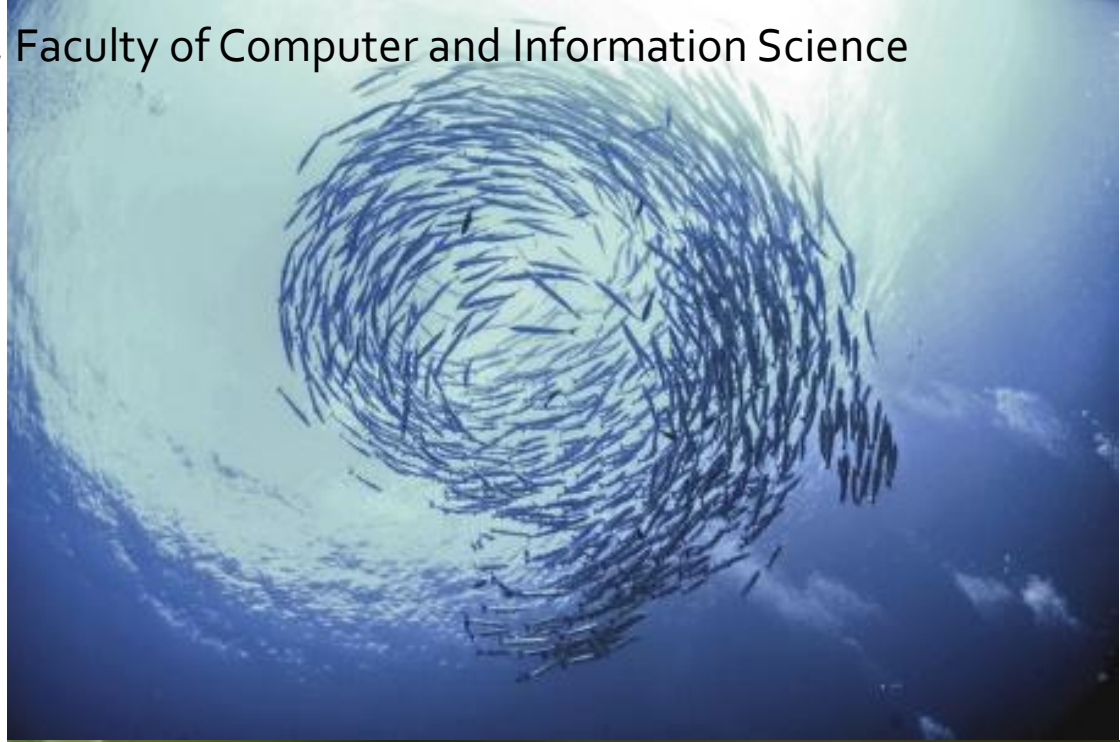


Swarm intelligence

Prof Dr Marko Robnik-Šikonja

Analysis of Algorithms and
Heuristic Problem Solving
Version 2026



Nature inspired methods

- ✿ Besides evolutionary computation, nature is an inspiration for many other computational algorithms.
- ✿ Swarm intelligence (SI) is the collective behavior of decentralized, self-organized systems, natural or artificial.
- ✿ A population of simple agents interacting locally with one another and with their environment.
- ✿ The agents follow very simple rules, and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergence of "intelligent" global behavior, unknown to the individual agents.
- ✿ Examples in natural systems of SI include ant colonies, bird flocking, animal herding, bacterial growth, fish schooling and microbial intelligence.



Computational SI

✿ Computational properties

- ✧ Fixed population
- ✧ Autonomous individual
- ✧ Communication between agents

✿ We will cover

- ✧ Particle swarm optimization
- ✧ Ant colony optimization

Swarming – the definition

- ✱ Aggregation of similar animals, generally cruising in the same direction
- ✱ Termites swarm to build colonies
- ✱ Birds swarm to find food
- ✱ Bees swarm to reproduce



Swarming is powerful

- ✿ Swarms can achieve things that an individual cannot



Human swarms



Powerful ... but simple

All evidence suggests:

- ✿ No central control
- ✿ Only simple rules for each individual
- ✿ Emergent phenomena
- ✿ Self-organization

Harness this power out of simplicity

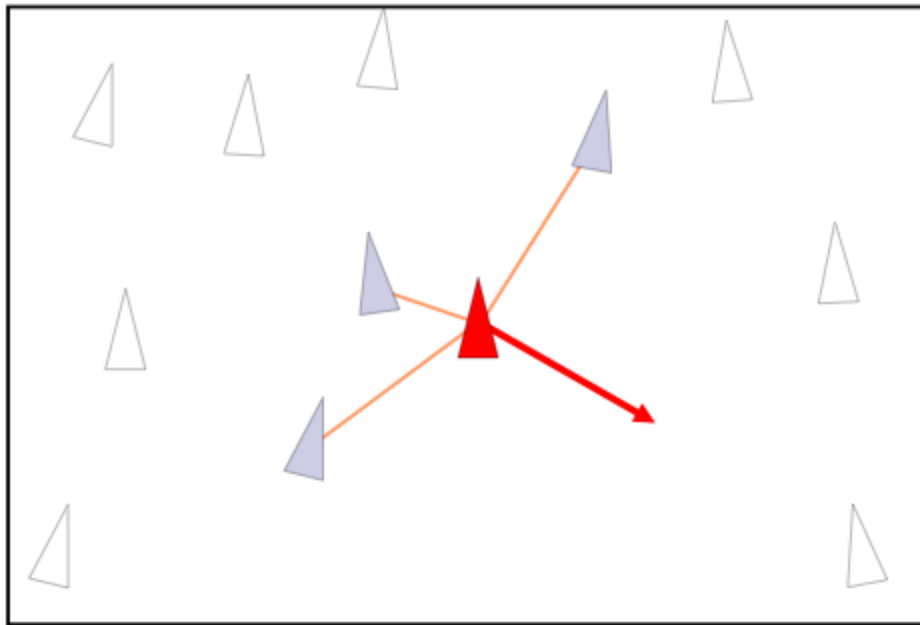
- ★ Technical systems are getting larger and more complex
 - ✧ Global control hard to define and program
 - ✧ Larger systems lead to more errors
- ★ Swarm intelligence systems are:
 - ✧ Robust
 - ✧ Relatively simple (How to program a swarm?)

Swarming – example

- ✿ Bird flocking
- ✿ “Boids” model was proposed by Reynolds (1985)
 - ✧ Boids = Bird-oids (bird like)
- ✿ Only three simple rules

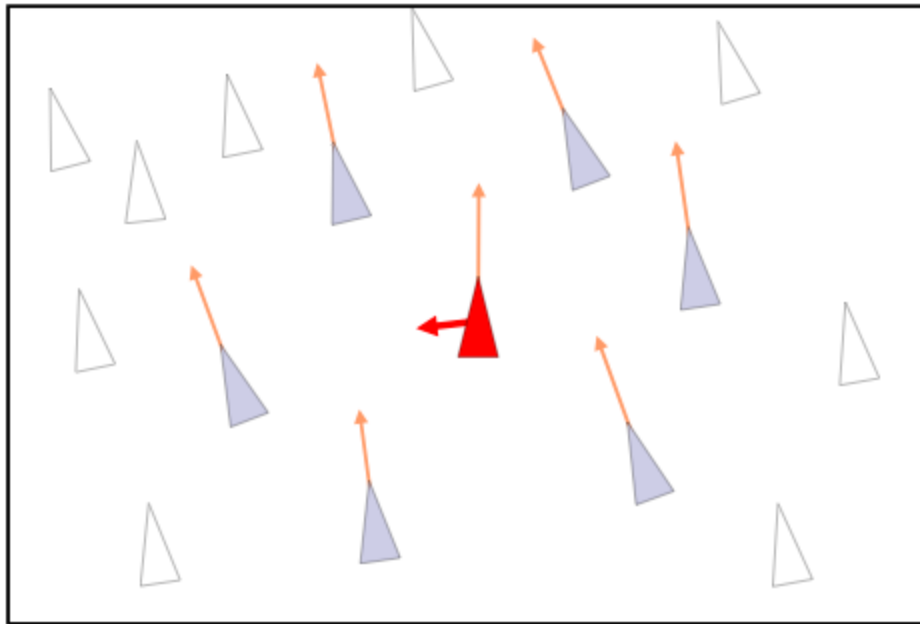
Collision Avoidance

- ✱ Rule 1: Avoid Collision with neighboring birds



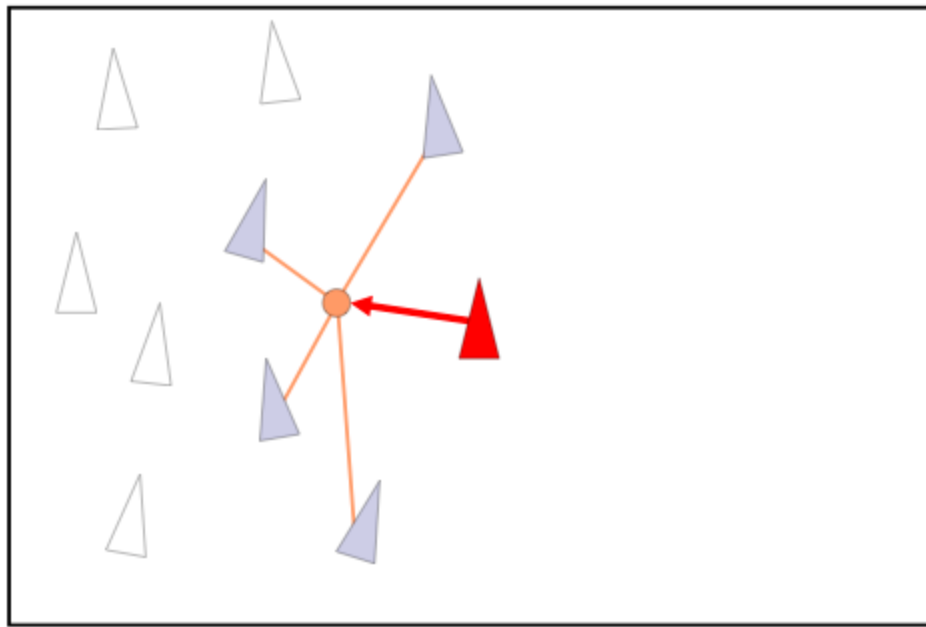
Velocity matching

- ✱ Rule 2: Match the velocity of neighboring birds



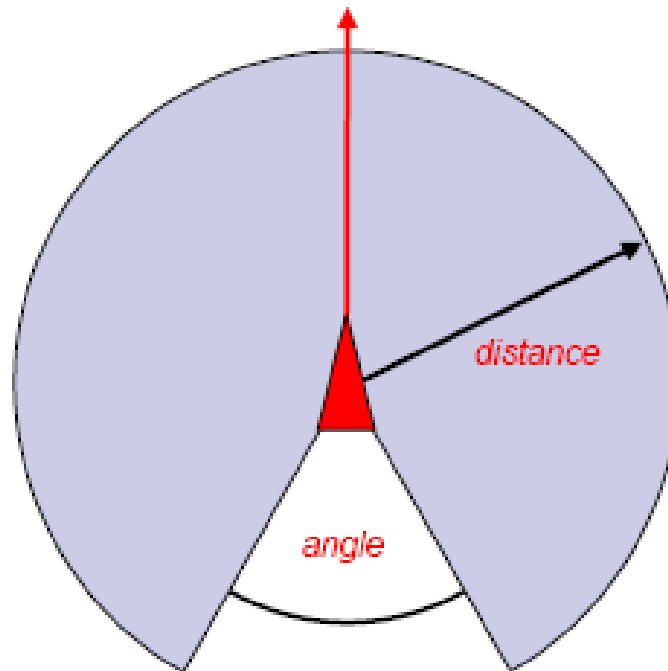
Flock centering

- ✱ Rule 3: Stay near neighboring birds



Define the neighborhood

- ✱ Model the view of a bird
- ✱ Only local knowledge, only local interaction
- ✱ Affects the swarm behavior (fish vs. birds)



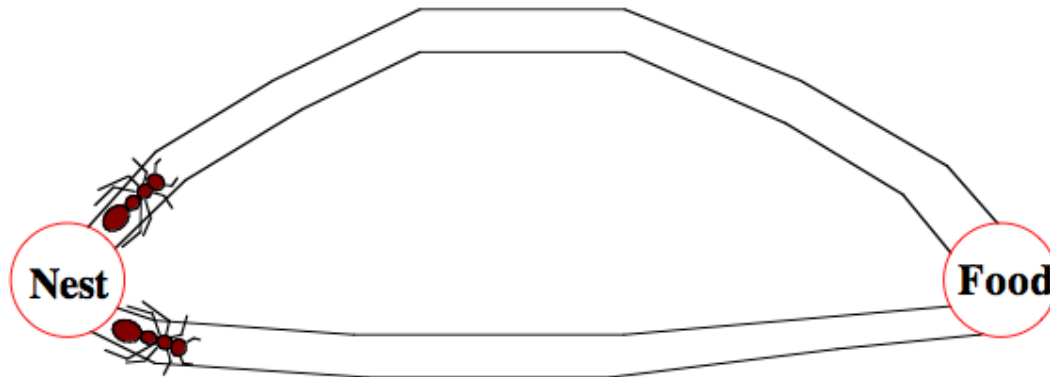
Swarming - characteristics

- ✿ Simple rules for each individual
- ✿ No central control
 - ✗ Decentralized and hence robust
- ✿ Emergent
 - ✗ Performs complex functions

Ant Colony Optimization - Biological Inspiration

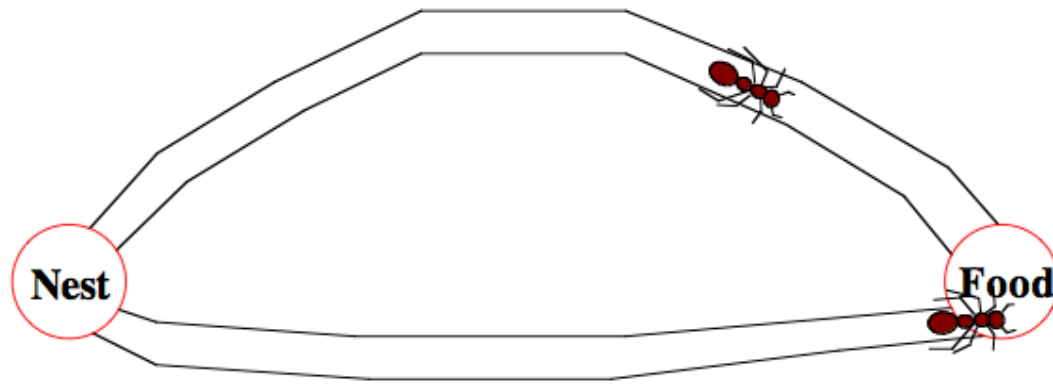
- ✿ Inspired by foraging behavior of ants.
- ✿ Ants find shortest path to food source from nest.
- ✿ Ants deposit pheromone along traveled path which is used by other ants to follow the trail.
- ✿ This kind of indirect communication via the local environment is called stigmergy.
- ✿ Has adaptability, robustness and redundancy.

Foraging behavior of Ants



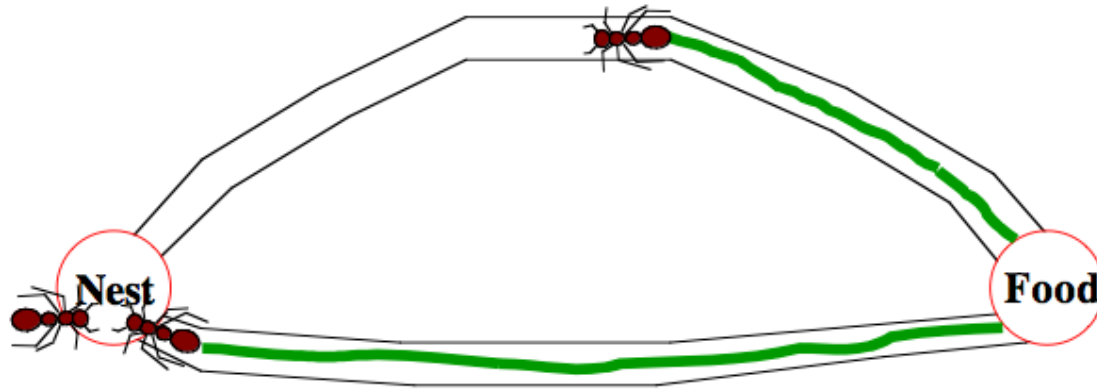
-
- ✱ 2 ants start with equal probability of going on either path.

Foraging behavior of Ants



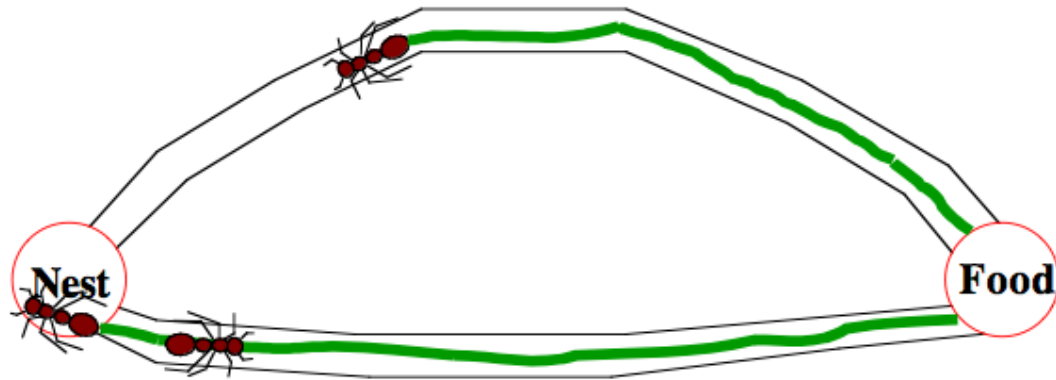
- ✱ The ant on shorter path has a shorter to-and-fro time from it's nest to the food.

Foraging behavior of Ants



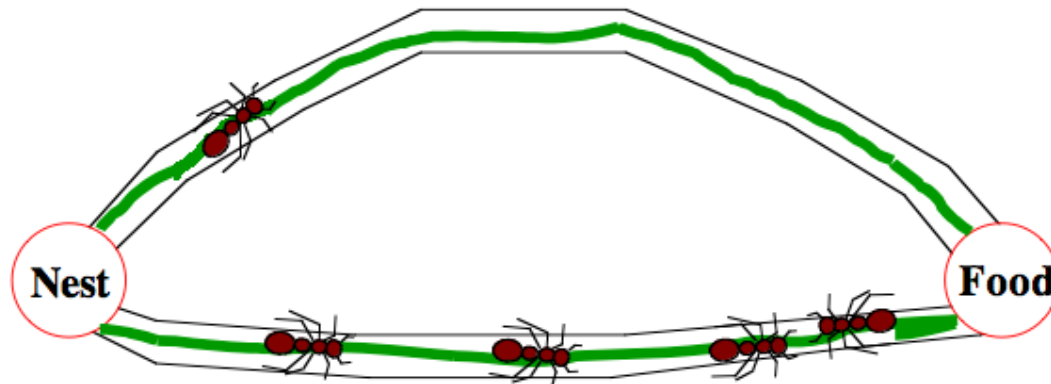
- ✳ The density of pheromone on the shorter path is higher because of 2 passes by the ant (as compared to 1 by the other).

Foraging behavior of Ants



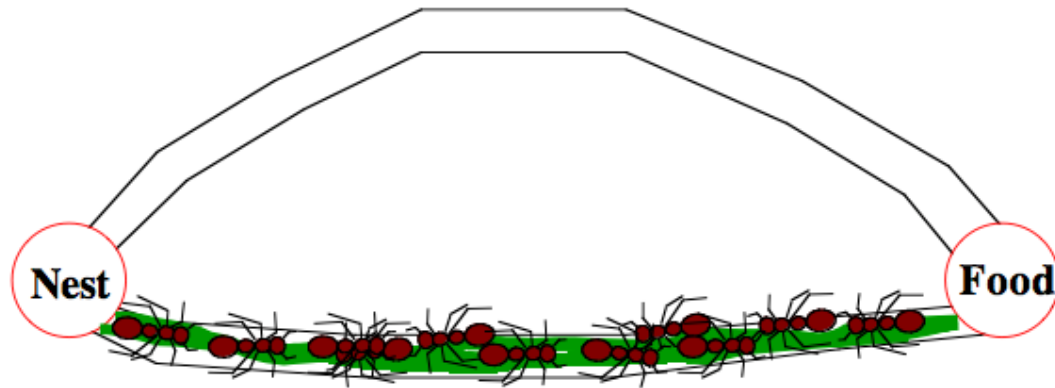
-
- ✱ The next ant takes the shorter route.

Foraging behavior of Ants



- ★ Over many iterations, more ants begin using the path with higher pheromone, thereby further reinforcing it.

Foraging behavior of Ants



-
- ✱ After some time, the shorter path is almost exclusively used.

Ant colony

- ★ Pheromones
- ★ Ants lead their sisters to food source
- ★ Evaporation
- ★ Moving targets



Illustration of the dynamic adaptation

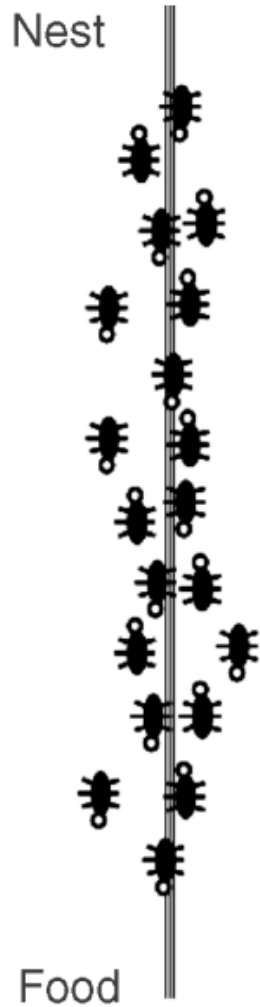


Illustration of the dynamic adaptation

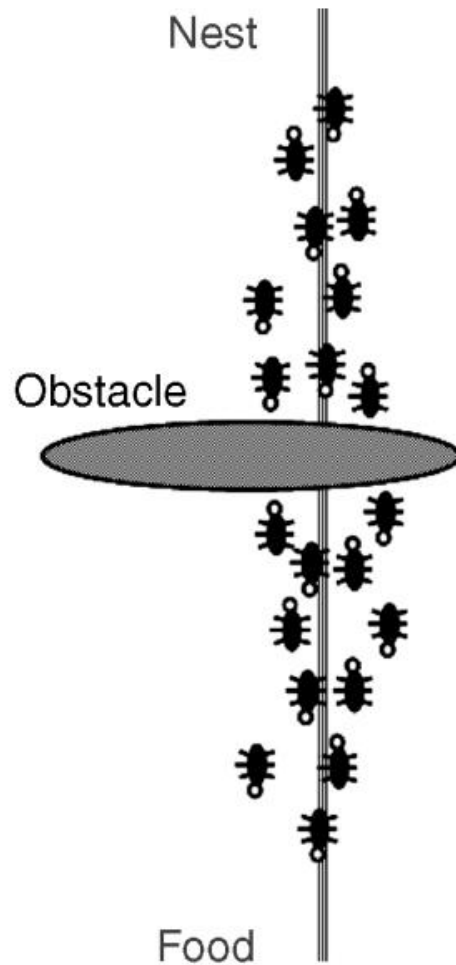


Illustration of the dynamic adaptation

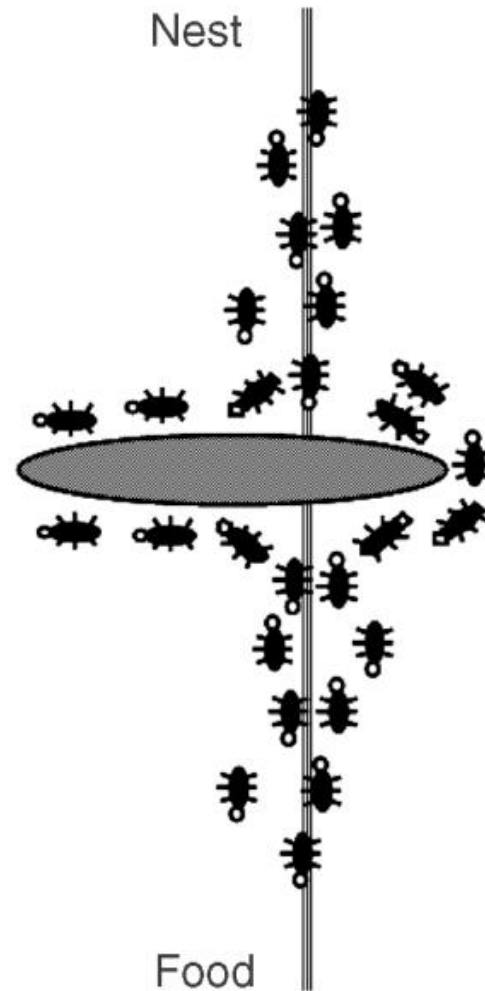


Illustration of the dynamic adaptation

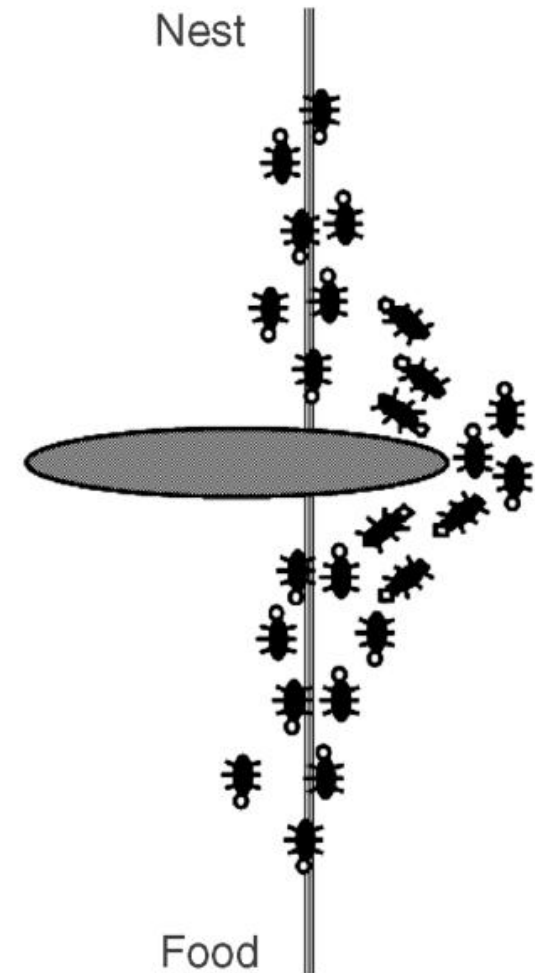
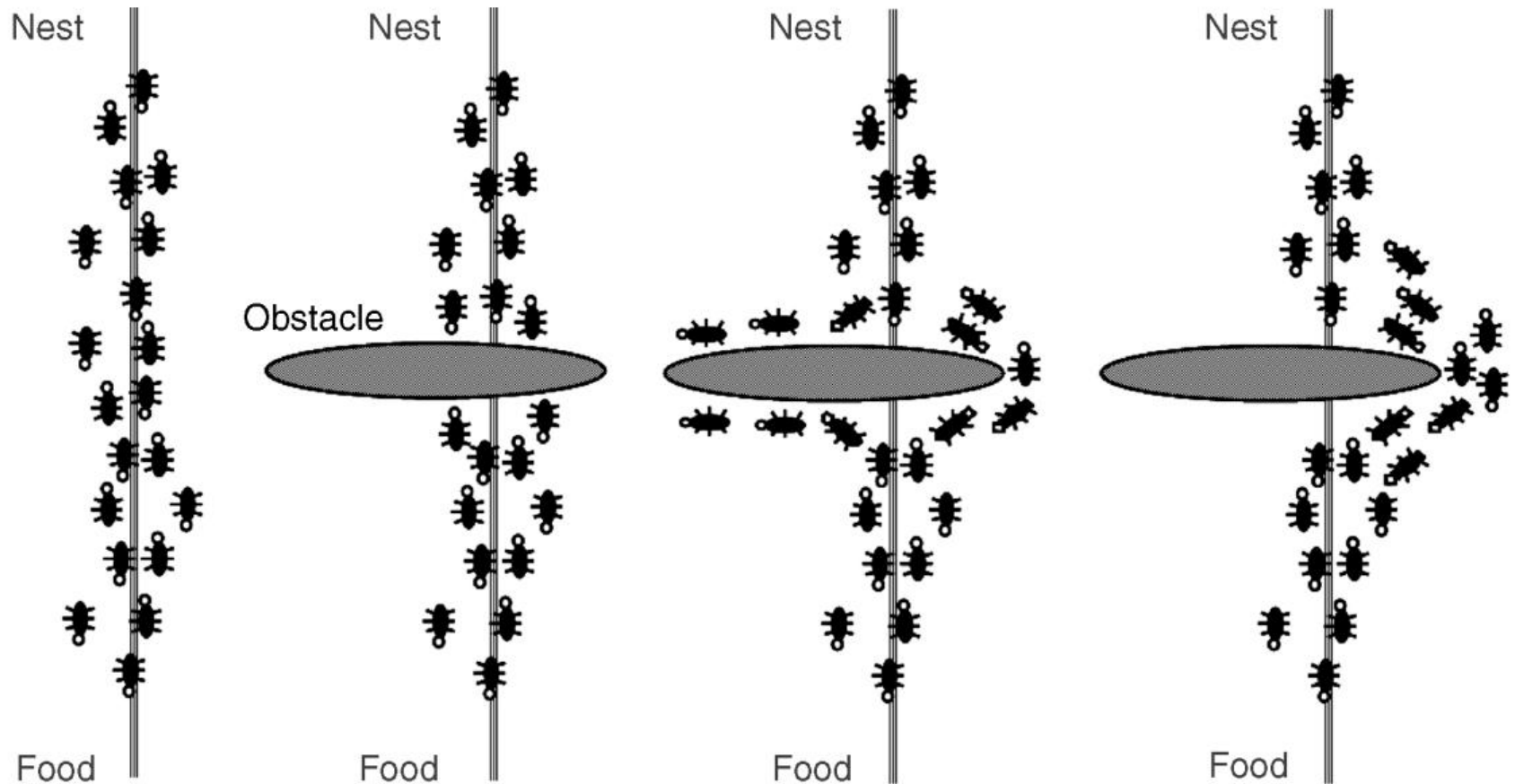


Illustration of the dynamic adaptation



Generic ACO

- ✿ Formalized into a metaheuristic.
- ✿ Artificial ants build solutions to an optimization problem and exchange info on their quality vis-à-vis real ants.
- ✿ A combinatorial optimization problem reduced to a construction graph.
- ✿ Ants build partial solutions in each iteration and deposit pheromone on each edge.

ACO pseudo code

Initialization of pheromones

```
do {  
  for each ant  
    find solution: use pheromones and cost of path to select  
    route  
    apply local optimization (optional)  
    update pheromones: enforcement, evaporation  
} while (! satisfied)  
return best overall solution
```

ACO details

- ★ Pheromones updates
 - ★ ρ speed of evaporation
- ✧ Trails updates
- ✧ Many variants

$$\tau_{i,j} = (1 - \rho)\tau_{i,j} + \Delta\tau_{i,j}$$

$$\Delta\tau_{i,j} = \begin{cases} 1/C & \text{if ant takes the connection between } i,j \\ 0 & \text{otherwise} \end{cases},$$

where C is a cost of edge i,j

ACO for TSP

- ✿ Cities $1, 2, \dots, n$
- ✿ Cost $c_{i,j}$
- ✿ Construct the cheapest Hamiltonian tour through cities
- ✿ Attractiveness $\eta_{i,j} = 1 / c_{i,j}$
- ✿ Probability of ant's transition
 - ✕ α - impact of pheromones
 - ✕ β - impact of transition cost

$$p_{i,j} = \frac{\tau_{i,j}^{\alpha} \eta_{i,j}^{\beta}}{\sum \tau_{i,j}^{\alpha} \eta_{i,j}^{\beta}}$$

A simple TSP example

A

B

C

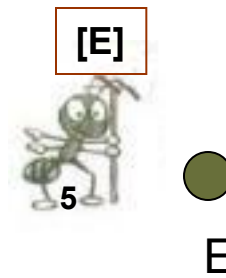
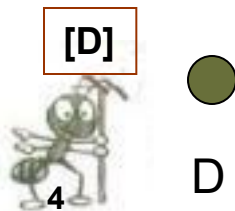
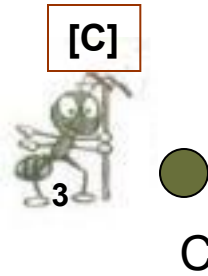
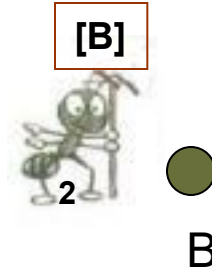
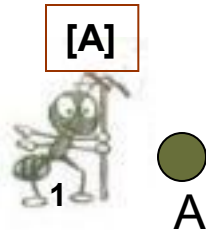
D

E

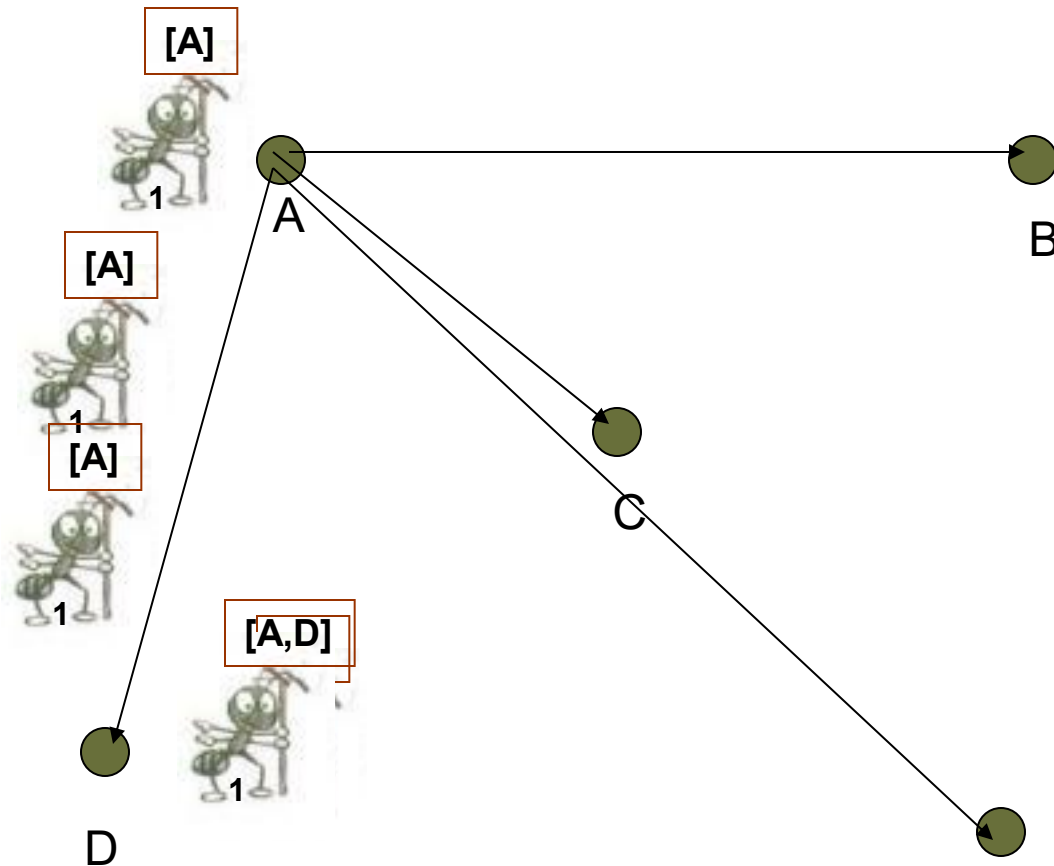
$d_{AB}=100; d_{BC}=60; \dots; d_{DE}=150$



Iteration 1

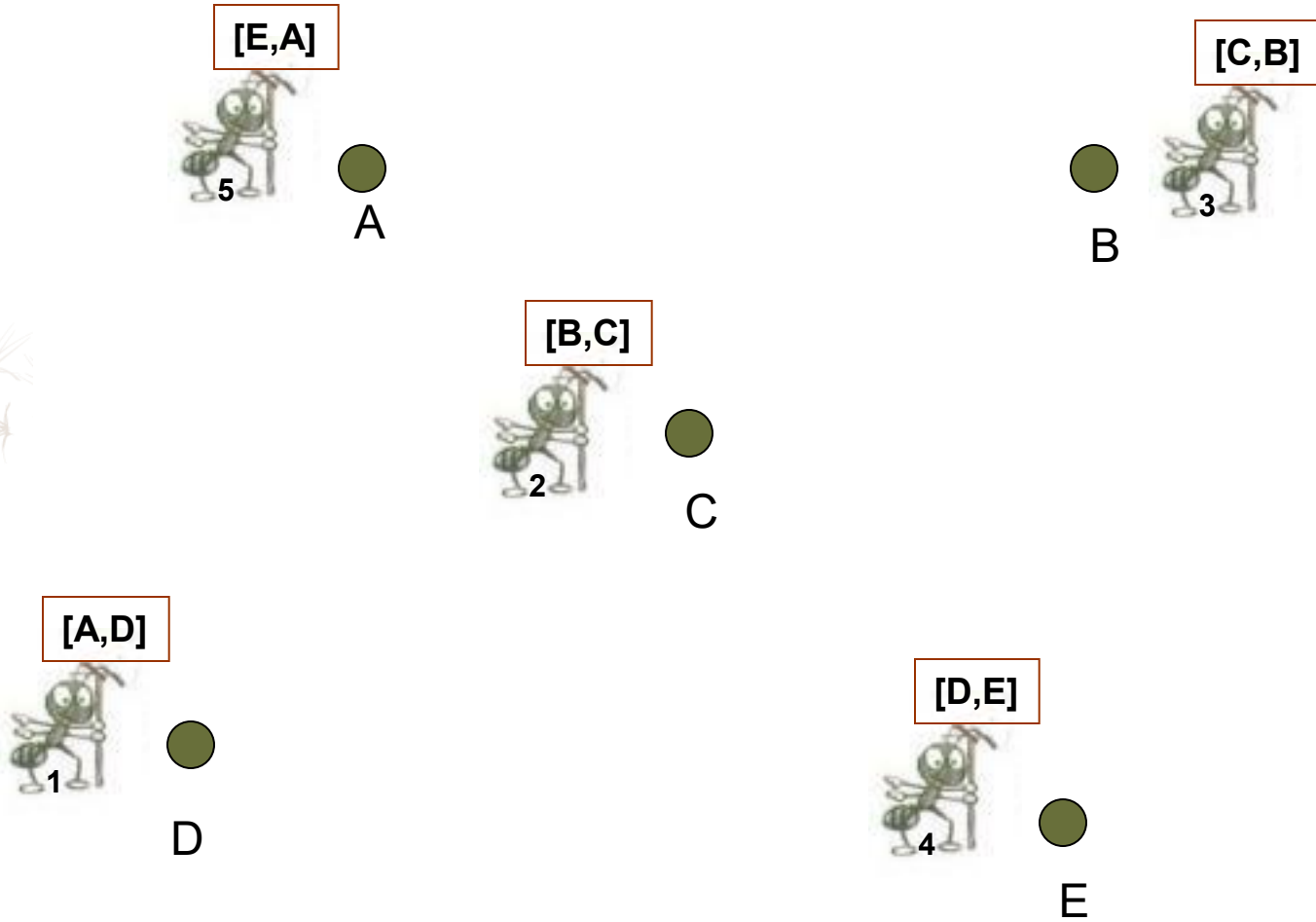


How to build next sub-solution?

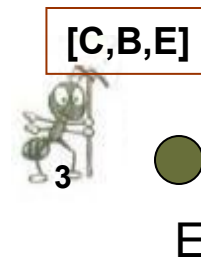
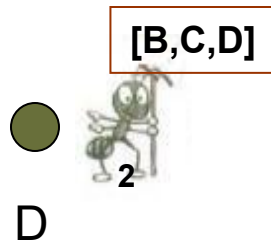
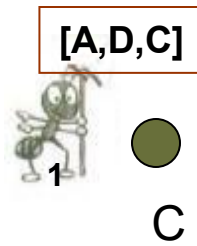
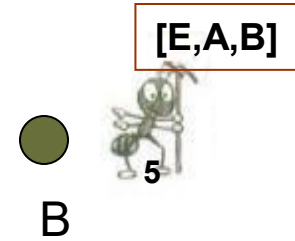
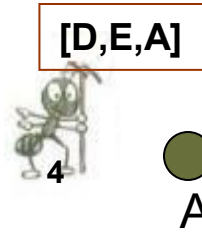


$$p_{ij}(t) = \begin{cases} \frac{\tau_{ij}(t)^\alpha \eta_{ij}^\beta}{\sum_{k \in \text{allowed}_k} \tau_{ik}(t)^\alpha \eta_{ik}^\beta} & \text{if } j \in \text{allowed} \\ 0 & \text{otherwise} \end{cases}$$

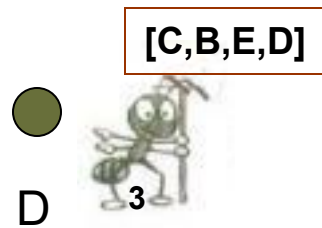
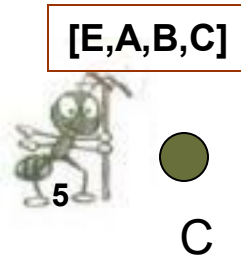
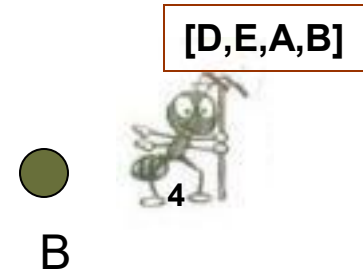
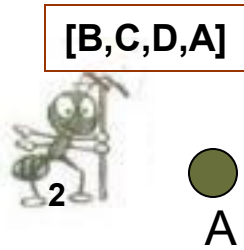
Iteration 2



Iteration 3



Iteration 4



Iteration 5

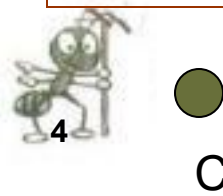
[C,B,E,D,A]



[A,D,C,E,B]



[D,E,A,B,C]



[E,A,B,C,D]



[B,C,D,A,E]



Path and Pheromone Evaluation

[A,D,C,E,B]



$$L_1 = 300$$

[B,C,D,A,E]



$$L_2 = 450$$

[C,B,E,D,A]



$$L_3 = 260$$

[D,E,A,B,C]



$$L_4 = 280$$

[E,A,B,C,D]



$$L_5 = 420$$

$$\Delta\tau_{i,j}^k = \begin{cases} \frac{Q}{L_k} & \text{if } (i,j) \in \text{tour} \\ 0 & \text{otherwise} \end{cases}$$

$$\Delta\tau_{A,B}^{total} = \Delta\tau_{A,B}^1 + \Delta\tau_{A,B}^2 + \Delta\tau_{A,B}^3 + \Delta\tau_{A,B}^4 + \Delta\tau_{A,B}^5$$



End of First Run

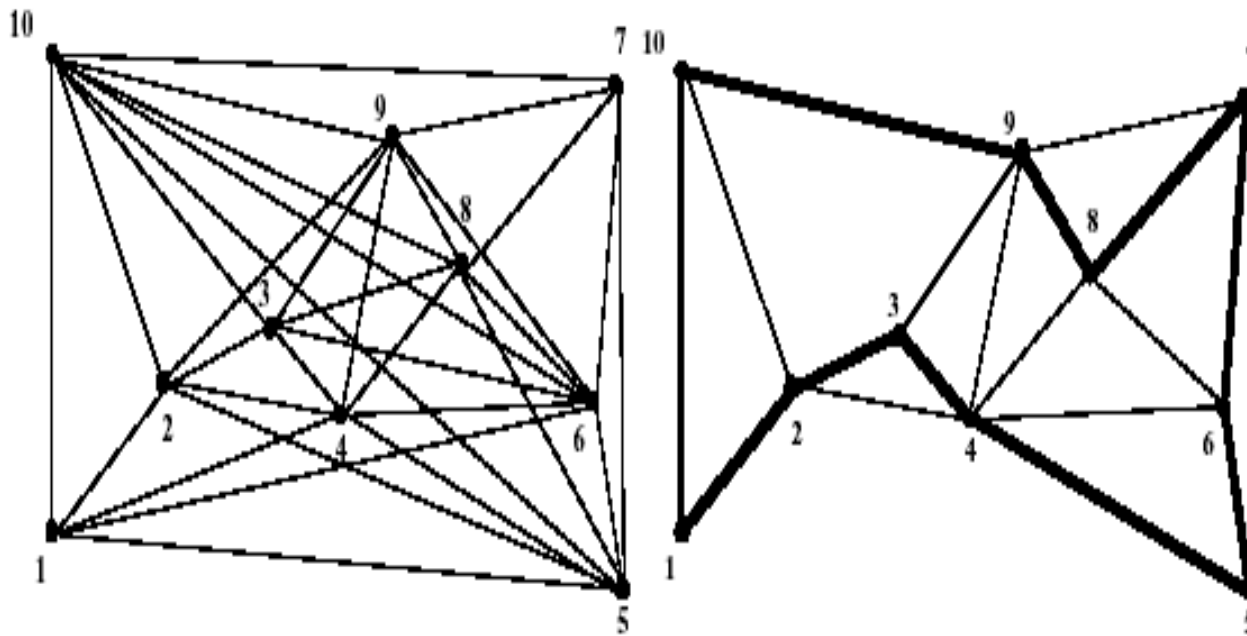
Save Best Tour (Sequence and length)

Do Next Run



Stopping criteria

- ✱ Stagnation (use, e.g., leaderboard)
- ✱ Max iterations



General ACO

- ✱ A stochastic construction procedure
- ✱ Probabilistically build a solution
- ✱ Iteratively add solution components to partial solutions
 - Heuristic information
 - Pheromone trail
- ✱ Reinforcement Learning reminiscence
- ✱ Modify the problem representation at each iteration

General ACO

- ★ Ants work concurrently and independently
- ★ Collective interaction via indirect communication leads to good solutions

Some advantages

- ✱ Positive feedback accounts for rapid discovery of good solutions
- ✱ Distributed computation avoids premature convergence
- ✱ The greedy heuristic helps find acceptable solution in the early stages of the search process.
- ✱ The collective interaction of a population of agents.

Disadvantages in Ant Systems

- ✱ Possibly slow convergence
- ✱ No centralized processor to guide the AS towards good solutions

Improvements to Ant Systems

- ✿ Also apply centralized actions

- ✿ ACO is a local optimization procedure

- ✿ Improve by biasing the search process with the global information

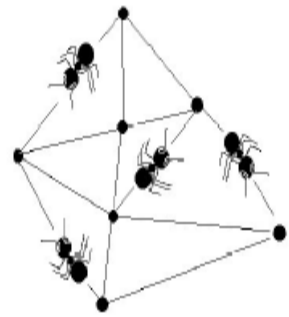
- ✿ Max-Min Ant System

- ✿ Pheromone values are limited $\tau_{\min} \leq \tau_{ij} \leq \tau_{\max}$

- ✿ Only the best ant(s) can add pheromones

- ✿ Sometimes uses local search to improve its performance

Quadratic Assignment Problem(QAP)

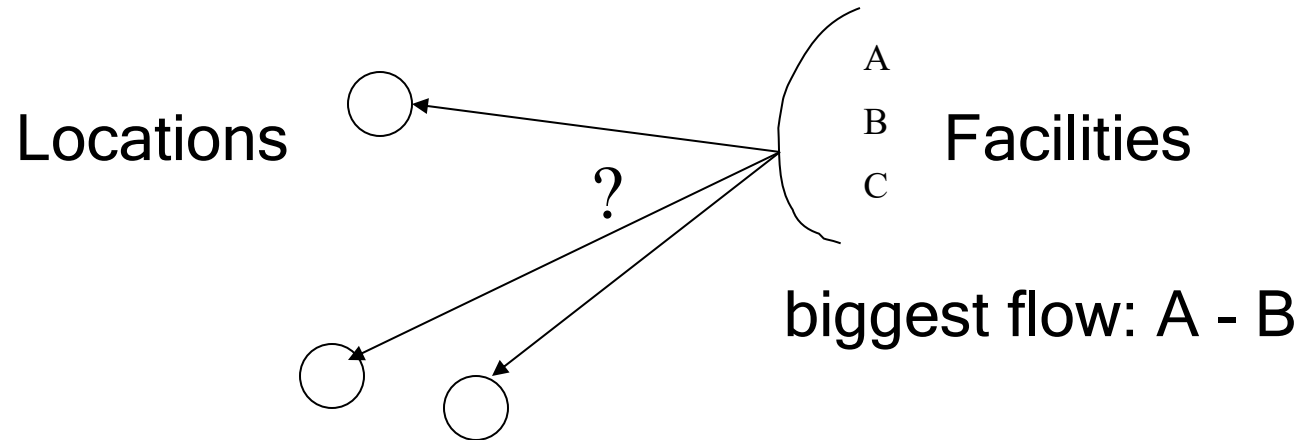


NP-hard problem defined as

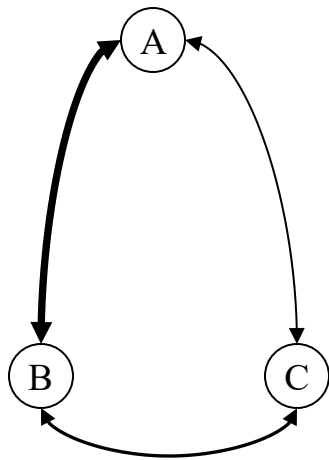
- Assign n activities to n locations (campus and mall layout).
- $D = [d_{i,j}]_{n,n}$, where $d_{i,j}$ is the distance from location i to location j
- $F = [f_{h,k}]_{n,n}$, where $f_{h,k}$ is the flow from activity h to activity k
- Assignment is a permutation π
- Minimize:

$$C(\pi) = \sum_{i,j=1}^n d_{ij} f_{\pi(i)\pi(j)}$$

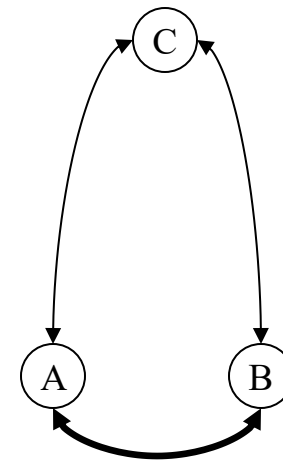
QAP Example



How to assign facilities to locations ?



Higher cost



Lower cost

SIMPLIFIED QAP

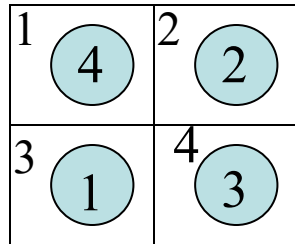
Simplification Assume all departments have equal size

Notation $d_{i,j}$ distance between **locations** i and j

$f_{k,h}$ travel frequency between **departments** k and h

$X_{i,k} \begin{cases} 1 & \text{if department } k \text{ is assigned to location } i \\ 0 & \text{otherwise} \end{cases}$

Example



Location



Department („Facility“)

Distance* $d_{i,j}$

	1	2	3	4
1	-	1	1	2
2	1	-	2	1
3	1	2	-	1
4	2	1	1	-

Frequency* $f_{k,h}$

	1	2	3	4
1	-	1	3	2
2	2	-	0	1
3	1	4	-	0
4	3	1	1	-

Ant System (AS-QAP)

Constructive method:

step 1: choose a facility j

step 2: assign it to a location i

Characteristics:

- each ant leaves trace (pheromone) on the chosen couplings (i,j)
- assignment depends on the probability (function of pheromone trail and a heuristic information)
- already coupled locations and facilities are inhibited (e.g., Tabu list)

AS-QAP Heuristic information

Distance and Flow Potentials

$$D_{ij} = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 4 & 5 \\ 2 & 4 & 0 & 6 \\ 3 & 5 & 6 & 0 \end{bmatrix} \Rightarrow D_i = \begin{bmatrix} 6 \\ 10 \\ 12 \\ 14 \end{bmatrix} \quad F_{ij} = \begin{bmatrix} 0 & 60 & 50 & 10 \\ 60 & 0 & 30 & 20 \\ 50 & 30 & 0 & 50 \\ 10 & 20 & 50 & 0 \end{bmatrix} \Rightarrow F_i = \begin{bmatrix} 120 \\ 110 \\ 130 \\ 80 \end{bmatrix}$$

The coupling Matrix:

$$S = \begin{bmatrix} 720 & 1200 & 1440 & 1680 \\ 660 & 1100 & 1320 & 1540 \\ 780 & 1300 & 1560 & 1820 \\ 480 & 800 & 960 & 1120 \end{bmatrix} \quad \begin{aligned} s_{11} &= f_1 \bullet d_1 = 720 \\ s_{34} &= f_3 \bullet d_4 = 960 \end{aligned}$$

Ants choose the location according to the heuristic desirability “Potential goodness”

$$\zeta_{ij} = \frac{1}{s_{ij}}$$

AS-QAP Constructing the Solution

- The facilities are ranked in decreasing order of the flow potentials
- Ant k assigns the facility i to location j with the probability given by:

$$p_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}(t)^\alpha \eta_{ij}^\beta}{\sum_{l \in N_i^k} \tau_{il}(t)^\alpha \eta_{il}^\beta} & \text{if } j \in N_i^k \end{cases}$$

where N_i^k is the feasible neighborhood of node i

When ant k chooses to assign facility j to location i , it leaves a trace “pheromone” on the coupling (i,j)

- Repeated until the entire assignment is found

AS-QAP Pheromone Update

- Pheromone trail update to all couplings:

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

$\Delta\tau_{ij}^k$ is the amount of pheromone ant k puts on the coupling (i,j)

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{J_{\psi}^k} & \text{if facility } i \text{ is assigned to location } j \text{ in the solution of ant } k \\ 0 & \text{otherwise} \end{cases}$$

J_{ψ}^k ... the objective function value

Q...the amount of pheromone deposited by ant k

Hybrid Ant System For The QAP

- Constructive algorithms often result in a poor solution quality compared to local search algorithms.
- Repeating local searches from randomly generated initial solution results for most problems in a considerable gap to optimal solution
- Hybrid algorithms combining solution constructed by (artificial) ant “probabilistic constructive” with local search algorithms yield significantly improved solution.

Hybrid Ant System For The QAP (HAS-QAP)

- HAS-QAP uses of the pheromone trails in a non-standard way. It is used to modify an existing solution
- Improves the ant's solution using the local search algorithm.
- Intensification and diversification mechanisms.

Hybrid Ant System For The QAP (HAS-QAP)

```
Generate  $m$  initial solutions, each one associated to one ant
Initialise the pheromone trail
For  $I_{max}$  iterations repeat
    For each ant  $k = 1, \dots, m$  do
        Modify ant  $k$ 's solution using the pheromone trail
        Apply a local search to the modified solution
        new starting solution to ant  $k$  using an intensification mechanism
    End For
    Update the pheromone trail
    Apply a diversification mechanism
End For
```

HAS-QAP Intensification & diversification mechanisms

- The intensification mechanism is activated when the best solution produced by the search so far has been improved.
- The diversification mechanism is activated if during the last S iterations no improvement to the best generated solution is detected.

Particle Swarm Optimization (PSO)

- A population based stochastic optimization technique
- Searches for an optimal solution in the computable search space
- Developed in 1995 by Eberhart and Kennedy
- Inspired by social psychology
- Inspiration: swarms of bees, flocks of birds, schools of fish

PSO principles

- In PSO individuals strive to improve themselves and often achieve this by observing and imitating their neighbors
- Each PSO individual has the ability to remember
- PSO has simple algorithms and low overhead
 - Making it more popular in some circumstances than Genetic/Evolutionary Algorithms
 - Has only one operation calculation:
 - Velocity: a vector of numbers that are added to the position coordinates to move an individual

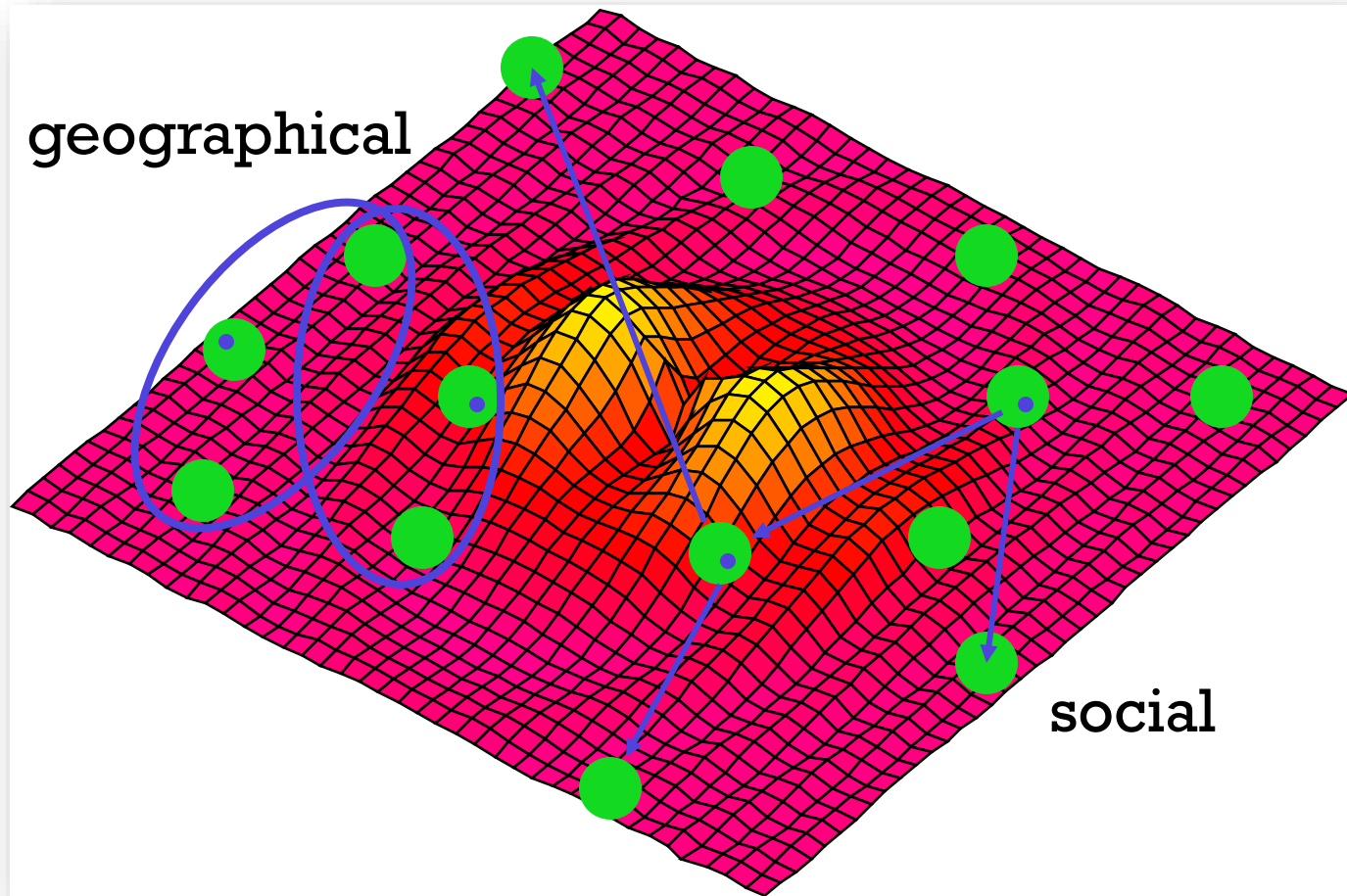
PSO and social psychology

- Individuals (points) tend to
 - Move towards each other
 - Influence each other
 - Why?
 - Individuals want to be in agreement with their neighbors
- Individuals (points) are influenced by:
 - Their previous actions/behaviors
 - The success achieved by their neighbors

What Happens in PSO

- Individuals in a population learn from previous experiences and the experiences of those around them
- The direction of movement is a function of:
 - Current position
 - Velocity (or in some models, probability)
 - Location of individuals “best” success
 - Location of neighbors “best” successes
 - Location of globally “best” success
- Therefore, each individual in a population will gradually move towards the “better” areas of the problem space
- Hence, the overall population moves towards “better” areas of the problem space

PSO: Neighborhood



Particle Swarm Optimization (PSO)

- ★ One can imagine that each particle is represented with two vectors, location and velocity

- ✧ Location $x = (x_1, x_2, \dots)$

- ✧ Velocity $v = (v_1, v_2, \dots)$

- ✧ For locations $x(t-1)$ and $x(t)$ in time $t-1$ and t :

$$\vec{v} = \vec{x}(t) - \vec{x}(t-1)$$

- ★ Initialization of locations and velocities (small initial values, e.g., one half of distance to the neighboring particle, random, or 0)

Information exchange in the swarm

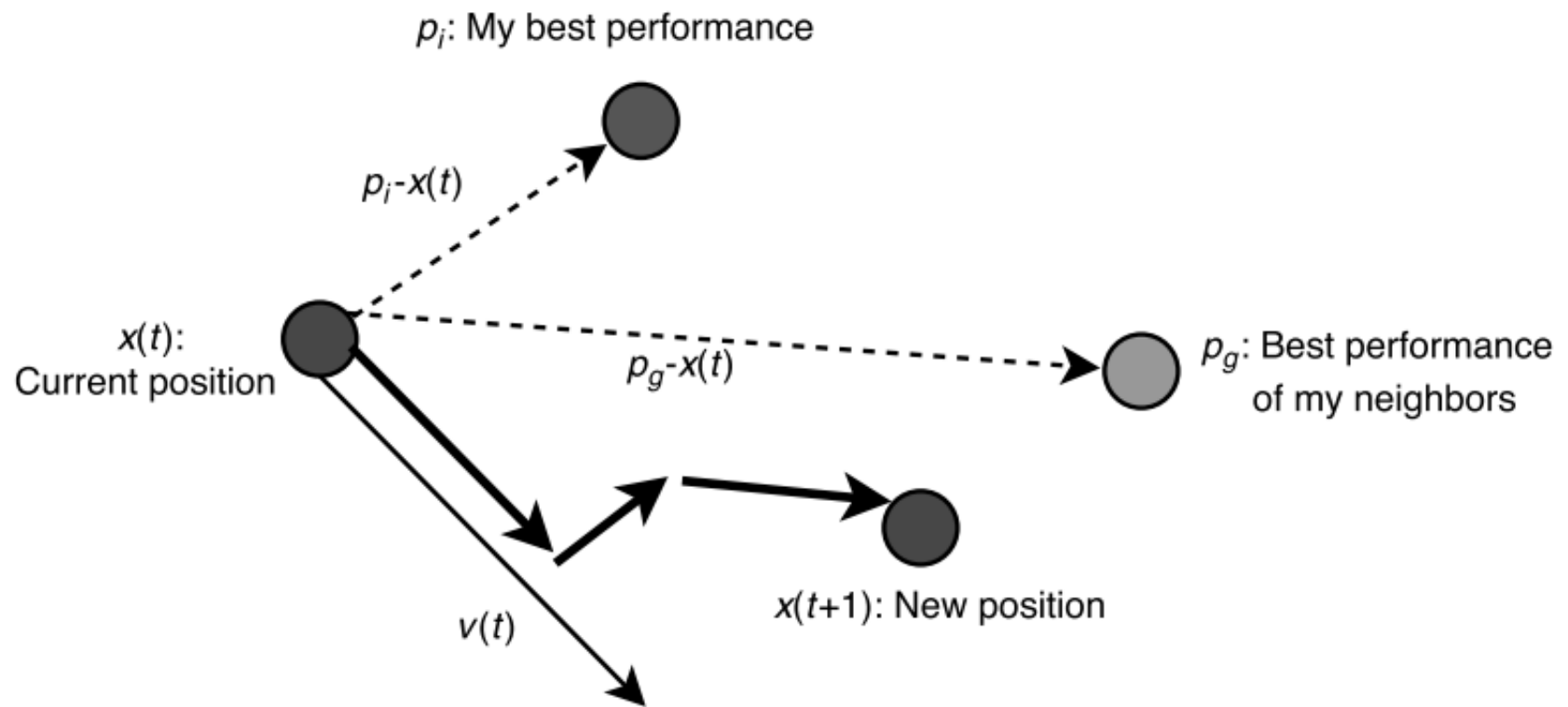
- ✧ Historically best location x^*
- ✧ Best location of informants x^+
- ✧ Globally best location $x^!$

Moving particles

✱ In each time step, the following operations are executed

1. compute the fitness of each particle and update x^* , x^+ in $x^!$
2. update the representation of particle
 - ✧ velocity vector takes into account updated directions x^* , x^+ in $x^!$
 - ✧ each direction is updated with some random noise
3. move the particle in the direction of velocity vector

Computing new position



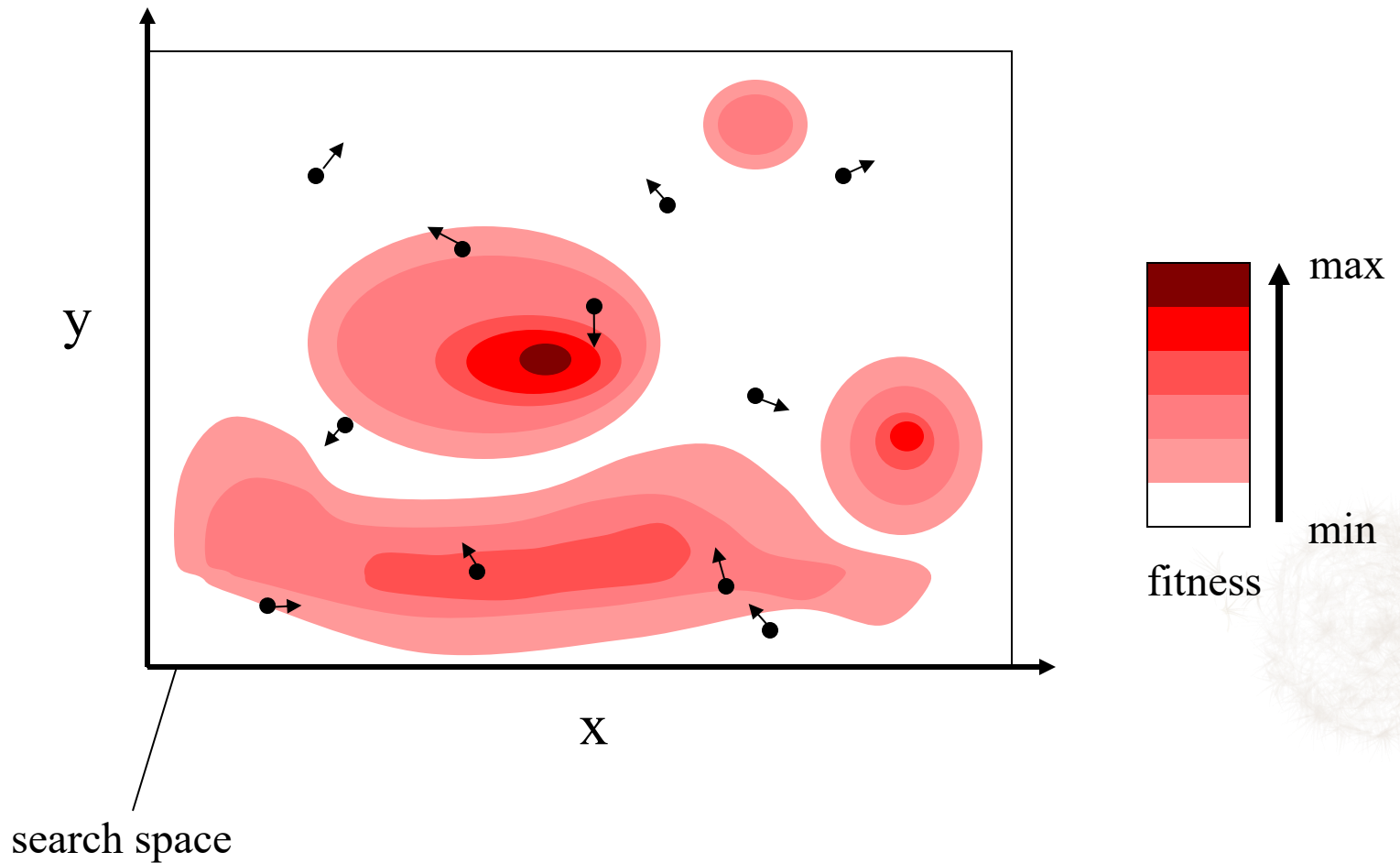
PSO - parameters

- ✱ α - proportion of current velocity vector v
- ✱ β - proportion of the best value of location x^*
too large value pushes towards its maximum and we get a swarm of greedy searchers and no group dynamics
- ✱ δ - proportion of the best global location $x^!$
too large value pushes particles towards the current global maximum and we get a single greedy search, instead of several local searches (often we set this parameter to 0)
- ✱ γ - proportion of the best value of informants x^+
the effect between β and δ , depends also on the number of informants:
more informants emphasize global, less informants emphasize effect of local information
- ✱ ε - speed of particle movement
too large speed may cause too fast convergence without enough search
(default value is 1)
- ✱ swarmsize – size of swarm (between 20 and 50)

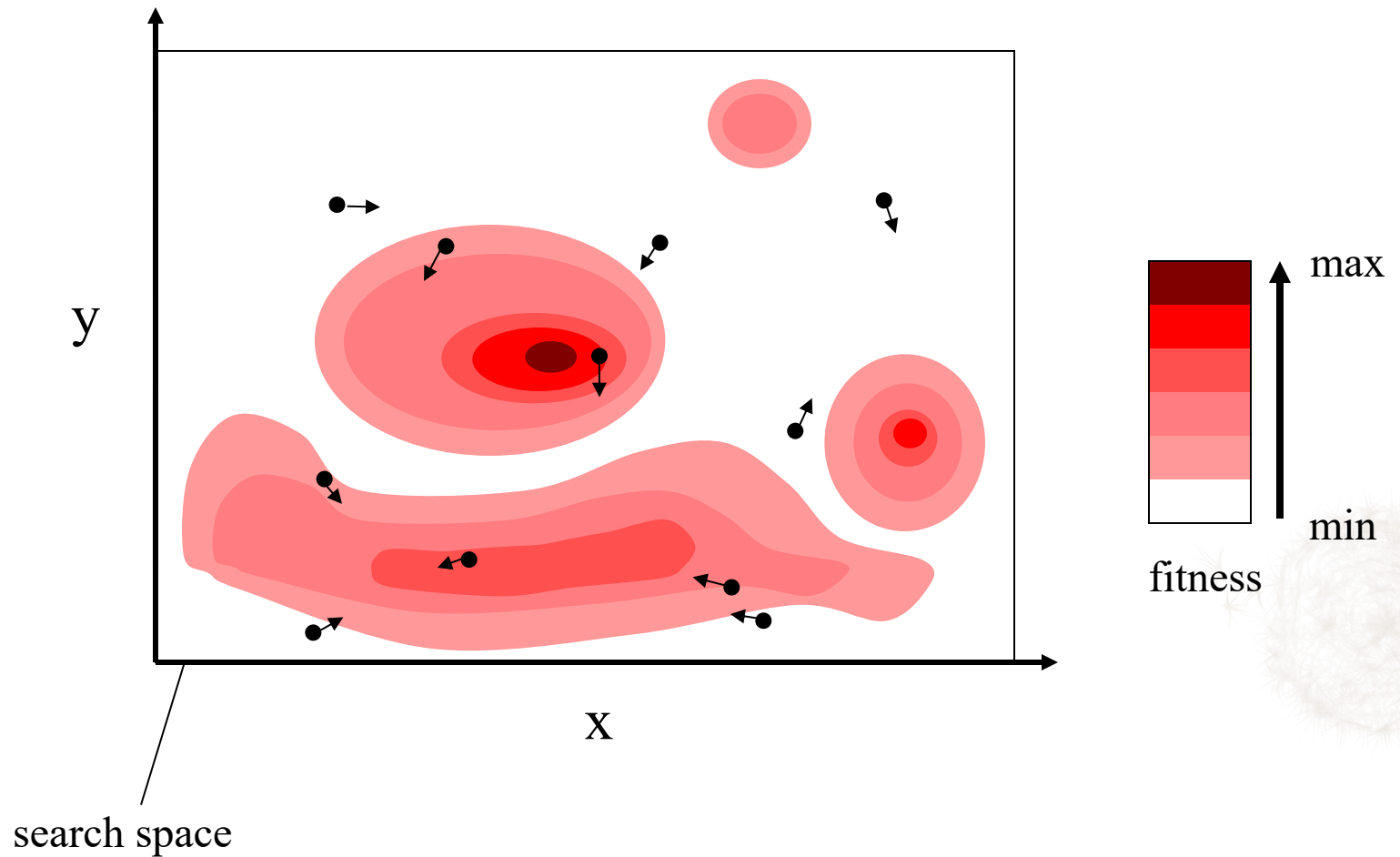
PSO pseudocode

```
P = []
for (i=0 ; i < swarmsize ; i++)
    Pi = new particle with random position x and random velocity v
best = null
do {
    for (i=0 ; i < swarmsize ; i++) {
        compute fitness(Pi)
        if ( fitness(Pi) >  fitness(best) )
            best = Pi
    }
    for (i=0 ; i < swarmsize ; i++) {
        x* = update location of the best fitness of xi
        x+ = update location of the best fitness of informants of xi
        x! = update location of the best fitness of all particles
        for (j=0; j < #dimensions; j++) {
            b = random between 0 and β
            c = random between 0 and γ
            d = random between 0 and δ
            vj = αvj + b(x*j - xj) + c(x+j - xj) + d(x!j - xj)
        }
        xi = xi + ε·v
    } while (!satisfied with best or out of time)
return best
```

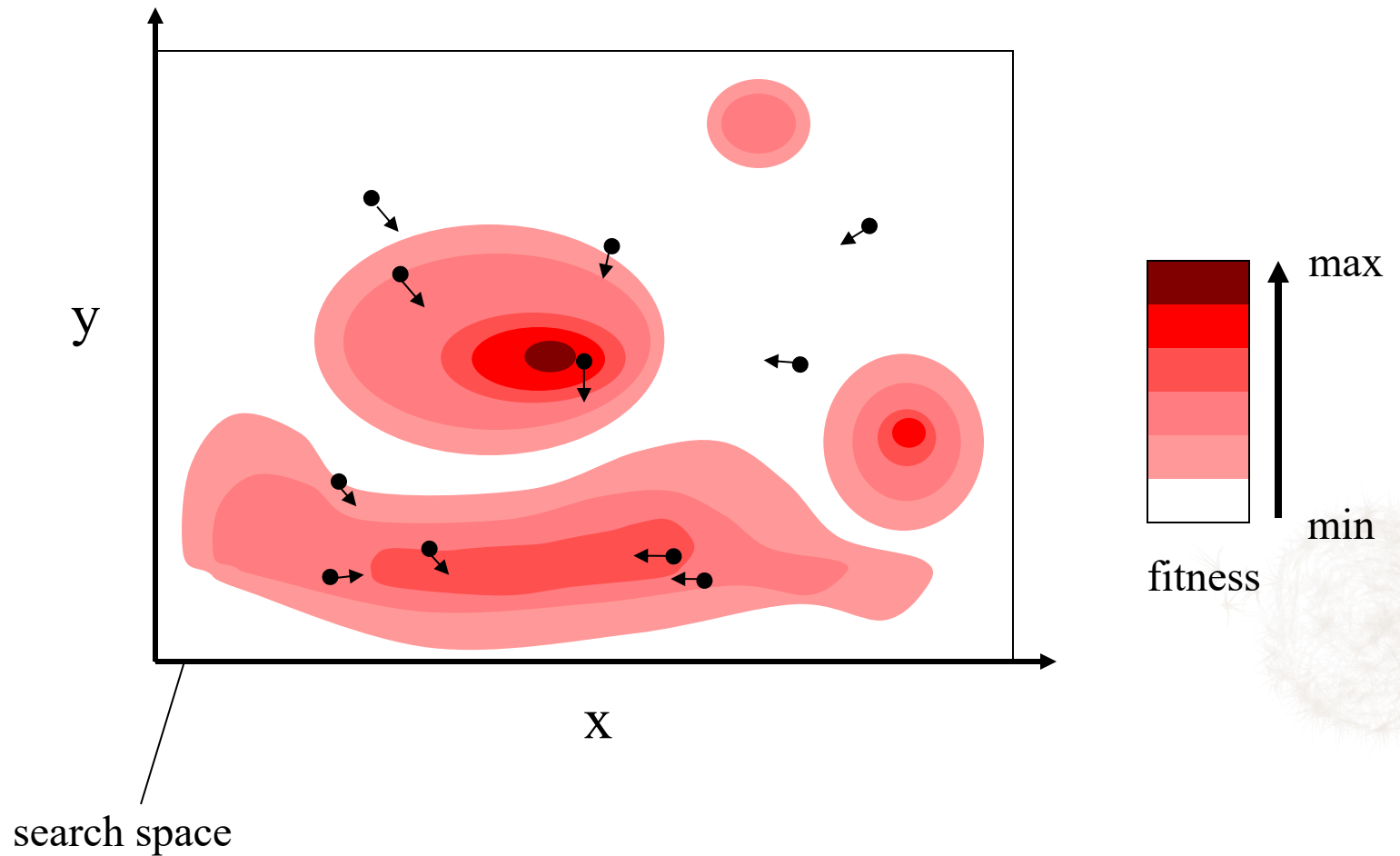
simulation₁



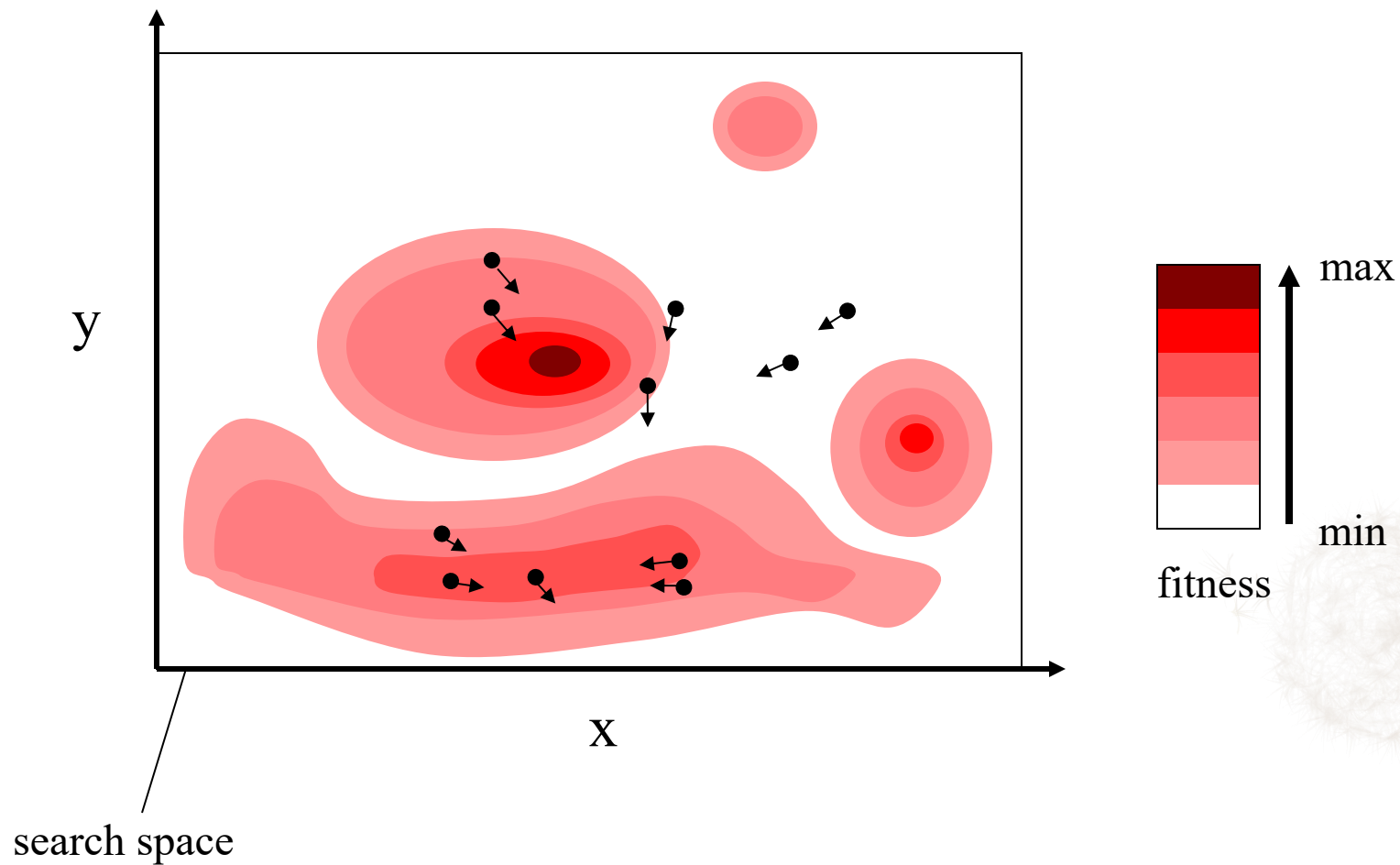
simulation₂



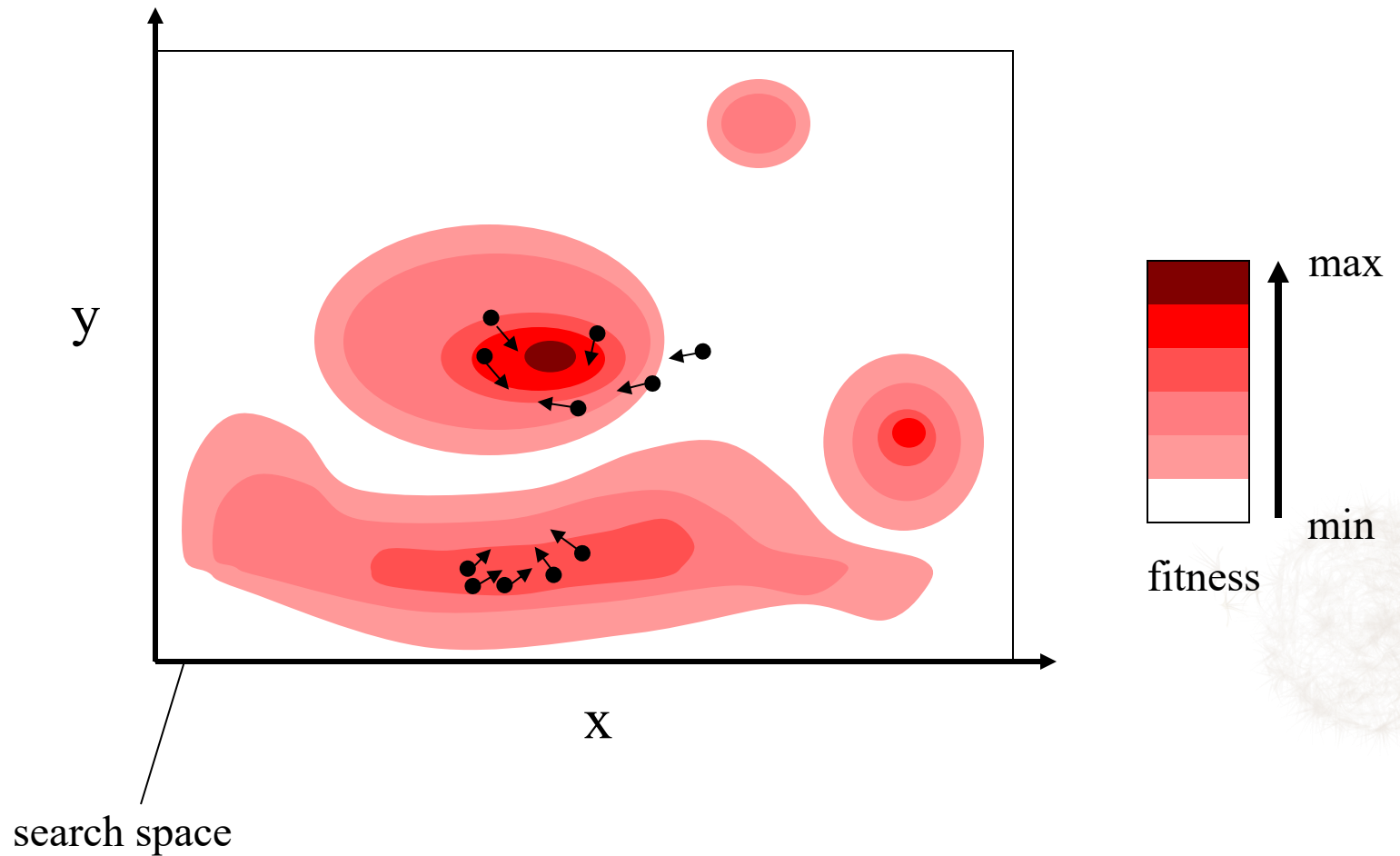
simulation₃



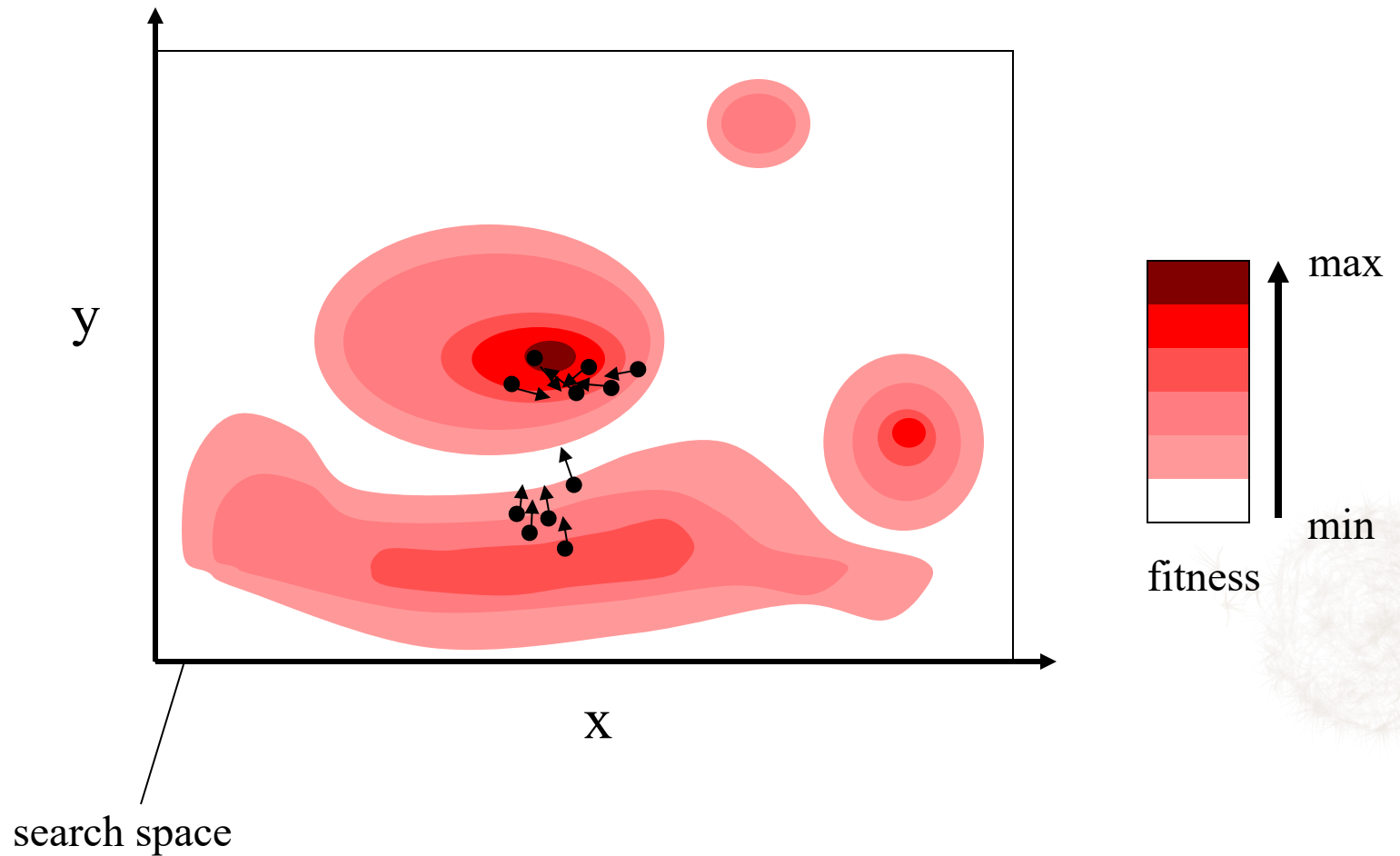
simulation₄



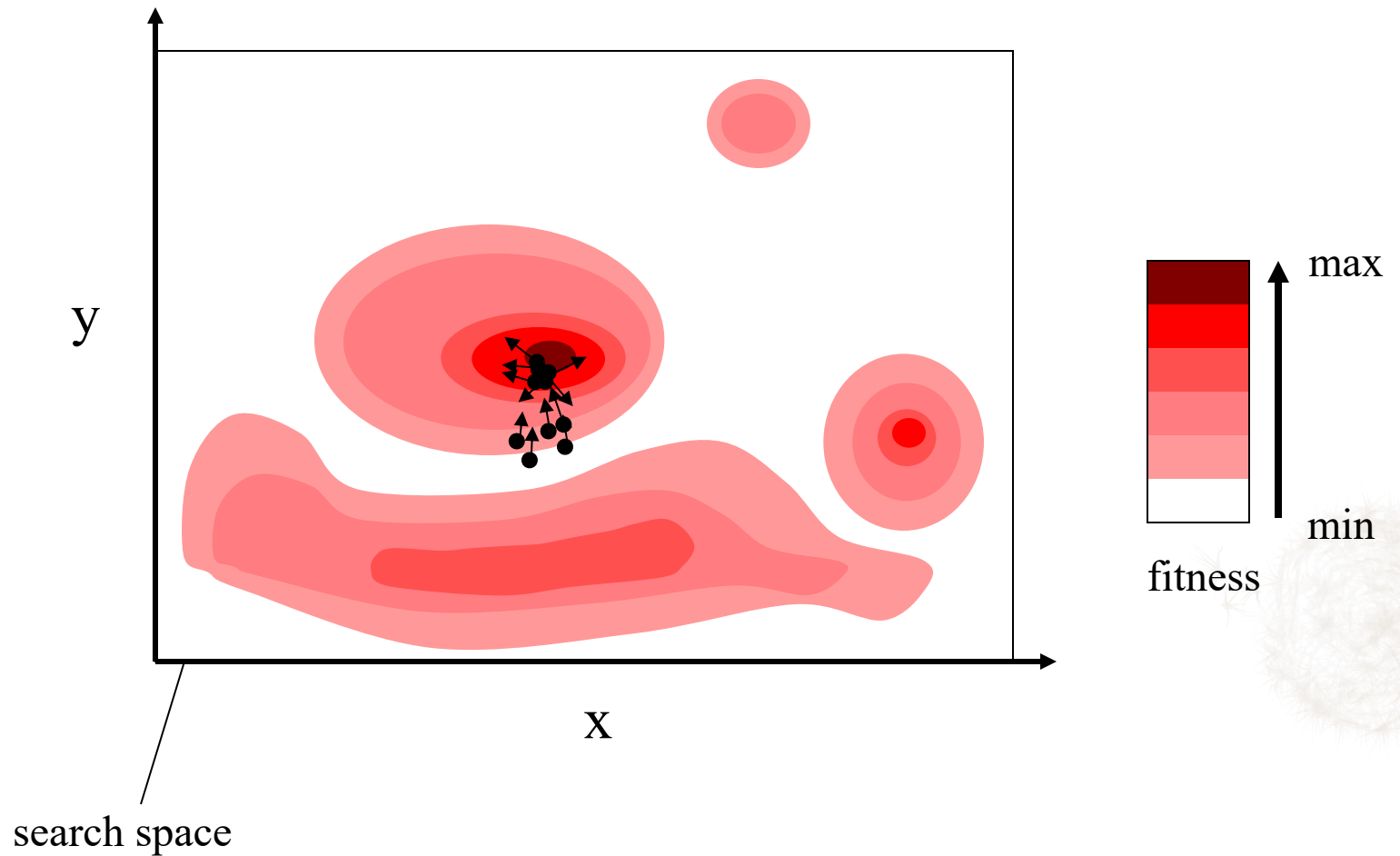
simulation₅



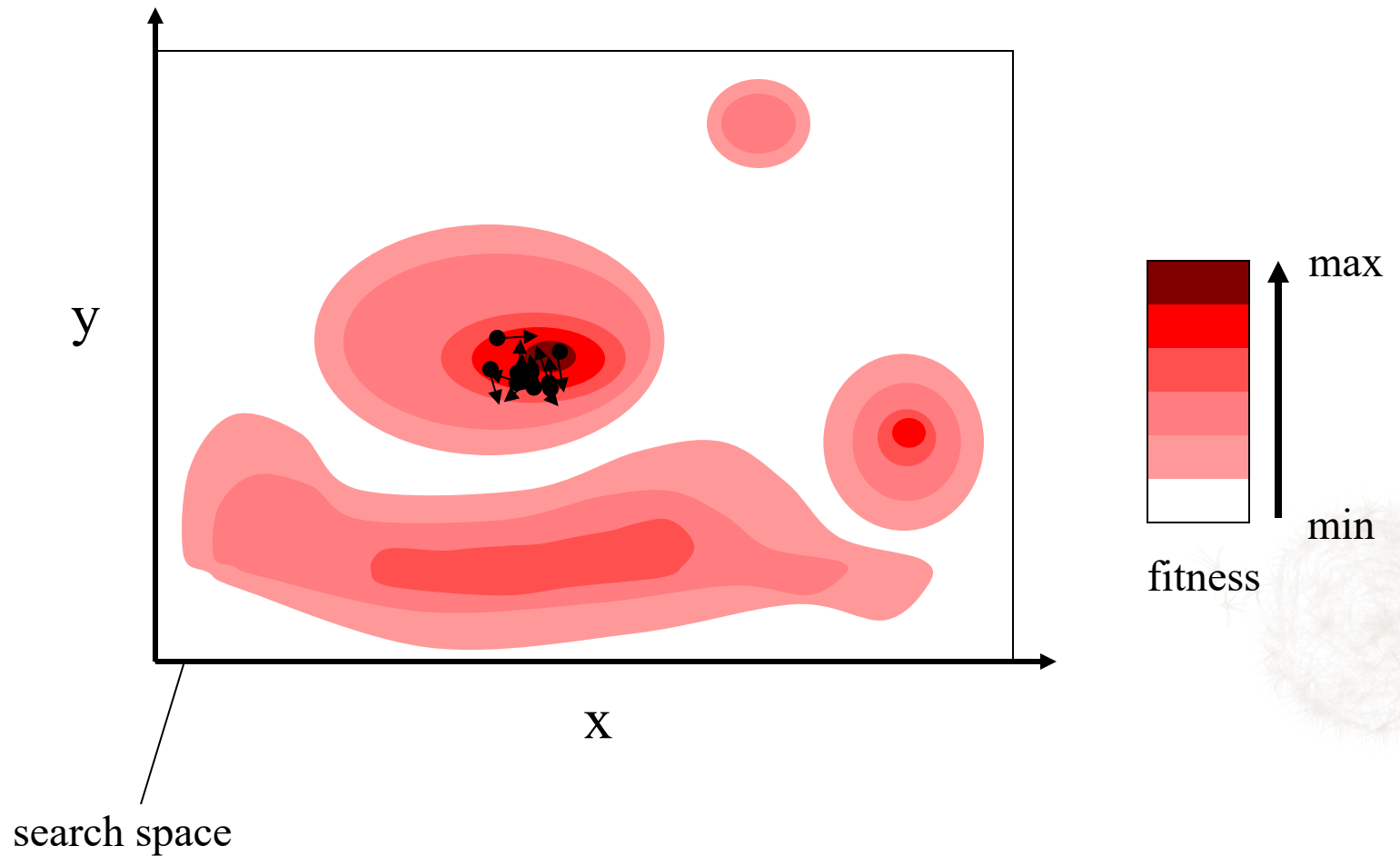
simulation₆



simulation₇



simulation₈



PSO characteristics

✱ Advantages

- ✧ Insensitive to scaling of design variables
- ✧ Simple implementation
- ✧ Easily parallelized for concurrent processing
- ✧ Derivation free
- ✧ Very few algorithm parameters
- ✧ Very efficient global search algorithm

✱ Disadvantages

- ✧ Tendency to a fast and premature convergence in mid optimum points
- ✧ Slow convergence in refined search stage (weak local search ability)