SWARM INTELLIGENCE

- *Swarm intelligence (SI) is an artificial intelligence technique based around the study of collective behavior in decentralized, self-organized systems.*
- *SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. Although there is normally no centralized control structure dictating how individual agents should behave, local interactions between such agents often lead to the emergence of global behavior. Examples of systems like this can be found in nature, including ant colonies, bird flocking, animal herding, bacteria molding and fish schooling (from Wikipedia).*

MIND IS SOCIAL

- **Human Intelligence is the result of social interaction**
- Evaluate, compare and imitate other individuals, learn from experience emulating the successful behavior of others, enables people to adapt to complex environments and situations through the discovery of optimal patterns, beliefs and behaviors (Kennedy & Eberhart, 2001).
- **Culture and cognition are consequences of human social choices**
- Culture emerges when individuals become similar through a process of social learning
- To model human intelligence there is need to model individuals in a social context.

FEATURES OF A SI SYSTEM

- A set of simple individuals with limited capacities
- Individuals are not aware of the system in its global view
- Local communication patterns (direct or indirect - stigmergic)
- Distributed computation: no centralized coordination of individual activities
- Robustness
- Adaptivity

- NATURAL METAPHORS
  - Ant colonies
  - Bee colonies
  - Fish shoals
  - Bird flocks

SELF-ORGANISATION

- Ingredients:
  - Multiple interactions among agents
    - Simple agents (es. rule based)
    - Multi-agent systems
  - Positive Feedback
    - Reinforcement of common behaviours
    - Amplification di random fluctuations and structure formation
  - Negative Feedback
    - Saturation
    - Competition
    - Exhaust resources

STIGMERGY

- Stigmergy is a form of indirect communication
- An agent modifies the environment and the other react to this chance
  - Example: ants communicate through pheromone

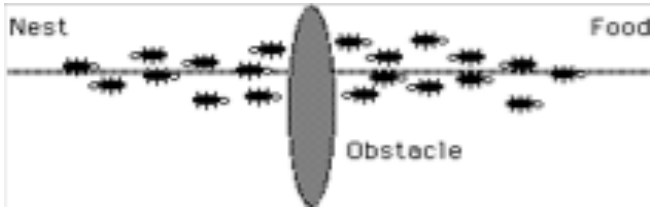
SWARM INTELLIGENCE ALGORITHMS

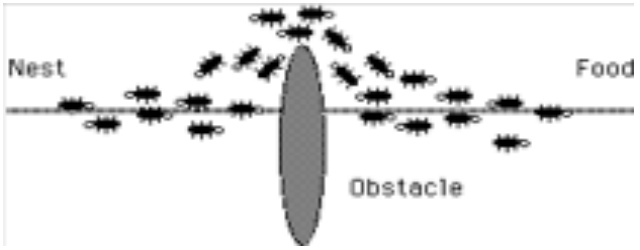
- Many algorithms exist based on the swarm intelligent concept
- Ant Colony Optimization ACO (Dorigo, 1992)
  - Algorithm based on the behaviour of ants. Positive feedback based on pheromone trails. Positive feedback based on pheromon trails that reinforce components that contribute to the problem solution
- Artificial Bee Colony Algorithm (Karaboga in 2005)
  - Algorithm based on the behaviour of bees. Population of bees that look for nectar.
- Particle Swarm Optimization PSO (Kennedy, Eberhart 1995)
  - Algorithm based on the observation of bird flocks or fish shouls. Stigmergy as communication and imitation of neighborhoods.

ANT COLONY

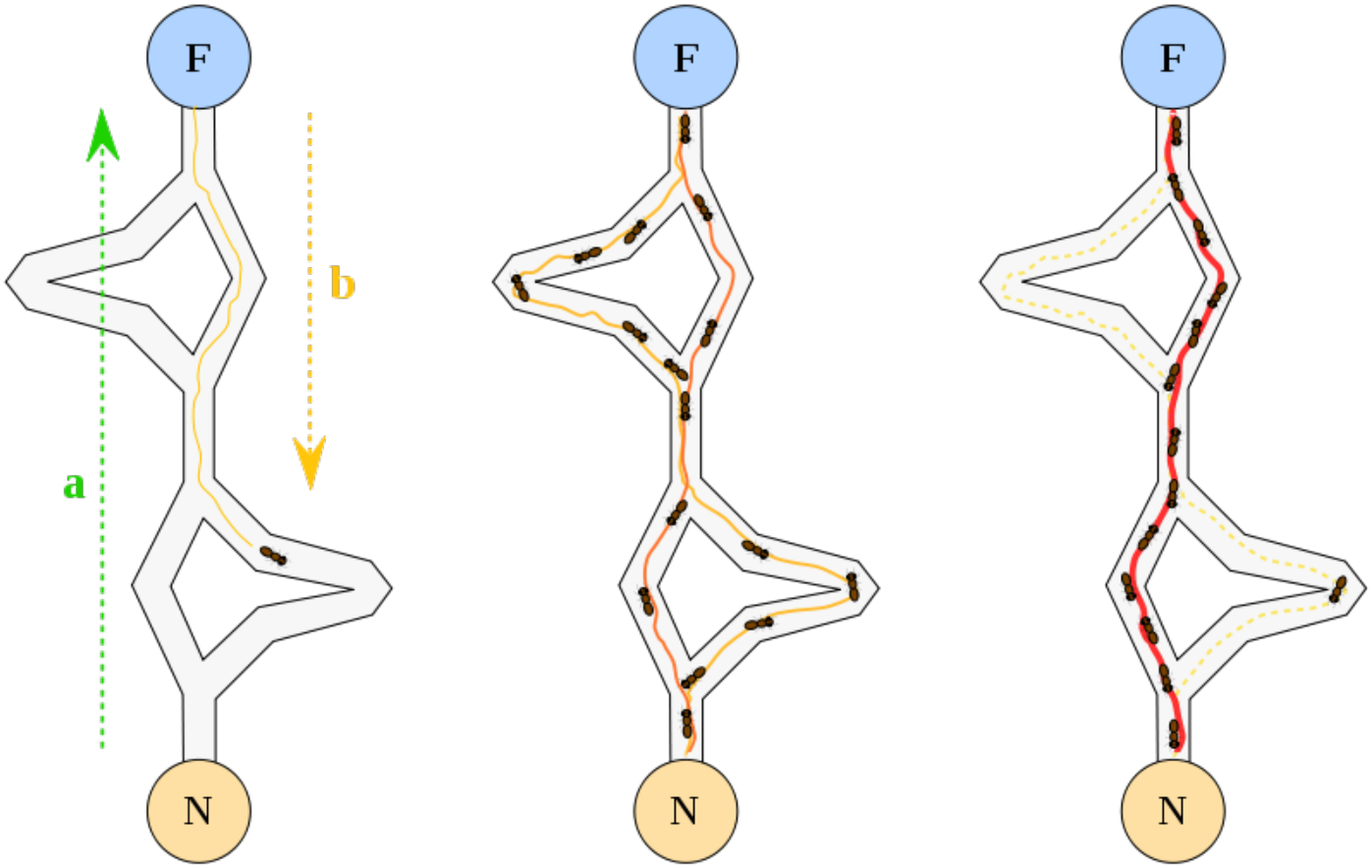
- From the observation of the ants we discover that:
  - Ants deposit pheromone trails while walking from the nest to the food and vice versa
  - Ants tend to choose (more likely) the paths marked with higher pheromone concentrations.
  - Cooperative interaction leads to the emergent behavior to find the shortest path

ANT BEHAVIOUR





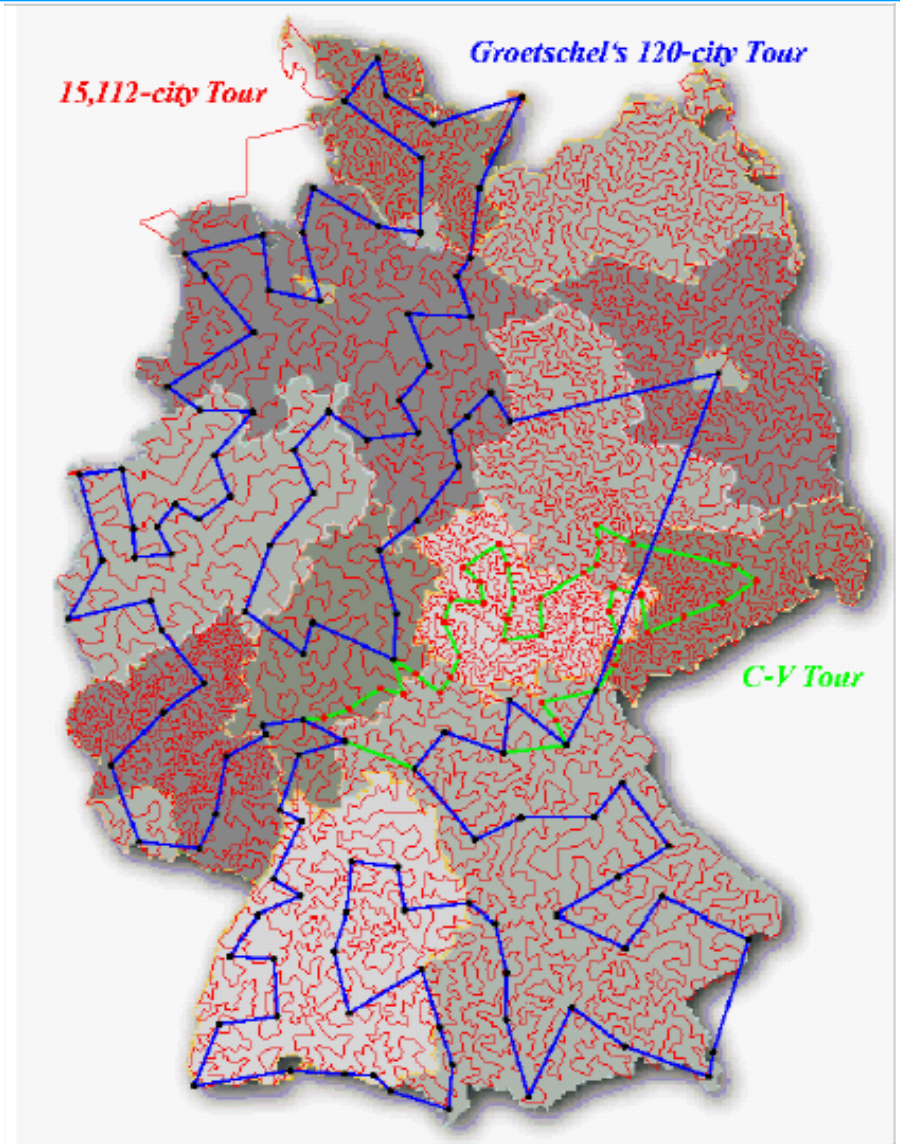
DOUBLE BRIDGE



ANT COLONY OPTIMIZATION

- Probabilistic parametrized model – the *pheromone model* – used to model pheromon trails
- Ants build solution components in an incremental way.
- Stochastic steps on a fully connected graph called *construction graph:* G = (C,L).
  - Vertexes C are solution components
  - Arcs L are connections
  - States are paths on G
- Constraints can be represented to define what is a consistent solution

EXAMPLE: TSP



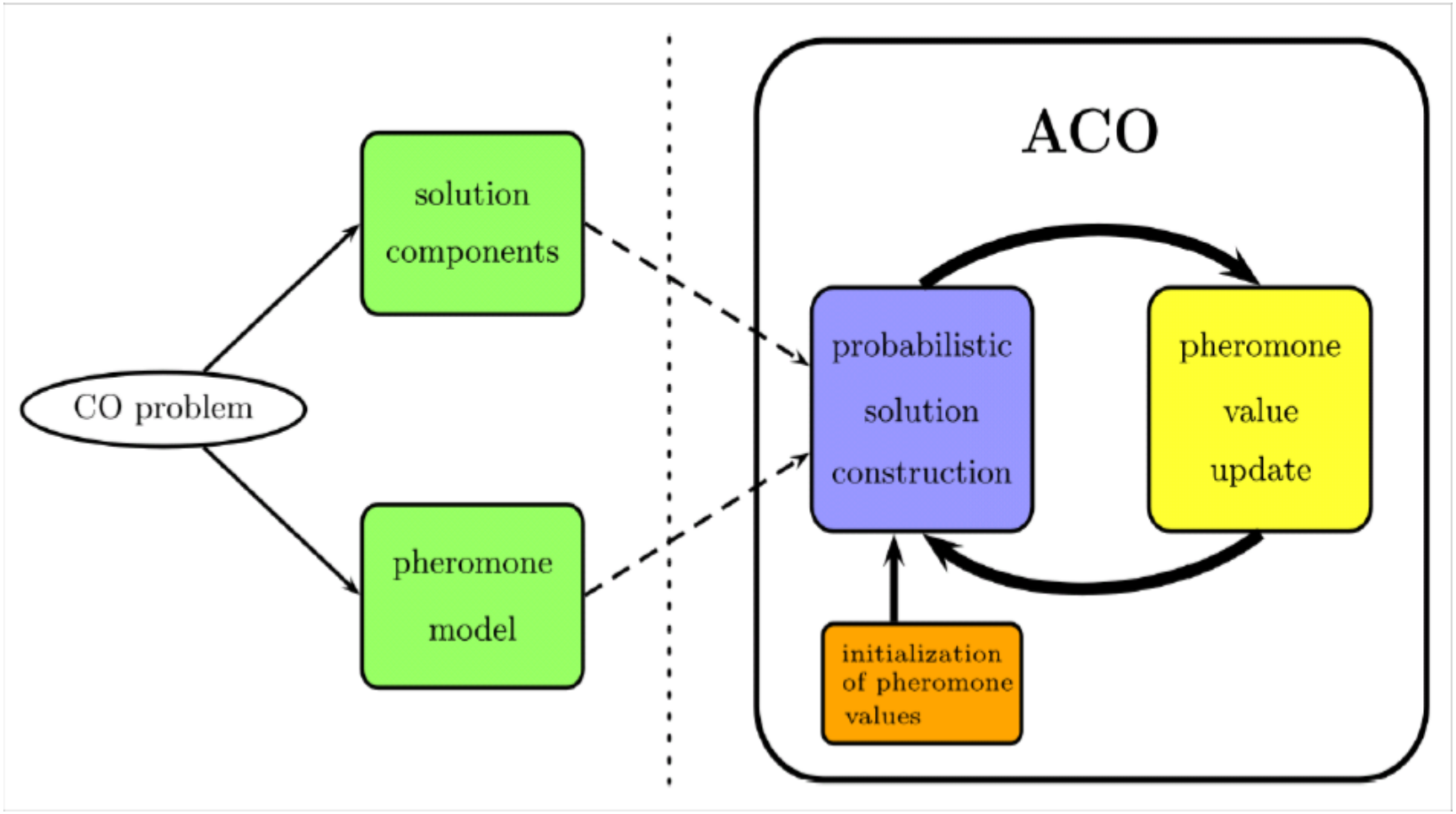
EXAMPLE: TSP

- A TSP model in ACO
  - Nodes of G (solution components) are cities to be visited;
  - Arcs are connections between cities
  - A solution is an Hamiltonian path in the graph
  - Constraints are used to avoid sub-cycles: each ant can visit a city once.

INFORMATION SOURCES

- Connections, solution components or both have the following associated information:
  - Pheromone τ
  - Heuristic value η
- The pheromone value abstracts natural pheromone trails and codes the long term memory of the global search process.
- The heuristic value represents the prior background knowledge on the problem.

BASIC SCHEMA



ACO SYSTEM

- First example of Ant Colony Optimization

- Ants build a solution following a path on the construction graph.
- A transition rule is followed to choose the next node to visit.
- The heuristics and the pheromone are used.
- Pheromone values are updated on the basis of the QUALITY OF THE SOLUTION FOUND by the ants.

ANT-SYSTEM ALGORITHM

InitializePheromoneValues()

while *termination conditions not met* ***do***

for all *ants* ***a*** $\in$ ***A do***

$s_a \leftarrow$ *ConstructSolution*(τ ,η)

end for

ApplyOnlineDelayedPheromoneUpdate()

end while

ANT-SYSTEM

- Memory is used to remind past paths
- Starting from node i, we have to probabilistically chose the next consistent node to visit
- The probabilistic choice depends on
  - pheromone trail τ_{ij}
  - heuristics $\eta_{ij} = 1/d_{ij}$

α and β weight the importance of the pheromone and the heuristics

$$p_{ij} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha} \; [\eta_{ij}]^{\beta}}{\sum_{k \text{ feasible}} [\tau_{ij}]^{\alpha} \, [\eta_{ij}]^{\beta}} & \text{if j consistent} \\ 0 & \text{otherwise} \end{cases}$$

ANT SYSTEM

- The pheromone is updated with the following rule
- $\tau_{ij} \leftarrow (1 - \rho)\,\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$ (ρ: evaporation coefficient)
- $\Delta\tau_{ij} = \begin{cases} 1/L_k & \text{if ant k used arc (i,j)} \\ 0 & \text{otherwise} \end{cases}$
- L_k: length of the path followed by ant k

ALGORITHM

while termination conditions not met do
ScheduleActivities
AntBasedSolutionConstruction()
PheromoneUpdate()
DaemonActions() {optional}
end ScheduleActivities
end while

ALGORITHM STEPS

- AntBasedSolutionConstruction()
  - Ants move by applying a stochastic local decision policy that uses values of pheromone and heuristic on graph components.
  - While moving, ants take track of the partial solutions (paths) that have been built

- PheromoneUpdate()
- Ants update the pheromone during the solution construction (*online step-by-step pheromone update).*
- Ants can update backward the pheromone on components used on the basis of the quality of the overall solution (*online delayed pheromone update).*
- Evaporation is applied all time.

ALGORITHM STEPS

- DemonActions()
  - Are “centralised” actions that cannot be executed by the single ants:
  - local search procedure applied to each solution built
  - Collection of global information to decide whether to leave additional pheromone to guide search from a global perspective

ACO- SYSTEM

- Papers on the course web-site
  - MAX-MIN Ant System
  - Hyper-cube Framework
  - Multi-level ACO
  - Beam ACO

- EU Project: SWARM-BOT, SWARMANOID
- http://www.swarm-bots.org/
- Video youtube
- http://www.youtube.com/watch?v=seGqyO32pv4m/watch?v=3YDkbltzMmA

HONEY BEE-COLONY

- Artificial Bee Colony Algorithm: ABC Algorithm
- Artificial bees of three types**:**
  - **employed bees** that are associated with a specific nectar source,
  - **onlookers** that observing the employed bees chose a nectar source and
  - **scouts** that discover new food sources

- Initially, food sources are discovered by scout bees. Then food is consumed and the source exhausted. The employed bees in that source become scout.
- Food position: solution (we have as many solutions as employed bees)
- Food quantity: fitness

ABC ALGORITHM

InitializationPhase()
repeat
EmployedBeePhase()
OnlookerBeePhase()
ScoutBeePhase()
Sol=BestSolutionSoFar
Until (Cycle=MaxCycleNum or MaxCPUtime)

ABC ALGORITHM

InitializationPhase()

- A set of food source positions are randomly selected by the bees and their nectar amounts are determined.

- Each solution X_m (m=1..Npopol.) is composed of n variables X_{mi} (i=1..n). Each variable is subject to a lower and upper bound and initialized to

$$X_{mi} = lb_i + rand(0,1)\ (ub_i - lb_i)$$

ABC ALGORITHM

EmployedBeePhase()

- After sharing the information about the nectar amount, every employed bee goes to the food source area visited by herself at the previous cycle since that food source exists in her memory, and then chooses a new food source by means of visual information in the **neighbourhood** of the present one.
- Fitness function of X_m : we use the objective function of the problem to compute the fitness as follows

$$ftn(X_m) = \begin{cases} 1/(1+obj(X_m)) & \text{se } obj(X_m) \geq 0 \\ 1+|obj(X_m)| & \text{se } obj(X_m) < 0 \end{cases}$$

ABC ALGORITHM

OnlookerBeePhase()

- An onlooker bee chooses a food source depending on the probability value associated with that food source

$$pm = \frac{ftn(X_m)}{\sum_{i=1}^{N_{popol.}} ftn(X_i)}$$

Positive feedback

ABC ALGORITHM

ScoutPhase()

- Scouts chose nectar sources random.

- The employed bees that cannot improve the solution after a given number of attempts become scout and abandon the food source.
- Negative feedback

ABC APPLICATIONS

- Training neural network: optimization function that represents the mean squared error
- Karaboga D. Basturk B. Ozturk C. (2007) Artificial bee colony (ABC) optimization algorithm for training feed-forward neural network, Modeling Decisions for Artificial Intelligence, LNCS 4617. 318-319.

- Numerical Optimization
- http://chern.ie.nthu.edu.tw/gen/4a-Karaboga-tr06_2005-original.pdf

- Combinatorial Optimization
- Karaboga D. Basturk B. (2007) Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems, Advances in Soft Computing, LNCS 4529. 789-798.

PARTICLE SWARM OPTIMIZATION

- Particle Swarm Optimization *PSO* has been proposed in1995 by:
  - A psyco-social scientist James Kennedy
  - An electrical engineer Russel Eberart
- The research activity starts from the analysis of interaction mechanisms between individuals that compose the swarm
- Particularly interesting when the whole swarm has a common goal such as food search

PARTICLE SWARM OPTIMIZATION

- The observation of rules that guide the bird flock moves show that each individual entity has three driving trends
  - Follow neighbours
  - Stay in the flock
  - Avoid collisions

- With these rules it is possible to describe and model the collective move of a flock with NO COMMON OBJECTIVE
- PSO adds a common objective: food search

PARTICLE SWARM OPTIMIZATION

- With a common objective, a single individuals that finds a food source has two alternatives:
  - Move away from the group to reach the food (individualistic choice)
  - Stay in the group (social choice)

- If more than one individual entity moves toward the food other flock members do the same

- Gradually the whole group changes direction toward promising areas. The information propagates to all members.

PARTICLE SWARM OPTIMIZATION

- With PSO we can solve optimization problems with the following analogy:
  - individuals: tentative configurations that move and sample the solution N-dimensional space.
  - Social interaction: each individual agent takes advantage from other searches moving toward promising regions (best solution globally found)

- Search strategy can be found as a balance between exploration and exploitation:
  - exploration: individual agents that search for a solution
  - Exploitation: social behaviour which is the exploitation of other individual successful behaviour

NEIGHBORHOOD

- A feature that appears to be important is the concept of proximity
  - individuals are affected by the actions of other individuals that are closer to them (sub-groups)
  - individuals are part of more sub-groups, and then the spread of information is globally guaranteed

- Sub-groups are not tied to the physical proximity of the configurations in the parameter space but are a priori defined and may take into account also considerable shifts between individuals

ALGORITHM

- PSO optimizes a problem by setting a population (Swarm) of candidate solutions (particles).
- PSO moves these particles in the search space through simple mathematical formulas.
- The movement of particles is guided by the best position found so far in search space (from individual to population) and it is updated when better solutions are discovered
- f: $\mathbb{R}^n \rightarrow \mathbb{R}$ fitness or cost function to minimize: it takes a solution (vector) and produces a fitness value. The gradient of f is not known.
- Goal:find solution **a** such that $f(\mathbf{a}) \leq f(\mathbf{b})$ for all **b** in the search space

ALGORITHM

- S number of particles in population.
- Each particle has
  - a position $\mathbf{x}_i \in \mathbb{R}^n$ in the search space
  - a speed $\mathbf{v}_i \in \mathbb{R}^n$.
- $\mathbf{p}_i$ is the best solution found so far by particle i
- $\mathbf{g}$ is the best solution found so far by the entire swarm.

- For each particle $i = 1, ..., S$ do:
  - Initialize the particle's position with a uniformly distributed random vector: $\mathbf{x}_i \sim U(\mathbf{b}_{lo}, \mathbf{b}_{up})$, where $\mathbf{b}_{lo}$ and $\mathbf{b}_{up}$ are the lower and upper boundaries of the search-space.
  - Initialize the particle's best known position to its initial position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
  - If ($f(\mathbf{p}_i) < f(\mathbf{g})$) update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
  - Initialize the particle's velocity: $\mathbf{v}_i \sim U(-|\mathbf{b}_{up}-\mathbf{b}_{lo}|, |\mathbf{b}_{up}-\mathbf{b}_{lo}|)$

ALGORITHM

- Until a termination criterion is met (e.g. number of iterations performed, or adequate fitness reached), repeat:
  - For each particle $i = 1, ..., S$ do:
    - Pick random numbers: $r_p, r_g \sim U(0,1)$
    - Update the particle's velocity: $\mathbf{v}_i \leftarrow \omega\, \mathbf{v}_i + \varphi_p\, r_p\, (\mathbf{p}_i - \mathbf{x}_i) + \varphi_g\, r_g\, (\mathbf{g} - \mathbf{x}_i)$
    - Update the particle's position: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
    - If $(f(\mathbf{x}_i) < f(\mathbf{p}_i))$ do:
      - *Update the particle's best known position:* $\boldsymbol{p}_i \leftarrow \boldsymbol{x}_i$
      - *If* $(f(\boldsymbol{p}_i) < f(\boldsymbol{g}))$ *update the swarm's best known position:* $\boldsymbol{g} \leftarrow \boldsymbol{p}_i$
- Return **g** : best found solution.
- Parameters ω, φ_p, is φ_g should be carefully selected as they strongly influence the effectiveness and the efficiency of the PSO method.

ROBOT BASED ON PSO

- Article which is introduced PSO
- Kennedy, J .; Eberhart, R. (1995). "Particle Swarm Optimization". *Proceedings of the IEEE International Conference on Neural Networks*. **IV**. pp. 1942-1948.http://www.engr.iupui.edu/~shi/Coference/psopap4.html.

- movie robots
- http://www.youtube.com/watch?v=RLIA1EKfSys
- Sorgente in movimento
- http://www.youtube.com/watch?v=nul8nYIQ8ug&feature=related

PARAMETER TUNING

- These algorithms are very simple but require an accurate parameter tuning activity
- Tedious and error prone operation. It is hard to find the optimal parameter configuration
- Techniques for automatic parameter tuning
  - ParamILS http://www.cs.ubc.ca/labs/beta/Projects/ParamILS/
  - Available on line