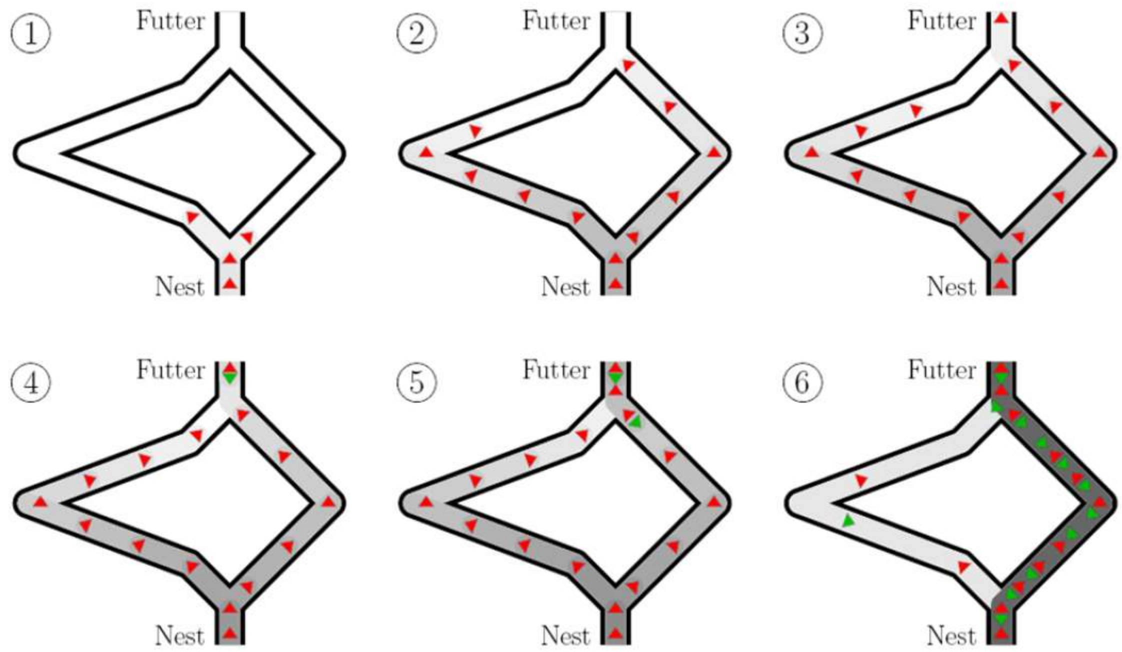
Double-Bridge Experiment (2)

- Experiment run with Argentinian Ants Iridomyrmex
  - These ants are almost blind so they cannot “see“ which bridge is shorter



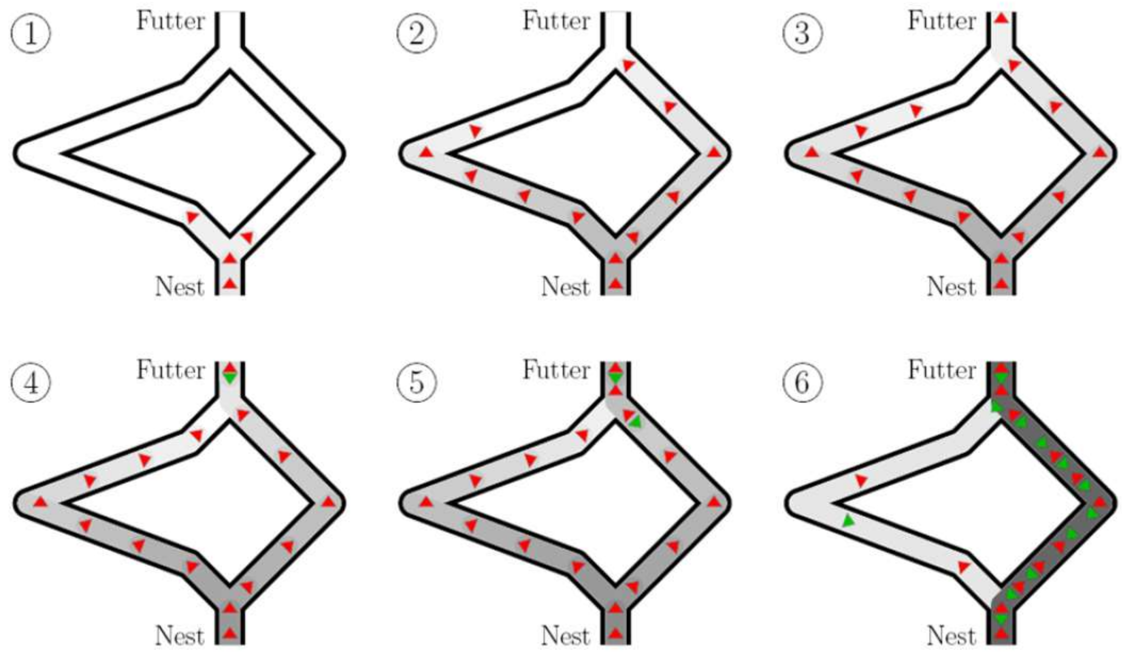
Double-Bridge Experiment (3)

- In most runs: after just several minutes most ants were using the shorter bridge



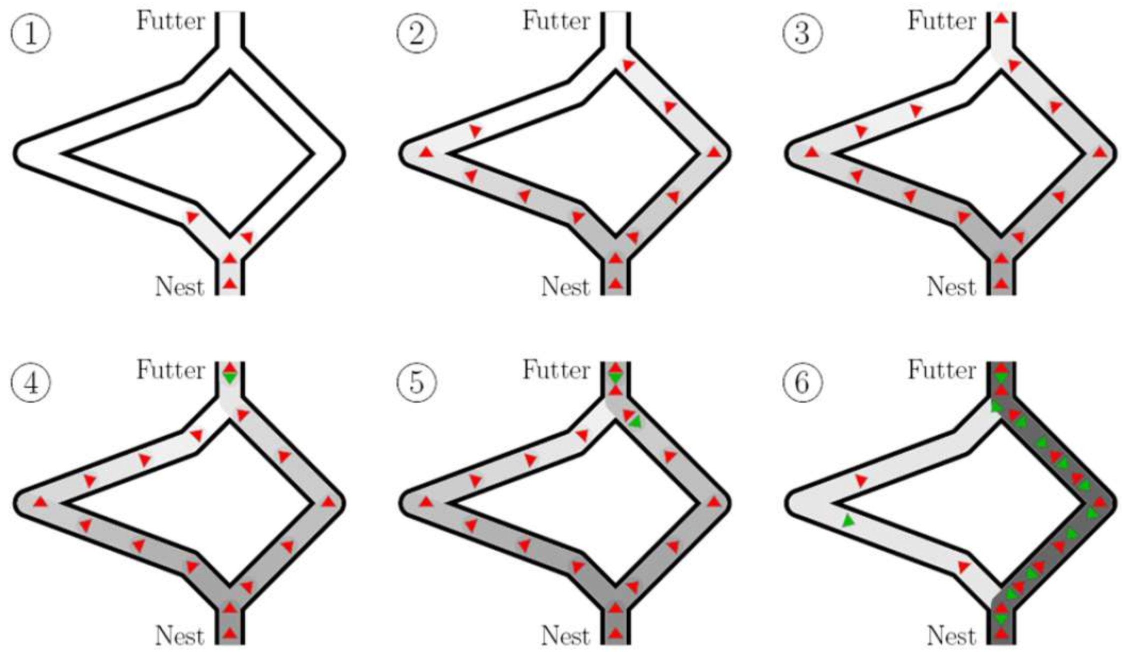
Double-Bridge Experiment (4)

- Ants travelling the shorter bridge are reaching the food earlier so in the first place the end of the shorter bridge gets more pheromon



Double-Bridge Experiment (5)

- When returning to the nest, choosing the shorter bridge again is more probable for it is labeled with more pheromone now, thus increasing the difference in pheromone even more



Double-Bridge Experiment (6)

- Shorter route is intensified automatically (autocalysis)
  - More pheromon ←→ more ants will choose this route
- Ants are able to find shortest path only because they return on it while placing pherormons again
  - At the nest there is no way to decide between both paths as there is no difference in pheromone
  - At the junction of both bridges the ration decreases slowly and finally disappears

Double-Bridge Experiment (7)

- What if a new even shorter route is added later by changing the environment?
  - Once a solution route has been established, the ants will stick to it
  - Proof by a second bridge experiment
    - Initializing the experiment with only one (longer) path, later adding a second (shorter) one
    - Most ants go on using the longer path, only few ants change

Natural and Artificial Ants (1)

- Search for the best path within a weighted graph
- Problem: self-intensifying cycles
  - Being visited by an ant a cycle becomes even more probable to be again visited by an ant
- Solution: labelling routes only after the ant has completed its whole tour
  - Cycles may be removed from the path
- Problem: early convergence to a candidate solution found in the very beginning
- Solution: pheromone evaporation

Natural and Artificial Ants (2)

- Extensions/improvements
  - Amount of pheromone dependent on the quality of the solution
  - Considering heuristics when choosing graph edges (e.g. their weights)

Algorithm for Ant Colony Optimization (1)

- Preconditions
  - Combinatorial optimization problem with constructive method for creating a solution candidate
- Procedure
  - Solutions are constructed according to a sequence of random choices, where every choice extends a partial solution
  - Sequence of choices = path in a decision graph
  - Ants ought to explore the paths through a decision graph and find the best (shortest, cheapest) one
  - Ants label the graph edges with pheromone
    - Other ants will be guided towards promising solutions
  - Pheromone "evaporates" after every iteration so once placed it won't affect the system for too long ("forgetting" outdated information)

Algorithm for Ant Colony Optimization (2)

Application to the TSP

- represent the problem by $n \times n$ matrix $\mathbf{D} = (d_{ij})_{1 \leq i,j \leq n}$
- n cities with distances d_{ij} between city i and j
- note: $\mathbf{D}$ may be asymmetrical, but $\forall i \in \{1, \ldots, n\} : d_{ii} = 0$
- pheromone information as $n \times n$ matrix $\Phi = (\phi_{ij})_{1 \leq i,j \leq n}$
- pheromone value $\phi_{ij} (i \neq j)$ indicates the desirability of visiting city j directly after visiting city i (ϕ_{ii} not used)
- there is no need in keeping Φ symmetrical
- initialize all ϕ_{ij} with the same small value (same amount of pheromone on all edges in the beginning)
- ants run Hamilton tour by labelling the edges of the Hamilton tour with pheromone (with the added pheromone value corresponding to the quality of the found solution)

Algorithm for Ant Colony Optimization (3)

Application to TSP: Constructing a solution

1. Every ant possesses a “memory” C where indices of not-yet visited cities are stored
   - Every visited city is removed from the set C
   - No such memory in the nature
2. Ant is put randomly to a city where it begins its cycle
3. Ant chooses not-yet visited city and goes there
   - Chooses a (not-yet visited) city j with probability

$$p_{ij} = \frac{\phi_{ij}}{\sum_{k \in C} \phi_{ik}}.$$

4. Repeat step 2 until every city has been visited

Algorithm for Ant Colony Optimization (4)

- Updating the pheromone
  - Evaporation
    - Reduced by a fraction η (evaporation)
  - Intensifying a constructed solution
    - Pheromone is put on all edges of the constructed solution corresponding to its quality
    - The better the solution, the more pheromone is added.

Algorithm for Ant Colony Optimization (5)

- Extensions
  - Prefer nearby entities
  - Tend to choosing the best edge (greedy)
  - Intensify best known solution (elitism)

Algorithm for Ant Colony Optimization (6)

- Extensions (cont’d)
  - Ranking based updates
    - Place pheromone only on edges of last iteration’s best solution
    - Amount of pheromone depends on the rank of the solution
  - Strict elite principles
    - Place pheromone only on the last iteration’s best solution
    - Place pheromone only on the best solution found so far

Algorithm for Ant Colony Optimization (7)

- Extensions (cont'd)
  - Minimal/maximal amount of pheromone
    - Set an upper or lower limit of pheromone for the edges
    - Sets an upper or lower limit for the probability of choosing an edge
    - Better search space exploration, but might lead to worse convergence
  - Limited evaporation
    - Pheromone evaporates only on edges, that have been used during this iteration
    - Better search space exploration

Improving a Tour Locally

- Considering local improvements of a candidate solution is promising
  - Before updating the pheromone, the generated tour is optimized locally
- Local optimizations
  - Recombination after removing 2 edges (2-opt)
  - Recombination after removing 3 edges (3-opt)
  - Limited recombination (2.5-opt)
  - Exchanging neighboring nodes
  - Permutation of neighboring node-triplets
- Apply “expensive” local optimization only to the best solution found so far

Application to Optimization Problems (1)

- Idea
  - Problem as searching a (decision) graph, with the candidate solutions being described by sets of edges
- General description
  - Nodes and edges of the decision/construction graph
  - Constraints
  - Significance of the pheromone on edges
  - Useful heuristics
  - Generation of a candidate solution

Application to Optimization Problems (2)

Example: TSP

- Nodes and edges of the decision/construction graph
  - The cities to be visited, and their weighted connections
- Constraints
  - Visit every city exactly once
- Meaning of pheromone on the edges
  - The desirability of visiting city j right after city i
- Useful heuristics
  - Distances between the cities, prefer close cities
- Generation of a candidate solution
  - Starting at a randomly chosen city always progress to another, not-yet visited city

Application to Optimization Problems (3)

Example: General Assignment Problem

- Assign n tasks to m working units minimizing the sum of assignment costs with respect to the maximal capacity
- Every task and every working unit = node of the construction graph
  - Edges are labeled with the costs of assignment
- Every task has to be assigned to exactly one working unit without exceeding their capacity
- Pheromones upon the edges are used for describing the desirability of assigning a task to a working unit
- Choose edges step by step, not necessarily creating a path. Skip edges of tasks that have already been assigned
- Penalize candidate solutions that violate constraints (e.g. by raising costs)

Application to Optimization Problems (4)

Example: Knapsack Problem

- Choose a subset of maximal value from a set of n objects with a volume, with respect to an upper limit for volume
- Every object = node within the construction graph, labeled with their value

Application to Optimization Problems (5)

- Swarm and population-based algorithms: heuristics for solving optimization problems
- Purpose: finding a good approximation of the solution
- Attempt to reduce the problem of local optima by improving exploration of the search space)
- Exchange of information between individuals

Application to Optimization Problems (6)

- Particle Swarm Optimization
  - Optimization of a function with real arguments
  - Exchange of information by watching the neighbors
- Ant Colony Optimization
  - Search for best routes
  - Exchange of information: manipulation of the environment (stigmergy)