Swarm Intelligence
(SI)

Recap on Evolutionary Computing

- Population-based stochastic algorithms, evolve candidate solutions through mutation, crossover and natural selection. Inspired by Darwin's theory of natural selection
  - Genetic algorithms: vector-based representation, developed for discrete optimization
  - Evolutionary Strategies: vector-based representation, evolve also the strategy parameters, developed for continuous optimization
  - Genetic Programming: tree-based representation, originally developed for evolving programs
- Direct search, i.e. gradient free

Swarm intelligence in nature

- Principle underlying swarm intelligence: imitate self and the others in a social group
- Life in social groups is easier:
  - facilitates the retrieval of food
  - reduces the probability of attack by predators
  - allows for a division of labor
  - facilitates hunting

Swarm intelligence in nature: examples

- Confusing predator by:
  - united erratic maneuvers (e.g. zigzagging)
  - separation into subgroups (e.g., flash expansion & fountain effect)
- Coordinated movements to trap prey: e.g. parabolic formation of tuna
- Fish schooling increases hydrodynamic efficiency:
  - endurance may be increased up to 6×
  - school acts like "group-level vehicle"
- V-formation increases aerodynamic efficiency of geese:
  - Range 70% greater than that of individual
- Lobsters line up in single file by touch:
  - move 40% faster than when isolated
  - decreased hydrodynamic drag

Swarm Intelligence

- Swarm Intelligence models the emergent global behavior of a population of simple agents
- Agents interact locally with one another and with their environment
- No centralized control structure dictating how individual agents should behave
- Metaheuristics inspired by Swarm Intelligence:
  - Particle Swarm Intelligence
  - Ant Colony Optimization

Particle Swarm Optimization (PSO)

Motivation

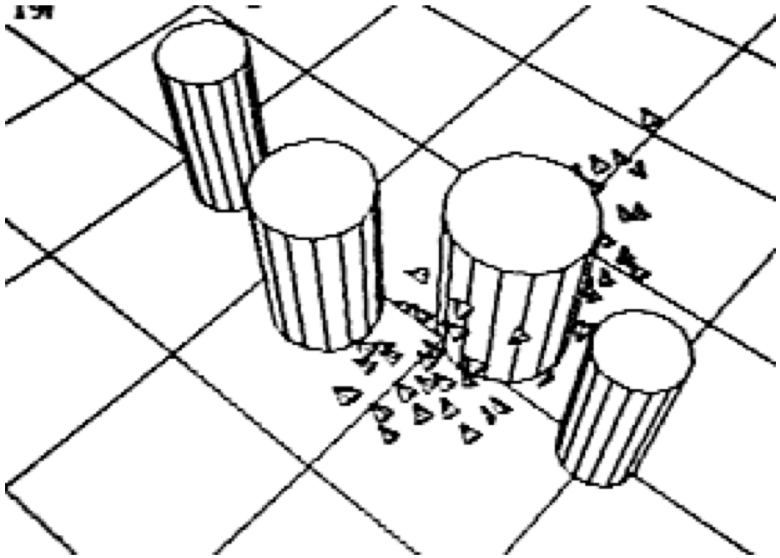
- Hypothesis: There are two major sources of cognition, namely, own experience and communication from others
- Approach: Imitate self and others

Leon Festinger, 1954/1999, Social Communication and Cognition

Early study: model flocking behavior in birds

- A computer model of coordinated animal motion: the simulated flock is an elaboration of a particle system, with the simulated birds being the particles (Craig Reynolds, 1986)

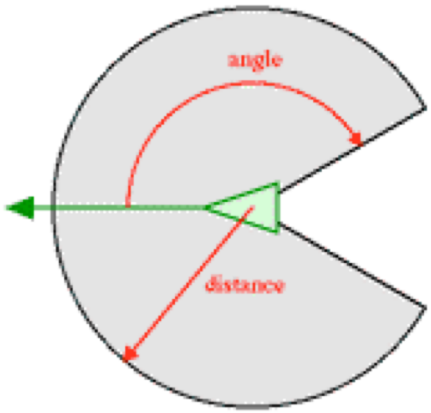
The aggregate motion of the simulated flock is the result of the dense interaction of the relatively simple behaviors of the individual simulated birds (called boids)



http://www.red3d.com/cwr/papers/1987/boids.html

Reynolds' boids

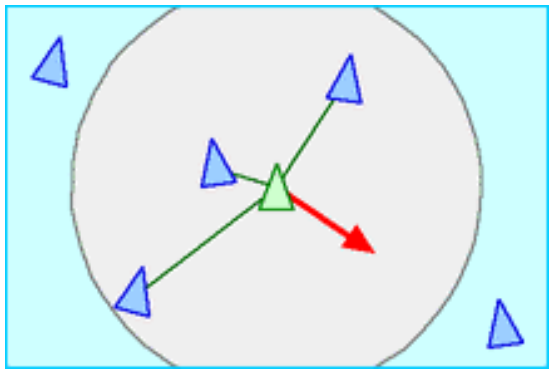
- Boid: an individual agent equipped with a position and a velocity
- A boid reacts only to flockmates within a certain small neighborhood around itself
- Boid's neighborhood: characterized by a *distance*, measured from the center of the boid, and an *angle*, measured from the boid's direction of flight



A boyd's neighborhood (grey area)

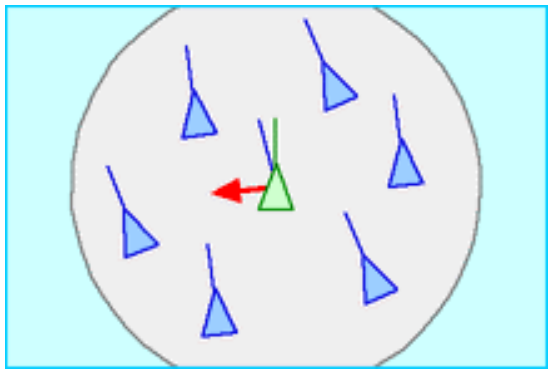
Reynolds' boids

Boids have three simple steering behaviors: separation, alignment and cohesion



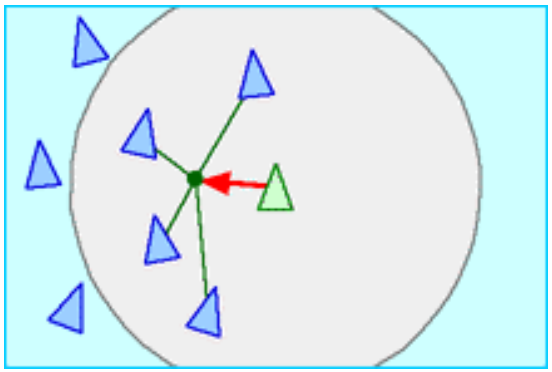
Separation

avoid crowding local agents



Alignment

move towards the average heading of local agents

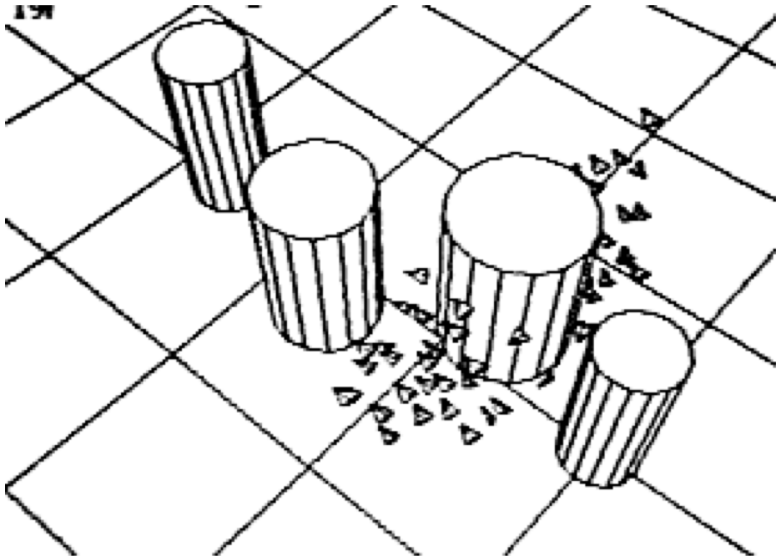


Cohesion

move towards the average position of local agents

Reynolds' flock simulation model

- The flock simulation model included obstacle avoidance and goal seeking
  - Obstacle avoidance allowed the boids to fly through simulated environments while avoiding static objects
  - Goal seeking behavior caused the flock to follow a scripted path, used for applications in computer animation



https://www.youtube.com/watch?v=86iQiV3-3IA

Particle Swarm Optimization (PSO)

- In 1995 Kennedy and Eberhart developed a realistic visual simulation of bird flock behaviour and *included a point in the environment that attracted the virtual bird agents, e.g. food.*
- Particle swarm modeling (e.g. Reynolds' model) became Particle Swarm Optimization

Kennedy, J.; Eberhart, R.; , "Particle swarm optimization," *Neural Networks, 1995. Proceedings., IEEE International Conference on* , vol.4, no., pp.1942-1948 vol.4, Nov/Dec 1995.

Kennedy, Jim, and R. Eberhart. "Tutorial on particle swarm optimization." *2002 World Congress on Computational Intelligence WCCI*. 2002.

PSO

- Stochastic population-based methods for continuous optimization
- Simple agents imitate self and others
- Asynchronous, decentralized control
- Direct search, i.e. gradient free
- Redundancy leads to reliability and adaptation

Particle Swarm Optimization (PSO)

- Main characteristics:
  - Simple and quasi-identical units, called particles
  - 'Intermediate' number of particles:10 to 1000 (or more)
  - Each particle is described by a velocity and a position

Kennedy, J.; Eberhart, R.; , "Particle swarm optimization," *Neural Networks, 1995. Proceedings., IEEE International Conference on* , vol.4, no., pp.1942-1948 vol.4, Nov/Dec 1995.

Kennedy, Jim, and R. Eberhart. "Tutorial on particle swarm optimization." *2002 World Congress on Computational Intelligence WCCI*. 2002.

The (canonical) PSO algorithm

- Initialize parameters $n, \omega, \alpha_1, \alpha_2$
- Initialize the particles positions $x_1, \dots, x_n$
- Initialize velocities $v_1 := 0, \dots, v_n := 0$
- Initialize current best $\hat{x}_1, \dots, \hat{x}_n$ and global best $\hat{g}$
- WHILE (termination crit. not satisfied)
  - FOR i=1:n
  - Create random vectors r_1, r_2 with components in $U[0,1]$
  - Update velocities (° componentwise multiplication)

$$v_i := \omega v_i + \alpha_1 r_1 {}^\circ (\hat{x}_i - x_i) + \alpha_2 r_2 {}^\circ (\hat{g} - x_i)$$

  - Update positions

$$x_i := x_i + v_i$$

  - FOR i=1:n
  - Update local bests (for a minimization problem)

$$\hat{x}_i := x_i \text{ if } f(x_i) < f(\hat{x}_i)$$

  - Update global best (for a minimization problem)

$$\hat{g} := x_i \text{ if } f(x_i) < f(\hat{g})$$

The values of $\omega, \alpha_1, \alpha_2$ of PSO should be chosen properly in order to balance between exploration and exploitation. Values used in practice: $\omega \sim 0.73, \alpha_1 = \alpha_2 \sim 1.5$

Velocity update

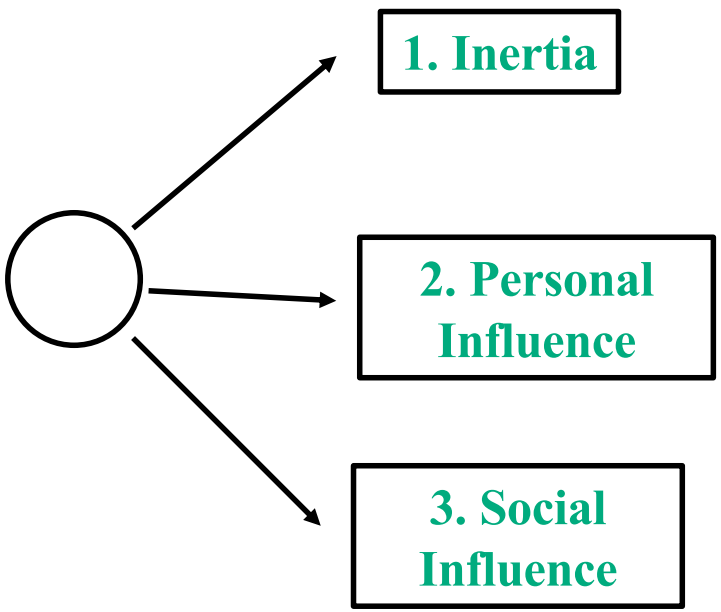
Velocity of particle *i*:

$$v_i \leftarrow \omega v_i + \alpha_1 r_1 \circ (\hat{x}_i - x_i) + \alpha_2 r_2 \circ (\hat{g} - x_i)$$

Inertia parameter

Local best: personal influence

Global best: social influence



- Makes the particle move in the same direction and with the same velocity
- Improves the individual
- Makes the particle return to a previous position better than the current
- Makes the particle follow the overall best direction

Inertia weight ω

$$v_i \leftarrow \boxed{\omega} v_i + \alpha_1 r_1 \circ (\hat{x}_i - x_i) + \alpha_2 r_2 \circ (\hat{g} - x_i)$$

- A large inertia weight favors the diversification of the search process while a small inertia weight favors its intensification
- ω influences convergence of the swarm:
  - $\omega > 1$: velocities increase over time causing divergent behavior, so that particles fail to change direction in order to move back towards promising areas
  - $\omega < 1$: particles may decelerate until their velocities reach zero

Eberhart, R. C., Shi, Y.: Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization, Proceedings of the IEEE Congress on Evolutionary Computation , San Diego, USA, 2000.

van den Bergh, F.: An Analysis of Particle Swarm Optimizers , Ph.D. Thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.

Parameters, Conditions and Tweaks

- Initialization methods
- Population size
- Neighborhood topology (used to compute global best)
- Sub-swarms
- Births, deaths, migration

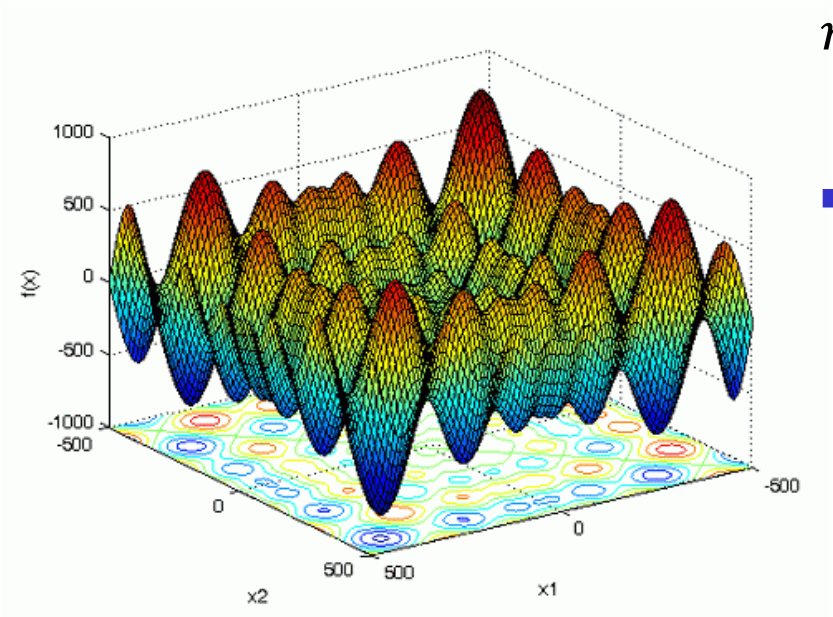
Kennedy, Jim, and R. Eberhart. "Tutorial on particle swarm optimization." *2002 World Congress on Computational Intelligence WCCI*. 2002.

Example: a benchmark function

$$f(x) = \sum_{i=1}^{n} (-x(i)) \cdot \sin\sqrt{|x(i)|}$$

$$-500 \le x(i) \le 500$$

n denotes the input dimension



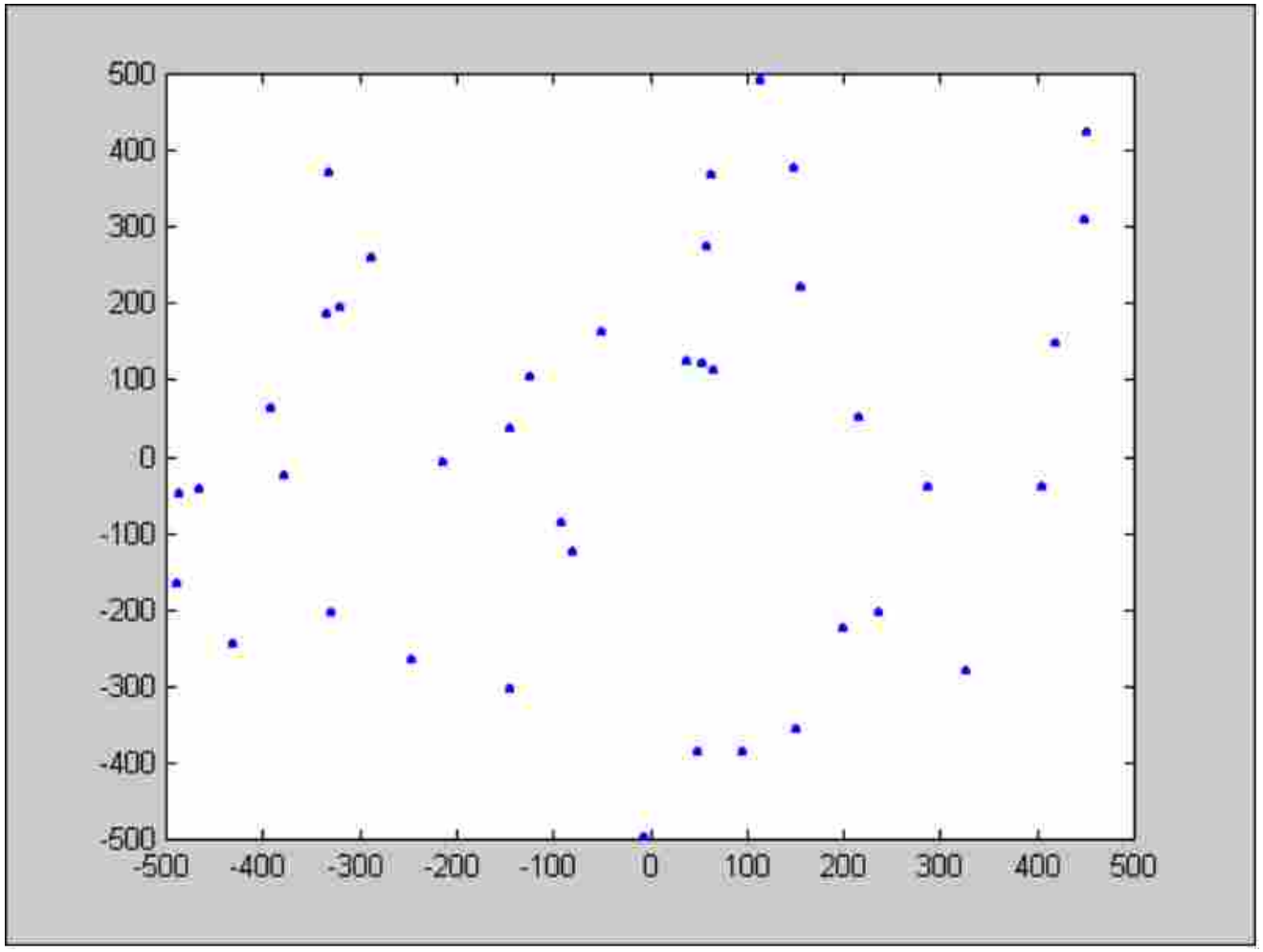
$n = 2$

- Global maximum:

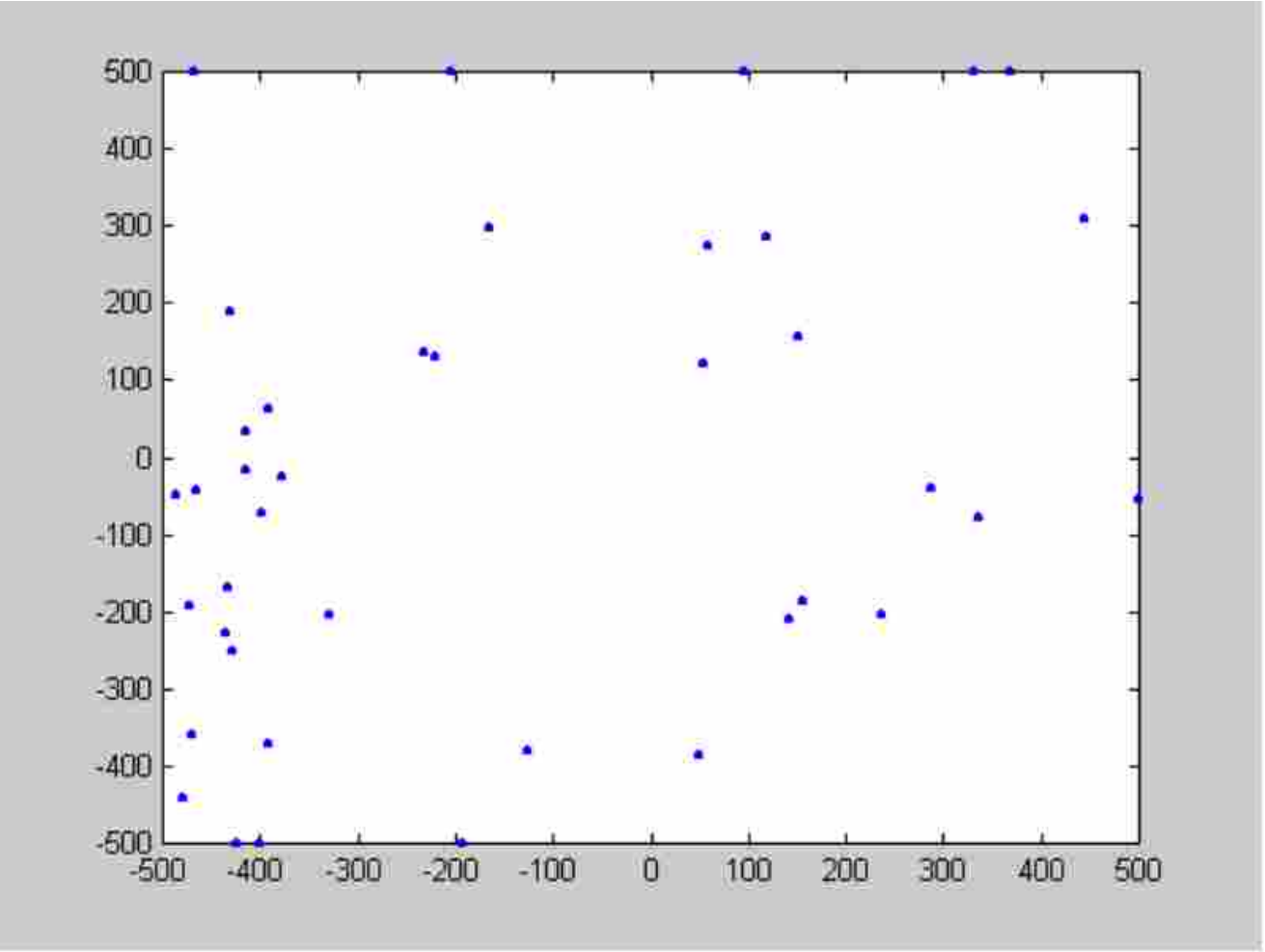
$$f(x) = n \cdot 418.9829$$

$$x(i) = -420.9687, i = 1{:}n$$

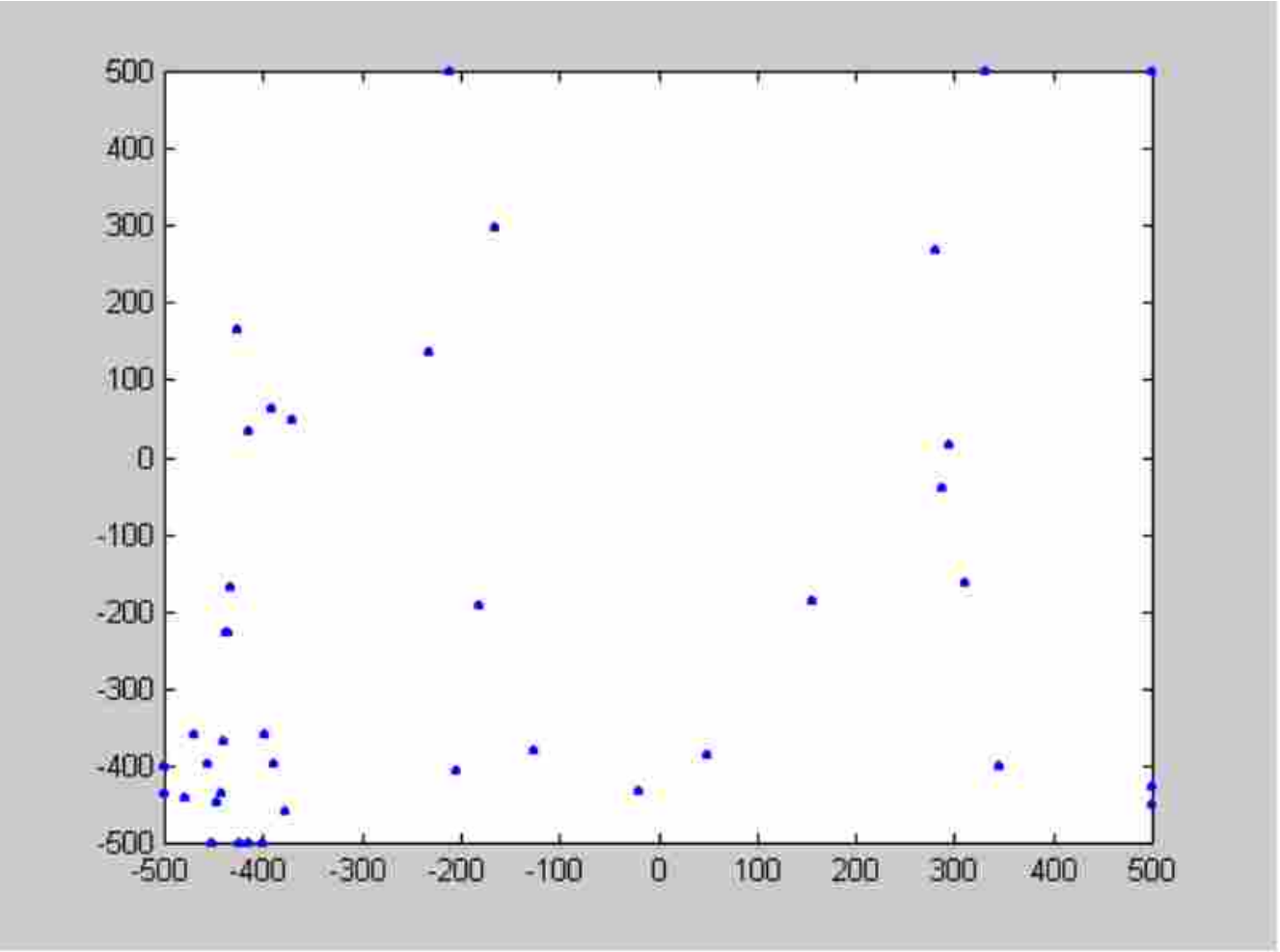
A PSO run (2-dimensional input)



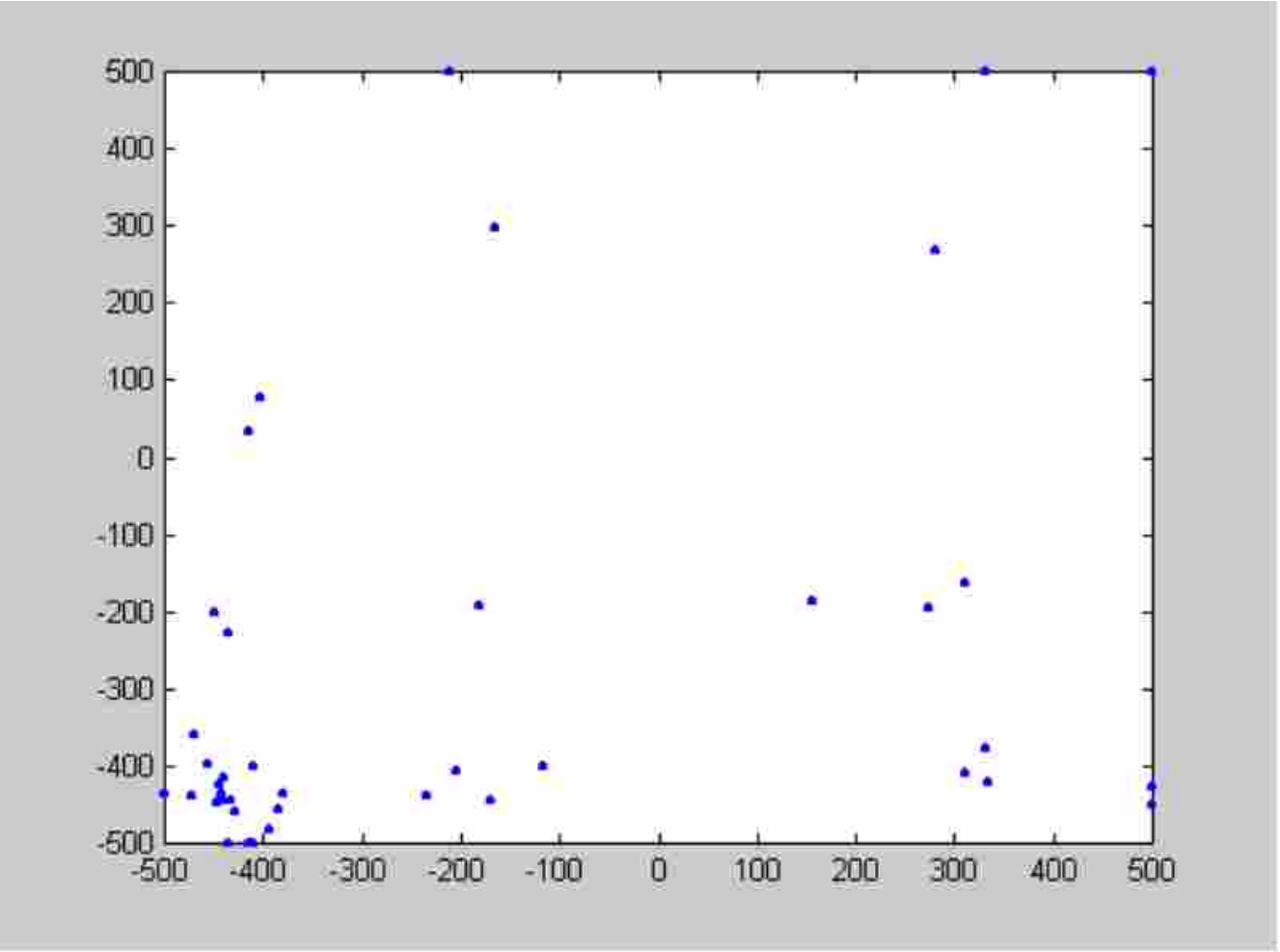
Evolution – 5th iteration



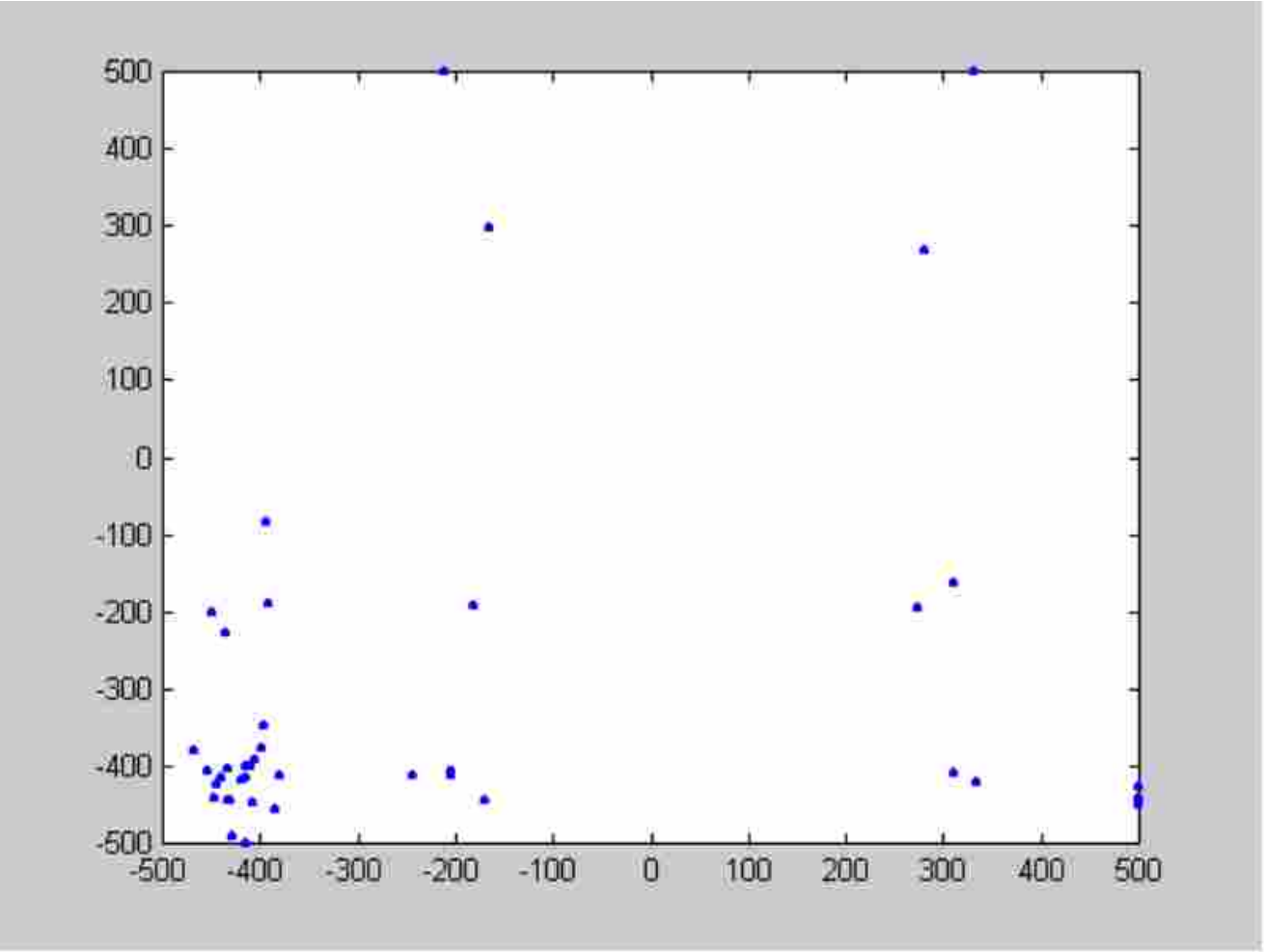
Evolution – 10th iteration



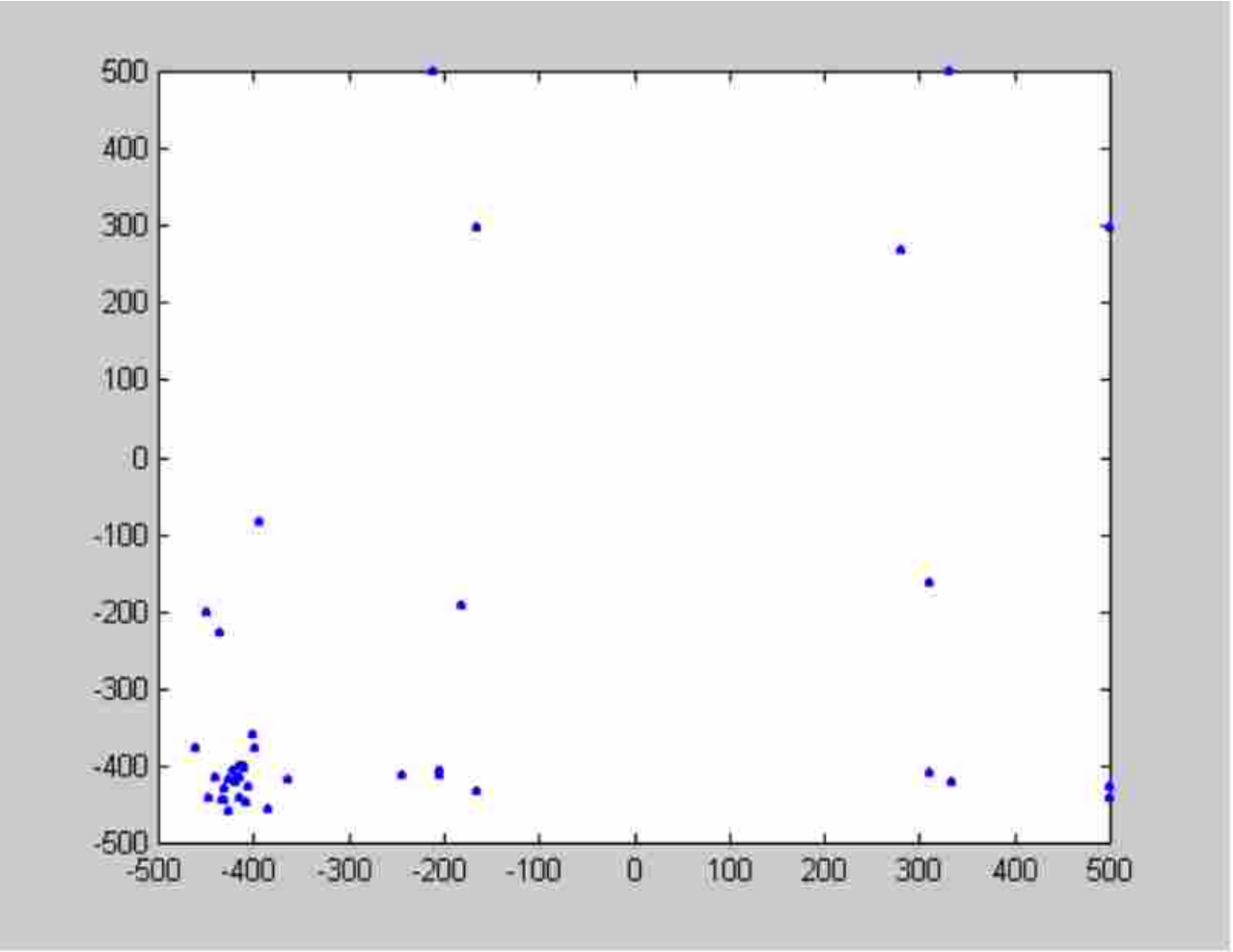
Evolution – 15th iteration



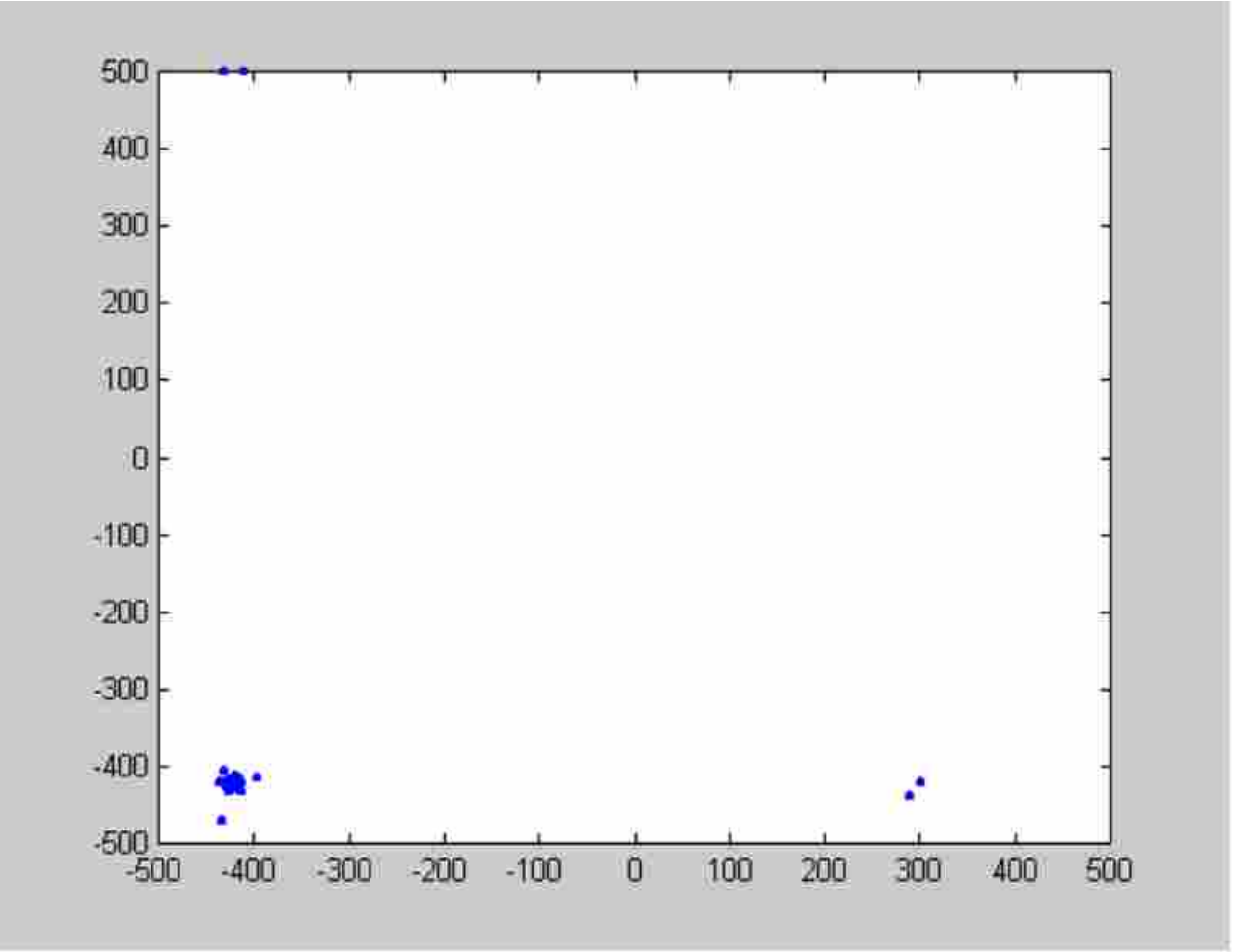
Evolution – 20th iteration



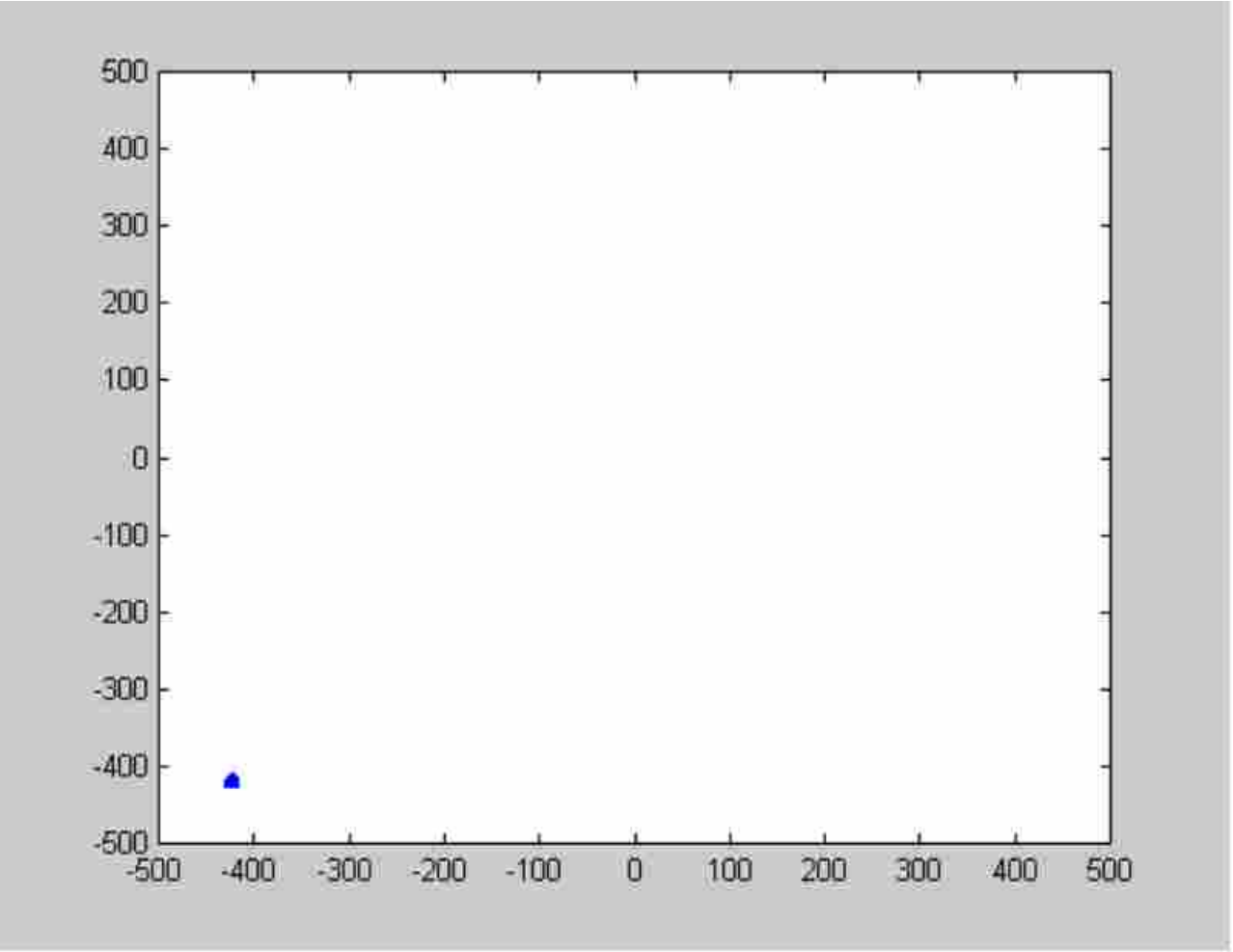
Evolution—25th iteration



Evolution – 100th iteration

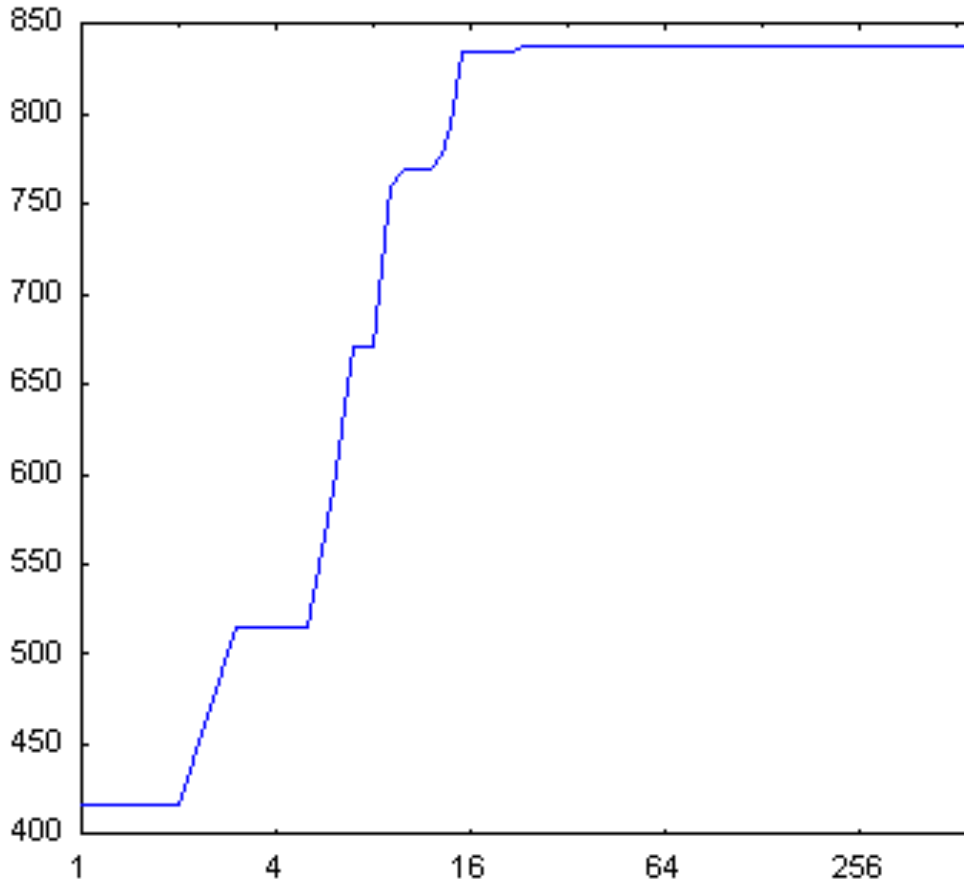


Evolution – 500th iteration



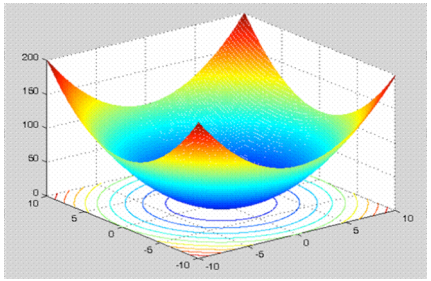
Search result

Iteration	Swarm best
0	416.245599
5	515.748796
10	759.404006
15	793.732019
20	834.813763
100	837.911535
5000	837.965771
Global	837.9658



Other benchmark functions

http://www.cil.pku.edu.cn/resources/benchmark_pso/

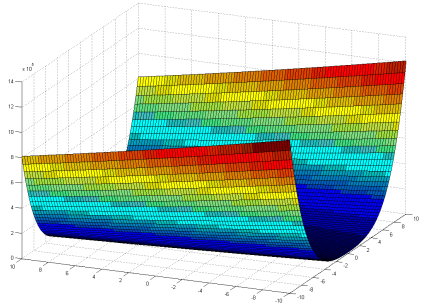


1）Sphere Function

$$f(x)=\sum_{i=1}^{n} x_i^{\,2}, x\in[-5,5]^n$$

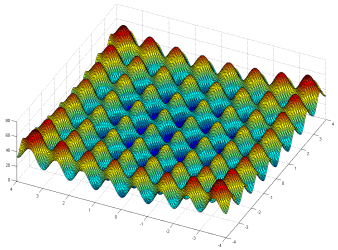
2）Rosenbrock Function

$$f(x)=\sum_{i=1}^{n-1}\left(100\left(x_{i+1}-x_i^{\,2}\right)^2+\left(x_i-1\right)^2\right), x\in[-10,10]^n$$



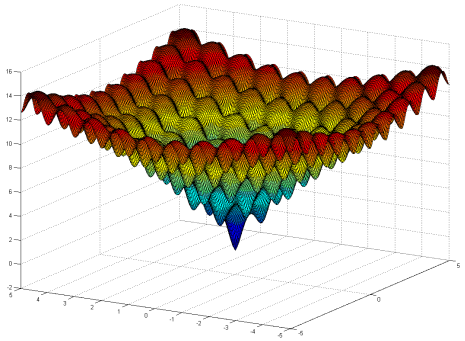
3）Rastrigin Function

$$f(x)=\sum_{i=1}^{D}\left(x_i^2-10\cos(2\pi x_i)+10\right)$$



4）Ackley Function

$$f(x)=20+e-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right), x\in[-32,32]^n$$



Convergence behavior

- Worst: Swarm diverges or remains itinerant
- Optimal: Global best approaches global optimum (swarm may still oscillate)
- Typical: Global best approaches a local optimum (premature collapse of the swarm). Empirically $\omega = 0.7298$ and acceleration coefficients $\alpha_1 = \alpha_2 = 1.49618$ provide good convergence behavior.

Proof of convergence to global optimum for a 1-particle swarm after infinite time (simple setting) was given by van den Bergh

van den Bergh, F.: An Analysis of Particle Swarm Optimizers , Ph.D. Thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.

M. Clerc & J. Kennedy (2002) The particle swarm Explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 7:1, 58-73.

PSO: pros and cons

- **Advantages**
  - Simple implementation
  - Easily to parallelize for concurrent processing
  - Derivative free
  - Few parameters
- **Disadvantages**
  - Tendency to fast and premature convergence to local optima
  - Slow convergence in refined search stage (weak local search ability)

PSO versus GA: similarities

- Random generation of an initial population
- Calculation of a fitness value for each individual
- Reproduction of the population based on fitness values

PSO versus GA: differences

Update rule:

- In GA: by crossover and mutation
- In PSO: particles modify themselves via their own velocity and memory of their personal best and global best

Sharing of information:

- Mutual in GA: Population moves in groups to optimal areas
- One-way in PSO: Source of information is global best. All particles tend to converge to a single best solution quickly

Representation

- GA: usually discrete
- PSO: usually continuous

www.swarmintelligence.org/tutorials.php

Example: clustering data with PSO

Cluster a given dataset into N clusters using PSO (N is given).

- Which type of candidate solutions?
- Which fitness function?

A solution

Each particle represents a list of N centroids: $x_i = (c_1^i, \dots, c_N^i)$

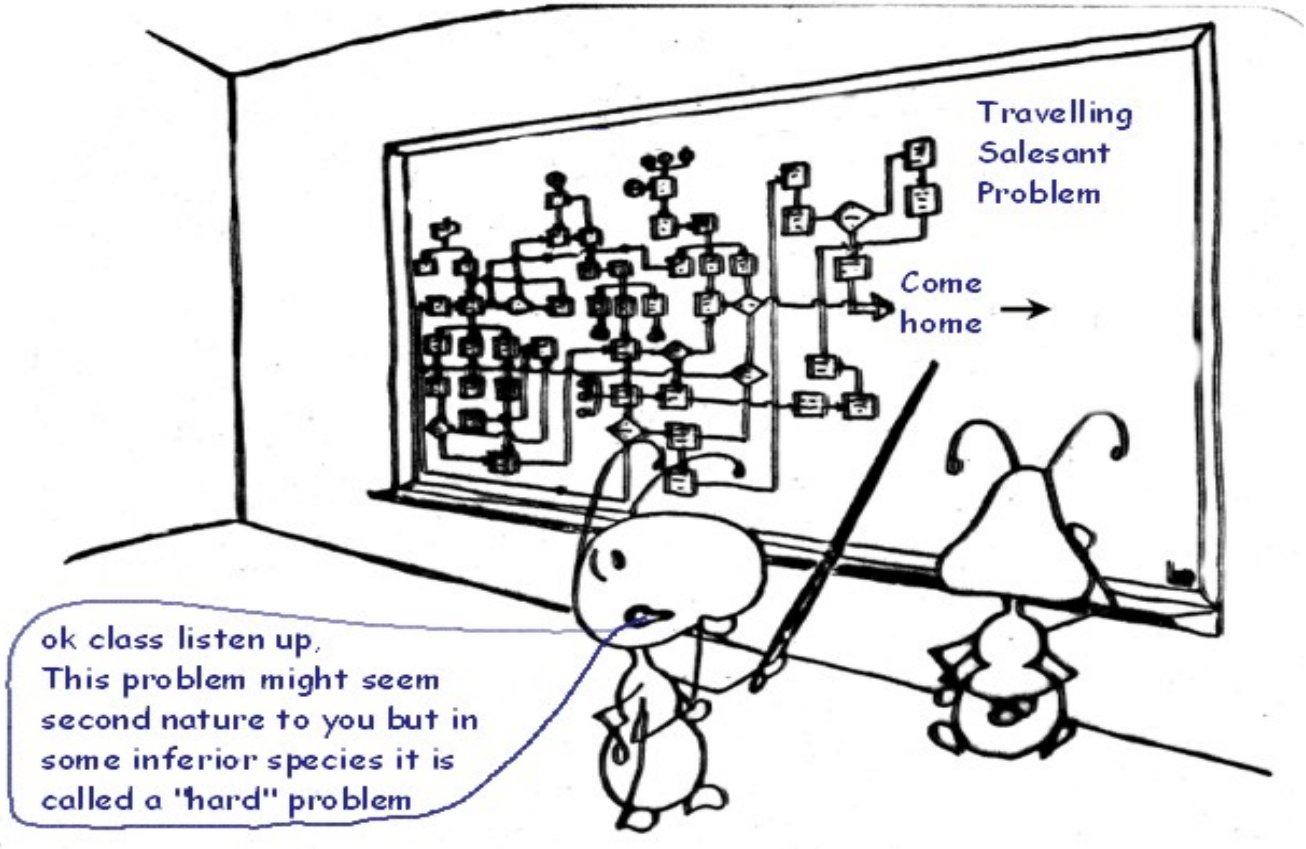
- Initialize each particle to contain N randomly selected centroids;
- For iter=1 to maxiter
  - For each particle $x_i = (c_1^i, \dots, c_N^i)$
    - For each data point z
      - Calculate the distance of z to each cluster centroid c_j^i
      - Assign z to the cluster (centroid) with minimal distance;
    - Compute the fitness of x_i (e.g. average weighted distance of data points from their centroid);
    - Update the local best;
  - Update the global best;
  - Update each particle x_i using PSO update rules:

$$v_i \leftarrow \omega v_i + \alpha_1 r_1 \circ (\hat{x}_i - x_i) + \alpha_2 r_2 \circ (\hat{g} - x_i)$$

$$x_i \leftarrow x_i + v_i$$

Van der Merwe, D. W., and Andries Petrus Engelbrecht. "Data clustering using particle swarm optimization." *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*. Vol. 1. IEEE, 2003.

Ant Colony Optimization (ACO)

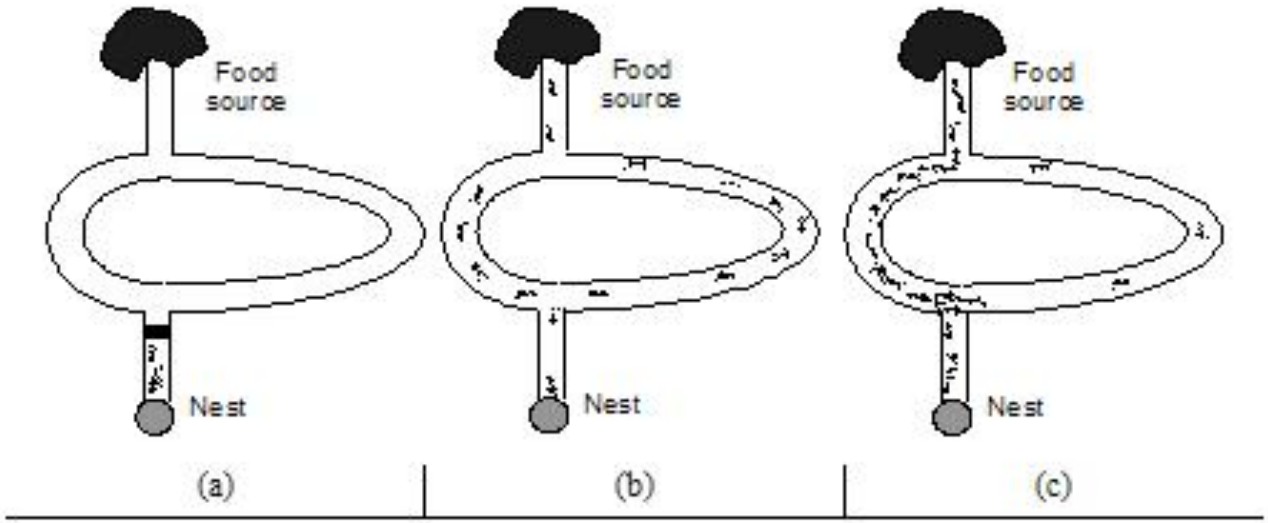


Motivating example from nature

(a) The bridge is initially closed

(b) Initial distribution of ants after the bridge is open

(c) Distribution of ants after some time has passed since they were allowed to exploit the food source



From real to artificial ant colonies

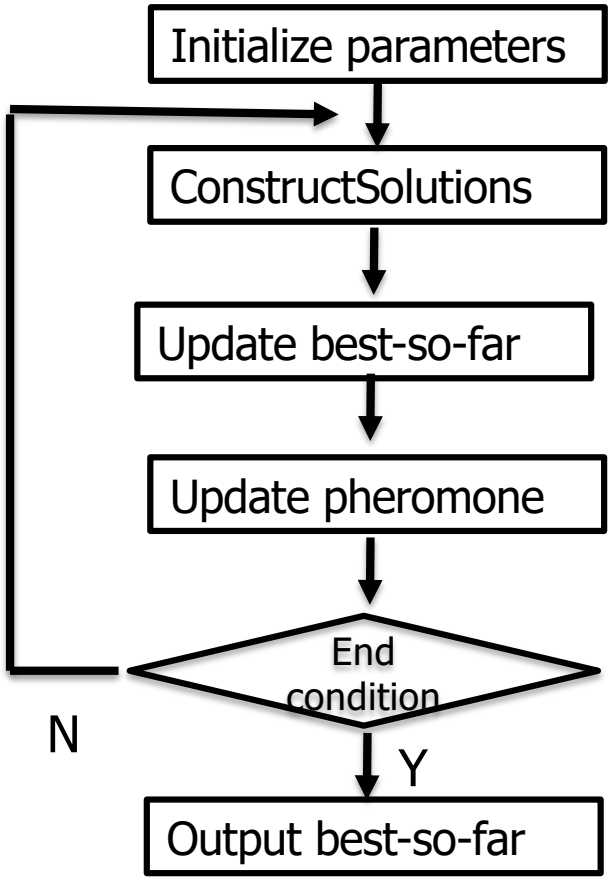
Biology (Ant Foraging)	ACO Algorithms
Ant	Individual (agent) used to build (construct) a solution to the problem
Ant colony	Population (colony) of cooperating individuals known as artificial ants
Pheromone trail	Modification of the environment caused by the artificial ants in order to provide an indirect mean of communication with other ants of the colony This allows the assessment of the quality of a given edge on a graph
Pheromone evaporation	Reduction in the pheromone level of a given path (edge) due to aging

Motivating example from nature

- Ants place pheromone on their way
- Routing depends on the concentration of pheromone
  - Ants using the shorter bridge return faster
  - Pheromone concentration higher on shorter bridge
  - Ants choose shorter bridge more frequently
  - Pheromone concentration of shorter bridge increases
  - Even more ants choose shorter bridge
  - And so forth

Positive feedback loop

ACO algorithm

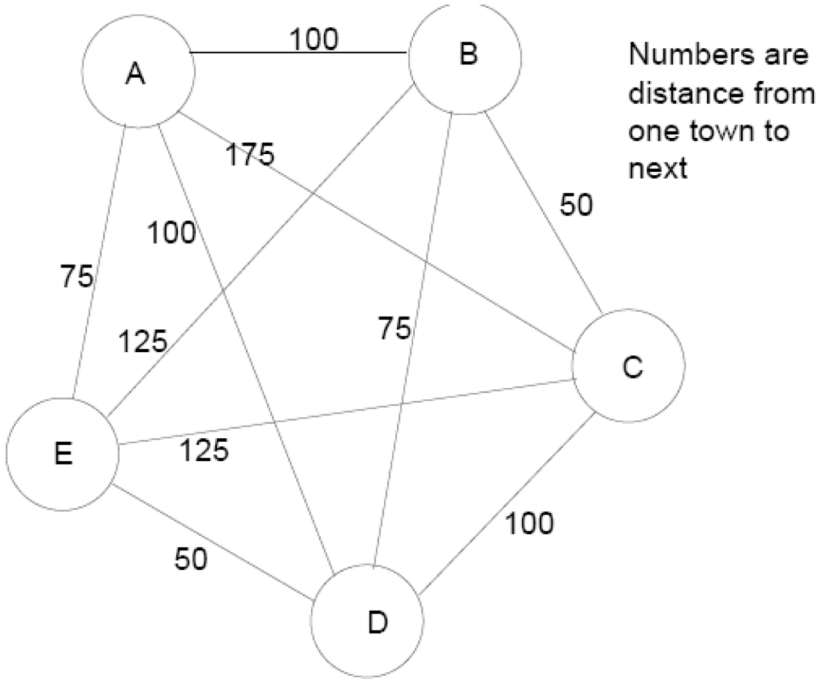


ConstructSolutions

- Expand partial solutions by iteratively selecting and adding components.
- Constructing solutions can be viewed as a walk on the so-called construction graph whose nodes are solution components, and edge weights are pheromone values (they change during execution)

Example: ACO for the TSP

Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city once and returns to the origin city?



ACO for the TSP

- Each ant constructs its own tour from a starting city
- Ant k at city i chooses city j with probability p_{ij}^k that depends on the distance d_{ij} between i and j and the amount τ_{ij} of pheromone on (i, j):

$$p_{ij}^k = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{l\ neighbor\ of\ i} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta}} \qquad j \text{ neighbor of } i$$

$$\eta_{ij} = 1/d_{ij}$$

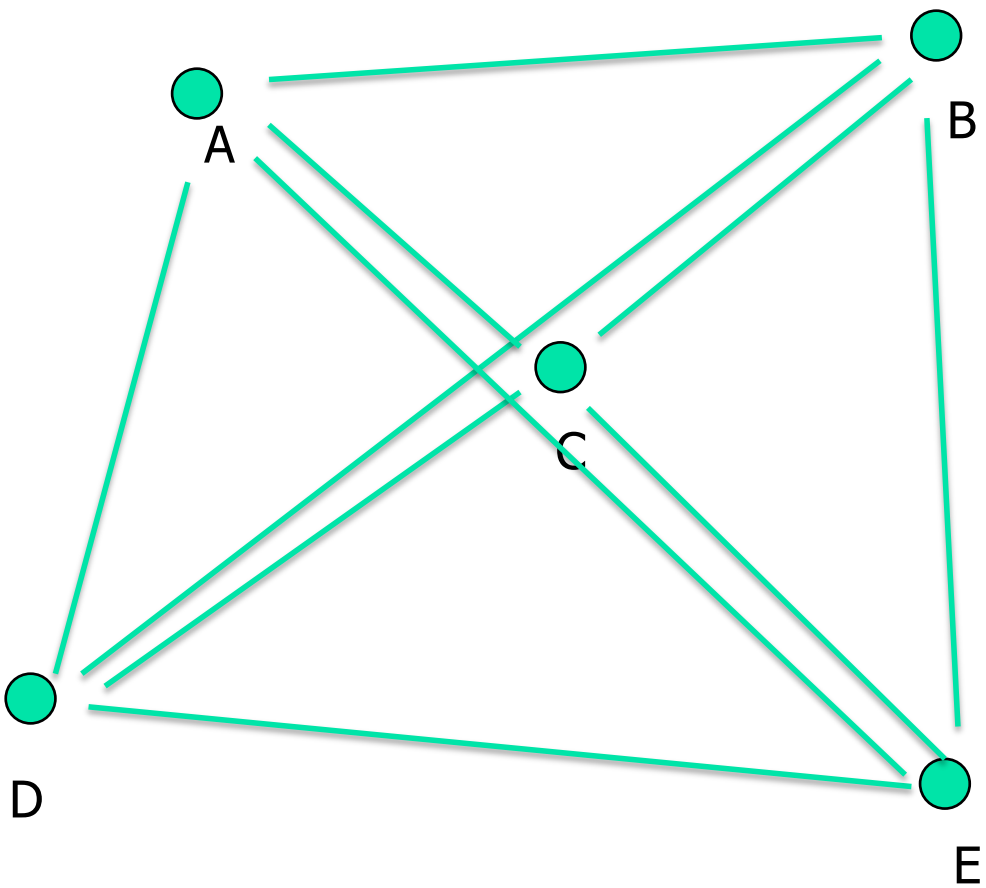
- Legal tours are enforced: choosing already visited cities is disallowed (by means of a tabu list of cities already visited)

Parameters α, β

$$p_{ij}^{k} = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{l\ neighbor\ of\ i} \tau_{il}^{\alpha} \cdot \eta_{il}^{\beta}}$$

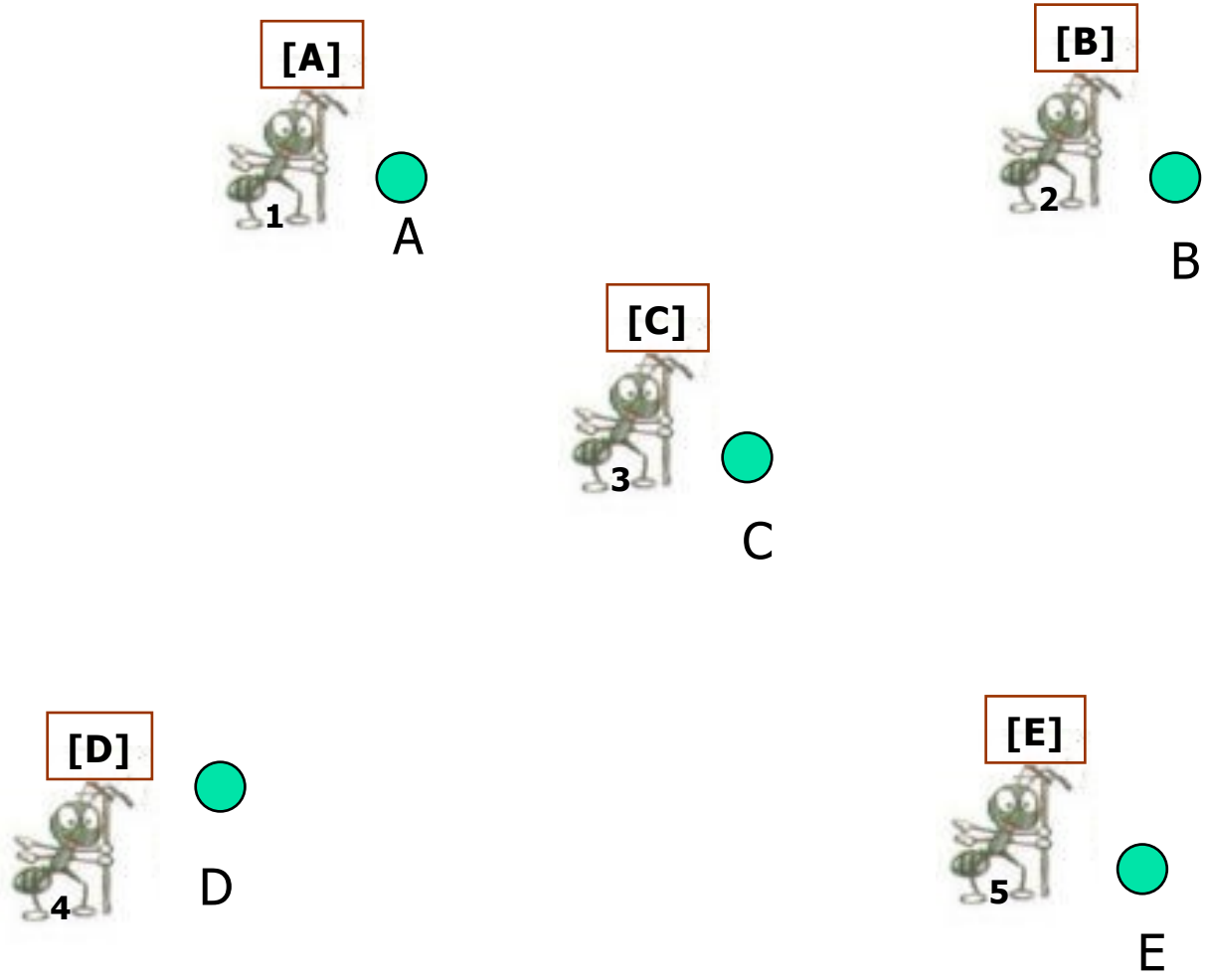
- If $\alpha=0$ then ant k chooses nearest cities more likely (search resembles stochastic greedy search)
- If $\beta=0$ then ant k uses only pheromone amplification (search favors stagnation, where all ants follow the same path)

A toy example simulation



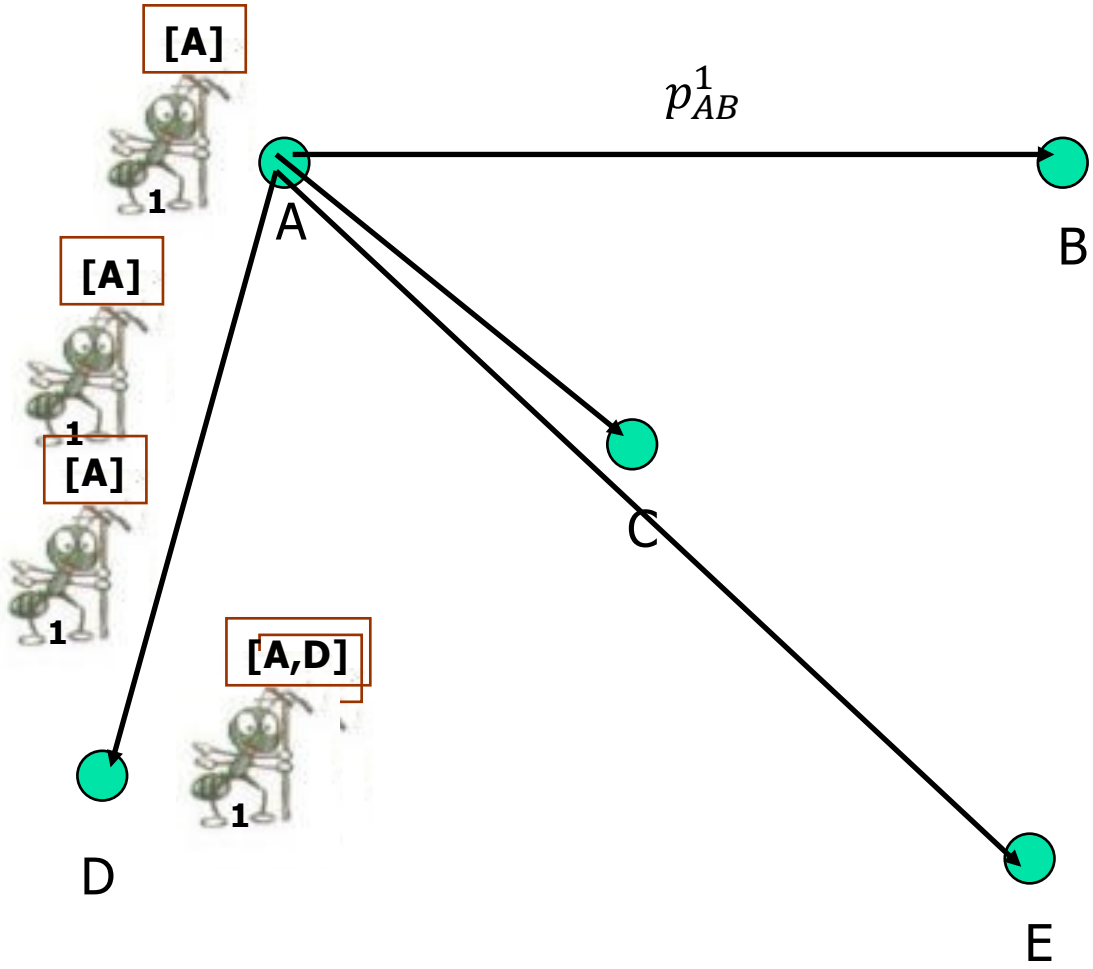
d_{AB} =100;d_{BC} = 60…;d_{DE} =150

Initialization

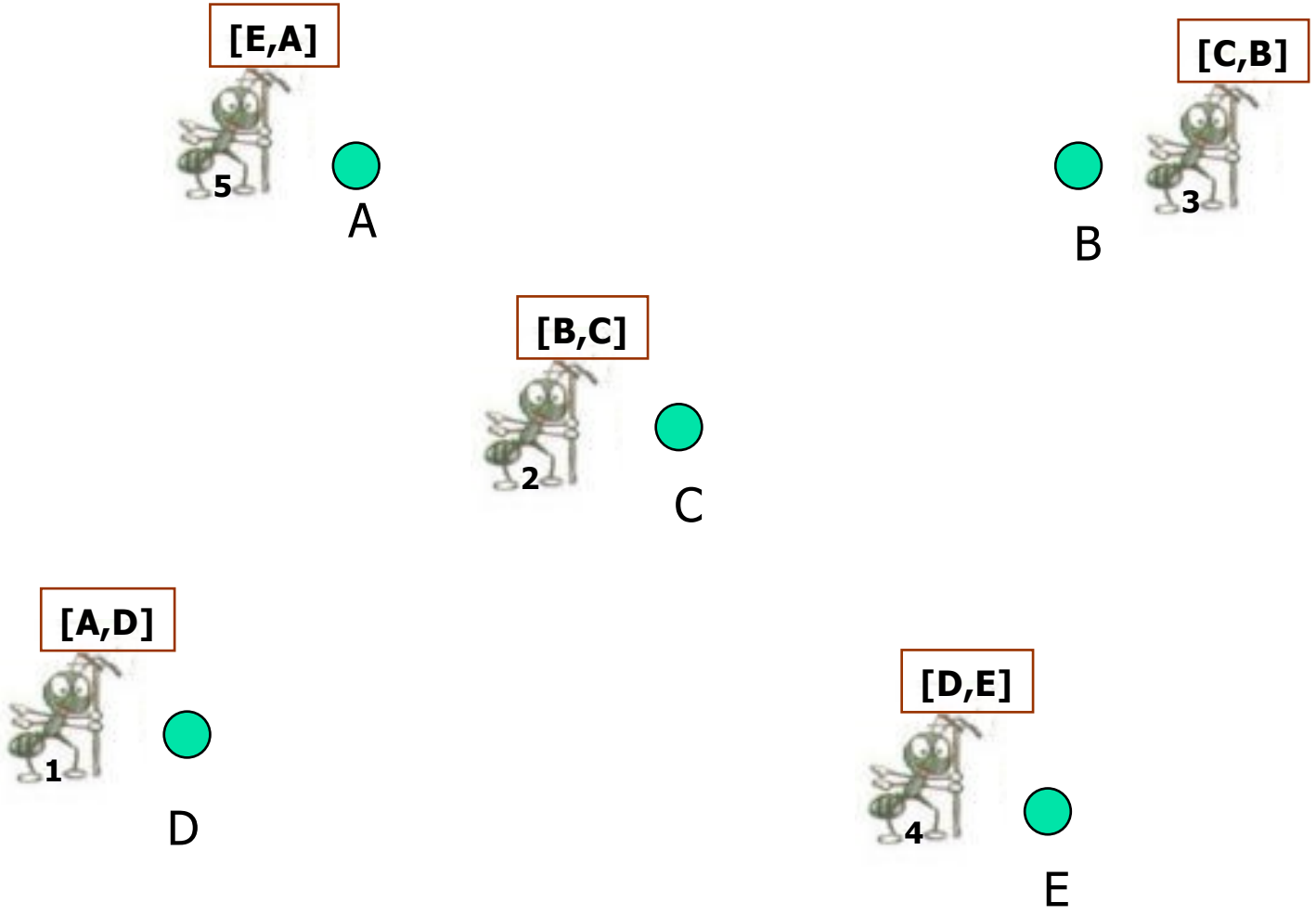


Select next solution component

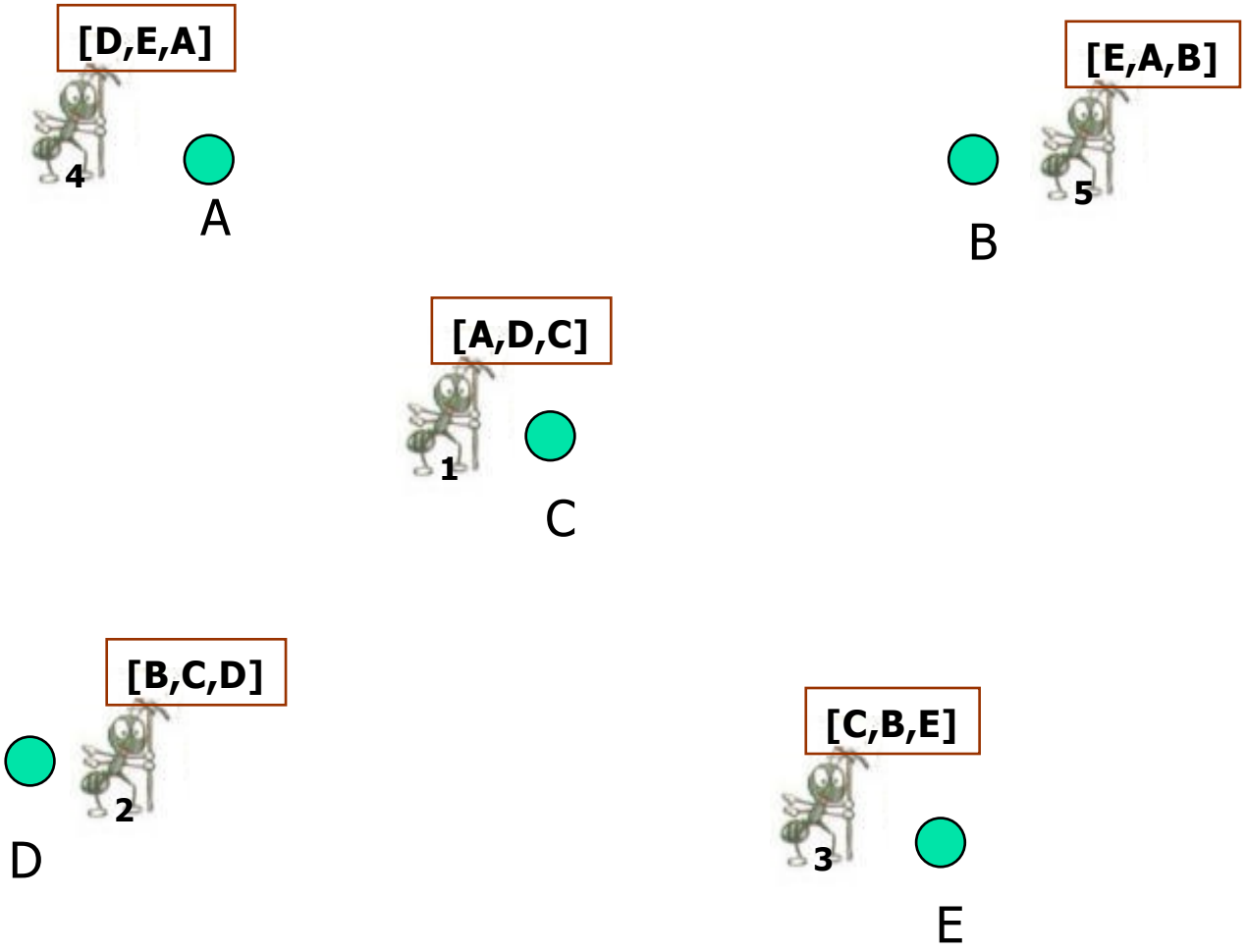
Probabilistic selection



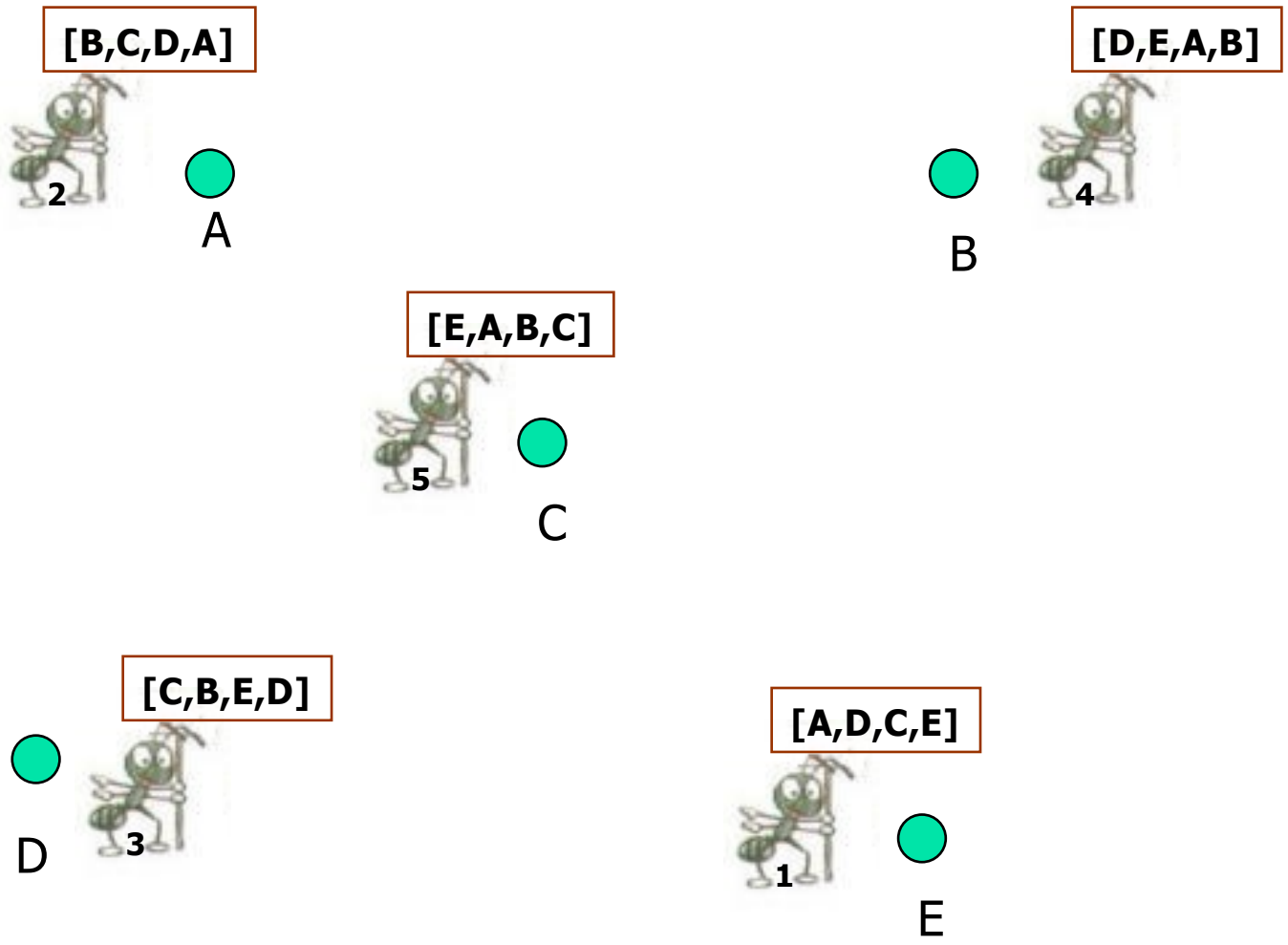
Select next solution component



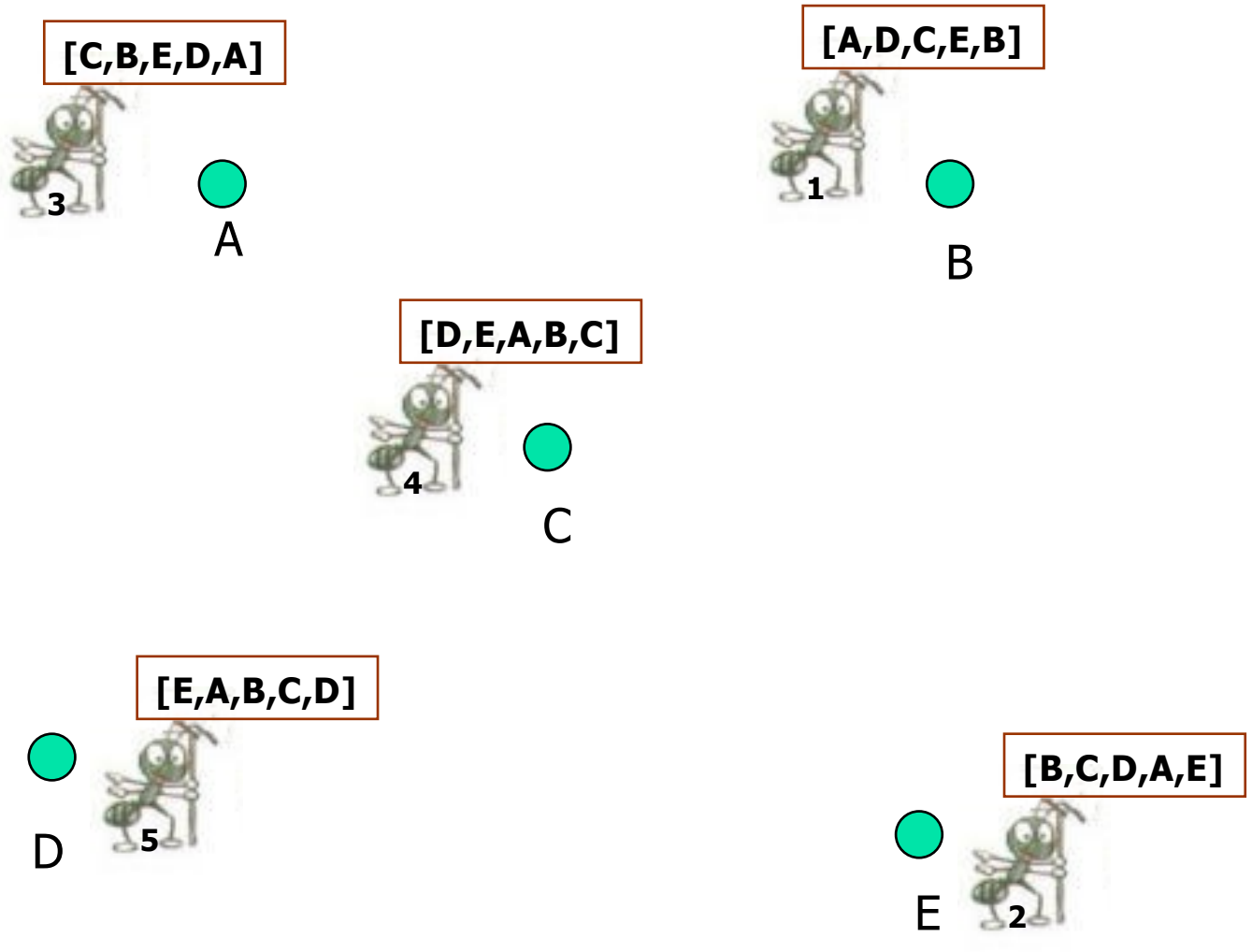
Select next solution component



Select next solution component

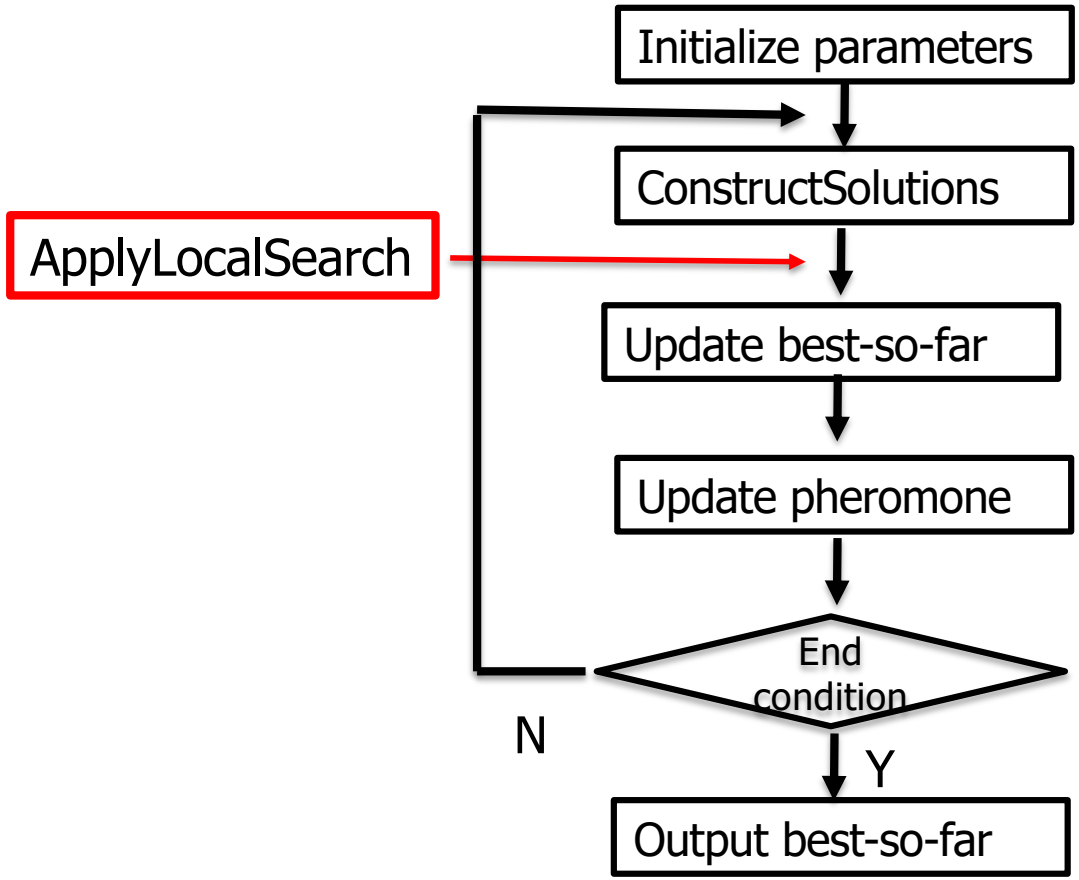


ConstructAntSolutions completed



ApplyLocalSearch

Once candidate solutions have been constructed, and before updating the pheromone, they can be improved through local search (an optional, problem-dependent, step)

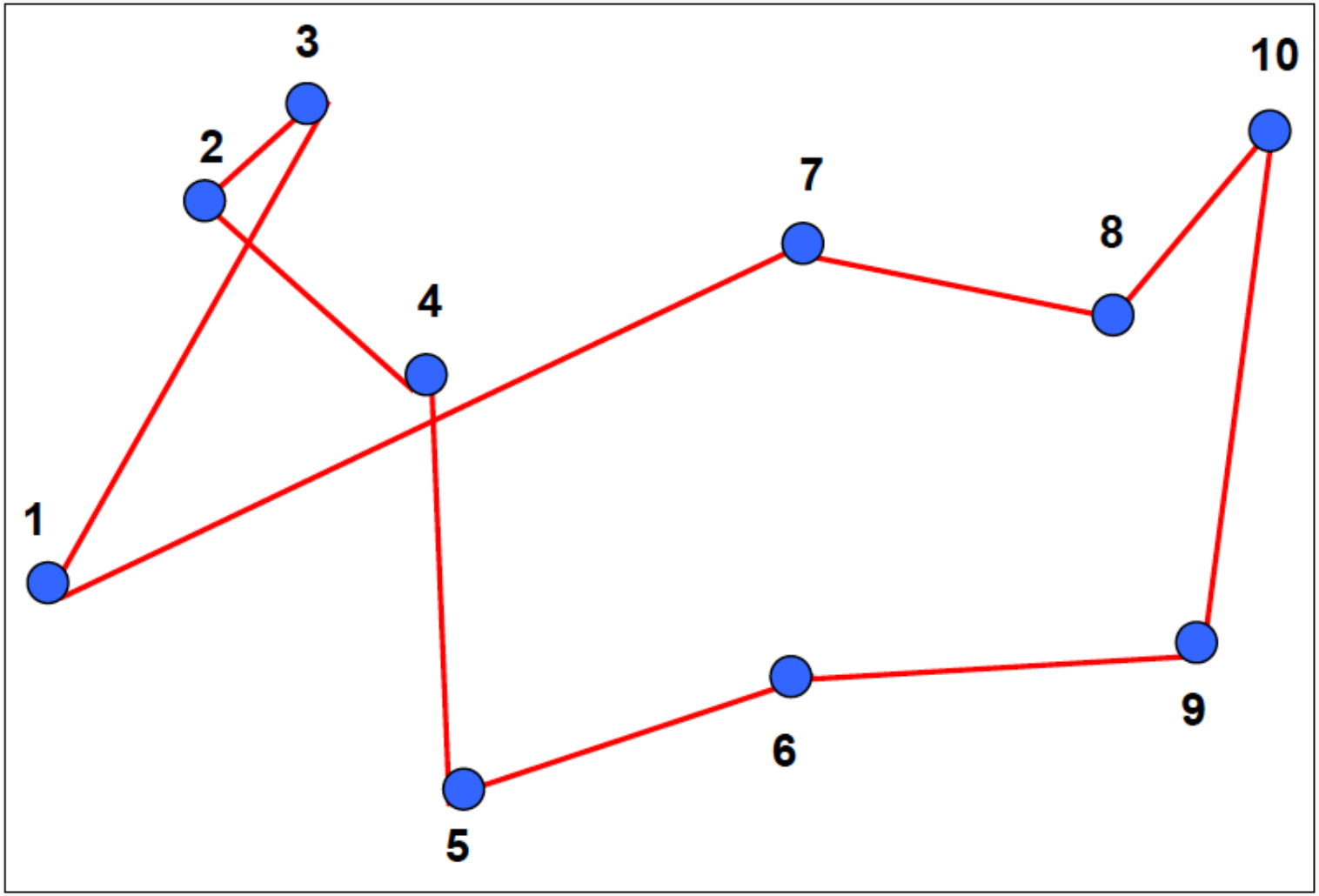


2-opt local search for TSP

Input: tour T; output: a 2-optimal tour T'

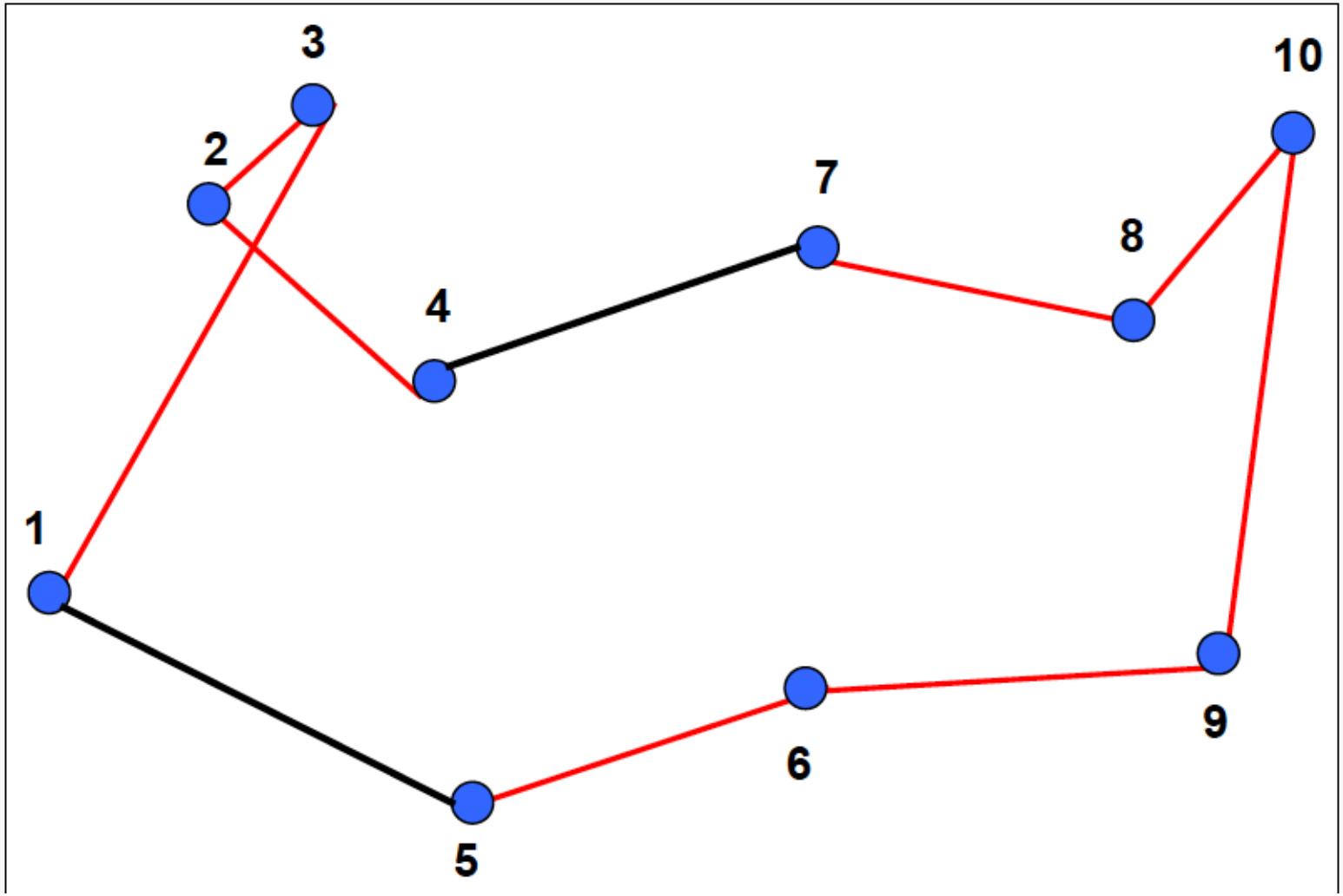
- Look for tour T' obtained from T by deleting 2 edges and adding 2 edges, with lower cost than T
- If one is found, then replaces T with T' and continue, otherwise stop

2-opt local search for TSP: example



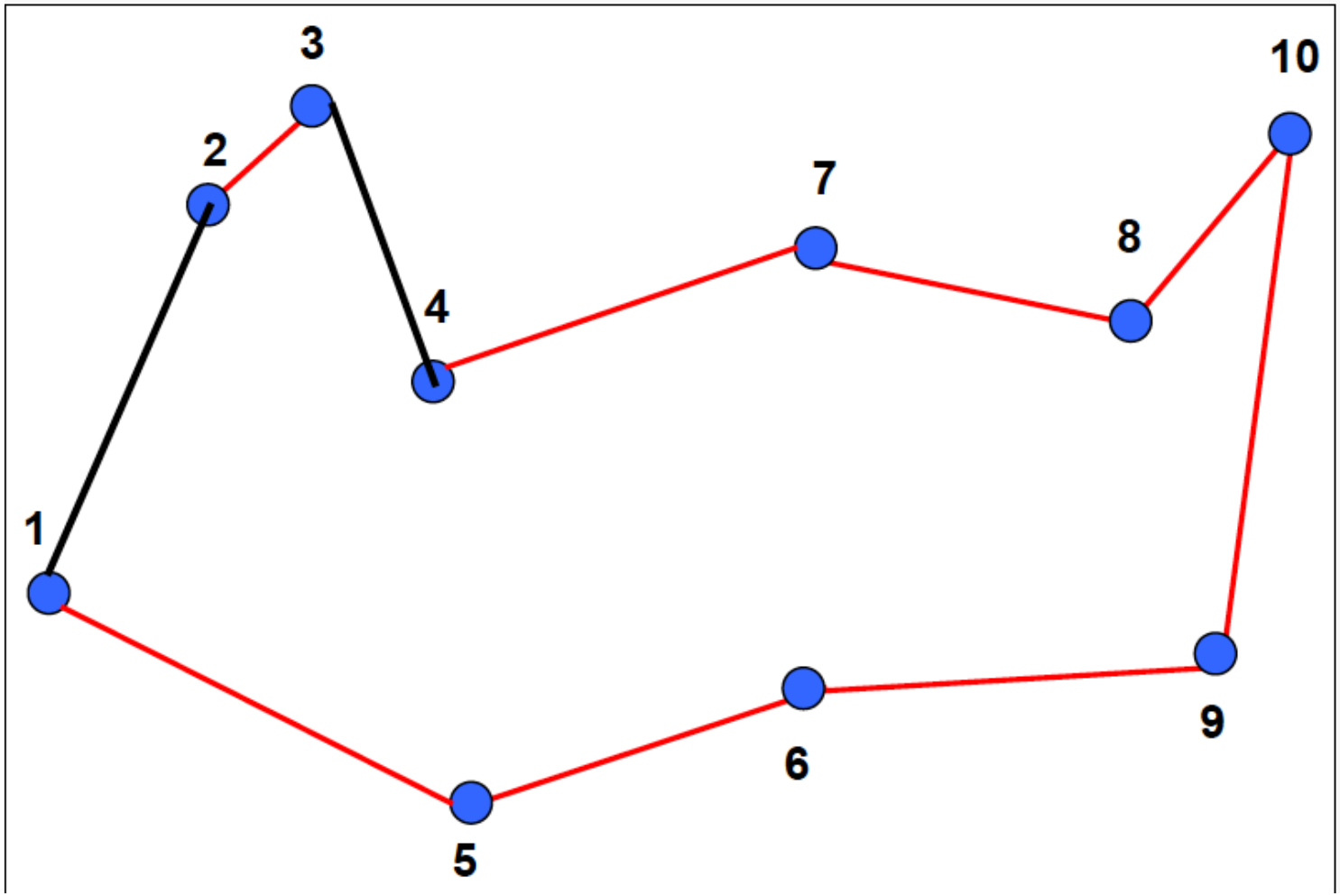
T
1
3
2
4
5
6
9
10
8
7

2-opt local search for TSP: example



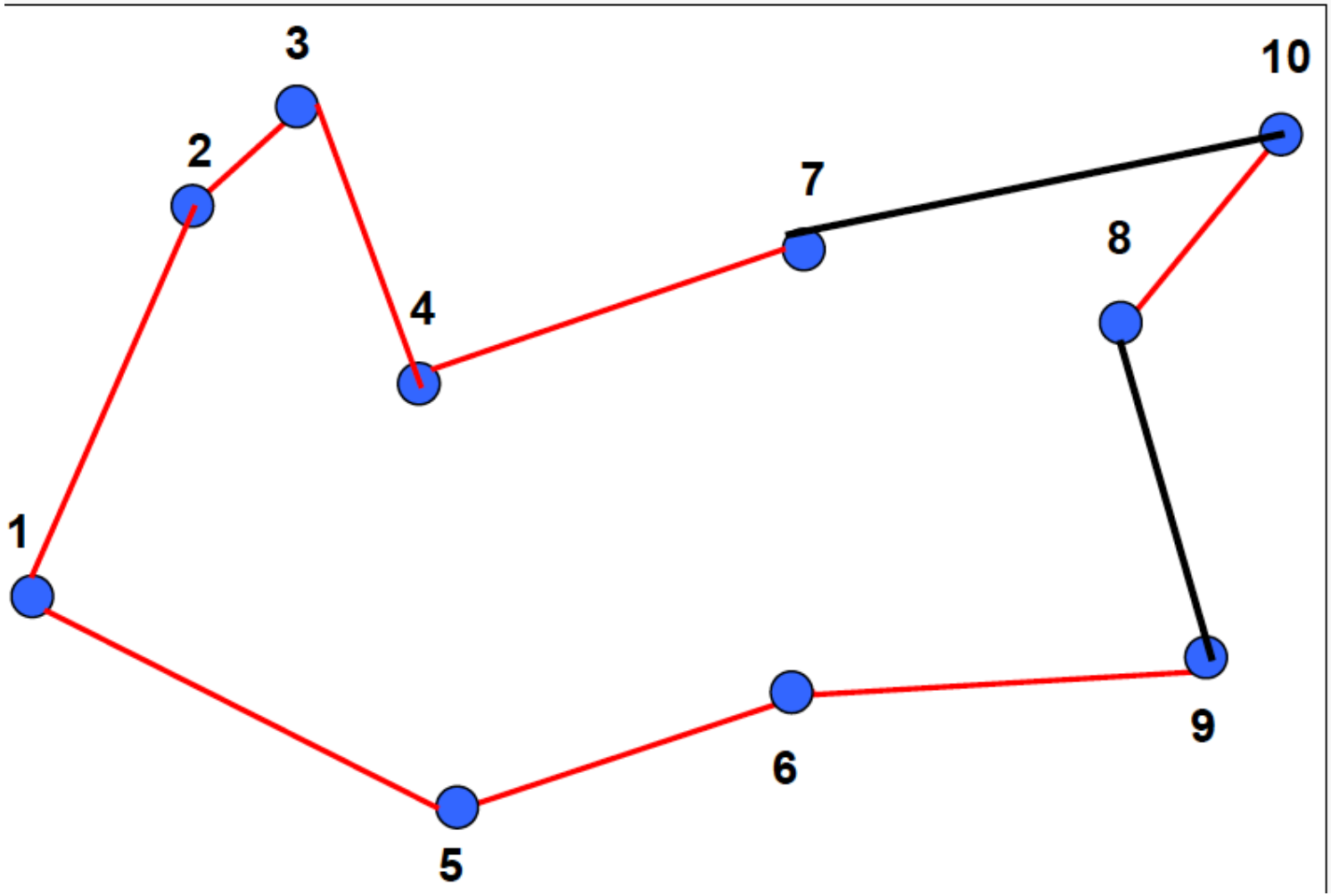
T_1	T_2
1	1
3	3
2	2
4	4
5	7
6	8
9	10
10	9
8	6
7	5

2-opt local search for TSP: example



T_2	T_3
1	1
3	2
2	3
4	4
7	7
8	8
10	10
9	9
6	6
5	5

2-opt local search for TSP: example



T_3	T_4
1	1
2	2
3	3
4	4
7	7
8	10
10	8
9	9
6	6
5	5

UpdatePheromones

- Decrease all pheromone values through pheromone evaporation AND
- Increase pheromone levels associated with edges occurring in good solutions

ACO algorithms mainly differ in how the pheromone update rule is specifically defined

Ant System (AS)

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

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{L_k} & \text{if ant } k \text{ used edge } (i,j) \text{ in its tour} \\ 0 & \text{otherwise} \end{cases}$$

- L_k length of tour built by ant k
- ρ: evaporation rate parameter, values in (0,1], used to forget (bad decisions)
- AS update rule uses all solutions constructed by the ants in the current iteration

Max-Min Ant System (MMAS)

- Values of pheromone are limited τ_{min}, τ_{max} (by truncation)

$$\tau_{ij} = \rho\tau_{ij} + (1-\rho)\Delta\tau_{ij}^{best}$$

$$\Delta\tau_{ij}^{best} = \begin{cases} 1/L_{best} & if\ ij\ in\ best\ tour \\ 0 & otherwise \end{cases}$$

L_{best} length of the tour of the best ant (two variants: iteration-best or best-so-far)

- MMAS update rule uses only the best solution constructed by the ants in the current iteration (IB-update) or over all iterations so far (BS-update)

Ant Colony System (ACS)

- Pseudorandom proportional transition rule:
  - with probability q_0 select component *j* from *i* that maximizes $\tau_{ij}\eta_{ij}$ otherwise use standard probabilistic transition rule
- 2-steps pheromone update:
  - After each solution construction step decrease the pheromone concentration: $\tau_{ij} = (1-\varphi)\tau_{ij} + \varphi\tau_0$ each time *ij* traversed by an ant, τ_0 very small positive constant
  - After solution construction: $\tau_{ij} = \rho\tau_{ij} + (1-\rho)\Delta\tau_{ij}^{best}$ (BS-update)
- ACS update rule decreases pheromone of visited edges during construction of solutions, and updates pheromone using the best solution so far (BS-update)

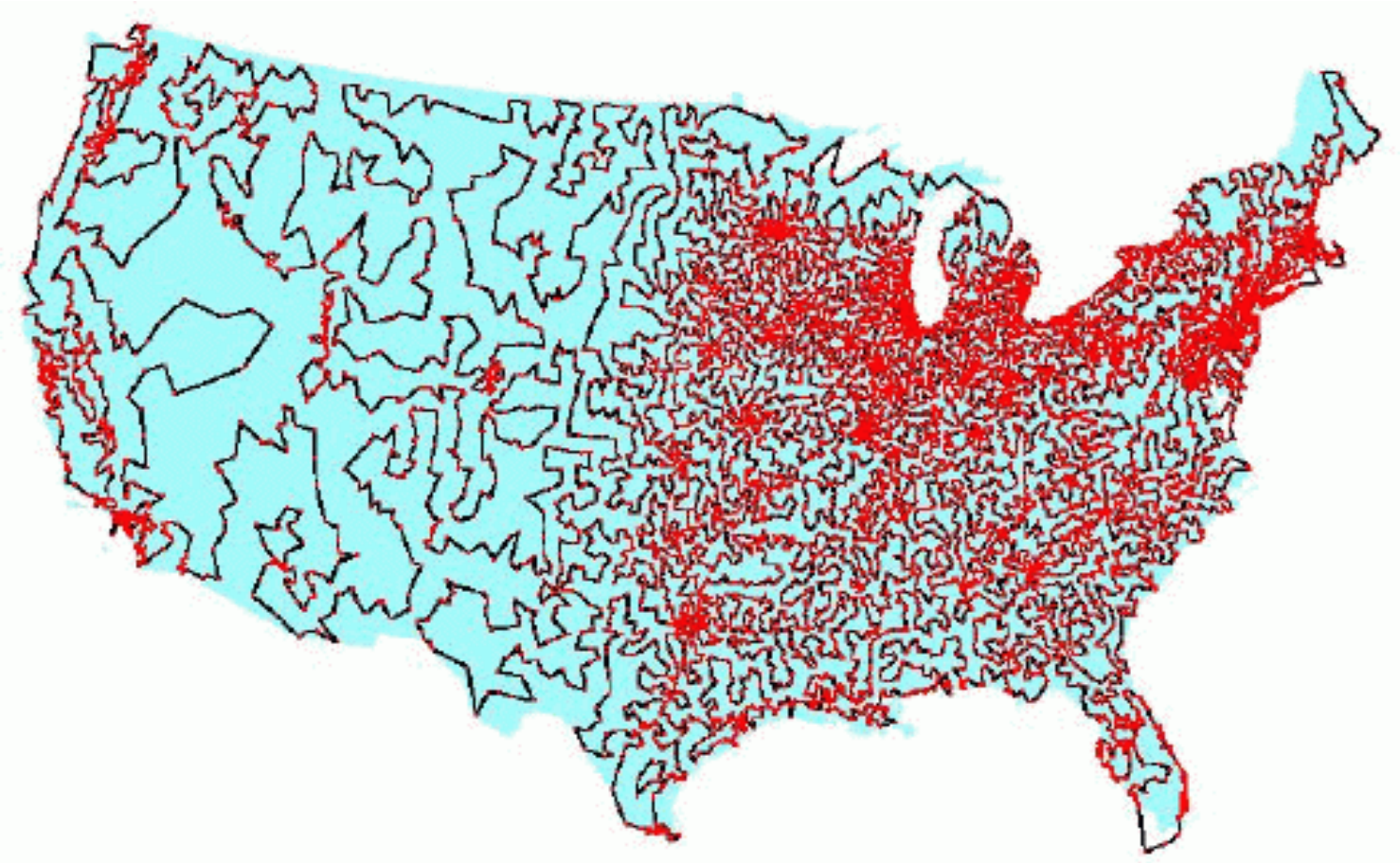
ACO algorithms

Algorithm	*Authors*	*Year*
Ant System (AS)	Dorigo et al.	1991
Elitist AS	Dorigo et al.	1992
Ant-Q	Gambardella & Dorigo	1995
Ant Colony System	Dorigo & Gambardella	1996
$\mathcal{MAX}$–$\mathcal{MIN}$ AS	Stützle & Hoos	1996
Rank-based AS	Bullnheimer et al.	1997
ANTS	Maniezzo	1999
BWAS	Cordón et al.	2000
Hyper-cube AS	Blum et al.	2001

Dorigo, Marco, and Mauro Birattari. "Ant colony optimization." Encyclopedia of machine learning. Springer, Boston, MA, 2011. 36-39.

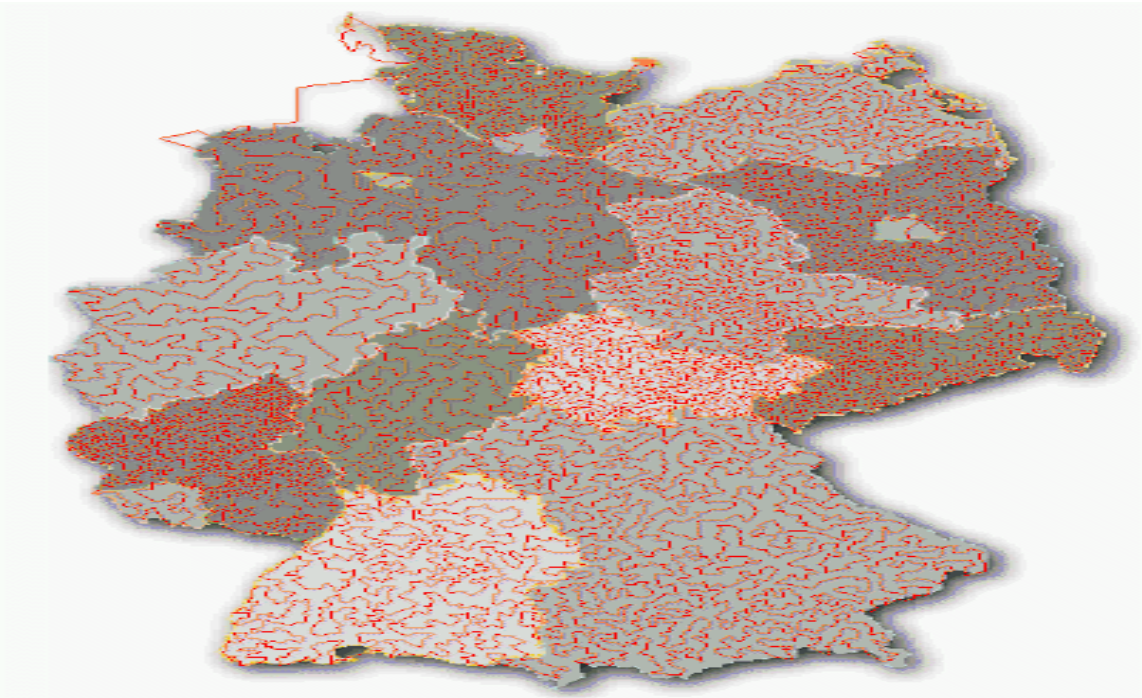
Real life TSP benchmark instances (1/2)

USA Towns of 500 or more population: 13509 cities (1998 Applegate, Bixby, Chvátal and Cook)



Real life TSP benchmark instances (2/2)

Towns in Germany: 15112 cities
(2001Applegate, Bixby, Chvátal and Cook)



TSP benchmark instances
http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

Convergence results

- For a particular ACO algorithm the objective function of the best solution so far converges with probability one to the optimal objective function value: weak result, holds also for random search (Stutzle andDorigo 2002)
- For a particular ACO algorithm, its current solutions converge to an optimal solution with probability one: the optimal paths get attractors of the stochastic dynamic process realized by the algorithm (Gutjahr 2002)

Stutzle, Thomas, and Marco Dorigo. "A short convergence proof for a class of ant colony optimization algorithms." *IEEE Transactions on evolutionary computation* 6.4 (2002): 358-365.

Gutjahr, Walter J. "ACO algorithms with guaranteed convergence to the optimal solution." *Information processing letters* 82.3 (2002): 145-153.

Search bias in ACO

- Induced selection bias occurs if some solution components occur in a larger number of feasible solutions (in terms of sequences built by the algorithm)
- If this occurs there can be unfair competition between components

Blum, Christian, and Marco Dorigo. "Search bias in ant colony optimization: On the role of competition-balanced systems." *IEEE Transactions on Evolutionary Computation* 9.2 (2005): 159-174.

Competition-Balanced System

- An ACO and a problem instance are Competition-Balanced System (CBS) if given a feasible partial solution s_p and the set C of solution components that can be added to extend s_p, all c in C occur in the same number of *feasible* solutions
- CBS implies fair competition between components

Blum, Christian, and Marco Dorigo. "Search bias in ant colony optimization: On the role of competition-balanced systems." *IEEE Transactions on Evolutionary Computation* 9.2 (2005): 159-174.

CBS: example

- ACO and the TSP are a Competition-Balanced System because each city occurs exactly once in each feasible solution

ACO: applications

- ACO are mainly applied to discrete optimization problems:
  - Scheduling problems
  - Graph coloring problem
  - Maximum clique problem
  - Bioinformatics, e.g. protein folding
  - Dynamic optimization applications (it adapts to changes such as new distances, etc)
  - etc

Blum, Christian. "Ant colony optimization: Introduction and recent trends." Physics of Life reviews 2.4 (2005): 353-373.

ACO: pros and cons

- Pros:
  - Inherent parallelism
  - Positive feedback accounts for rapid discovery of good solutions
- Cons:
  - Theoretical analysis is difficult: sequences of random decisions (not independent), probability distribution changes by iteration, time to convergence uncertain.

ACO versus PSO

Criteria	ACO	PSO
Communication Mechanism	ACO uses an indirect communication mechanism among ants, called stigmergy, which means interaction through the environment.	The communication among particles in PSO is rather direct without altering the environment.
Problem Types	ACO was originally used to solve combinatorial (discrete) optimization problems, but it was later modified to adapt continuous problems.	PSO was originally used to solve continuous problems, but it was later modified to adapt binary/ discrete optimization problems.
Problem Representation	ACO's solution space is typically represented as a weighted graph, called construction graph.	PSO's solution space is typically represented as a set of n-dimensional points.
Algorithm Applicability	ACO is commonly more applicable to problems where source and destination are predefined and specific.	PSO is commonly more applicable to problems where previous and next particle positions at each point are clear and uniquely defined.
Algorithm Objective	ACO's objective is generally searching for an optimal path in the construction graph.	PSO's objective is generally finding the location of an optimal point in a Cartesian coordinate system.
Examples of Algorithm Applications	Sequential ordering [32], scheduling [33], assembly line balancing [34], probabilistic TSP [35], DNA sequencing [36], 2D-HP protein folding [37], and protein–ligand docking [38].	Track dynamic systems [60], evolve NN weights [61], analyze human tremor [62], register 3D-to-3D biomedical image [63], control reactive power and voltage [64], and even play games [65].

Swarm Intelligence: *Concepts*, *Models* and *Applications*. Technical Report 2012-585. Hazem Ahmed. Janice Glasgow. School of Computing. Queen's University.

Ideas for your project

- PSO: Evolve DNN architectures, Junior, Francisco Erivaldo Fernandes, and Gary G. Yen. "Particle swarm optimization of deep neural networks architectures for image classification." *Swarm and Evolutionary Computation* 49 (2019): 62-74.
- ACO: Optimize Long Short-Term Memory Recurrent Neural Networks, ElSaid, AbdElRahman, et al. "Using ant colony optimization to optimize long short-term memory recurrent neural networks." *Proceedings of the Genetic and Evolutionary Computation Conference*. 2018.
- Analyze ACO extensions in the context of a real-life application or over benchmark datastes, e.g. recent survey by Dorigo, M., & Stützle, T. (2019). Ant colony optimization: overview and recent advances. In *Handbook of metaheuristics* (pp. 311-351). Springer, Cham.

- Next lecture (23/2): Artificial Immune Systems
- Deadline assignment on SI: February 20