Evolutionary Computing

Swarm Intelligence

Dr. Petr Musilek

Department of Electrical and Computer Engineering
University of Alberta

Fall 2019

Motivation: Treasure Hunt

- **Who?** You and a group of friends
- **What do you know?** Knowledge of the approximate area of the treasure, but not exactly where it is located
- **What do you have?**
  - Metal detector
  - Strength and position of signal of your neighbors' metal detectors
  - Your friends!
- **What is your goal?** To find the treasure, or at least part of it

Motivation: Treasure Hunting (cont)

The agreement: You have agreed on some sharing mechanism:

- all who have taken part in the search will be rewarded, but with the person who found the treasure getting a higher reward than all others
- the rest are rewarded based on distance from the treasure at the time when the first one finds the treasure

What will be your actions?

- Ignore your friends?
- Use the information from your neighboring friends?

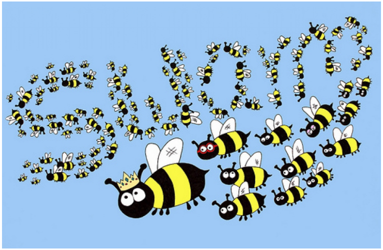
Swarm Intelligence

What is a Swarm?

- A loosely structured collection of interacting agents
- Agents:
  - Individuals that belong to a group (but are not necessarily identical)
  - They contribute to and benefit from the group
  - They can recognize, communicate, and/or interact with each other
- A swarm is better understood if thought of as agents exhibiting a collective behavior

Examples of Swarms in Nature

Classic Example: Swarm of Bees



Can be extended to other similar systems:

- Ant colony [agents: ants]
- Flock of birds [agents: birds]
- Crowd [agents: humans]
- Immune system [agents: cells and molecules]

Examples of Swarms in Nature

Examples of Swarms in Nature

Examples of Swarms in Nature

Swarm Intelligence (SI)

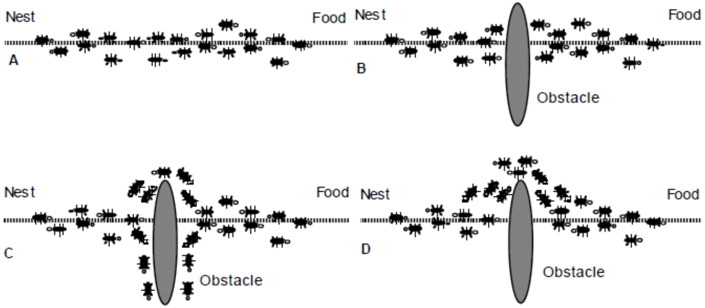
- Computational intelligence (CI) technique based on the collective behavior in decentralized, self-organized systems
- Generally made up of agents who interact with each other and the environment
- No centralized control structures
- No need to have very complex and superior intelligent agents
- Based on group behavior found in nature

Ant Colonies in Nature

Ant Colonies in Nature

- Lack of central control
- Intelligent behavior is emergent from the collective actions of simple agents
- Ants communicate via altering the environment (deposit pheromone)
- Ants are more likely to follow routes with more pheromone

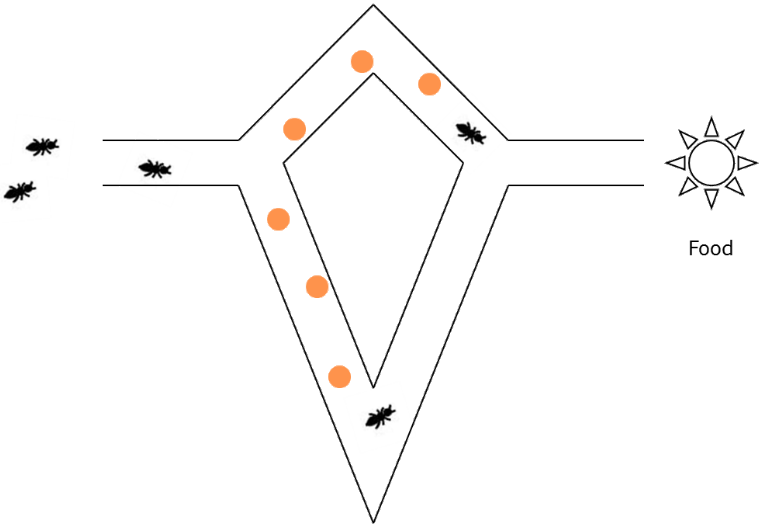
Ant Colonies - Foraging



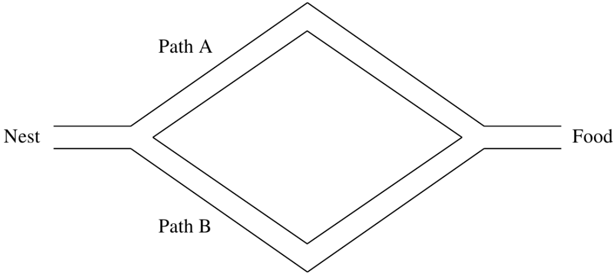
Foraging Strategies

- Ants deposit pheromone on the paths that they cover and this results in the building of a solution (optimal path).
- A way of knowledge sharing and communication through the environment (stigmergy)
- In implementation concept of pheromone evaporation is used – avoiding local optima.

Pheromone



Foraging: Bridge experiment



Probability of the next ant to choose path A:

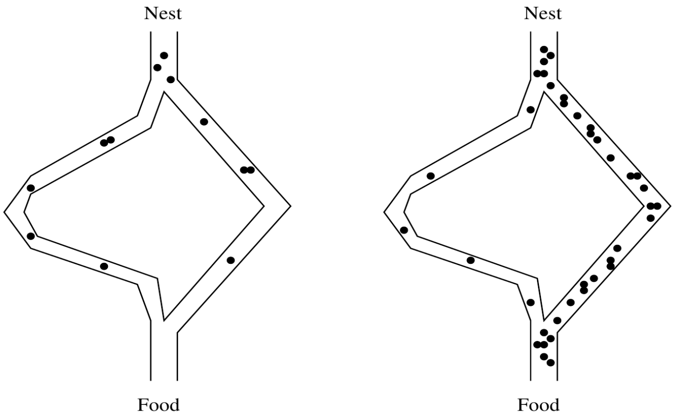
$$P_A(t+1) = \frac{(c+n_A(t))^\alpha}{(c+n_A(t))^\alpha+(c+n_B(t))^\alpha} = 1 - P_B(t+1)$$

- $n_A(t)$ and $n_B(t)$ are the number of ants on paths A and B respectively at time step t
- c quantifies the degree of attraction of an unexplored branch
- α biases towards pheromone deposits in the decision process

Foraging: Bridge experiment

- The larger the value of α, the higher the probability that the next ant will follow the path with a higher pheromone concentration
- The larger the value of c, the more pheromone deposits are required to make the choice of path non-random
- The decision rule:
  if $U(0,1) \leq P_A(t+1)$ then follow path A otherwise follow path B

Foraging: Extended binary bridge



Differential path length effect:

- The probability of selecting the shorter path increases with the length ratio between the two paths

Foraging Behavior of Ants (cont)

Each ant is a stimulus-response agent, following simple production rules :

Let $r \sim U(0,1)$
for each potential path A **do**
    Calculate P_A;
    if $r \leq P_A$ **then**
        Follow path A;
        Break;
    end if
end for

Stigmergy and Artificial Pheromone

- Generally stated, stigmergy is a class of mechanisms that mediate animal-to-animal interactions
- A form of indirect communication mediated by modifications to the environment
- Forms of stigmergy:
  - *Sematectonic stigmergy* refers to communication via changes in the physical characteristics of the environment - nest building, brood sorting
  - *Sign-based stigmergy* facilitates communication via a signaling mechanism, implemented via chemical compounds deposited by ants - foraging

Artificial Stigmergy

Artificial Stigmergy

The indirect communication mediated by numeric modifications of environmental states which are only locally accessible by the communicating agents

- The essence of modeling ant behavior is to find a mathematical model that accurately describes the stigmergic characteristics of the corresponding ant individuals
- Define stimergic variables which encapsulate the information used by artificial ants to communicate indirectly:
  - for foraging behavior, artificial pheromone

Ant Algorithms

- Ant algorithms are population-based systems inspired by observations of real ant colonies
- Cooperation among individuals in an ant algorithm is achieved by exploiting the stigmergic communication mechanisms observed in real ant colonies
- Foraging-based ant algorithms are generally referred to as

Ant Colony Optimization Meta-Heuristics

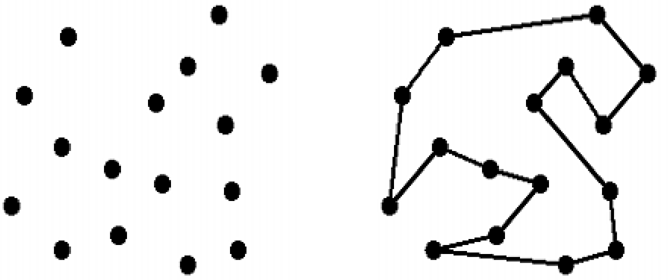
Ant Colony Optimization (ACO)

- Introduced in 1992 by Marco Dorigo
- Population-based meta-heuristic to find approximate solutions to difficult search and optimization problems
- A set of software agents
- Stochastic
- Incrementally build solutions by moving on a graph
- Constraints of the problem are built into the search process of the ants

Traveling Salesman Problem (TSP)

- **Informally:** The problem of finding the shortest tour through a set of cities starting at some city and going through all other cities once and only once, returning to the starting city
- **Formally:** Finding the shortest Hamiltonian path in a fully-connected graph
- TSP belongs to the class of NP-complete combinatorial optimization problems. Thus, it is assumed that there is no efficient algorithm for solving TSPs.

Simple TSP Instance



- Every edge has a distance and the answer would be the shortest tour having the minimum sum of the edges in the tour
- The direct solution will have a running time of $O(n!)$

16 cities: 16! = 209,227,898,88,000

Computing a solution

Approximate methods - iterative methods that try to improve the search over the course of a specific number of steps

- **Exploration:** Making choices contrary to previous experience in order to explore new tours
- **Exploitation:** Making choices considering previous experience in order to find tours close to the ones we already have

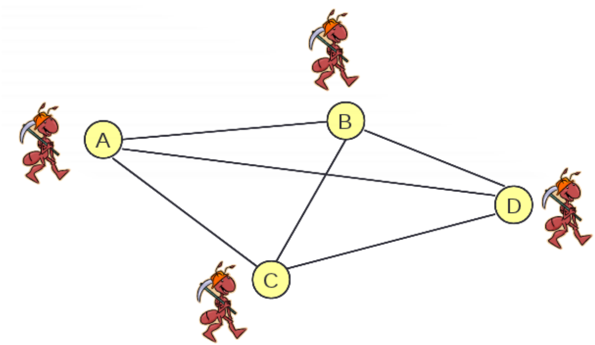
Ant System (AS)

Ant Systems for TSP

Graph (N,E): where N = cities/nodes, E = edges

d_{ij} = the tour cost from city i to city j (edge weight)

Ants "Independently" move from one city i to the next j with certain transition probability. Let the number of ants be m and cities n.



Ant System (AS)

Transition from city i to j depends on

- Pheromone intensity $\tau_{ij}(t)$ for each edge – represents the learned desirability to visit city j when in city i
- Heuristic information (visibility) $\eta_{ij} = 1/d_{ij}$ represents local information – heuristic desirability to visit city j when in city i.

Generally, have several ants searching the solution space, e.g.

$$(m = n)$$

Ant System (AS)

- Transition Rule
- Probability of ant k going from city i to j
- α and β are adjustable parameters tuning the balance between exploration and exploitation

$$p_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{u \in \mathcal{N}_i^k(t)} \tau_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)} & \text{if } j \in \mathcal{N}_i^k \\ 0 & \text{if } j \notin \mathcal{N}_i^k(t) \end{cases}$$

State Transition Rule: example

$$
\begin{aligned}
\tau(A,B) &= 150;\ \ \eta(A,B) = 1/10 \\
\tau(A,C) &= 35;\ \ \eta(A,C) = 1/7 \\
\tau(A,D) &= 90;\ \ \eta(A,D) = 1/15
\end{aligned}
$$

$$p_{ij}^{k}(t) = \frac{\tau_{ij}(t)^{\alpha}\eta_{ij}^{\beta}}{\sum_{u \in \mathcal{N}_i^k(t)} \tau_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)}; p_{AB}^{k}(t) = \frac{15}{15+5+6}$$

i.e. $p_{AB} = 15/26$, $p_{AC} = 5/26$, $p_{AD} = 6/26$

Exploration-Explotation Trade-off

- A balance between pheromone intensity, τ_{ij} , and heuristic information, η_{ij}
- If $\alpha = 0$:
  - no pheromone information is used, i.e. previous search experience is neglected
  - the search then degrades to a stochastic greedy search
- If $\beta = 0$:
  - the attractiveness of moves is neglected
  - the search algorithm is similar to SACO
- Heuristic information adds an explicit bias towards the most attractive solutions, e.g.

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

Pheromone Update

Pheromone update:

$$\Delta\tau_{ij}^{k} = Q/d_{ij}^{k}(t) \;\text{ if }\; (i,j) \in T^{k}(t) \;\text{ else }\; 0$$

where T is the tour done at time t by ant k, d is the deistance, Q is a heuristic parameter (positive constant).
Pheromone decay:

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

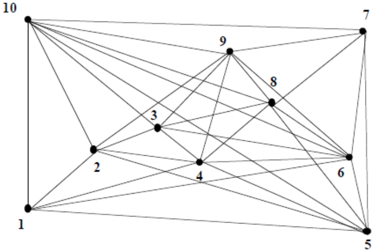
where $\rho = (0, 1]$ is the evaporation rate, m is the number of ants and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone laid on edge (i, j) by ant k.

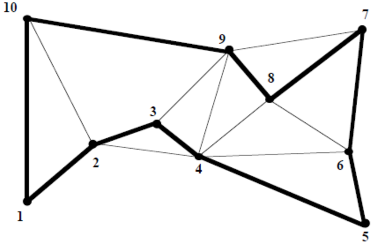
ACO for TSP

```
Loop
    Randomly position m artificial ants on n cities
    For city=1 to n
        For ant=1 to m
            {Each ant builds a solution by adding one city after
            the other}
            Select probabilistically the next city according to
                        exploration and exploitation mechanism
            Apply the local trail updating rule
        End for
    End for
    Apply the global trail updating rule using the best ant
Until End_condition
```

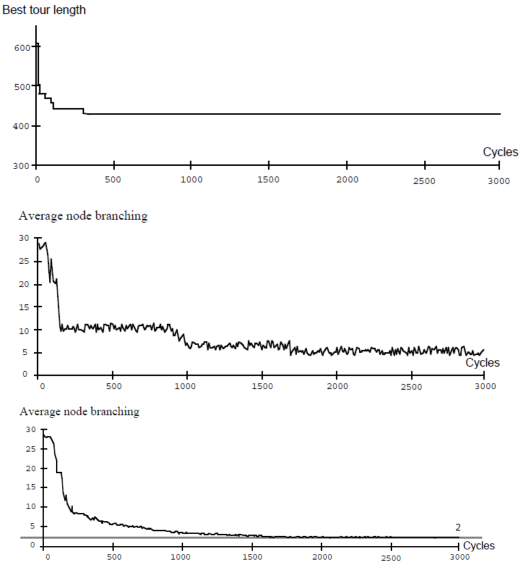
Stopping Criteria

- Stagnation
- Max Iterations
- Desired objective (lower than a threshold)



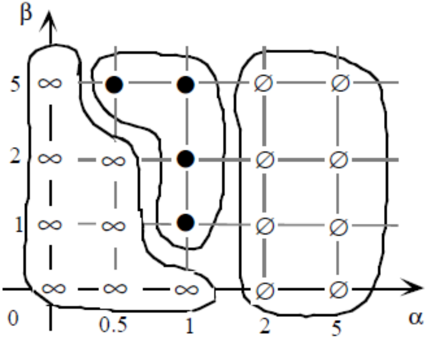


Search Process



Role of Parameters

- Experimentally tuned
- Bad solutions and stagnation-(Φ)
- Bad solutions and no stagnation- (∞)
- Good solutions ($\bullet$)



ACO Applications

- TSP
- Quadratic Assignment Problems (QAP)
- Scheduling
- Vehicle Routing Problems (VRP)
- Telecommunication Networks
- Graph Coloring
- Data Mining (classification)
- etc.

Particle Swarm Optimization

Introduction

Particle swarm optimization (PSO):

- developed by Kennedy & Eberhart,
- first published in 1995,
- exponential increase in the number of publications since then.

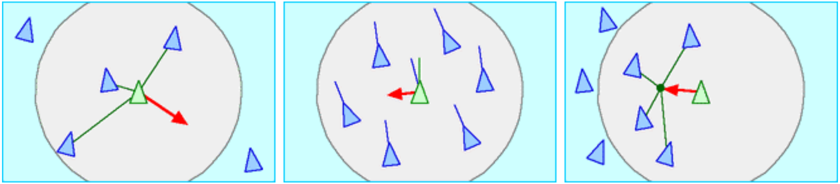
What is PSO?

- a simple, computationally efficient optimization method
- population-based, stochastic search
- based on a social-psychological model of social influence and social learning
- individuals follow a very simple behavior: emulate the success of neighboring individuals
- emergent behavior: discovery of optimal regions in high dimensional search spaces

Introduction: What are the origins of PSO?

- In the work of Reynolds on “boids”: Flocking is an emergent behavior which arises from the interaction of simple rules:
  - Collision avoidance
  - Velocity matching
  - Flock centering
- The work of Heppner and Grenander on using a “roost” as an attractor for a bird flock
- Simplified social model of determining nearest neighbors and velocity matching

Boids: flocks, herds, and schools



Separation: steer to avoid crowding local flockmates

Alignment: steer towards the average heading of local flockmates

Cohesion: steer to move toward the average position of local flockmates

(`http://www.red3d.com/cwr/boids/` ©Craig Reynolds)

Origins of PSO (cont)

- Initial objective: to simulate the graceful, unpredictable choreography of collision-proof birds in a flock
- At each iteration, each individual determines its nearest neighbor and replaces its velocity with that of its neighbor
- Resulted in synchronous movement of the flock
- Random adjustments to velocities prevented individuals to settle too quickly on an unchanging direction
- Adding roosts as attractors:
  - personal best
  - neighborhood best

  $\rightarrow$ particle swarm optimization

Overview of basic PSO

What are the main components?

- A swarm of particles
- Each particle represents a candidate solution
- Elements of a particle represent parameters to be optimized

The search process:

- Position updates

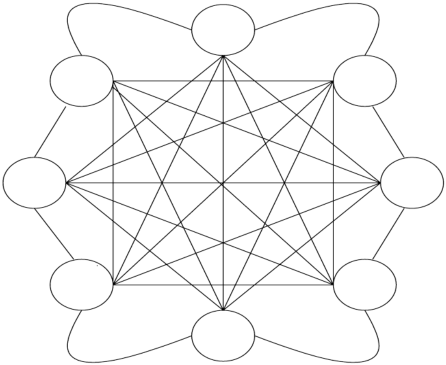
$$x_i(t+1) = x_i(t) + v_i(t+1)$$

  where

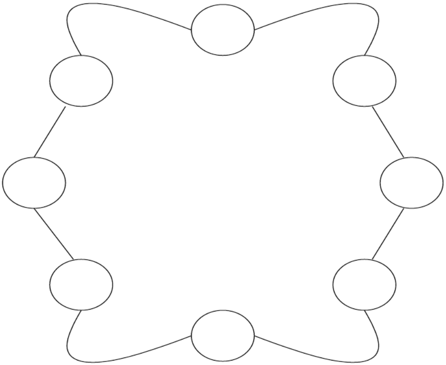
$$x_{ij}(0) \sim U(x_j^{\min}, x_j^{\max})$$

- Velocity
  - drives the optimization process
  - step size
  - reflects experiential knowledge and socially exchanged information

Social interaction based on neighborhoods - topologies



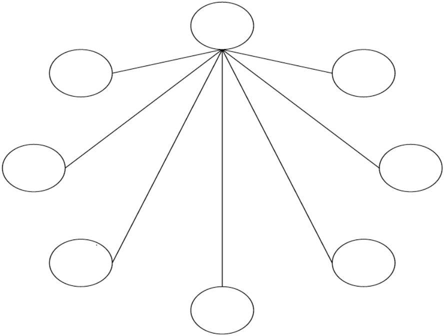
Star



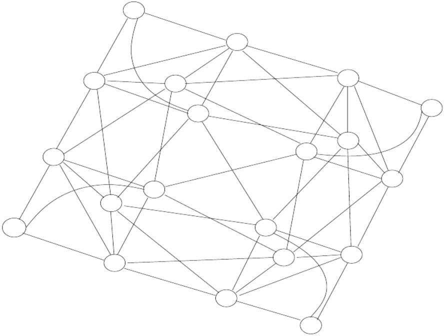
Ring

(A. P. Engelbrecht, Computational Intelligence, ©2007 Wiley)

Social interaction based on neighborhoods - topologies



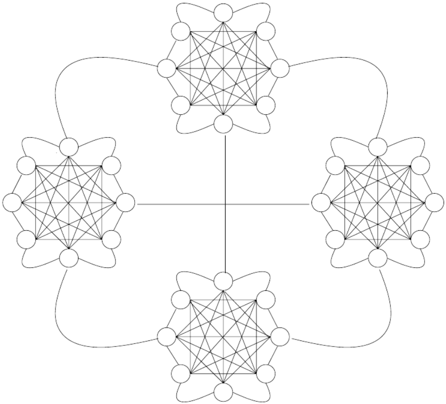
Wheel



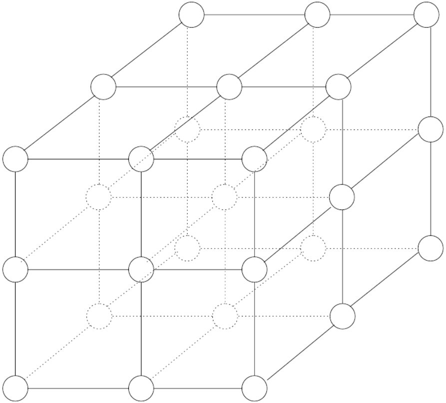
Pyramid

(A. P. Engelbrecht, Computational Intelligence, ©2007 Wiley)

Social interaction based on neighborhoods - topologies



Clusters

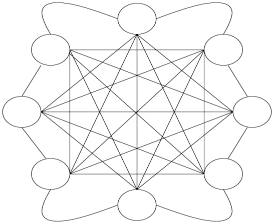


Von Neumann

(A. P. Engelbrecht, Computational Intelligence, ©2007 Wiley)

Global Best (`gbest`) PSO

Uses the **star** social network



Velocity update per dimension

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)]$$

$v_{ij}(0) = 0$ (usually, but can be random)
c_1, c_2 are positive acceleration coefficients
$r_{1j}(t), r_{2j}(t) \sim U(0, 1)$

Global Best PSO (cont)

$\boldsymbol{y}_i$ (t) is the **personal best** position (for minimization):

$$\boldsymbol{y}_i(t+1) = \begin{cases} \boldsymbol{y}_i(t) & \text{if } f\big(\boldsymbol{x}_i(t+1)\big) \geq f\big(\boldsymbol{y}_i(t)\big) \\ \boldsymbol{x}_i(t+1) & \text{if } f\big(\boldsymbol{x}_i(t+1)\big) < f\big(\boldsymbol{y}_i(t)\big) \end{cases}$$

$\hat{\boldsymbol{y}}(t)$ is the **global best** position calculated as

$$\hat{\boldsymbol{y}}(t) \in \{\boldsymbol{y}_0(t), ..., \boldsymbol{y}_{n_s}(t)\} | f(\hat{\boldsymbol{y}}(t)\big) = \min f\big(\boldsymbol{y}_0(t)\big), ..., f\big(\boldsymbol{y}_{n_s}(t)\big)$$

or

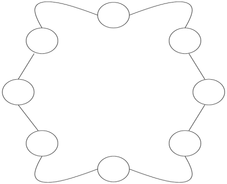
$$\hat{\boldsymbol{y}}(t) \in \{\boldsymbol{x}_0(t), ..., \boldsymbol{x}_{n_s}(t)\} | f(\hat{\boldsymbol{y}}(t)\big) = \min f\big(\boldsymbol{x}_0(t)\big), ..., f\big(\boldsymbol{x}_{n_s}(t)\big)$$

where n_s is the number of particles in the swarm.

gbest PSO Algorithm

Create and initialize an n_x-dimensional swarm S;
while stopping condition is not true **do**
    for each particle $i = \ldots, n_s$ **do**
        if $f(\boldsymbol{x}_i) < f(\boldsymbol{y}_i)$ **then**
            $\boldsymbol{y}_i = \boldsymbol{x}_i$;
        end if
        if $f(\boldsymbol{y}_i) < f(\hat{\boldsymbol{y}})$ **then**
            $\hat{\boldsymbol{y}} = \boldsymbol{y}_i$;
        end if
    end for
    for each particle $i = 1, \ldots, n_s$ **do**
        update the velocity and then the position;
    end for
end while

Local Best (`lbest`) PSO



Uses the **ring** social network

Velocity update per dimension (same as `gbest`)

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] \\ + c_2 r_{2j}(t)[\hat{y}_{ij}(t) - x_{ij}(t)]$$

but $\hat{\boldsymbol{y}}_i$ is now the **neighborhood best**, defined as

$$\hat{\boldsymbol{y}}_i(t+1) \in \{\mathcal{N}_i | f(\hat{\boldsymbol{y}}_i(t+1)) = \min\{f(\boldsymbol{x})\}, \forall \boldsymbol{x} \in \mathcal{N}_i\}$$

with the neighborhood defined as

$$\mathcal{N}_i = \{\boldsymbol{y}_{i-n_{\mathcal{N}_i}}(t), \boldsymbol{y}_{i-n_{\mathcal{N}_i}+1}(t), \ldots, \\ \boldsymbol{y}_{i-1}(t), \boldsymbol{y}_i(t), \boldsymbol{y}_{i+1}(t), \ldots, \boldsymbol{y}_{i+n_{\mathcal{N}_i}}(t)\}$$

where $n_{\mathcal{N}_i}$ is the neighborhood size

Local Best PSO (cont...)

Neighborhoods:

- Neighborhoods are based on particle indices, not spatial information
- Spatial approach is possible, but computationally expensive and not that effective in spreading information (indices allow leaps in search space)
- Neighborhoods overlap to facilitate information exchange

`gbest` PSO vs `lbest` PSO:

- Speed of convergence: larger interconnectivity of `gbest` PSO - it converges faster than the `lbest` PSO (but has less diversity)
- Susceptibility to local minima: larger diversity of `lbest` PSO (i.e. it covers larger parts of the search space) makes it less susceptible to being trapped

Velocity Components

Previous velocity, $\boldsymbol{v}_i(t)$

- inertia component
- memory of previous flight direction
- prevents particle from drastically changing direction

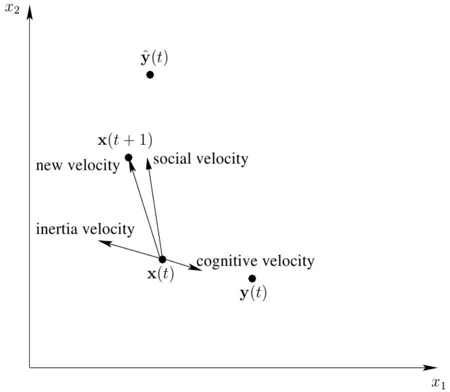
Cognitive component, $c_1 \boldsymbol{r}_1(\boldsymbol{y}_i - \boldsymbol{x}_i)$

- quantifies performance relative to past performances
- memory of previous best position
- a.k.a. nostalgia

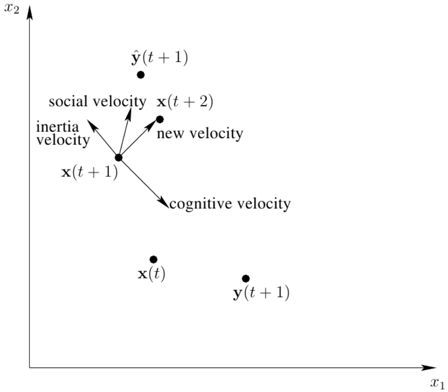
Social component, $c_2 \boldsymbol{r}_2(\hat{\boldsymbol{y}}_i - \boldsymbol{x}_i)$

- quantifies performance relative to neighbors
- a.k.a. envy

Geometric Illustration for a Single Two-Dimensional Particle



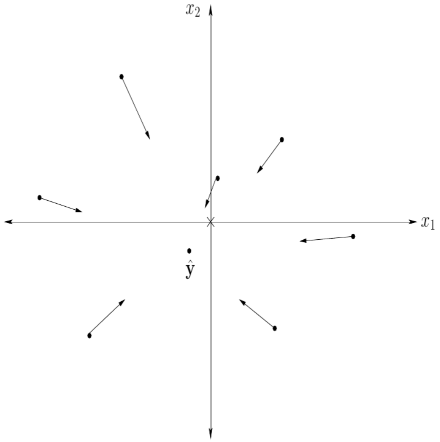
Time step t



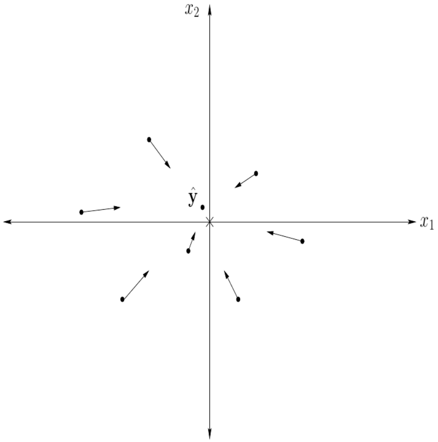
Time step $t + 1$

(A. P. Engelbrecht, Computational Intelligence, ©2007 Wiley)

Cumulative Effect of Position Updates (multi-particle `gbest` PSO)



Time step t



Time step $t + 1$

(A. P. Engelbrecht, Computational Intelligence, ©2007 Wiley)

Stopping Conditions

- Terminate when a maximum number of iterations, or function evaluations (FEs), has been exceeded
- Terminate when an acceptable solution has been found, i.e. when (assuming minimization)

$$f(\boldsymbol{x}_i) \leq |f(\boldsymbol{x}^*) - \epsilon|$$

- Terminate when no improvement is observed over a number of iterations

Stopping Conditions (cont)

- Terminate when the normalized swarm radius is close to zero, i.e.

$$R_{\text{norm}} = \frac{R_{\max}}{\text{diameter}\big(S(0)\big)},$$

  where

$$R_{\max} = ||\boldsymbol{x}_m - \hat{\boldsymbol{y}}||, m = 1, \ldots, n_s,$$

  with

$$||\boldsymbol{x}_m - \hat{\boldsymbol{y}}|| \geq ||\boldsymbol{x}_i - \hat{\boldsymbol{y}}||, \forall i = 1, \ldots, n_s.$$

- Terminate when the objective function slope is approximately zero, i.e.

$$f^{'}(t) = \frac{f(\hat{\boldsymbol{y}}(t)) - f(\hat{\boldsymbol{y}}(t-1))}{f(\hat{\boldsymbol{y}}(t))} \approx 0$$