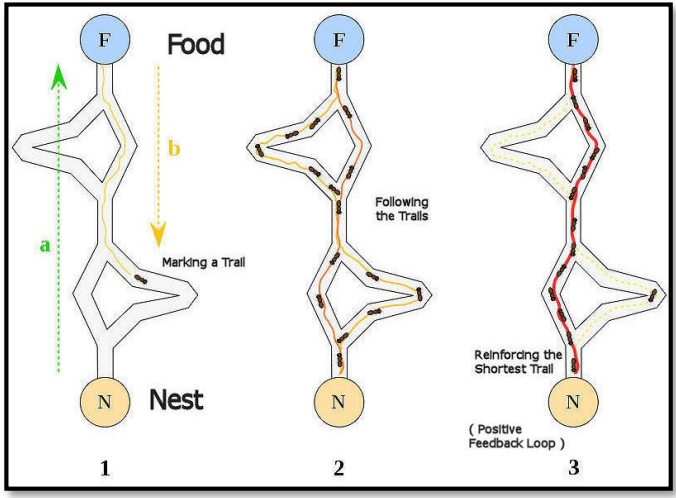
NTNU

Norwegian University of Science and Technology

Lecture 8

Swarm Intelligence



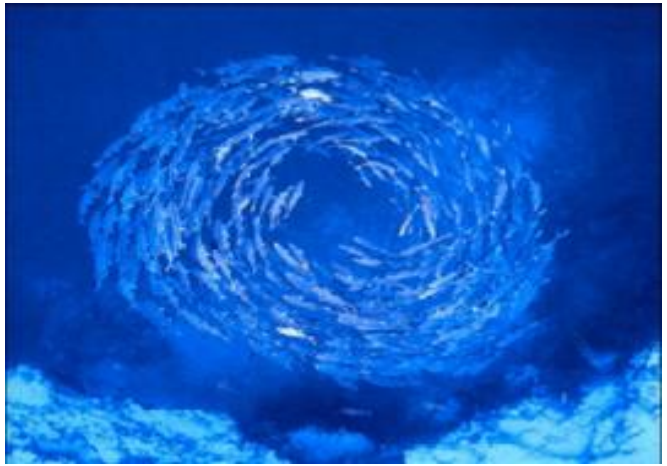
Kazi Shah Nawaz Ripon and Pauline Haddow

Outline

- Introduction to Swarm Intelligence (SI)
- Ant Colony Optimization (ACO)
- Particle Swarm Optimization (PSO)

Outline

- Introduction to Swarm Intelligence
- Ant Colony Optimization (ACO)
- Particle Swarm Optimization (PSO)



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**.

Swarming

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

Swarming is Powerful

- Swarms can achieve things that an individual cannot (**Collective Bahaviour**).

Powerful … but simple

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

Big Question ???

- Do Ants know what they are doing?
  - No.
  - Ants aren't clever little engineers, architects, or warriors.
    - **at least not as individuals.**
- When it comes to deciding what to do next, most ants don't have a clue.

The Big Picture

- Ants aren't smart.
  - Ant colonies are.
- A colony can solve problems unthinkable for individual ants.
- As individuals, ants might be tiny dummies,
  - As colonies they respond quickly and effectively to their environment.
- They do it with something called **Swarm Intelligence.**

Swarm Intelligence

- **Collective system** capable of accomplishing difficult tasks in dynamic and varied environments without any external guidance and with no central coordination.
- Generally made up of agents who interact with each other and the environment.
- Achieving a collective performance which could not normally be achieved by an individual acting alone.
- Particularly suited to distributed problem solving.

Why Insects?

- Insects have a few hundred brain cells.

- However, organized insects have been known for:
  - Architectural marvels.
  - Complex communication systems.
  - Resistance to hazards in nature.

Bees

- Colony cooperation.
- Regulate hive temperature.
- Efficiency via Specialization: division of labour in the colony.
- Communication: Food sources are exploited according to quality and distance from the hive.

Wasps

- Pulp foragers, water foragers & builders.
- Complex nests.
  - Horizontal columns.
  - Protective covering.
  - Central entrance hole.

Termites

- Cone-shaped outer walls and ventilation ducts.
- Brood chambers in central hive.
- Spiral cooling vents.
- Support pillars.

Ants

- Organizing highways to and from their foraging sites by leaving pheromone trails.
- Form chains from their own bodies to create a bridge to pull and hold leafs together with silk.
- Division of labour between major and minor ants.

Social Insects

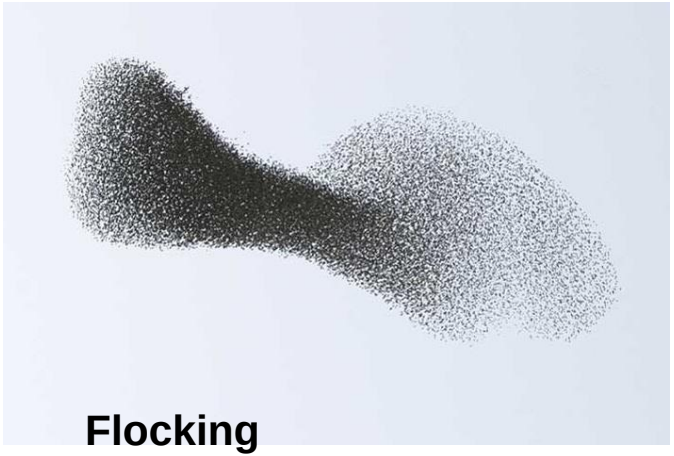
- Flexible.
  - Static, dynamic, online, offline, stationary, time-varying, centralized, distributed, Monolithic, modular, distributed, parallel, adaptive
- Robust.
  - Always finds an optimal solution
  - Deals well with change
- Decentralized.
  - Point-to-point
  - Broadcast like
  - indirect
- Self-organized.
  - The way complex systems/patterns arise out of a many simple interactions.
  - It is often triggered by positive feedback.

Self-Organization

- Self-organizing systems often display **emergent properties.**
  - An emergent behaviour can appear as a result of the interaction of components of the system.
  - E.g. flocking, or organisation of ant colony.
- Real life example of self-organized behaviour in humans
  - Emergence of paths across grassy area.
  - Most popular paths are reinforced.

Emergent Collective Behavior

- Some animal societies display **coordinated and purposeful navigation** of several individuals (from tens to thousands).
- Each individual uses **only local information** about the presence of other individuals and of the environment.
- There is no predefined group leader.

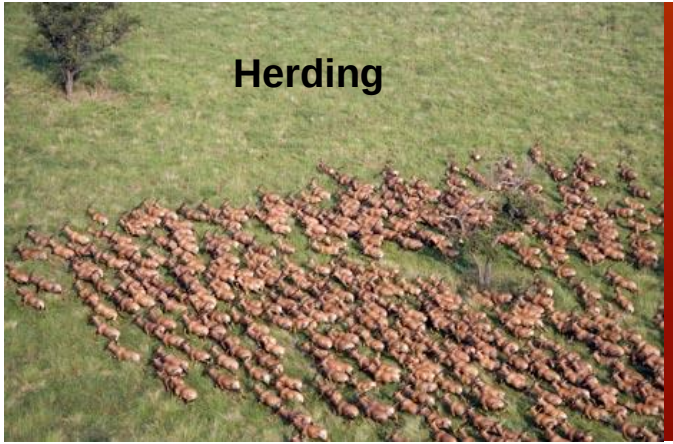


Flocking

Schooling

Emergent Collective Behavior

In some cases there is a leader and more restrictive rules on relative motion, but individuals still use local information to decide how to move.



Emergent Collective Behavior

individual swarm (ex: ant)

- stereotyped,
- unreliable,
- unintelligent,
- simple.

Collective swarm (ex: ant colony)

- flexible,
- reliable,
- intelligent,
- complex level performance

Swarm Intelligence

Main principles:

- Simple individuals (computational or physical) with simple abilities.
- Solve complex problems that a single individual could not solve.
- Robust despite loss of individuals or failure.
- Individuals have:
  - Local sensory information
  - Little/no memory
  - NO global information

Some SI Applications

- U.S. military is investigating SI for controlling unmanned vehicles.

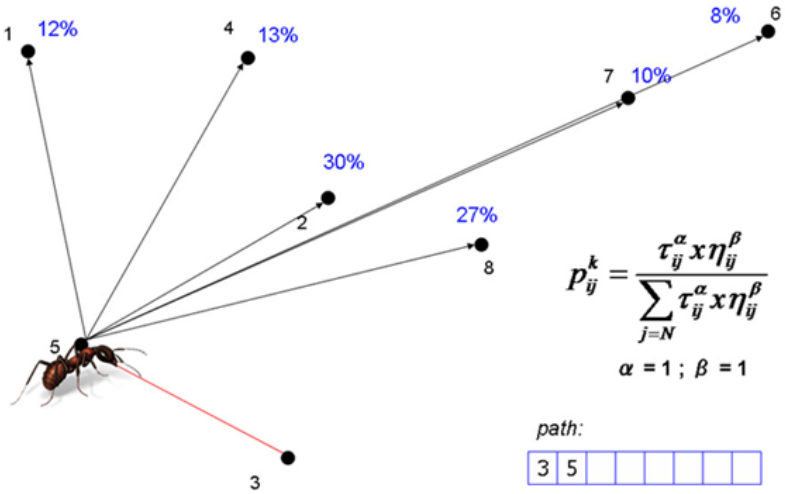
- NASA is investigating the use of SI for planetary mapping.

Application

A swarm robotic demonstration using 278 miniature e-puck robots at the Ecole Polytechnique Federale de Lausanne (EPFL) in Lausanne, Switzerland. Robots in the video are all real, not computer generated.

Outline

- Introduction to Swarm Intelligence
- Ant Colony Optimization (ACO)
- Particle Swarm Optimization (PSO)

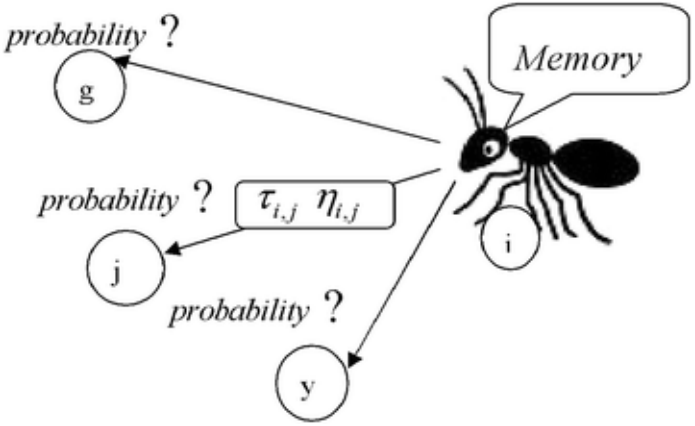


I am lost! Where is the line?!

- A Bug's Life, Walt Disney, 1998

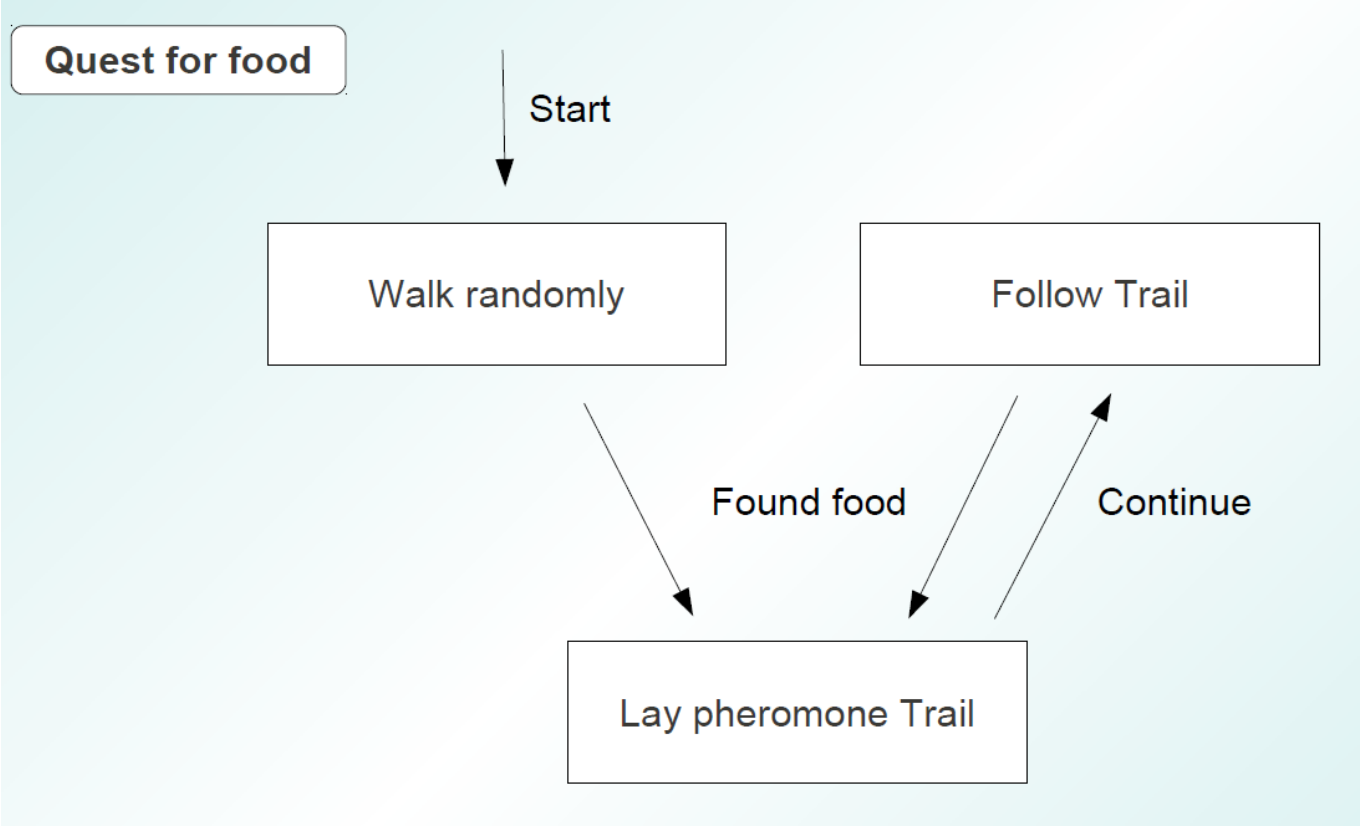
Ant Colony Optimization (ACO)

- Inspired by the behavior of real ants.
- ACO is a **population-based**, general search technique which is inspired by the **pheromone trail** laying behavior of real ant colonies.
- Proposed by Marco Dorigo in 1991.

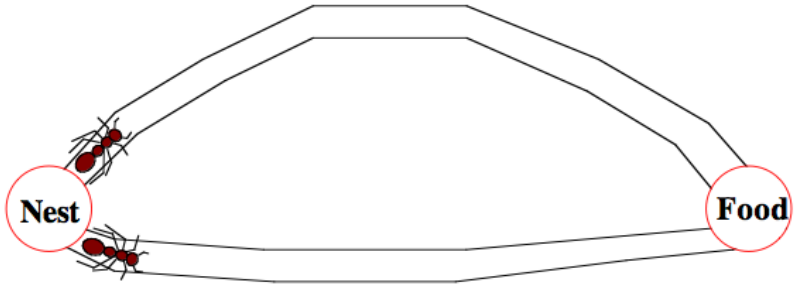


- **Multi-agent approach** for solving difficult combinatorial optimization problems.

Ant Behavior

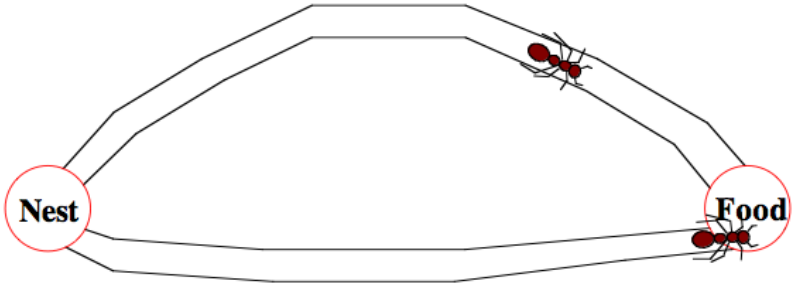


Foraging Behavior of Ants



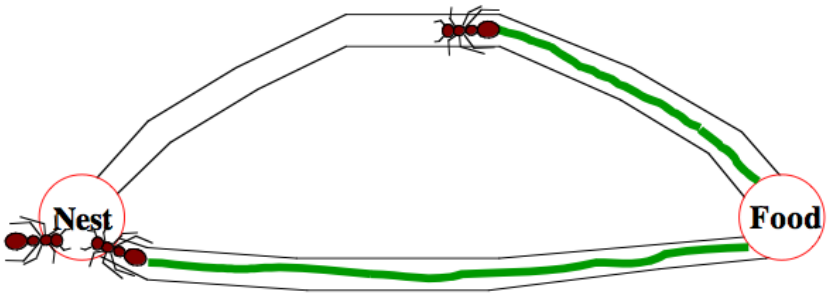
Two ants start with equal probability of going on either path.

Foraging Behavior of Ants



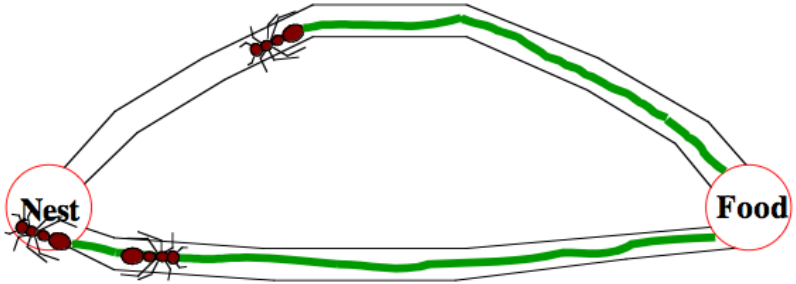
The ant on shorter path has a shorter to-and-fro time from it’s nest to the food.

Foraging Behavior of Ants



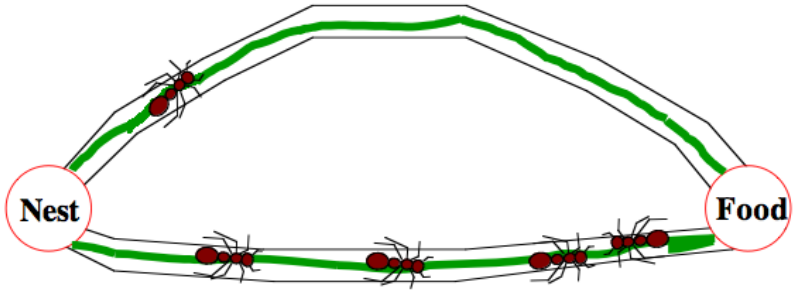
The density of pheromone on the shorter path is higher because of 2 passes by the ant (as compared to 1 by the other).

Foraging Behavior of Ants



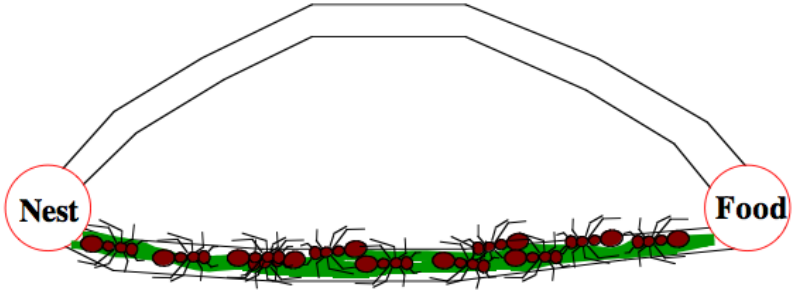
The next ant takes the shorter route.

Foraging Behavior of Ants



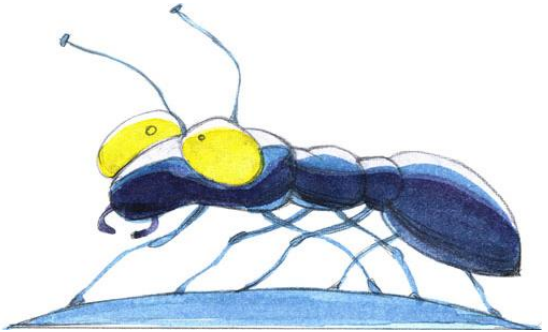
Over many iterations, more ants begin using the path with higher pheromone, thereby further reinforcing it.

Foraging Behavior of Ants



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

The Ants



- Almost blind.
  - Incapable of achieving complex tasks alone.
- Ant behavior is **stochastic.**
- The behavior is induced by **indirect communication.**
  - Use ***stigmergic*** communication via pheromone trails.
- Limited ability to sense local environment.
- Act concurrently and independently**.**
  - High quality solutions emerge via **global cooperation.**

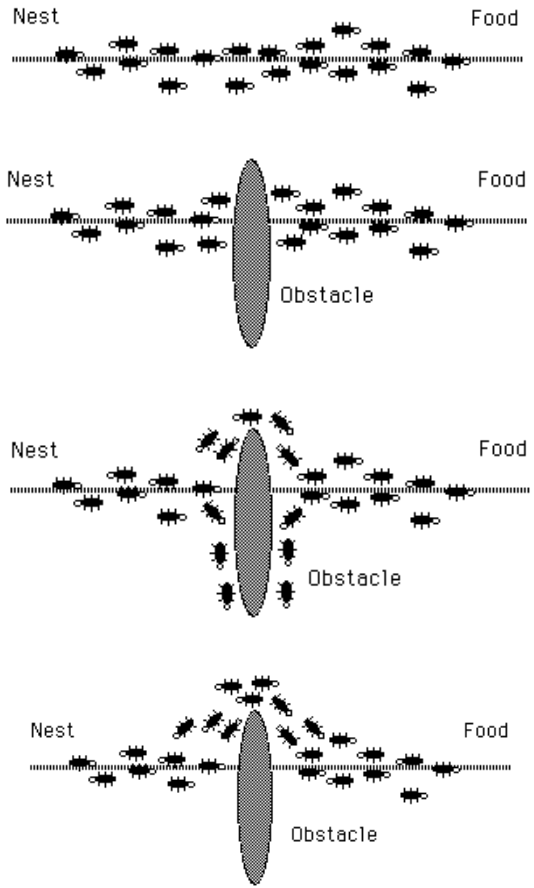
Stigmergy

The term indicates communication among individuals through modification of the environment.

Two individuals interact **indirectly** when one of them modifies the environment and the other responds to the new environment at a later time.

For example, some ants leave a chemical (pheromone) trail behind to trace the path. *The chemical decays over time.*

This allows other ants to find the path between the food and the nest. *It also allows ants to find the shortest path among alternative paths*.



Stigmergy in Humans

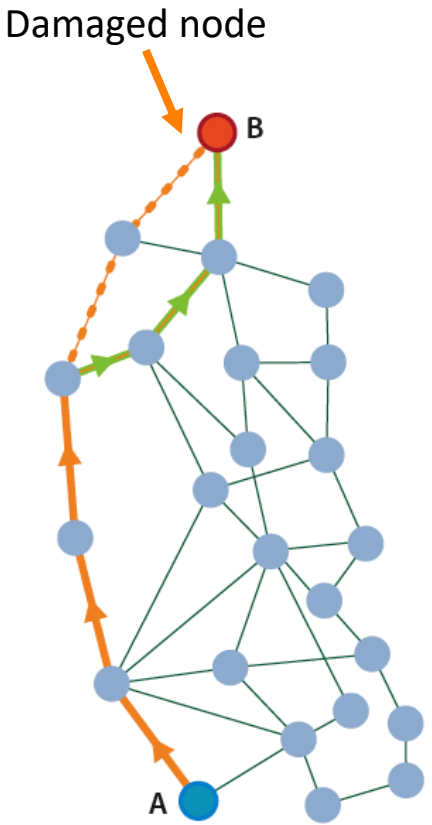
stigma (mark, sign) +
ergon (work, action)

Pierre-Paul Grassé
(1959)

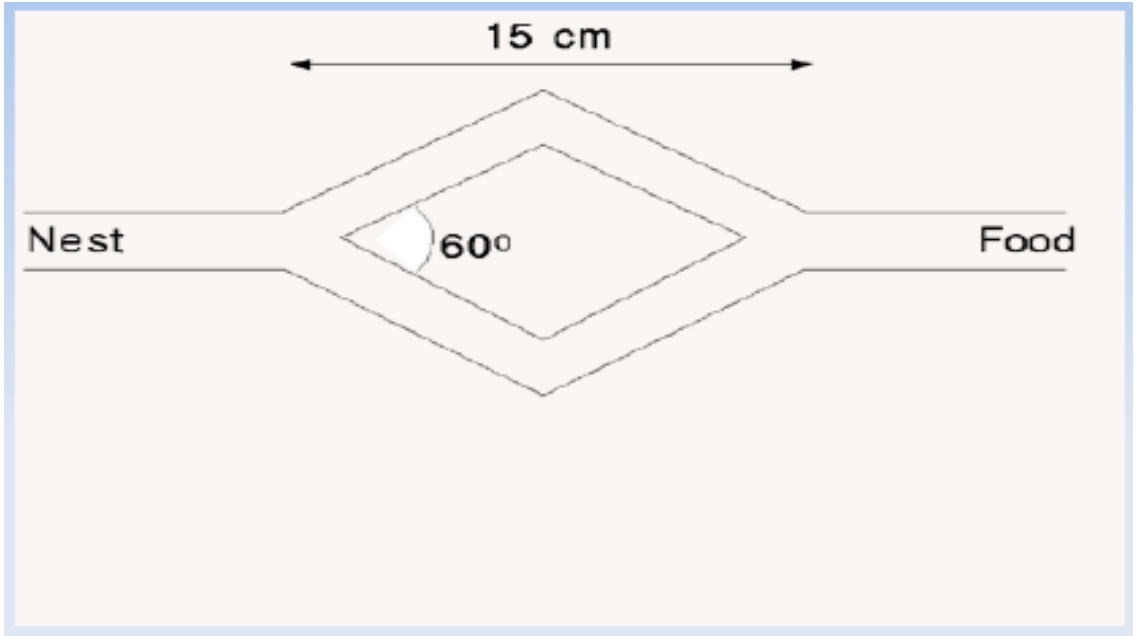
Ant Colony Optimization

- Typical examples are telephone, internet, and any problem that can be described as TSP.
  - Used/adopted by British Telecom, MCI Worldcom, Barilla, etc.

- Advantage of algorithm is that, as ants do, it allows **dynamic rerouting through shortest path** if one node is broken.
  - Most other algorithms instead assume that the network is static.

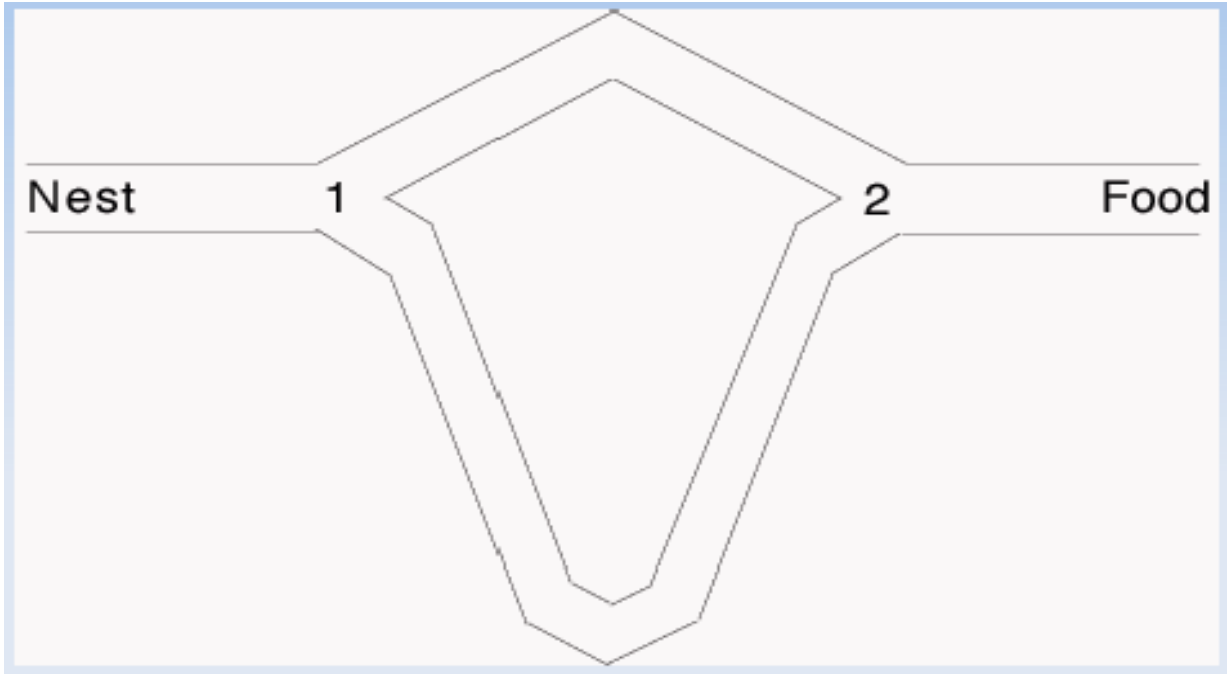


Double Bridge Experiments *(Deneubourg et al.)*



Branches have equal length.

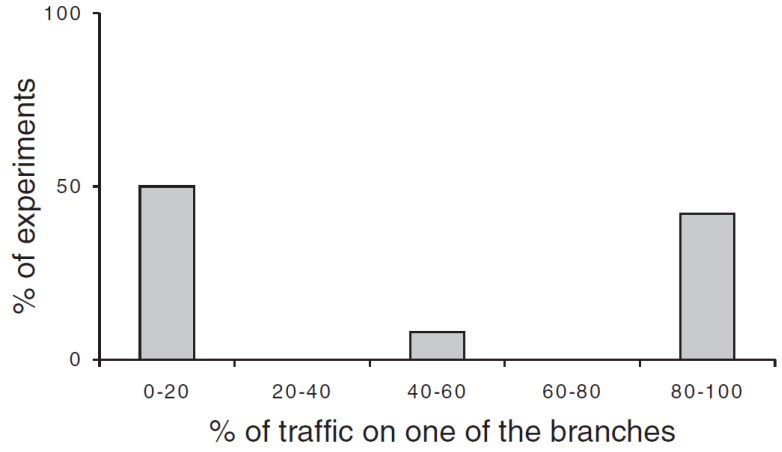
Double Bridge Experiments *(variant)*



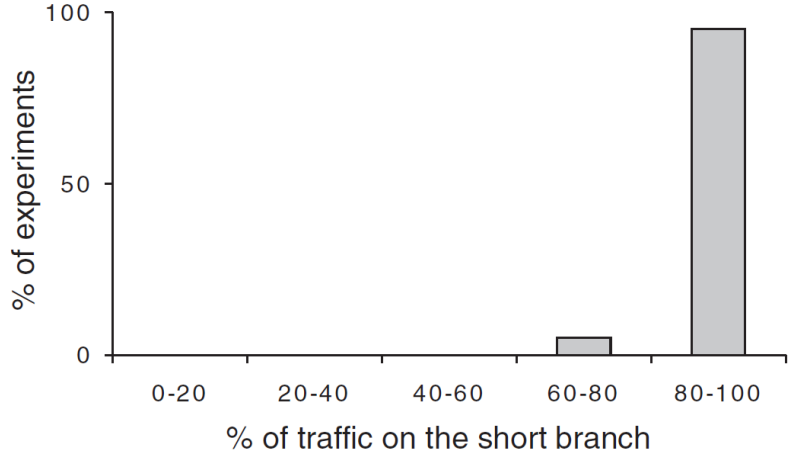
Branches have different length.

Experimental Results

- **Equal length bridges**
  - convergence to a single path (by random).
  - each path is selected 50% of the experiments.
- **Different length paths**
  - convergence to short path (*autocatalytic* or positive feedback).

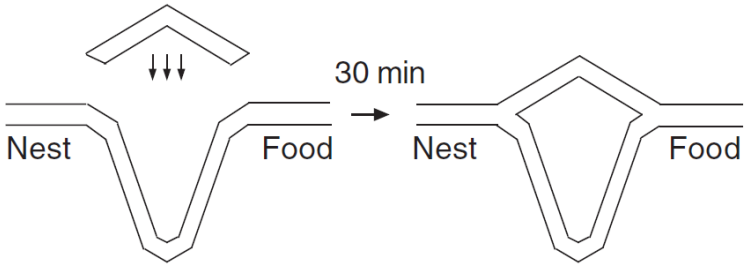


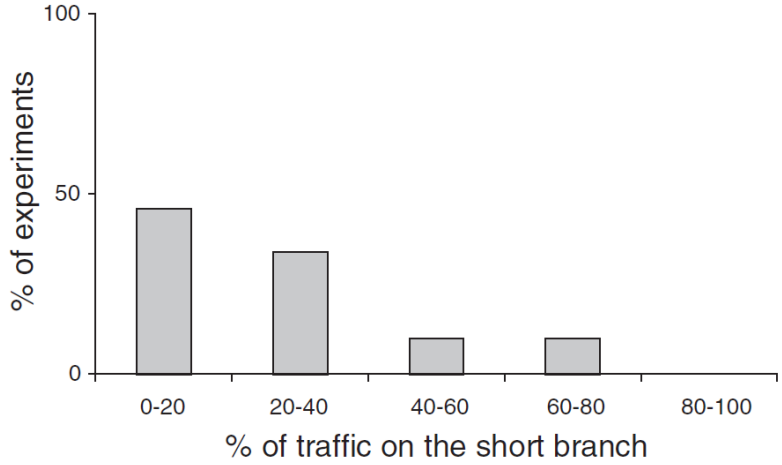
(a) Equal length bridges



(a) Shorter length bridges

High Concentration and Slow Evaporation of Pheromone





- Intitially only the long branch.
- After 30 minutes additional short branch.
- The short branch was selected sporadically.
- The colony was trapped on the long branch.

From Biological Ants to Artificial Ants

Problem:

- Adaptation to reality.

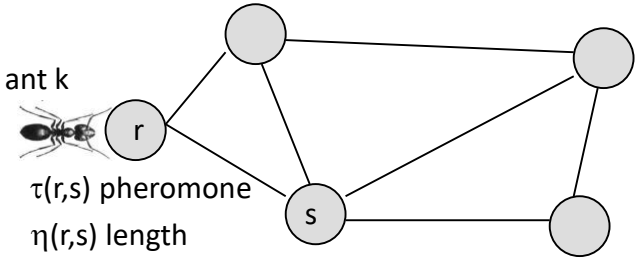
Solution:

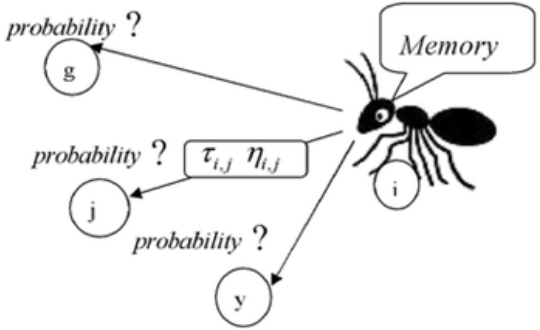
- Pheromone upgrade: **evaporation**.
- Ant aging: after a given time, ants are tired and have to come back to the nest.
- 2 different pheromones : away (from nest) and back (from source of food).

The ACO Meta-Heuristic

- Procedure ACO_Metheuristic
  - initialize parameters and pheromone trails
  - **while** termination condition not met **do**
    - generate ant solutions ()

generate ant solutions ()





- Each ant generates a complete tour of nodes using **probabilistic transition rule** encouraging choice of edge with
  - high pheromone, and
  - short distance.

The ACO Meta-Heuristic

- Procedure ACO_Metheuristic
  - initialize parameters and pheromone trails
  - **while** termination condition not met **do**
    - generate ant solutions ()
    - daemon activities() ((local search (optional))
      - optional and strictly implementation-dependent, which cannot be performed by single ants.

The ACO Meta-Heuristic

- Procedure ACO_Metheuristic
  - initialize parameters and pheromone trails
  - **while** termination condition not met **do**
    - generate ant solutions ()
    - daemon activities() ((local search (optional))
    - update pheromone()

update pheromone()

- The aim of the pheromone update is
    1. to increase the pheromone values associated with good solutions, and
    2. to decrease those that are associated with bad ones.

- Usually, this is achieved
    1. by decreasing all the pheromone values through pheromone evaporation, and
    2. by increasing the pheromone levels associated with a chosen set of good solutions.

The ACO Meta-Heuristic

- Procedure ACO_Metheuristic
  - initialize parameters and pheromone trails
  - **while** termination condition not met **do**
    - generate ant solutions ()
    - daemon activities() ((local search (optional))
    - update pheromone()
  - **end-while**
- End procedure

Local / Global Pheromone Update

Initialize

Loop

Each ant is positioned on a starting node.

For each ant

Loop

Each ant applies a **state transition rule** to incrementally build a solution.

A **local pheromone updating rule** (after complete path found by one ant).

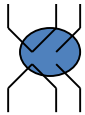
Until *all ants have built a complete solution.*

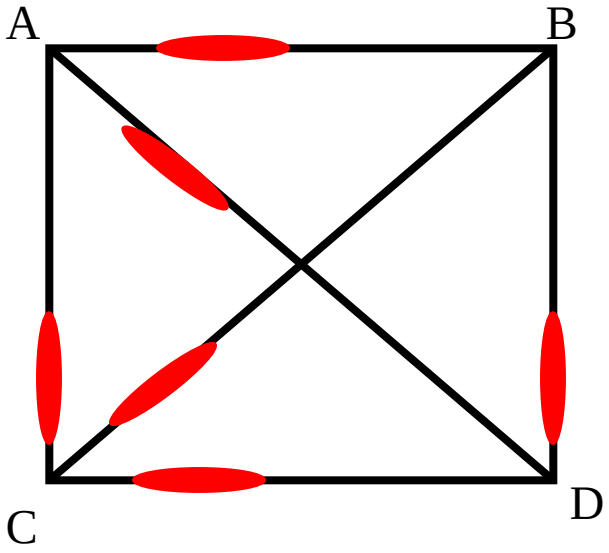
A **global pheromone updating rule** (after complete path found by all ants).

Until *End_condition*.

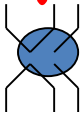
Example: A 4-city TSP

Initially, random levels of pheromone are scattered on the edges





Pheromone

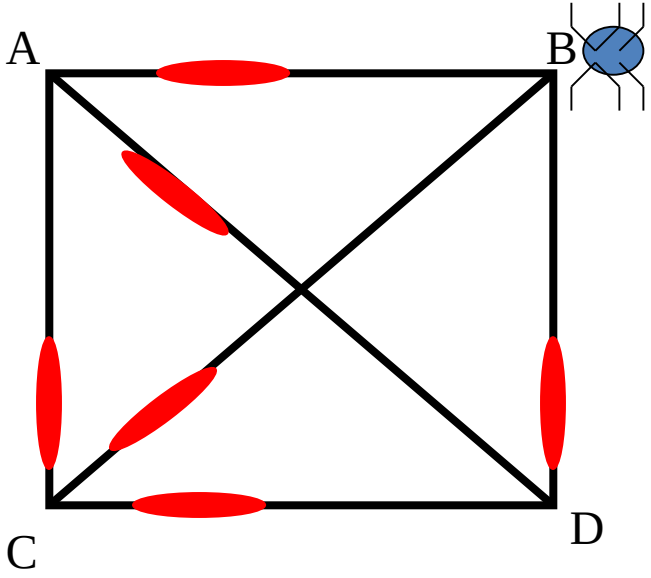


Ant

AB: 10, AC: 10, AD, 30, BC, 40, CD 20

Example: A 4-city TSP

An ant is placed at a random node



Pheromone

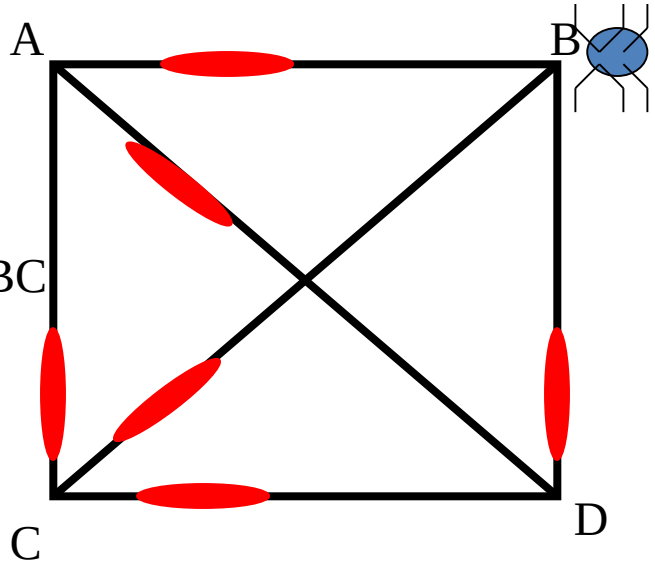
Ant

AB: 10, AC: 10, AD, 30, BC, 40, CD 20

Example: A 4-city TSP

The ant decides where to go from that node,
based on probabilities
calculated from:
- pheromone strengths,
- next-hop distances.

Suppose this one chooses BC



Pheromone

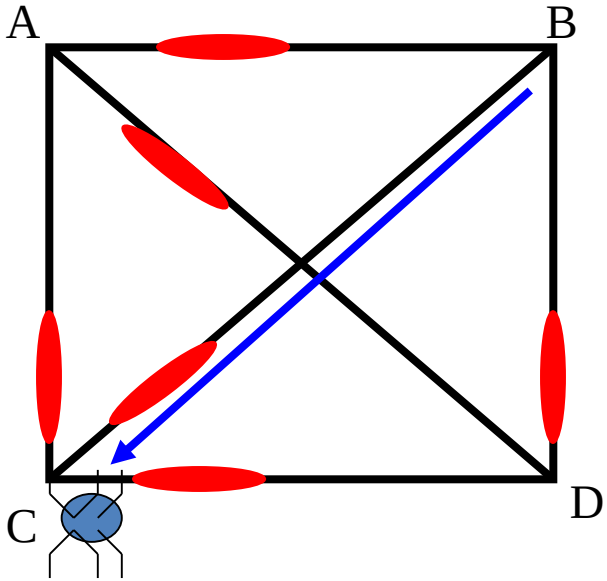
Ant

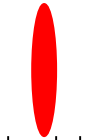
AB: 10, AC: 10, AD, 30, BC, 40, CD 20

Example: A 4-city TSP

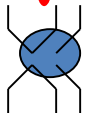
The ant is now at C, and has a `tour memory' = {B, C} – so he cannot visit B or C again.

Again, he decides next hop (from those allowed) based on pheromone strength and distance;
suppose he chooses CD





Pheromone

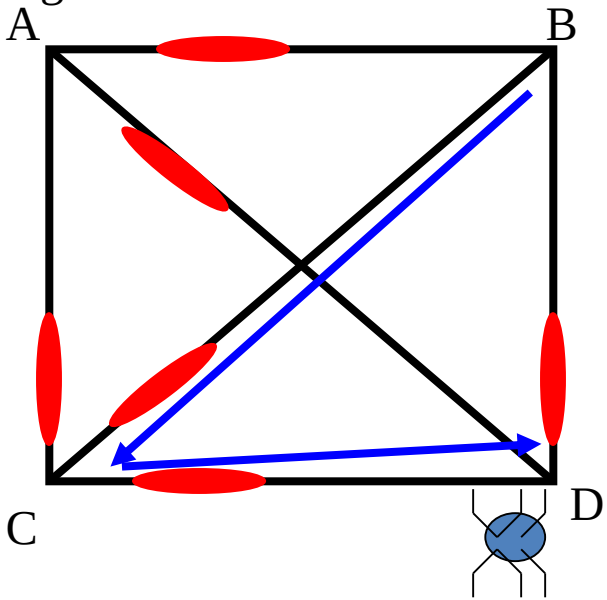


Ant

AB: 10, AC: 10, AD, 30, BC, 40, CD 20

Example: A 4-city TSP

The ant is now at D, and has a `tour memory' = {B, C, D}
There is only one place he can go now:



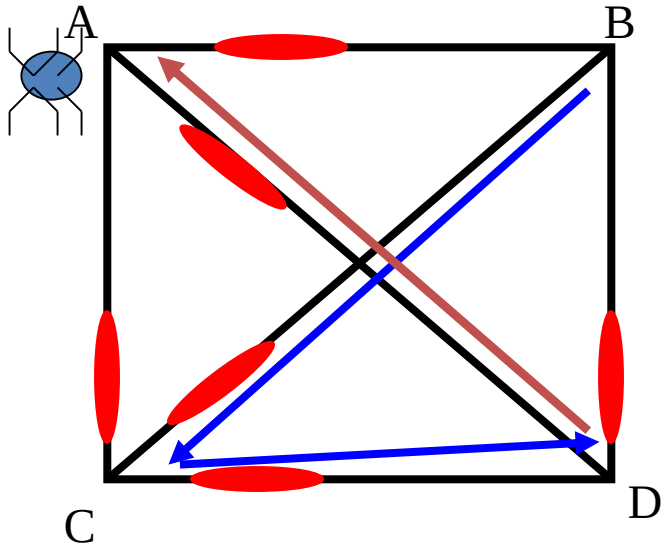
Pheromone

Ant

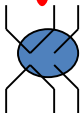
AB: 10, AC: 10, AD, 30, BC, 40, CD 20

Example: A 4-city TSP

So, he has nearly finished his tour, having gone over the links: BC, CD, and DA.



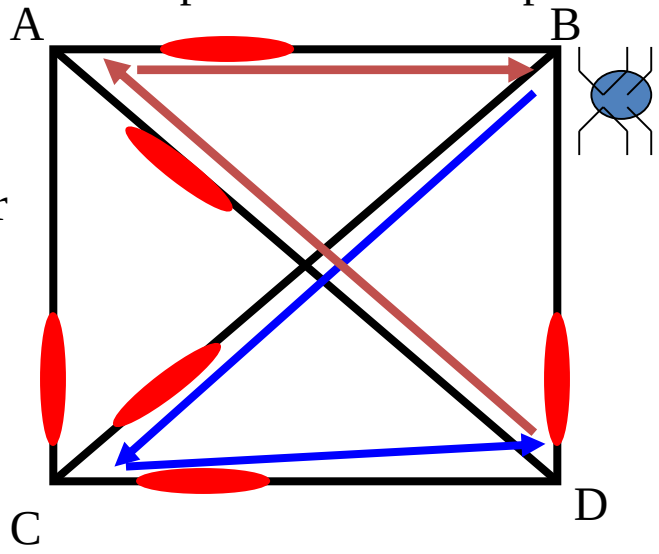

Pheromone


Ant

AB: 10, AC: 10, AD, 30, BC, 40, CD 20

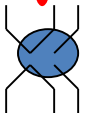
Example: A 4-city TSP

So, he has nearly finished his tour, having gone over the links: BC, CD, and DA. AB is added to complete the round trip.



Now, pheromone on the tour is increased, in line with the fitness of that tour.

Pheromone

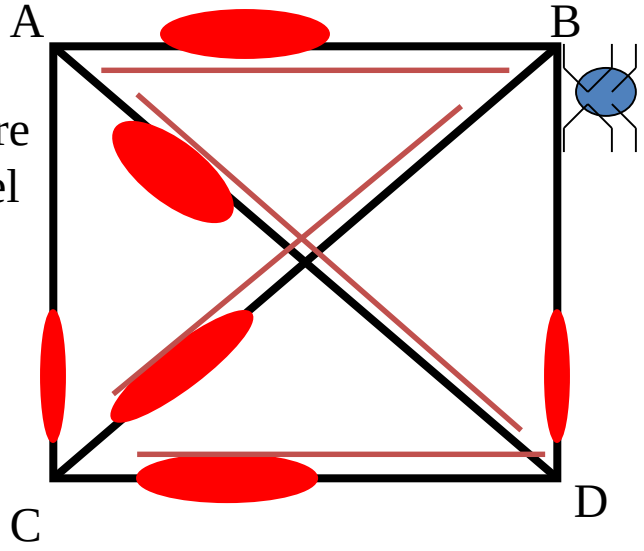


Ant

AB: 10, AC: 10, AD, 30, BC, 40, CD 20

E.g. A 4-city TSP

Next, pheromone everywhere is decreased a little, to model decay of trail strength over time



Pheromone

Ant

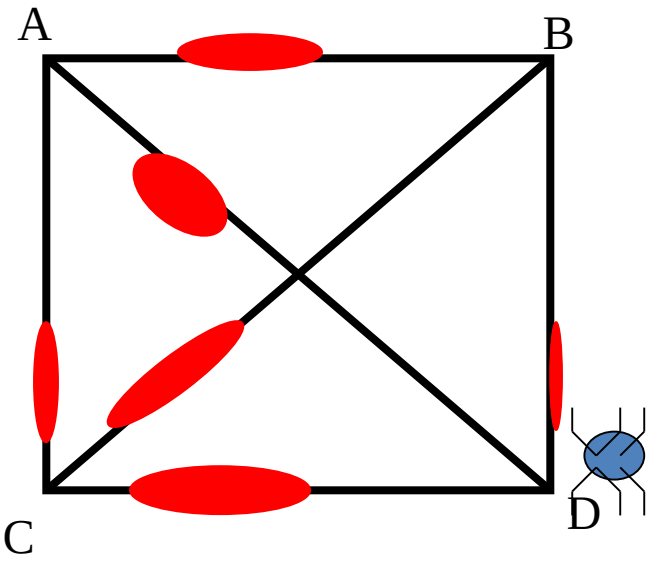
AB: 10, AC: 10, AD, 30, BC, 40, CD 20

Example: A 4-city TSP

We start again, with another ant in a random position.

Where will he go?

Next , the actual algorithm and variants.



Pheromone

Ant

AB: 10, AC: 10, AD, 30, BC, 40, CD 20

Variants of ACO

ALGORITHM	AUTHORS	YEAR
Ant System (AS)	Dorigo et al.	1991
Elitist AS	Dorigo et al.	1992
ANT-Q	Gambardella & Dorigo	1995
Ant Colony System (ACS)	Dorigo & Gambardella	1996
MAx-Min AS	Stutzle & Hoos	1996
Rank-Based AS	Bullnheimer et al.	1997
ANTS	Maniezzo	1999
BWAS	Cordon et al.	2000
Hyper-Cube AS	Blum et al.	2001

Parameters of ACO Algorithm

τ_{ij} : Pheromone trail of combination (i, j)

η_{ij} : Local heuristic of combination (i, j)

P_{ij} : Transition probability of combination (i, j)

α : Relative importance of pheromone trail

β : Relative importance of local heuristic

q_0 : Determines the relative importance of exploitation versus exploration

ρ : Trail persistence

Variants of ACO Algorithms

Ant System (AS) – the earliest version of ACO

State Transition Probability $P_{ij} = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{l \in U}(\tau_{il})^{\alpha}(\eta_{il})^{\beta}}$

Pheromone Update Rule $\tau_{ij}^{new} = \rho\tau_{ij}^{old} + \sum_{k=1}^{NA}\Delta\tau_{ij}^{k}$

$$\Delta\tau_{ij}^{k} = Q, \quad \Delta\tau_{ij}^{k} = \frac{Q}{d_{ij}} \quad or \quad \Delta\tau_{ij}^{k} = \frac{Q}{L^{k}}$$

Variants of ACO Algorithms

AS_{elite}

$$\tau_{ij}^{new} = \rho\tau_{ij}^{old} + \sum_{k=1}^{NA}\Delta\tau_{ij}^{k} + \Delta\tau_{ij}^{e} \qquad \Delta\tau_{ij}^{e} = e \cdot \frac{Q}{L^{e}}$$

AS_{rank}

$$\tau_{ij}^{new} = \rho\tau_{ij}^{old} + \sum_{r=1}^{e-1}\Delta\tau_{ij}^{r} + \Delta\tau_{ij}^{e} \qquad \Delta\tau_{ij}^{r} = (e-r) \cdot \frac{Q}{L^{r}}$$

Variants of ACO Algorithms

Ant-Q & Ant Colony System (ACS)

$$v = \begin{cases} \arg \max_{l \in U} [(\tau_{il})^{\alpha} (\eta_{il})^{\beta}] & q \le q_0 \\ V & q > q_0 \end{cases}$$

Exploitation

Exploration

$$P_{iv} = \frac{(\tau_{iv})^{\alpha} (\eta_{iv})^{\beta}}{\sum_{l \in U} (\tau_{il})^{\alpha} (\eta_{il})^{\beta}}$$

Local Updating (Online Updating) $\tau_{ij}^{new} = \rho \tau_{ij}^{old} + (1-\rho)\Delta\tau_0$

Global Updating (Offline Updating) $\tau_{ij}^{new} = \rho \tau_{ij}^{old} + (1-\rho)\Delta\tau_{ij}^{e}$

Variants of ACO Algorithms

Max-Min Ant System (MMAS)

$$\tau_{ij}^{new} = \rho\tau_{ij}^{old} + \Delta\tau_{ij}^{e} \qquad \tau_{\min} \le \tau_{ij} \le \tau_{\max}$$

ANTS

$$P_{ij} = \frac{\alpha \cdot \tau_{ij} + (1-\alpha)\cdot \eta_{ij}}{\sum_{l \in U}[\alpha \cdot \tau_{il} + (1-\alpha)\cdot \eta_{il}]}$$

$$\tau_{ij}^{new} = \tau_{ij}^{old} + \sum_{k=1}^{NA} \Delta\tau_{ij}^{k} \qquad \Delta\tau_{ij}^{k} = \tau_0 \cdot (1 - \frac{L^k - LB}{L_{avg} - LB})$$

Applications

- TSP
- QAP
- VRP
- Telecommunication Network
- Scheduling
- Graph Coloring
- Water Distribution Network
- etc

Some Inherent 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.

Applications to Industrial Problems

- The success on academic problems has raised the attention of a number of companies that have started to use ACO algorithms for real-world applications.

- EuroBios (www.eurobios.com).

- AntOptima (www.antoptima.com).

Current Research

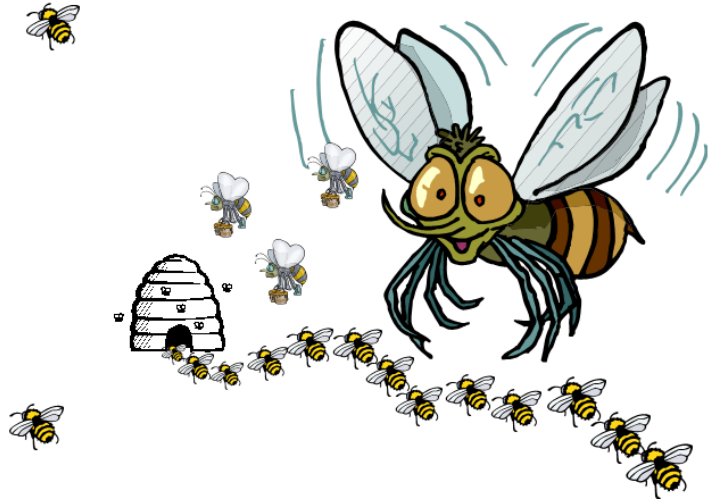
- Dynamic optimization problems.
- Parallel implementation.
- Multi-objective optimization.

NTNU

The Future?

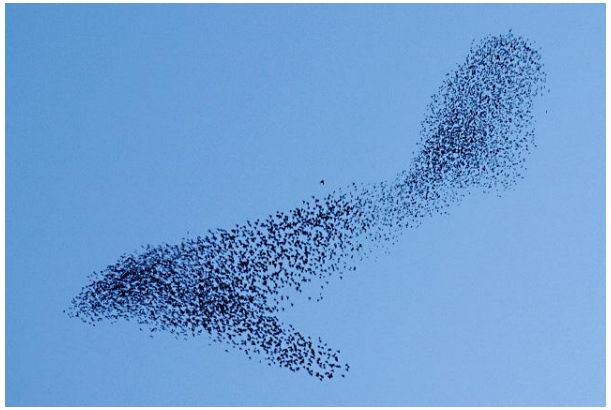
Outline

- Introduction to Swarm Intelligence
- Ant Colony Optimization (ACO)
- Particle Swarm Optimization (PSO)



Particle Swarm Optimization (PSO)

- Russ Eberhart (engineering Prof) and James Kennedy (social scientist) in 1995.
- Inspired by social behavior of bird flocking or fish schooling.
- PSO applies the concept of social interaction to problem solving.
- Individuals interact with one another while learning from their own experience, and gradually move towards the goal.
- Finds a global optimum.



Motivation: Coordinated Navigation

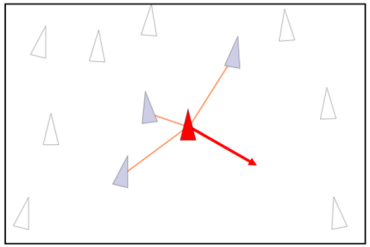
Motivation

- Inspired by simulation social behavior.
- It **combines local search methods with global search methods**, attempting to **balance exploration and exploitation**.
- It is easily implemented and has proven both very effective and quick when applied to a diverse set of optimization problems.

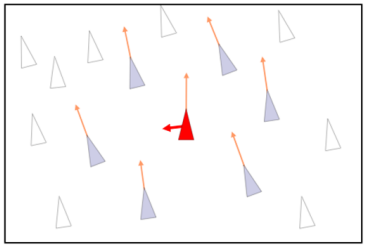
Simple Creatures Following Simple Rules

Only three simple rules (Example: Birds flocking)

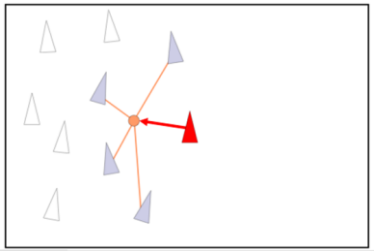
1. Avoid collision with neighboring birds.



2. Match the velocity of neighboring birds.

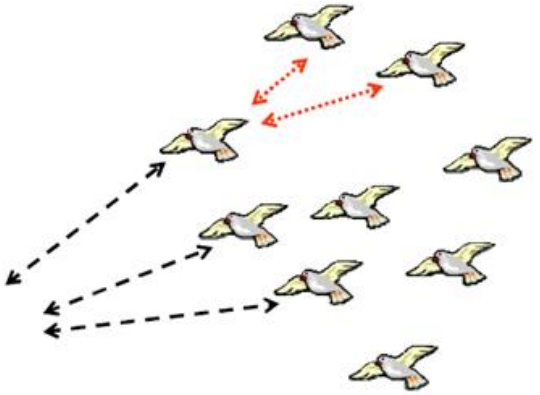


3. Steer toward the Center
   - Stay near neighboring birds

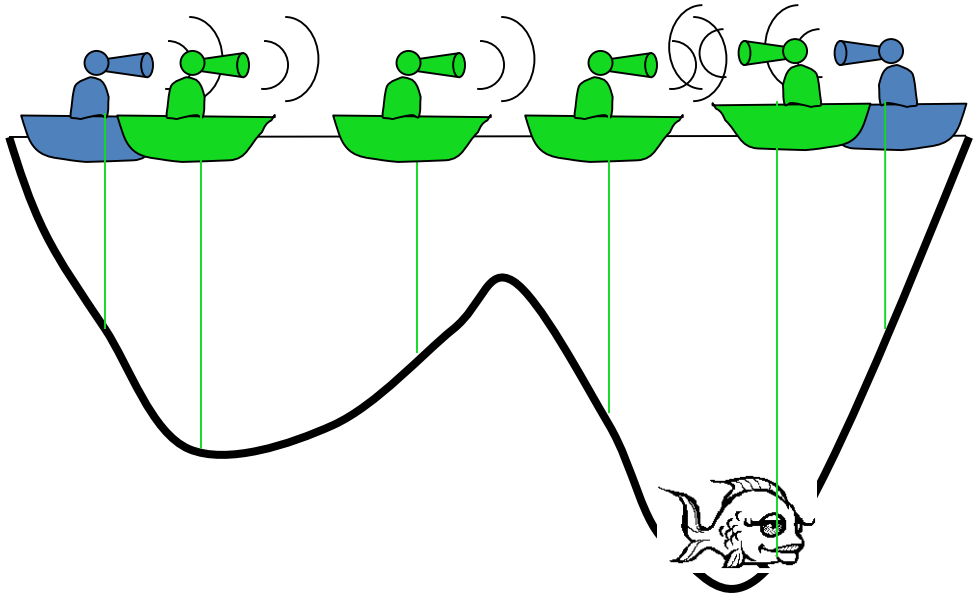


Concept

- Particles fly around in a **multidimensional** search space.
- During flight, each particle adjusts its position according to **its own experience**, and according to the **experience of a neighboring particle**,
  - making use of the best position encountered by itself and its neighbor.
- Bird flocking is one of the best example of PSO in nature.



Cooperation example



Cooperation example

- Of course, this example is a caricatural one, but it presents the main features of a particle in basic PSO:
  - A **position**,
  - A **velocity** (or, more precisely an operator which can be applied to a position in order to modify it),
  - The ability to **exchange information** with its neighbors,
  - The ability to **memorize a previous position**, and
  - The ability to **use information** to make a decision.

- Let’s now see more precisely these points.

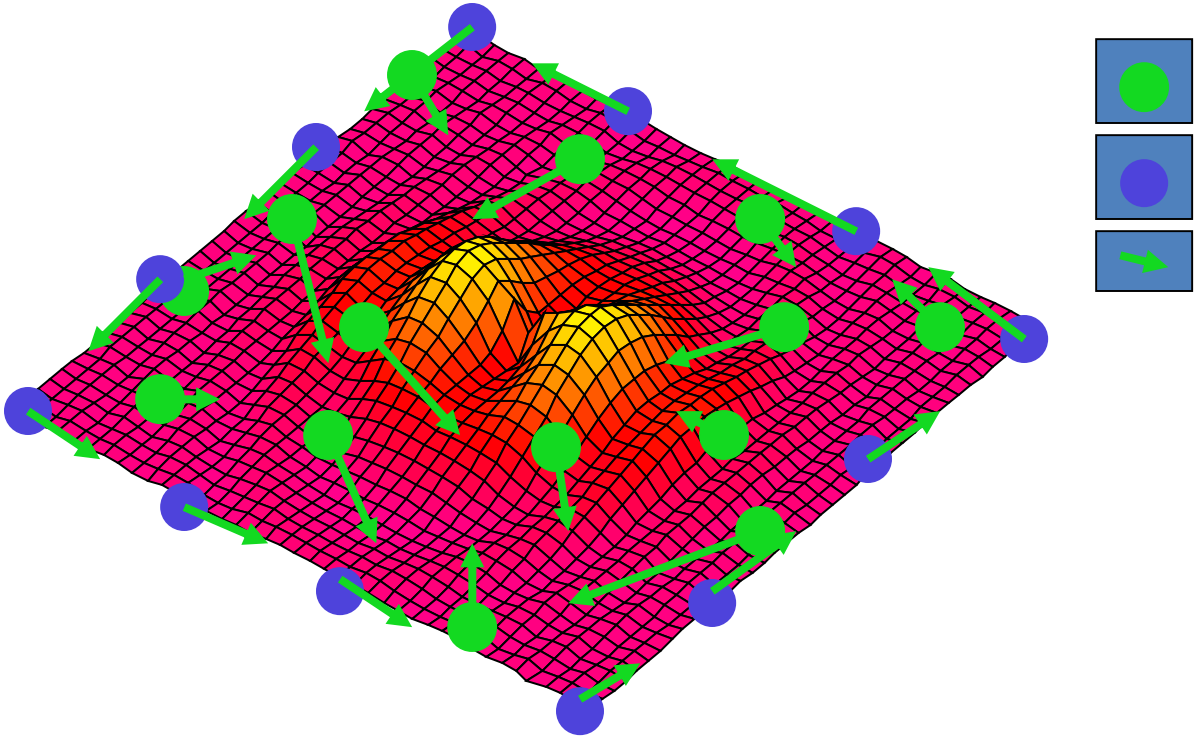
The Basic Idea I

- Each particle (or agent) is searching for the optimum.
- Each particle is **moving** and hence has a **velocity**.
- Each particle **remembers** the position it was in where it had its best result so far (its personal best - ***pbest***).
- *But this would not be much good on its own; particles need help in figuring out where to search*.
    - Each particle know the globally best position that one member of the flock had found, and its value (global best - ***gbest***).

The basic idea II

- The particles ***co-operate***.
  - They exchange information about what they've discovered in the places they have visited.
- The co-operation is very simple. In basic PSO it is like this:
  - A particle has a ***neighborhood*** associated with it.
  - A particle knows the ***fitnesses*** of those in its neighborhood, and uses the ***position*** of the one with ***global best fitness***.
  - This position is simply used to adjust the particle's velocity.

Initialization: Positions and Velocities



What A Particle Does

- In each timestep, a particle has to move to a new position.
- It does this by adjusting its *velocity*. *Having worked out a new velocity, its position is simply its old position plus the new velocity.*

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

- The adjustment is essentially of velocity is like this:
  - (i) The current velocity +
  - (ii) A weighted random portion in the direction of its personal best +
  - (iii) A weighted random portion in the direction of the neighborhood best.

$$v_i(t+1) = v_i(t) + c_1 * rand() * (\text{pbest}x_i - x_i(t)) + c_2 * rand() * (\text{gbest}x - x_i(t))$$
$$\text{where } 0 < rand() < 1$$

Algorithm – Phase 1 (1D)

- Using the co-ordinates of ***pbest*** and ***gbest***, each agent calculates its new velocity as:

$$v_i(t+1) = v_i(t) + c_1 * rand() * (\text{pbest}x_i - x_i(t)) + c_2 * rand() * (\text{gbest}x - x_i(t))$$

inertia

Social / global influence

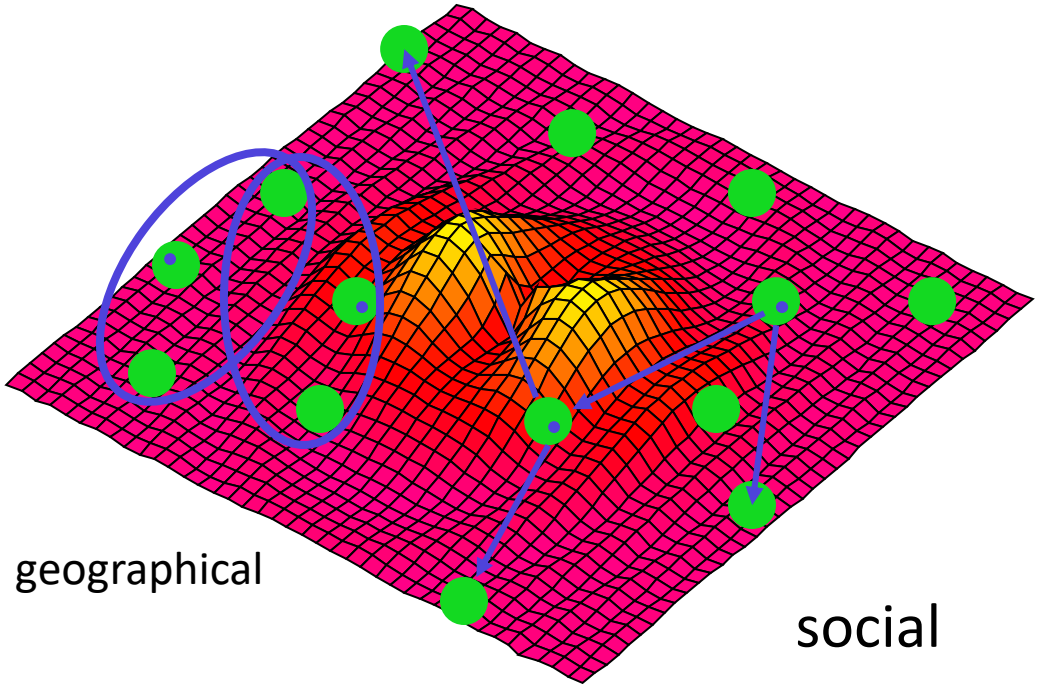
Personal / cognitive influence

where $0 < rand() < 1$

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

Acceleration constants c_1 and c_2, are real-valued and usually in the range $0 \leq (c_1 + c_2) \leq 4$

Neighbourhoods



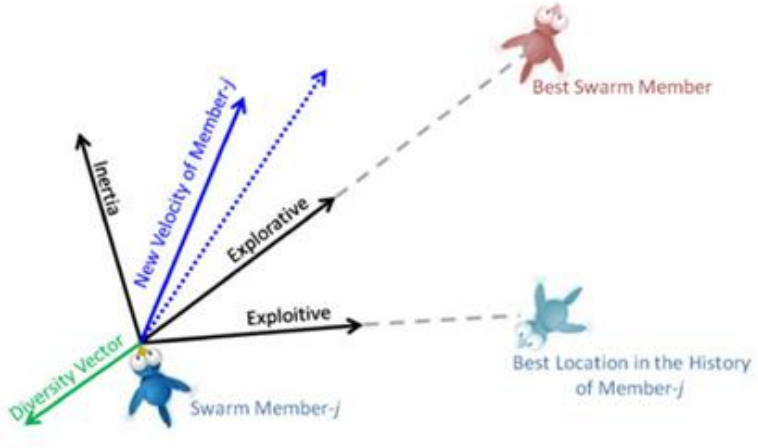
Neighbourhoods: *Size*

- Size of the neighborhood could be a problem.
- Fortunately, PSO is not very sensitive to this parameter and most of users just take a value of 3 or 5 with good results.
- Unlike for the swarm size, there is no mathematical formula,
  - but like for the swarm size, there are some adaptive variants.

Pseudocode

Equation (a)

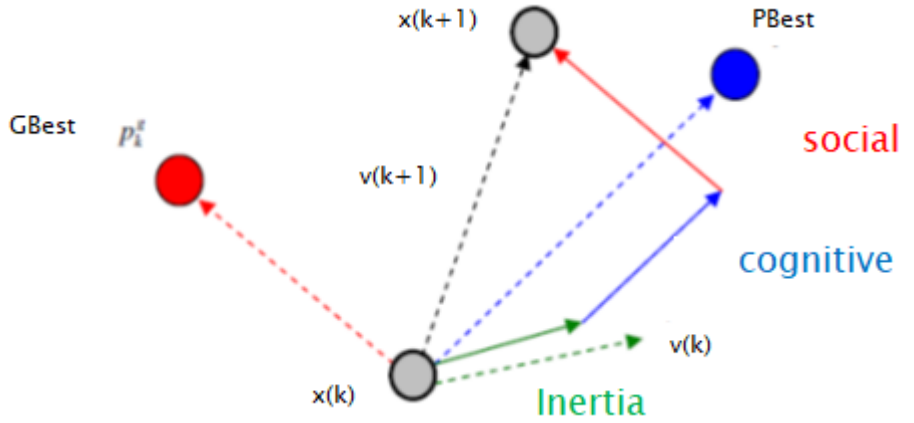
$$v_i(t+1) = c_0 * v_i(t) + c_1 * rand() * (\text{pbest}x_i - x_i(t)) + c_2 * rand() * (\text{gbest}x - x_i(t))$$



(in original method, $c_0=1$, but many researchers now play with this)

Equation (b)

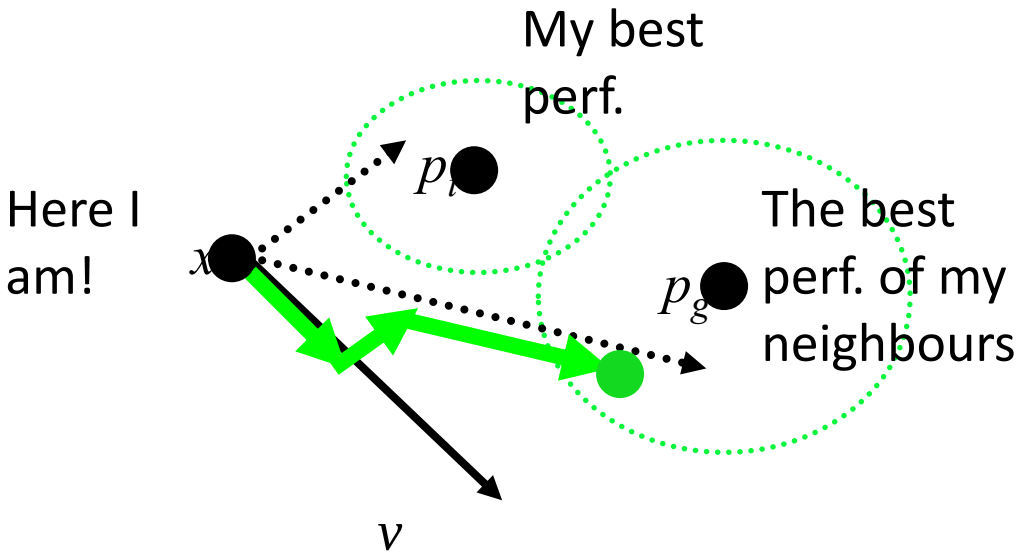
$$x_i(t+1) = x_i(t) + v_i(t+1)$$



Pseudocode

For each particle
    Initialize particle
END
Do
  For each particle
    Calculate fitness value
    If the fitness value is better than its peronal best
    set current value as the new ***pBest***
  End
  Choose the particle with the best fitness value of all as ***gBest***
  For each particle
    Calculate particle velocity according Equation (a)
    Update particle position according Equation (b)
  End
While maximum iterations or minimum error criteria is not attained

Position Adjustment



Particles adjust their positions according to a ``Psychosocial compromise'' between ***what an individual is comfortable with***, and **what society reckons**

PSO Solution Update in 2D

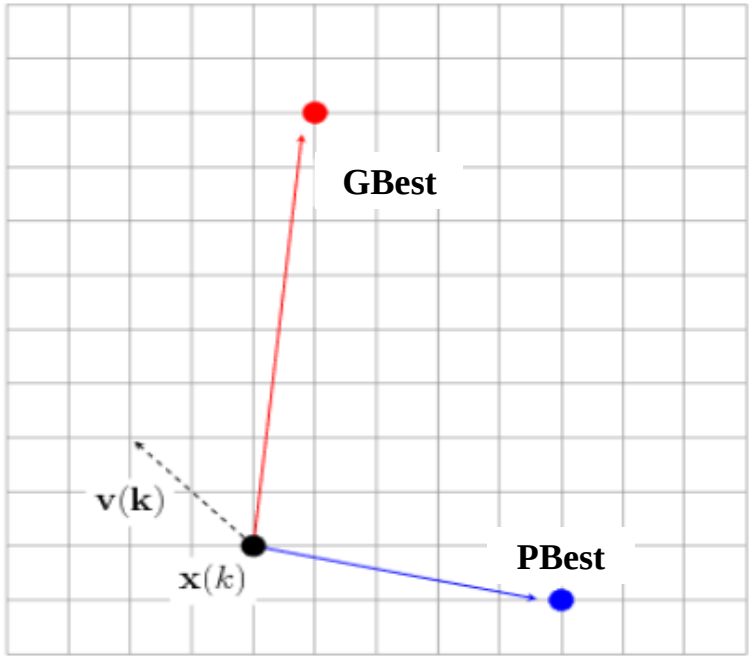
- **x**(k) - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

PSO Solution Update in 2D

Inertia: $\mathbf{v}(k)=(-2, 2)$

- $\mathbf{x}(k)$ - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

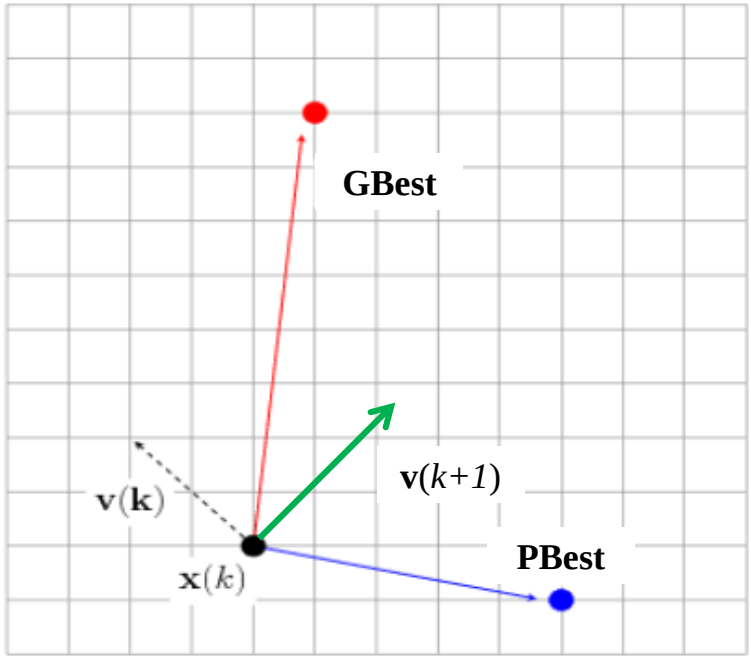
PSO Solution Update in 2D



- Inertia: **v**(*k*)=(-2,2)
- Cognitive:
  PBest-**x**(*k*)=(9,1)-(4,2)=(5,-1)
- Social:
  GBest-**x**(*k*)=(5,10)-(4,2)=(1,8)

- **x**(*k*) - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

PSO solution update in 2D

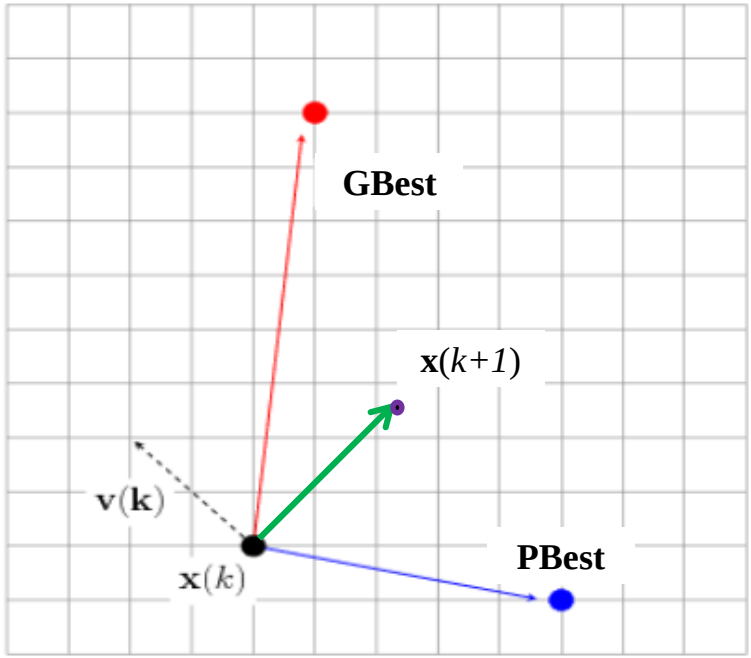


- Inertia: $\mathbf{v}(k)=(-2,2)$
- Cognitive:
  PBest-$\mathbf{x}(k)=(9,1)-(4,2)=(5,-1)$
- Social:
  GBest-$\mathbf{x}(k)=(5,10)-(4,2)=(1,8)$

$\mathbf{v}(k+1)=(-2,2)+0.8*(5,-1)+0.2*(1,8) = (2.2,2.8)$

- $\mathbf{x}(k)$ - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

PSO Solution Update in 2D

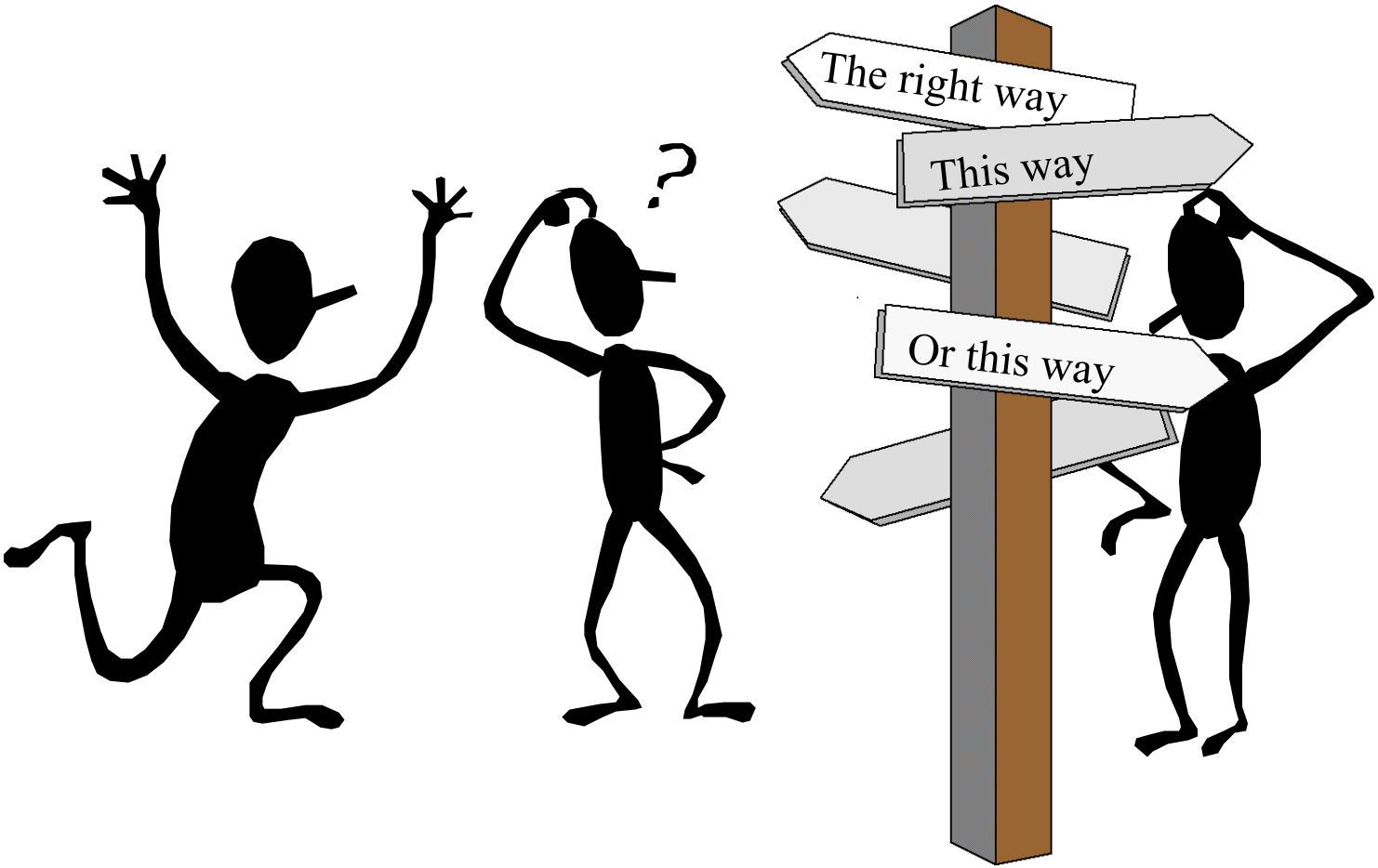


- Inertia: $\mathbf{v}(k)=(-2,2)$
- Cognitive:
  PBest-$\mathbf{x}(k)=(9,1)-(4,2)=(5,-1)$
- Social:
  GBest-$\mathbf{x}(k)=(5,10)-(4,2)=(1,8)$
- $\mathbf{v}(k+1)=(2.2,2.8)$

$\mathbf{x}(k+1)=\mathbf{x}(k)+\mathbf{v}(k+1)=(4,2)+(2.2,2.8)=(6.2,4.8)$

- $\mathbf{x}(k)$ - Current solution (4, 2)
- PBest - Particle's best solution (9, 1)
- GBest-Global best solution (5, 10)

How to Choose Parameters



Parameters

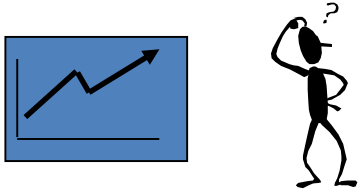
- Number of particles
  - (10—50) are reported as usually sufficient.
- C_1 (importance of personal best)
- C_2 (importance of neighbourhood best)
  - Usually $C_1 + C_2 = 4$.
  - No good reason other than empiricism
- V_{max}: limit on velocity

V_{max}

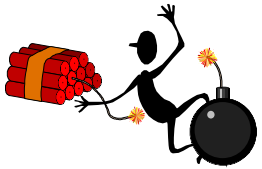
- An important parameter in PSO; typically the only once adjusted.
- Particles' velocities on each dimension are clamped to a maximum velocity $\boldsymbol{V_{max}}$.
  - If the sum of accelerations would cause the velocity on that dimension to exceed $\boldsymbol{V_{max}}$, the velocity on that dimension is limited to $\boldsymbol{V_{max}}$.
- Determines “fineness” with which regions are searched
  - if too high, can fly past optimal solutions
  - if too low, can get stuck in local minima

Adaptive Swarm Size

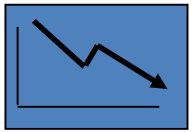
There has been enough improvement
although I'm the worst



I try to kill myself

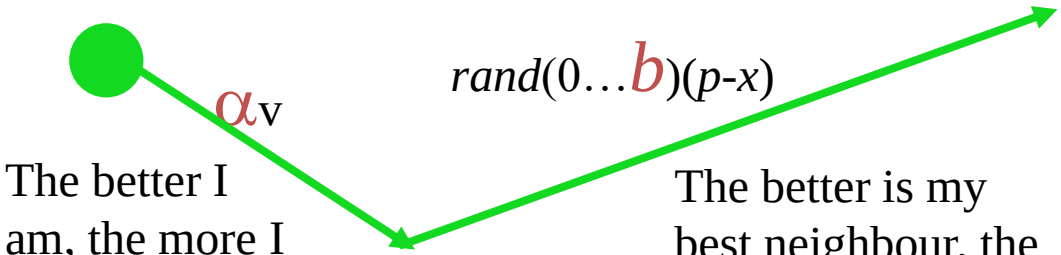


I'm the best
but there has been not enough improvement



I try to generate a new particle

Adaptive Coefficients



Explore PSO and Its Parameters

http://www.macs.hw.ac.uk/~dwcorne/mypages/apps/pso.html

PSO - Summary

- The method allows the motion of particles to explore the space of interest.
- Each particle updates its position in discrete unit time steps.
- Not “**what**” that best solution was, but “**where**” that best solution was.
- The velocity is updated by a linear combination of two terms:
  - The first along the direction pointing to the best position discovered by the particle.
  - The second towards the overall best position.

Additional Reading + References

- Dorigo, M., Birattari, M. and Stutzle, T., 2006. Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(*4*), pp. 28-39.

- Marini, F. and Walczak, B., 2015. Particle swarm optimization (PSO). A tutorial. *Chemometrics and Intelligent Laboratory Systems*, *149*, pp.153-165.