DIFFERENTIAL EVOLUTION, PARTICLE SWARM OPTIMISATION, AND ANT COLONY OPTIMISATION

Luca Manzoni

DIFFERENTIAL EVOLUTION

Differential evolution is a popular optimization algorithm used in solving complex engineering and scientific problems. It is a type of evolutionary algorithm that is inspired by the principles of natural selection and genetic algorithms.

In differential evolution, a population of candidate solutions (vectors) is iteratively improved through the processes of mutation, crossover, and selection. The mutation step involves the creation of new candidate solutions by perturbing the existing population. The crossover step combines the new candidate solutions with the original population to create offspring. The selection step then determines which candidate solutions will survive and be included in the next generation based on their fitness.

One of the key features of differential evolution is its simplicity and ease of implementation. It is known for its robustness, convergence speed, and ability to handle non-linear and multi-modal optimization problems.

DIFFERENTIAL EVOLUTION: IDEAS

- **Differential evolution** (DE) was invented in 1997 by Storn and Price

- Used for the solution of real-valued optimisation problems

- The approach is evolutionary, but different from GA for two main reasons:

  - The way new individuals are generated

  - The selection process

NOTATION

- The search space is $\mathbb{R}^m$ for some $m \in \mathbb{N}$
- We still have a population of solution consisting of n individuals
- The i^{th} individual is a vector $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,m}) \in \mathbb{R}^m$

DIFFERENTIAL MUTATION

- Select one candidate solution $\mathbf{x}_i$ from the current population

- Select three other vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ from the current population (in the original formulation $\mathbf{a}, \mathbf{b}, \mathbf{c}$, and $\mathbf{x}_i$ are all distinct)

- Compute the **"donor" vector** for $\mathbf{x}_i$ as $\mathbf{v}_i = a + F \cdot (b - c)$ where $F \in [0,2]$ is called the **mutation factor**

BINOMIAL CROSSOVER

- The solution $\mathbf{x}_i$ and the "donor" vector $\mathbf{v}_i$ are combined in a **"trial" vector $\mathbf{u}_i$**

- $\mathbf{u}_i$ is defined, for each coordinate $j \in \{1, \ldots, m\}$, as

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } \mathrm{rnd}_{i,j} \leq p_{\mathrm{CR}} \text{ or } I_{\mathrm{rnd}} = j \\ x_{i,j} & \text{otherwise} \end{cases}$$

I'M SAYING THAT AT LEAST ONE ELEMENT IS SELECTED FROM BOTH VECTORS

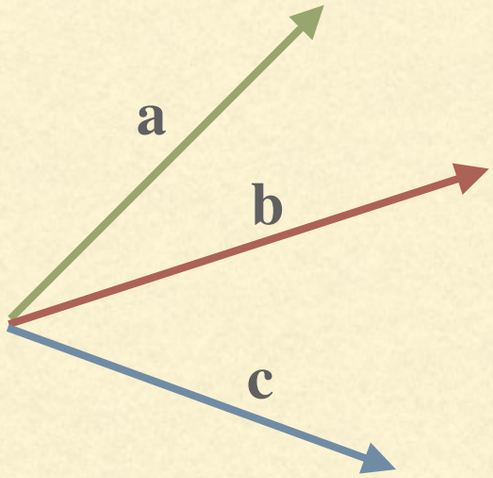
- Where p_{CR} is the crossover probability, $I_{\mathrm{rnd}} \in \{1, \ldots, m\}$ is a randomly selected index, and $\mathrm{rnd}_{i,j}$ are random numbers in $[0,1]$

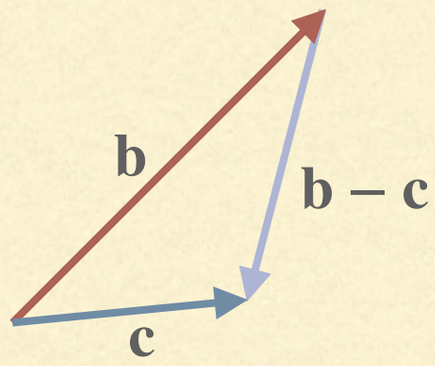
DE: SELECTION

- Given the parent $\mathbf{x}_i$ and the "trial" vector $\mathbf{u}_i$, selection is done by keeping only the individual with the best fitness:
  - $\mathbf{x}_i$ is kept if the $f(\mathbf{x}_i) < f(\mathbf{u}_i)$ (in a minimisation problem)
  - $\mathbf{u}_i$ replaces $\mathbf{x}_i$ otherwise
- This mutation/crossover/selection process is repeated for all individuals in the population to produce a new population of solutions

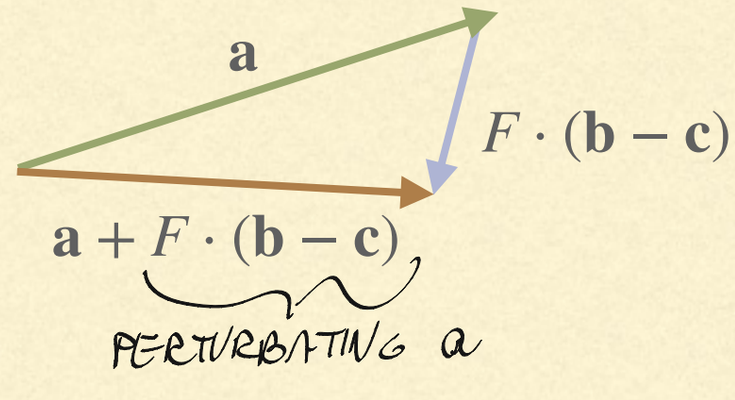
GRAPHICAL REPRESENTATION

WHAT IS THE EFFECT, GEOMETRICALLY?

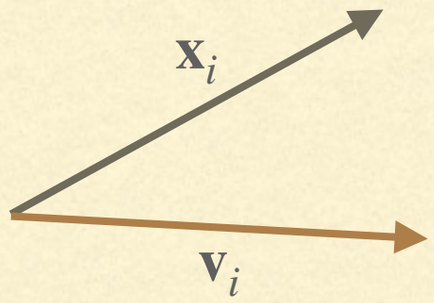




WE ADD SOME ADACTIVITY TO THE SOLUTION

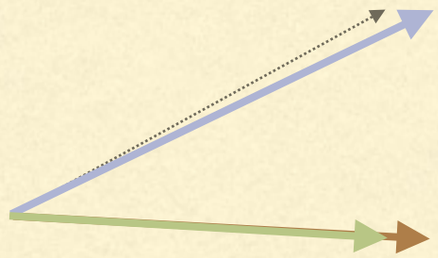


Extraction of three vectors from the population



Donor vector and $\mathbf{x}_i$

Computing the "donor vector"



Possible trial vectors and the original vector $\mathbf{x}_i$

↳ IN OPPOSITE DIRECTION

TAXONOMY

- The described scheme is sometimes called **DE/rand/1** with the meaning:
  - **DE**: self-explanatory
  - **rand**: the first vector of the differential mutation is selected uniformly at random across all the individuals
  - **1**: the donor vector is created using only one differential mutation

TAXONOMY

Best individual found so far

NAME	DIFFERENTIAL MUTATION
DE/best/1	$\mathbf{v}_i = \mathbf{x}_{\text{best}} + F \cdot (\mathbf{b} - \mathbf{c})$
DE/current-to-best/1	$\mathbf{v}_i = \mathbf{x}_i + F \cdot (\mathbf{x}_{\text{best}} - \mathbf{x}_i) + F \cdot (\mathbf{b} - \mathbf{c})$
DE/rand/2	$\mathbf{v}_i = \mathbf{a} + F \cdot (\mathbf{b} - \mathbf{c}) + F \cdot (\mathbf{d} - \mathbf{e})$
DE/rand-to-best/1	$\mathbf{v}_i = \mathbf{a} + F \cdot (\mathbf{x}_{\text{best}} - \mathbf{a}) + F \cdot (\mathbf{b} - \mathbf{c})$

ADAPTIVE DE (JADE)

MODIFICATION

- Introduced in 2009 by Zhang and Sanderson
- New mutation strategy
- External archive of sub-optimal solutions
- Dynamic update of hyper-parameters
- See https://ieeexplore.ieee.org/abstract/document/5208221

PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) is a popular metaheuristic optimization algorithm inspired by the social behavior of bird flocks or fish schools. In PSO, a population of potential solutions, called particles, is guided by their own best-known position and the best-known position of their neighbors towards finding the optimal solution to a problem.

The key components of the PSO algorithm include defining a fitness function to evaluate solution candidates, updating the velocity and position of each particle based on its own experience and the experience of its neighbors, and setting stopping criteria to terminate the optimization process when a satisfactory solution is found.

PSO: IDEAS

- Particle Swarm Optimisation (PSO) is part of a family of bio-inspired optimisation methods called **swarm intelligence**
  - Individuals are simple agents with limited capabilities (i.e., no "intelligent" agents).
  - Any intelligent behaviour emerges from the interactions of the simple agents
- See: swarm of birds, colony of ants, colony of bees. Linked to the ideas of "superorganism"

THEY HAVE A COMMON GOAL BUT ARE ALL INDIPENDENT ENTITIES

PSO: IDEAS

- PSO, like the other method we will see, is based on the exchange of information among different individuals (**particles**) that are part of the population (**swarm**)

- It is inspired by the collective movement of a group of animals (e.g., flock of birds, school of fishes, swarm of bees, etc)

PSO: DEFINITION

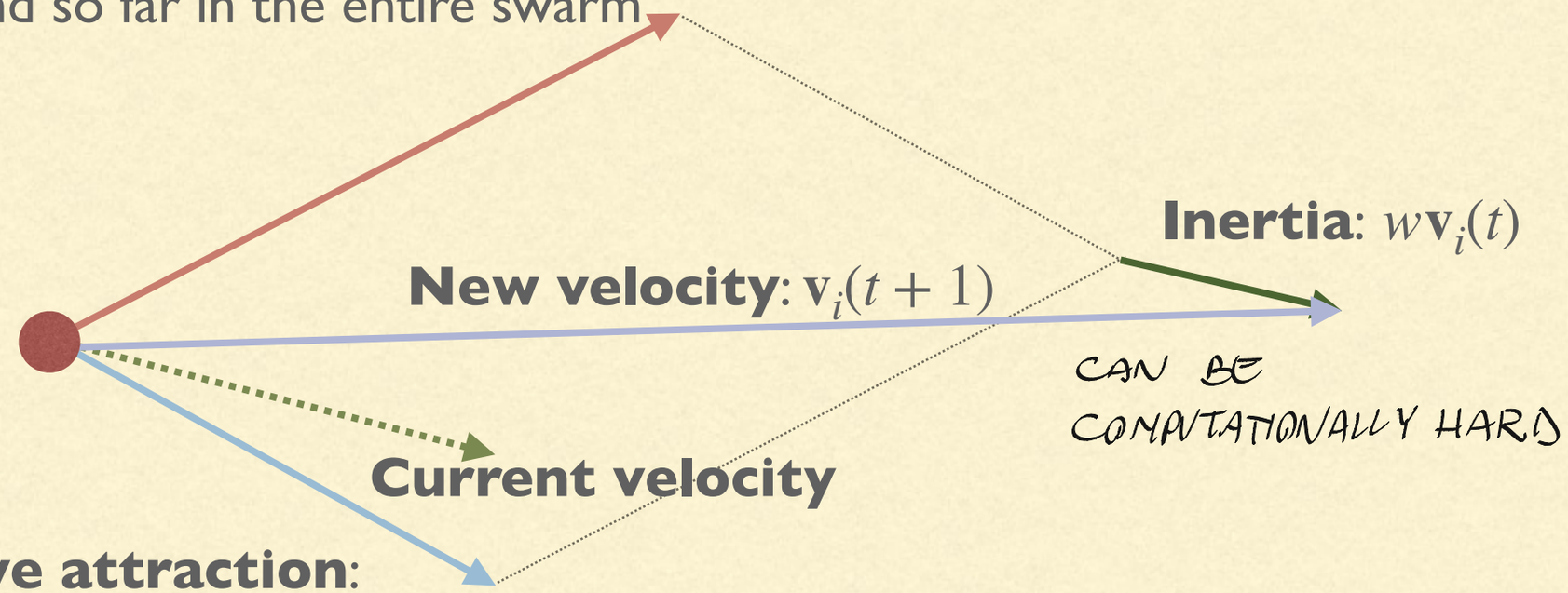
- The search space is m-dimensional, usually $\mathbb{R}^m$
- The swarm is composed of n particles. EACH WITH 2 ATTRIBUTES:
  - $\mathbf{x}_i(t)$ is the **position** of the i^{th} particle of the swarm at time t
  - $\mathbf{v}_i^t$ is the **velocity** of the i^{th} particle of the swarm at time t
- At each (discrete) time step, the position of the particle is updated as $\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t)$

PSO: DEFINITION

- The update of the velocity is more involved and keep track of three factors:
  - Inertia: the particle cannot change direction immediately
  - Social attraction: try to follow the swarm
  - Cognitive attraction: try to follow what the particle knows

VELOCITY COMPONENTS

Social attraction:
Attraction towards the best position $\mathbf{g}$
found so far in the entire swarm



Cognitive attraction:
Attraction towards the best position $\mathbf{b}_i$
found so far by the current particle

Usually each component of the velocity
is bounded (in absolute value) between $[v_{\min}, v_{\max}]$

PSO: VELOCITY UPDATE

The formula for the velocity update is:

$$\mathbf{v}_i(t+1) = w \cdot \mathbf{v}_i(t) + c_{\text{soc}} \cdot \mathbf{r}_1 \otimes (\mathbf{g} - \mathbf{x}_i(t)) + c_{\text{cog}} \cdot \mathbf{r}_2 \otimes (\mathbf{b}_i - \mathbf{x}_i(t))$$

- $\mathbf{r}_1$ and $\mathbf{r}_2$ are random vectors from $[0,1]^m$ and $\otimes$ denotes the Hadamard product
- $c_{\text{soc}} \in \mathbb{R}^+$ is the **social factor**
- $c_{\text{cog}} \in \mathbb{R}^+$ is the **cognitive factor**
- $w \in \mathbb{R}^+$ is the **inertia weight**

NEED TO BE TOONED

PSO: PARAMETERS

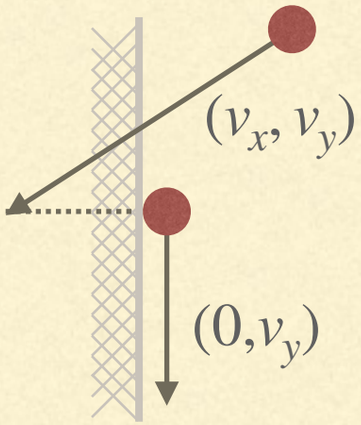
WHY THEESE COMPLEXITY? TO ALLOW EACH PARTICLE TO SEARCH WITHOUT LEAVING THE SWARM AND AT THE SEAN TIME PULL THE SWARM

- $c_{\text{soc}} \in \mathbb{R}^+$ is the **social factor**, it modulates the attraction towards the global best (i.e., the best solution found so far by the entire swarm). Usually $c_{\text{soc}} = 1.49445$. SWARM GOAL

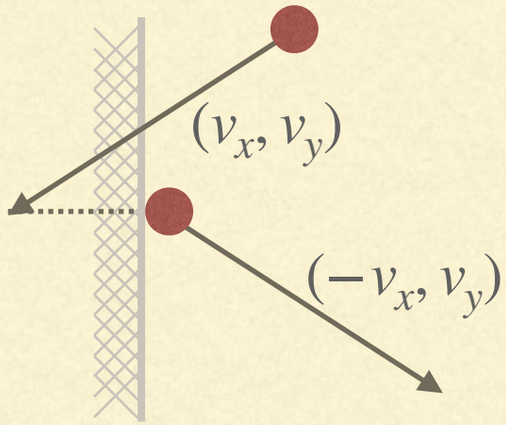
- $c_{\text{cog}} \in \mathbb{R}^+$ is the **cognitive factor**, it modulates the attraction towards the local best (i.e., the best solution found so far by the current particle). Usually $c_{\text{cog}} = 1.49445$. INDIVIDUA GOAL

- $w \in \mathbb{R}^+$ is the **inertia weight**, it is used to balance global and local search. It is usually decremented linearly from 0.9 to 0.4

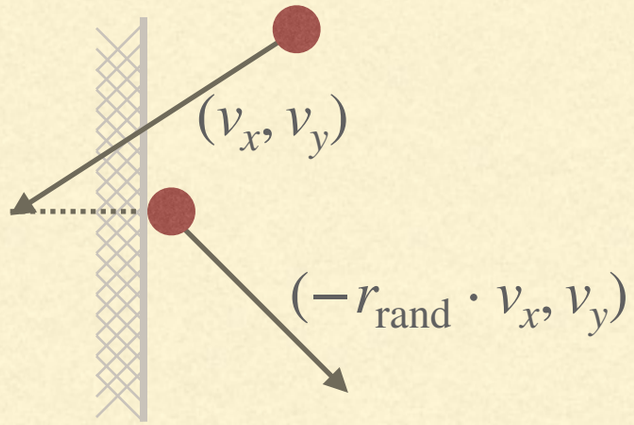
BOUNDARY CONDITIONS



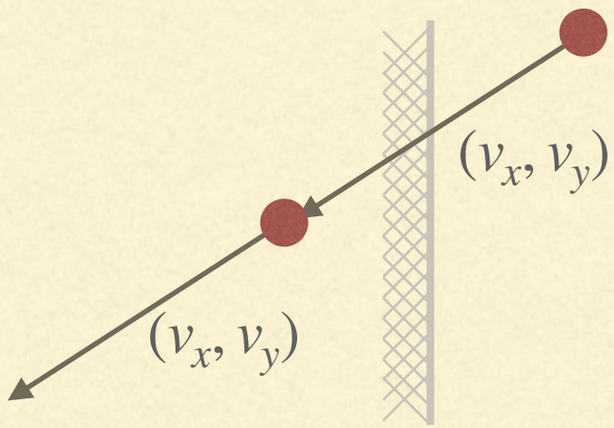
Absorbing



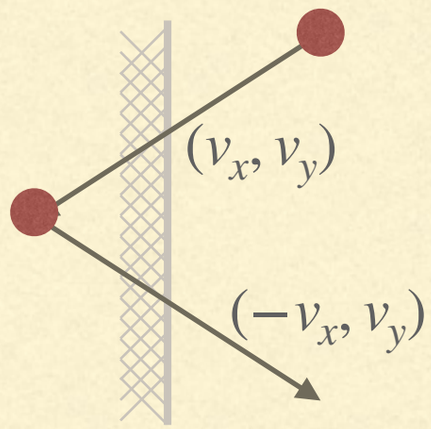
Reflecting



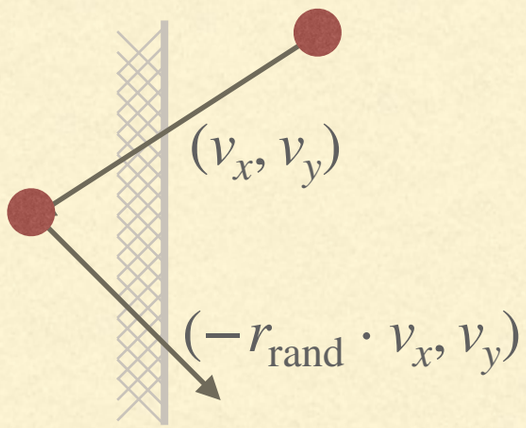
Damping



Invisible



Invisible/Reflecting



Invisible/Damping

SELECTION OF PSO PARAMETERS

- The settings for PSO are problem-dependent. To optimise them there are three main approaches:
  - Differential investigation of the impacts of the different settings
  - “Good enough” defaults
  - Self-tuning PSO that automatically adjust the parameters

ANT COLONY OPTIMIZATION

Ant colony optimization (ACO) is a metaheuristic algorithm inspired by the foraging behavior of ants. It is used to solve combinatorial optimization problems, such as the traveling salesman problem and the job scheduling problem.

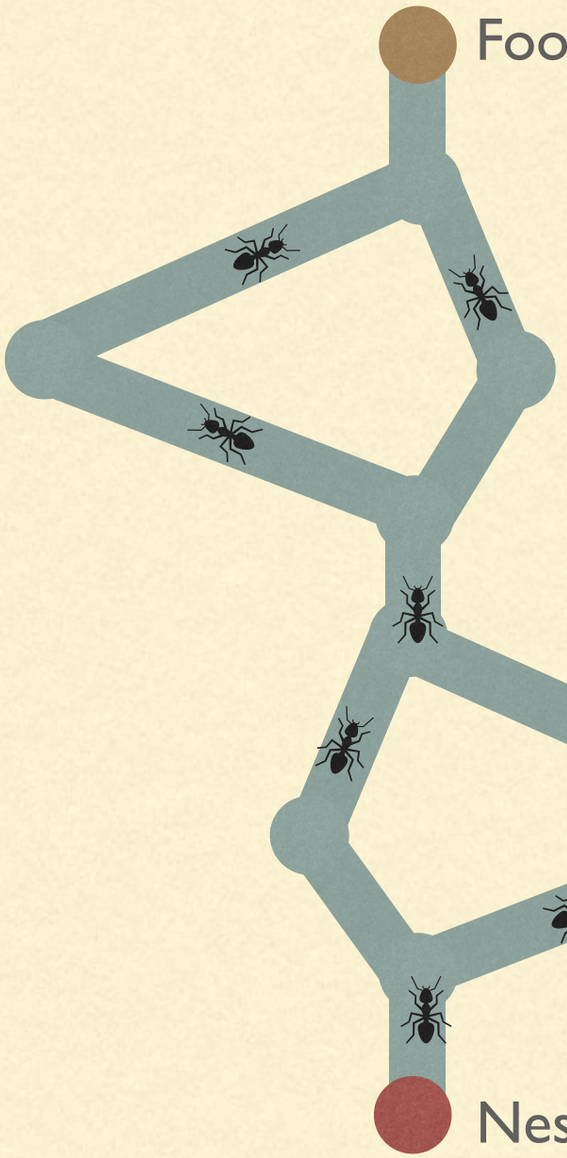
In ACO, a colony of artificial ants is used to iteratively construct solutions to the optimization problem by depositing pheromones on the edges of a graph representing the problem. The pheromone trails help guide other ants to choose better paths in the graph, with the amount of pheromone on an edge typically proportional to the quality of the solution found.

As the algorithm progresses, the pheromone trails are updated based on the quality of the solutions found by the ants, allowing the colony to converge towards optimal solutions. ACO is known for its ability to quickly converge to near-optimal solutions for complex problems with large search spaces.

ACO: IDEAS

- **Stigmergy** is a mechanism of *indirect coordination* mediated by the environment
  - Stigmergy can allow for complex behaviours without need of planning, control, or direct communication
  - Stigmergy allows the collaboration of very simple agents, lacking memory, intelligence, or even awareness of each other
- Standard example: *ants' pheromone trails*

STIGMERGY IN NATURE: THE DOUBLE BRIDGE EXPERIMENT

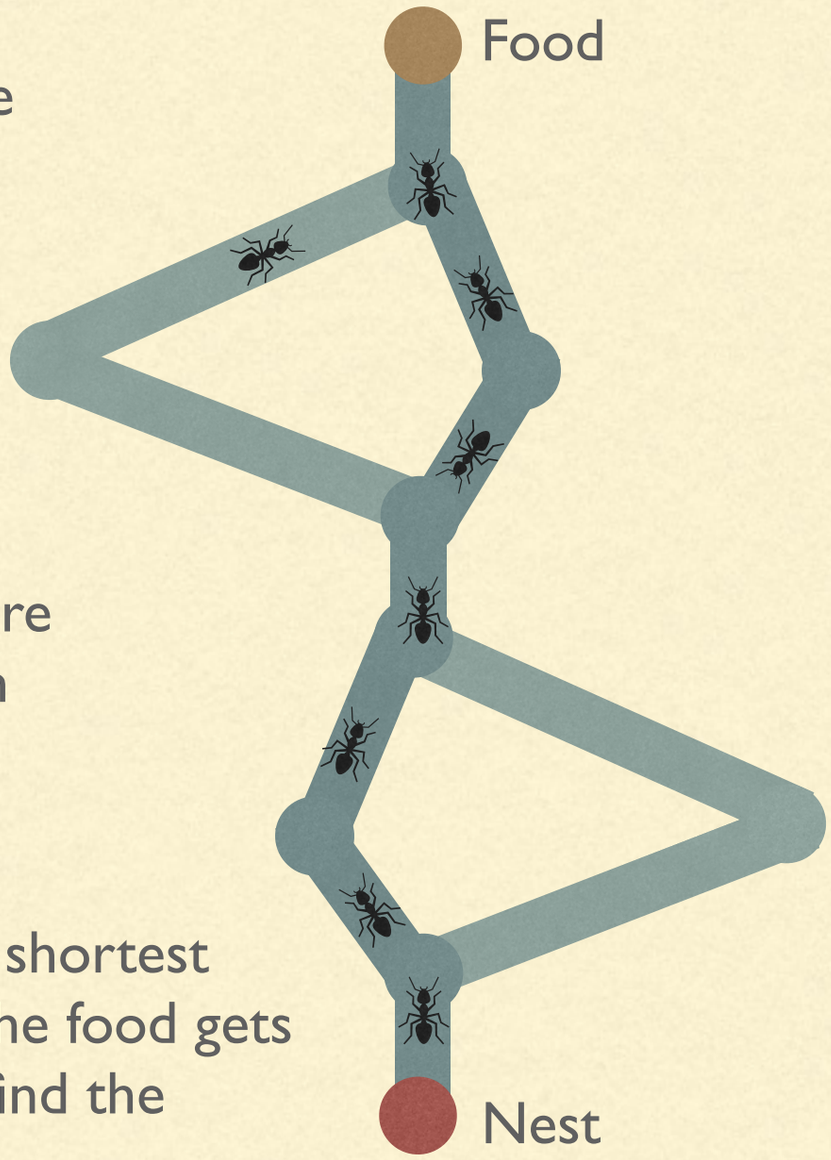


The ants initially explore the space randomly searching for food

Each ant leaves a pheromone trail that dissolves with time

The ants follow the trails where the pheromone concentration is higher

The pheromone trail on the shortest path between the nest and the food gets reinforced, allowing ants to find the shortest path to the food



STIGMERGIC SYSTEM

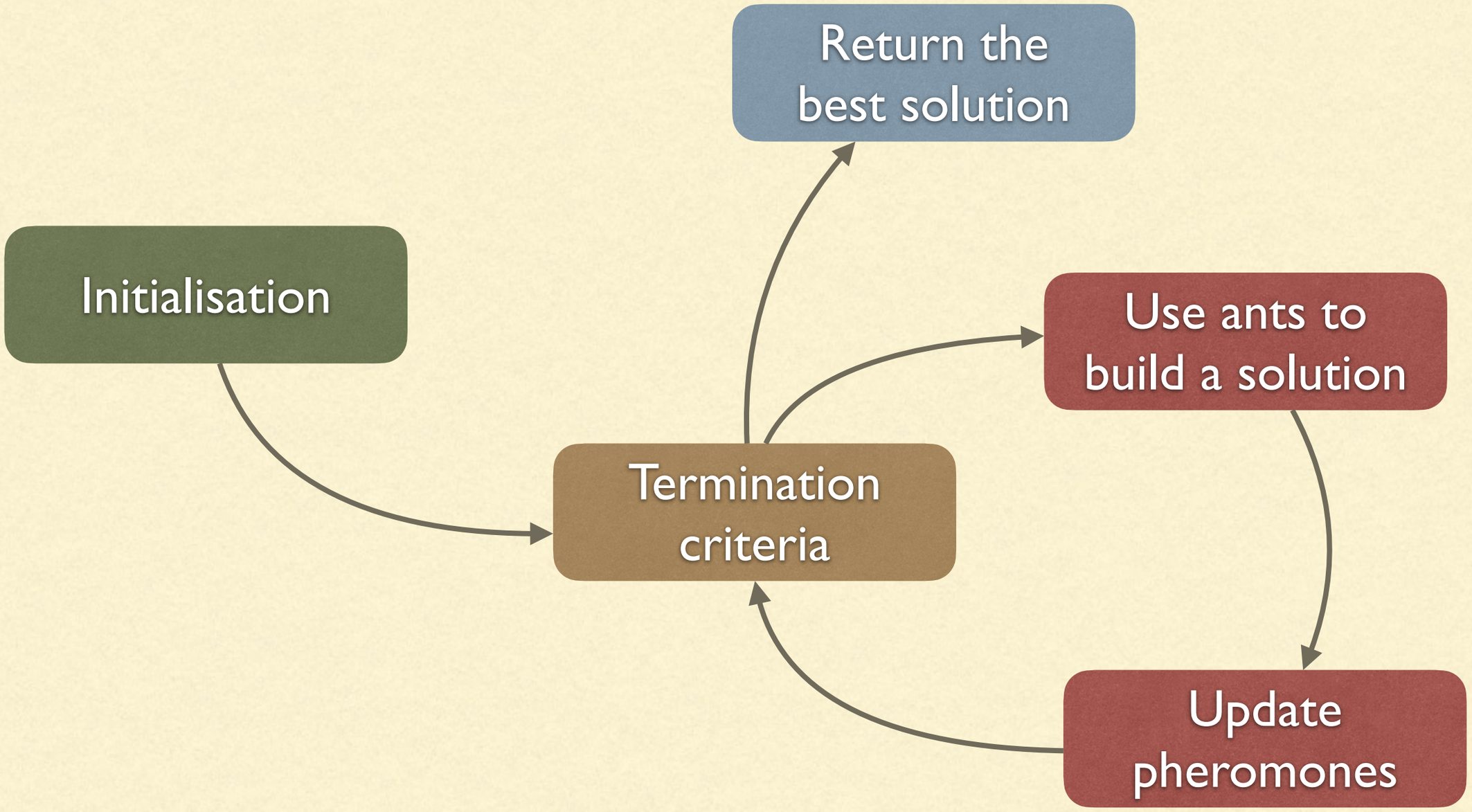
- A **Stigmergic system** has three main characteristics:
  - A **global environment** that is only partially perceivable by the agents
  - A **set of agents** that populate the environment with no single agent having complete knowledge of the system's state
  - **Complexity** emerging from the interaction of the previous to aspects that cannot be predicted or reduced to their simpler components

ANT COLONY OPTIMIZATION

- **Ant Colony Optimization (ACO)** is inspired by the collective behaviour of ants and usually employed to solve problems on graphs

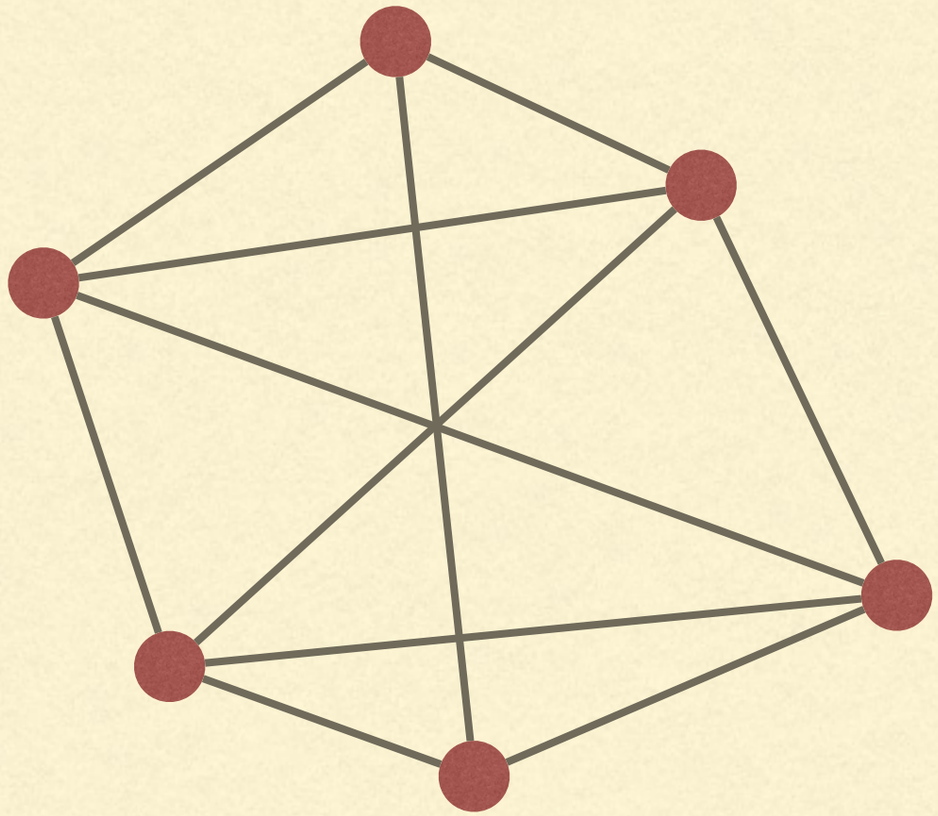
- ACO relies on stigmergy and simulated pheromone trails:
  - Pheromone left by ants in the search space evaporate over time
  - Pheromone traces guide the behaviour of the ants and, unless reinforced, evaporate over time
  - Ants are *stochastic solution building procedures* exploiting both simulated pheromone and available heuristic information about the problem being solved

ANT COLONY OPTIMIZATION



ACO FOR THE TSP

- We have a complete graph G of m nodes, representing m cities
- The edge between nodes i and j has weight/length $d_{i,j}$
- We want fo find the Hamiltonian circuit in G that minimises the sum of the weights of all the edges in the circuit (i.e., the shortest Hamiltonian circuit)



CONSTRUCTION OF THE SOLUTIONS

- $\tau_{i,j}(t)$ is the amount of pheromones on the edge (i, j) at time t
- $\eta_{i,j} = \frac{1}{d_{i,j}}$ is the heuristic information that we have on each edge
- $\mathcal{N}_i^k$ is the set of cities that the ant k has yet to visit when it is in city i
- An ant in city i will select which is the next city to visit according to a probability distribution based on some *heuristic information* and on the *amount of pheromones*.

CONSTRUCTION OF THE SOLUTIONS

- Each ant will start on a city and visit all cities selecting city j as the next one (in $\mathcal{N}_i^k$) with probability

- $$p_{i,j}^k(t) = \frac{(\tau_{i,j}(t))^\alpha (\eta_{i,j})^\beta}{\sum_{\ell \in \mathcal{N}_i^k} (\tau_{i,\ell}(t))^\alpha (\eta_{i,\ell})^\beta}$$

SEEMS COMPLEX BUT
- BOTTOM : NORMALIZATION FACTOR
- TOP : PRODUCT OF PHEROMONS TRAITS AND HEURISTIC INFORMATION TO POWER OF α AND β

- Where α and β are parameters controlling how important are the pheromones and the heuristic information, respectively

UPDATE OF THE PHEROMONES

- Let L_k be the length of the solution found by ant k (for $k \in \{1,\ldots,n\}$)
- We define $\Delta\tau_{i,j}^k$ as $\frac{1}{L_k}$ if (i,j) is part of the solution of ant k and 0 otherwise

THE PHEROME REMAIN LONGER DEPENDING ON THE LENGTH

- Then we update the pheromone trails as

$$\tau_{i,j}(t+1) = (1-\rho)\tau_{i,j}(t) + \sum_{k=1}^{n} \Delta\tau_{i,j}^k$$

- Where $\rho \in [0,1]$ is a parameter used to decide how fast the pheromones evaporates

EFFECT OF THE UPDATE

- Edges that are part of better solutions (shortest length) receive more pheromones
- Edges that are part of more solutions receives more pheromones
- More pheromones make the edge more desirable
- If an edge is more desirable its probability of being selected increases