Swarm Intelligence: COSC 3P71

Ant-based Algorithms

Reference: no assigned text, slides based on various research papers & online material

Swarm Intelligence

- Originated from the study of colonies, swarms of social organism
- Studies of the social behaviour of organisms (individuals) in swarms lead to the design of very efficient algorithms
  - the foraging behaviour of ants resulted in ant colony optimization algorithms
  - simulation studies of the graceful, but unpredictable, choreography of bird flocks results in particle swarm optimization (PSO)
- A very young field in computer science, with much potential
  - ..lots of possibilities to discover!

Ant-based algorithms

- Ant-based systems are a population-based stochastic search methods.
  - Sound familiar- it is similar to genetic algorithms

- There is a population of ants, with each ant finding a solution and then communicating with the other ants, *how*?

Ant-based algorithms

- Ant-based systems are a population-based stochastic search methods.
  - Sound familiar- it is similar to genetic algorithms

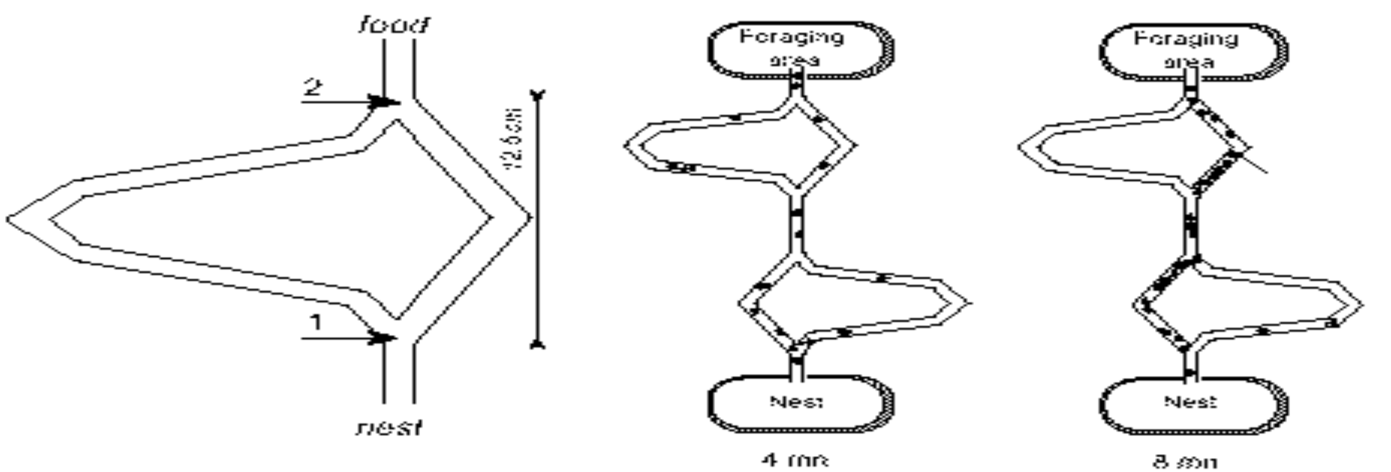
- There is a population of ants, with each ant finding a solution and then communicating with the other ants, *how?*

Ant System

- Ant System
  - Developed by Marco Dorigo, 1991
  - Modeled after real life ant colonies, based on results of experiment by Goss
  - Exhibit efficient ways to solve problems

Ant System

- Experiment by Goss et al ’89
  - Ants started at nest
  - Food placed some distance away
  - Paths of different length between nest and food
  - Ants found shortest path!

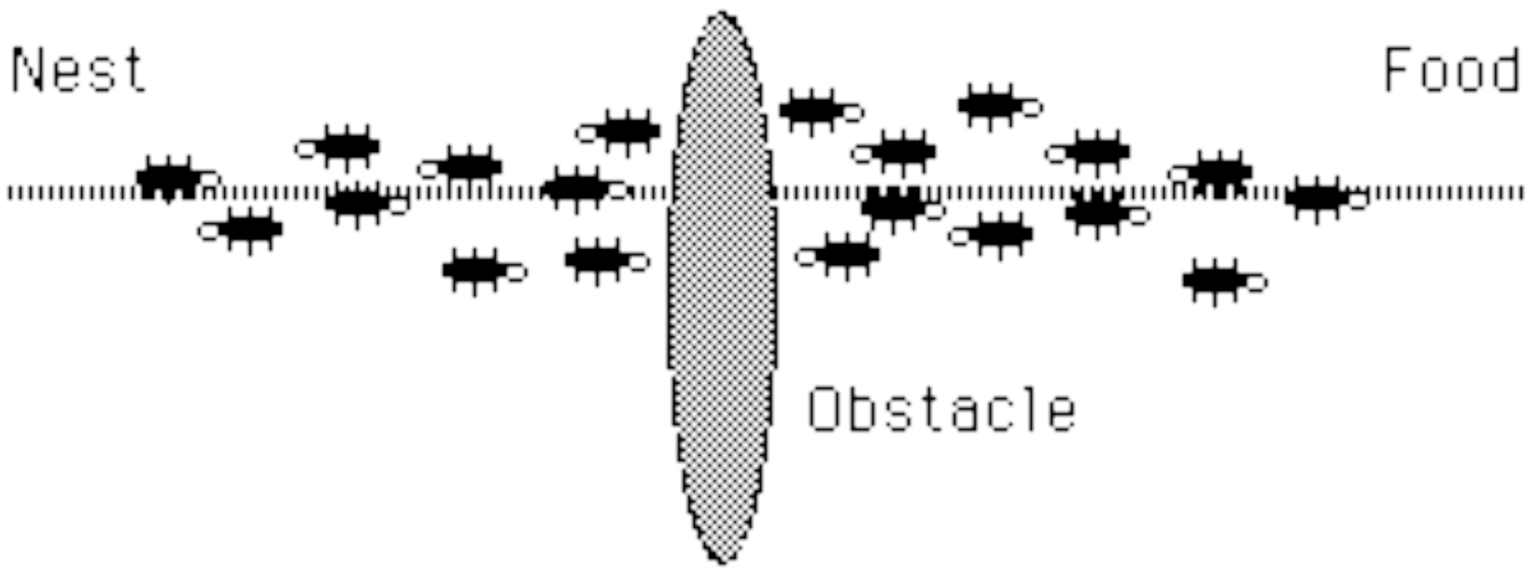


Ants

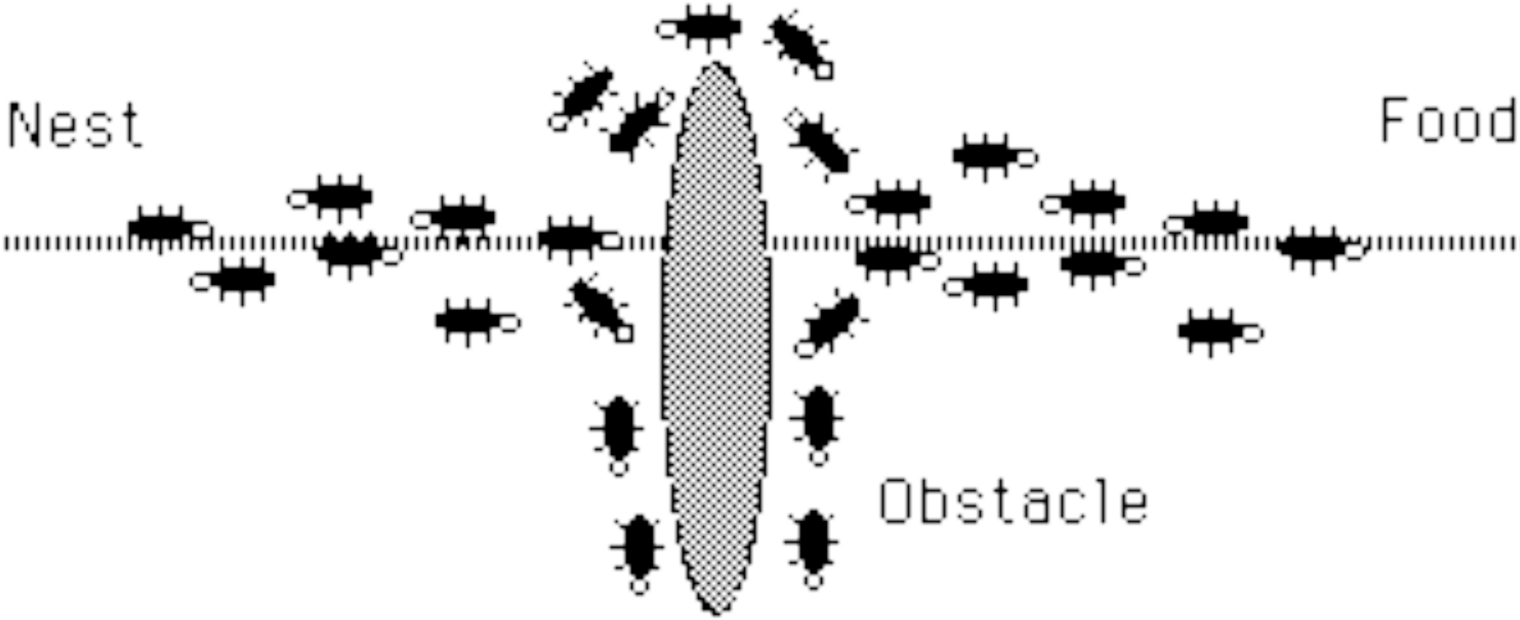
- Biological Inspiration
- Trail between nest and food
- Communicate via pheromone

Real Ant Optimization

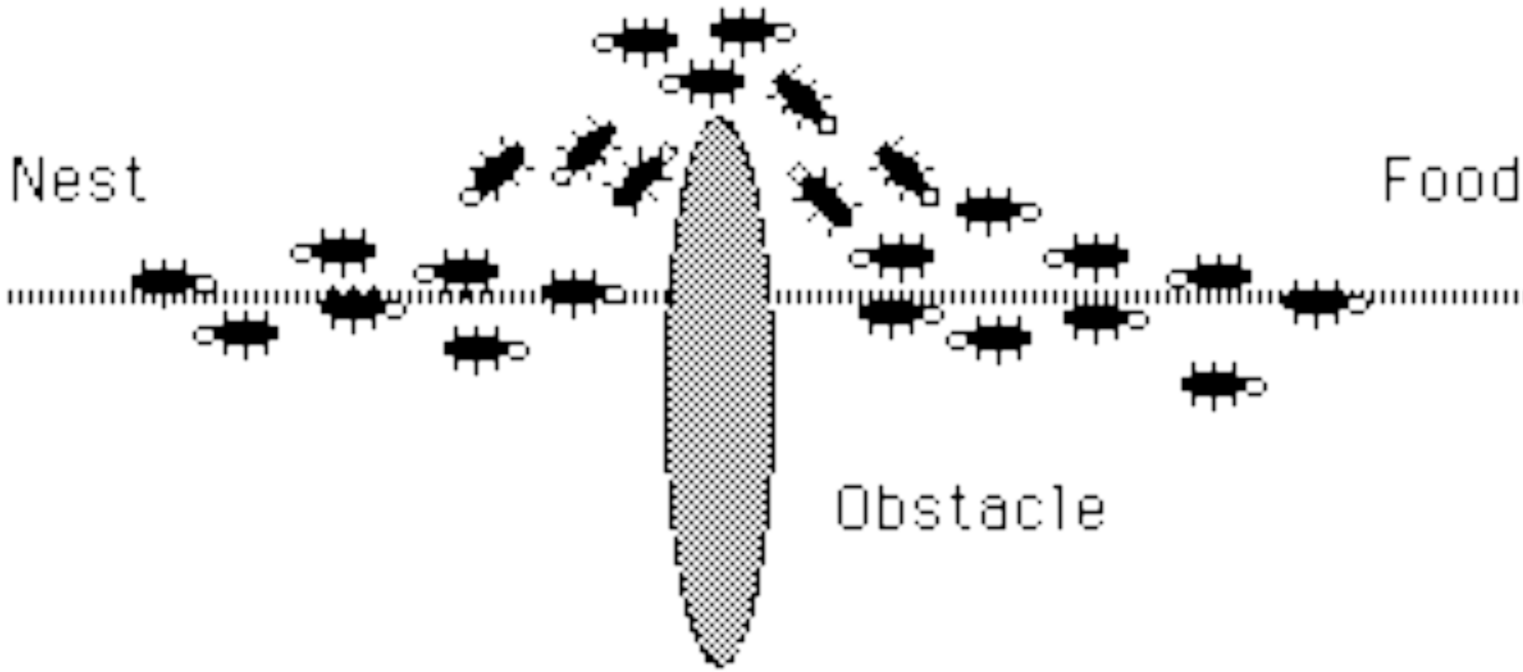
Real Ant Optimization



Real Ant Optimization



Real Ant Optimization



Ant System

- When ants travel they mark their path with substance called pheromone
  - Attracts other ants

- When an ant reaches a fork in its path the direction it follows is based on amount of pheromone it detects
  - Decision probabilistically made

- This causes positive feedback situation
  (i.e. choosing a path increases the probability it will be chosen)

Ant algorithms

- We need to explore the search space, rather than simply mapping a route

- Ants should be allowed to explore paths and follow the best paths with some probability in proportion to the intensity of the pheromone on a given edge/trail.

- If the ants simply follow the path with the highest amount of pheromone on it, our search will quickly likely settle on a very sub-optimal solution

Ant algorithms

- The probability of an ant following a certain route is a function of both the pheromone intensity, and of what the ant can see.

- Furthermore, the pheromone trail must not build unbounded, hence evaporation is needed.

Ant System

- Group of ants start at home/nest
  - An initial amount of pheromone already placed on edges
- Travel on edges
  - Edges contain pheromone amount
- Visit nodes
  - Probability of selecting next node
    - Based on distance between nodes and pheromone amount

Ant System

- Ants travel from node to node until end
  - decision based on transition probability (called state transition)

- Once all ants finished
  - Solutions compared
  - Pheromone evaporation applied to all edges
  - Pheromone increased along each edge of best/each ant's path
    - **Original ant system**: at each iteration, the pheromone values are updated by all the ants that have build a solution in the iteration itself.
  - Daemon activities can be run (like local search)

- Redo until termination criteria met

Requirements

- Problem being solved must be in graphical format
  - Since algorithm is based on path finding behavior
  - Not always apparent
- Must be finite (must have a start and end)

Algorithm

- While (termination not satisfied)
  - create ants
    - Starting point depends on problem constraints
    - Initial pheromone is > 0, but very small
  - Find solutions
  - Pheromone update
  - Daemon activities {optional}

Algorithm

- While (termination not satisfied)
  - create ants
  - Find solutions
    - Transition probability:

$$P_{ij}(t) = \frac{\tau_{ij}(t)^{\alpha}\left(\frac{1}{d_{ij}}\right)^{\beta}}{\sum_{j \in \text{allowed nodes}} \tau_{ij}(t)^{\alpha}\left(\frac{1}{d_{ij}}\right)^{\beta}}$$

Quantity of pheromone

Heuristic distance

α,β constants

  - Pheromone update
  - Daemon activities {optional}

Algorithm

- While (termination not satisfied)
  - create ants
  - Find solutions
  - Pheromone update

Evaporation rate

Pheromone laid by each ant that uses edge (i,j)

$$\boxed{\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{\substack{k \in Colony\ that \\ used\ edge\ (i,j)}} \frac{Q}{L_k}}$$

  - Daemon activities {optional}

Algorithm

- While (termination not satisfied)
  - create ants
  - Find solutions
  - Pheromone evaporation
  - Daemon activities {optional}
    - Usually, a local search algorithm is employed here
    - May also appear after “find solutions” stage

Ant System

- State Transition

$$P_{ij}(t) = \frac{\tau_{ij}(t)^{\alpha}\left(\frac{1}{d_{ij}}\right)^{\beta}}{\sum_{j \in \text{allowed nodes}} \tau_{ij}(t)^{\alpha}\left(\frac{1}{d_{ij}}\right)^{\beta}}$$

- Pheromone Evaporation

$$\tau_{ij}(t+n) = \rho\tau_{ij} + \Delta\tau_{ij}(t+n)$$

- Pheromone Update

$$\Delta\tau_{ij}(t+n) = \begin{Bmatrix} \frac{Q}{f_{evaluation}(best\_so\_far)} \\ 0, otherwise \end{Bmatrix}$$

- Where,

τ_{ij} – quantity of pheromone on edge from nodes i to j
d_{ij} – distance between nodes i and j
p_{ij} – probability to travel from node i to j
ρ – evaporation coefficient
Q – constant quantity of pheromone to deposit
α, β – user defined parameters

Problems

- Ant System tends to converge quickly
- This means that its exploitation of the best solution found is too high, it should be exploring solution space more
  - **Pheromone evaporation/update rule (better rule may exist)**
    - **what is the evaporation rate?**
- Led to extensions of the ant system
  - *MAX-MIN* Ant system
  - Ant colony system
  - Foot-Stepping
  - Others (will not be discussed)

MAX-MIN Ant System

- Developed by Stutzle and Hoos 2000
- An improvement over the original Ant System to allow for more exploration
  - Introduced use of tending to global best from iteration best solution over time
    - i.e., only the best ant updates the pheromone trails, and that,
    - the value of the pheromone is bound, upper and lower bound imposed
      - bounds on pheromone that are dependant on solution quality
  - Bounds set empirically…some guidelines available
- Has led to good results for many types of problems

Ant Colony System

- Most popular contribution of ACS is
- introduction of a local pheromone update in addition to the pheromone update performed at the end of the construction process (known as offline pheromone update)
- Local pheromone update is performed by all ants after each construction step
- Each ant applies it only to the last edge traversed:

$$\tau_{ij} = (1 - \varphi).\tau_{ij} + \varphi.\tau_0$$

where $\varphi \in (0,1)$ is the pheromone decay coefficient

Ant Colony System

- The offline pheromone update, similary to *MAX-MIN* is applied to the end of each iteration, by only one ant, which can either be the ***iteration-best*****,** or ***best-so-far***.
- The update rule is slightly different from *MAX-MIN* though (see paper)

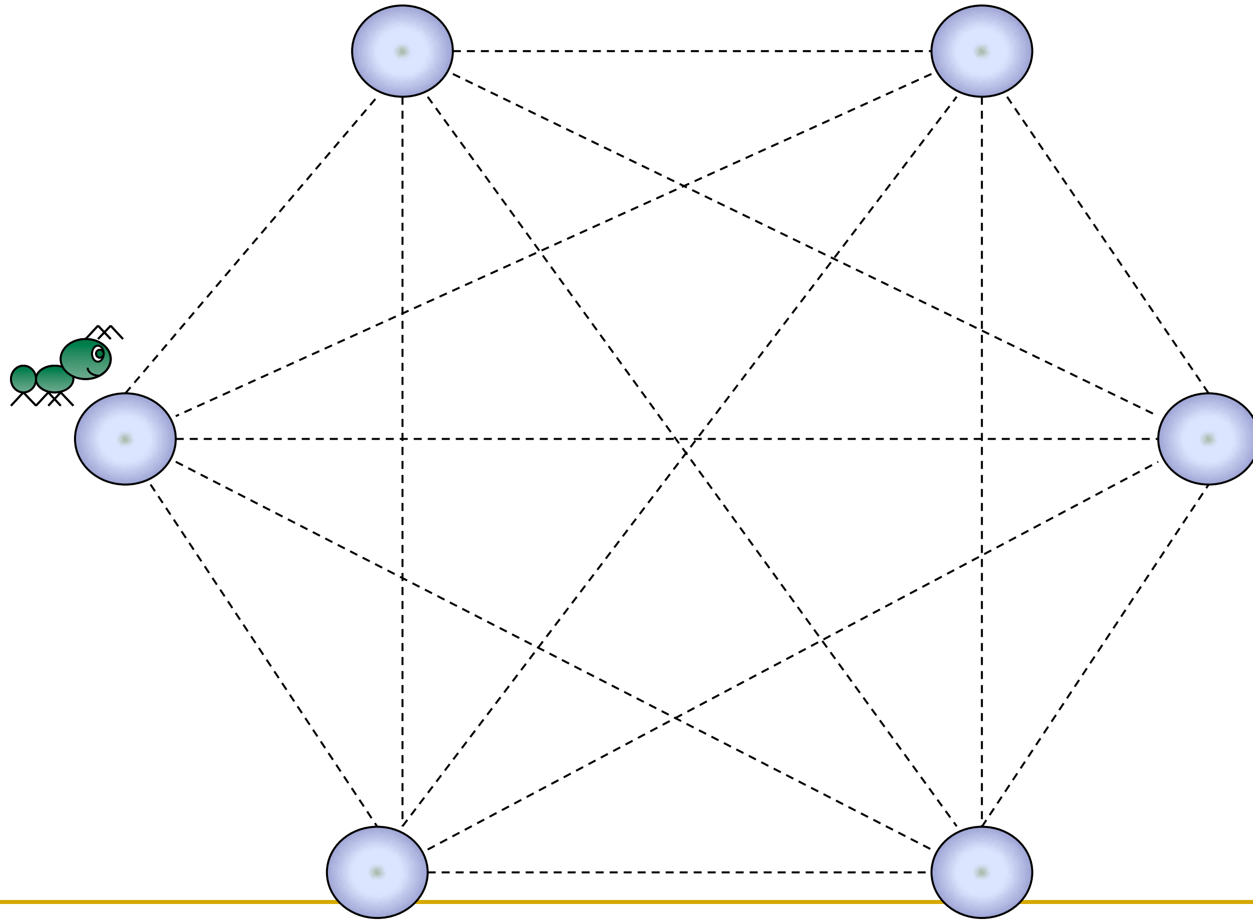
- Another important difference between ACS and Ant System is in the decision rule used by the ants during the construction process by using the so called pseudo random proportional rule.

ACO for TSP

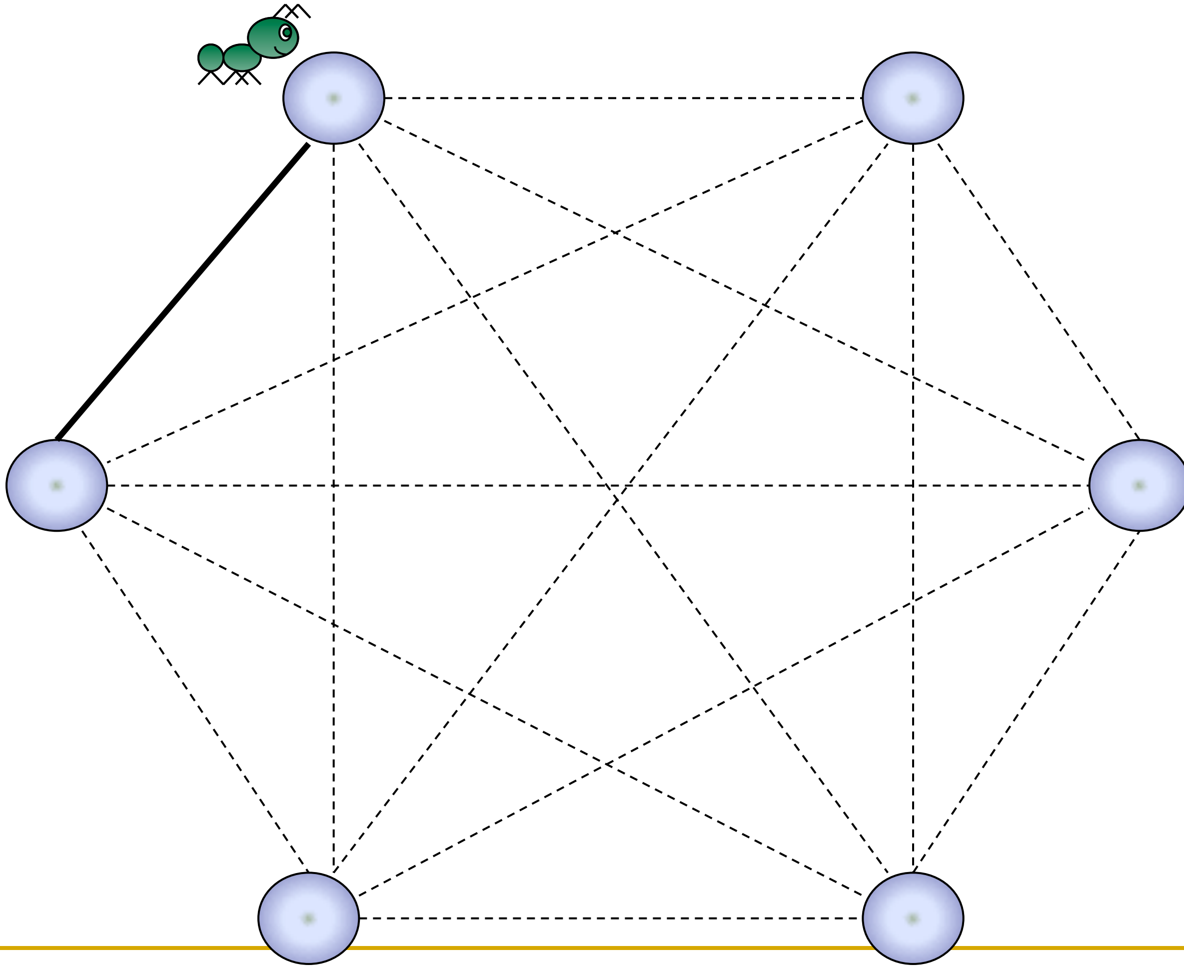
- **Input**: set of cities given, and distance between each city is known
- **Goal**: Find the shortest tour that allows each city to be visited exactly once.
- ACO algorithm

  set parameters, initiliaze pheromone trails
  while termination condition not met do
      ConstructAntSolution
      ApplyLocal search (optional)
      UpdatePheromones
  EndWhile

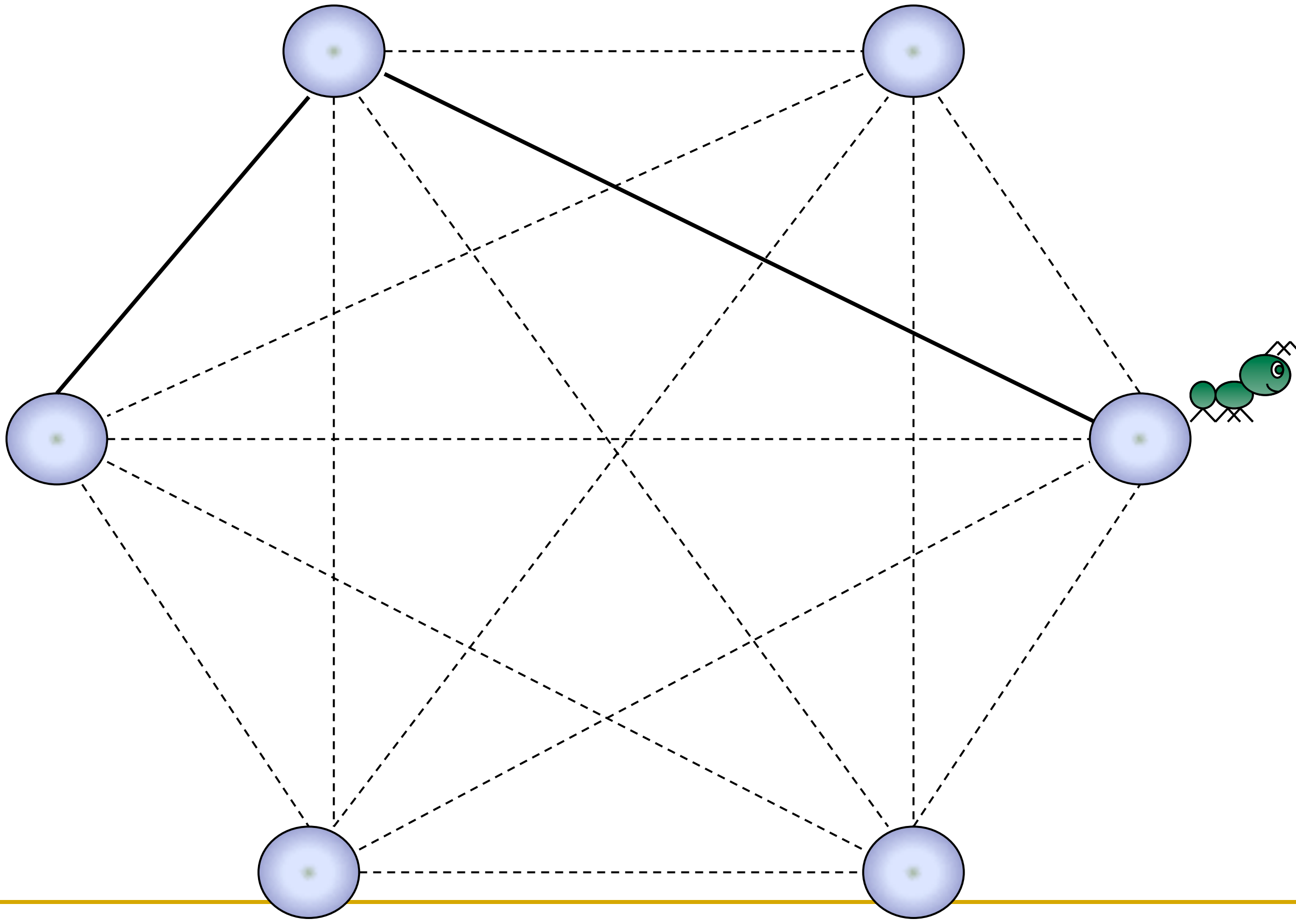
ACO for the TSP



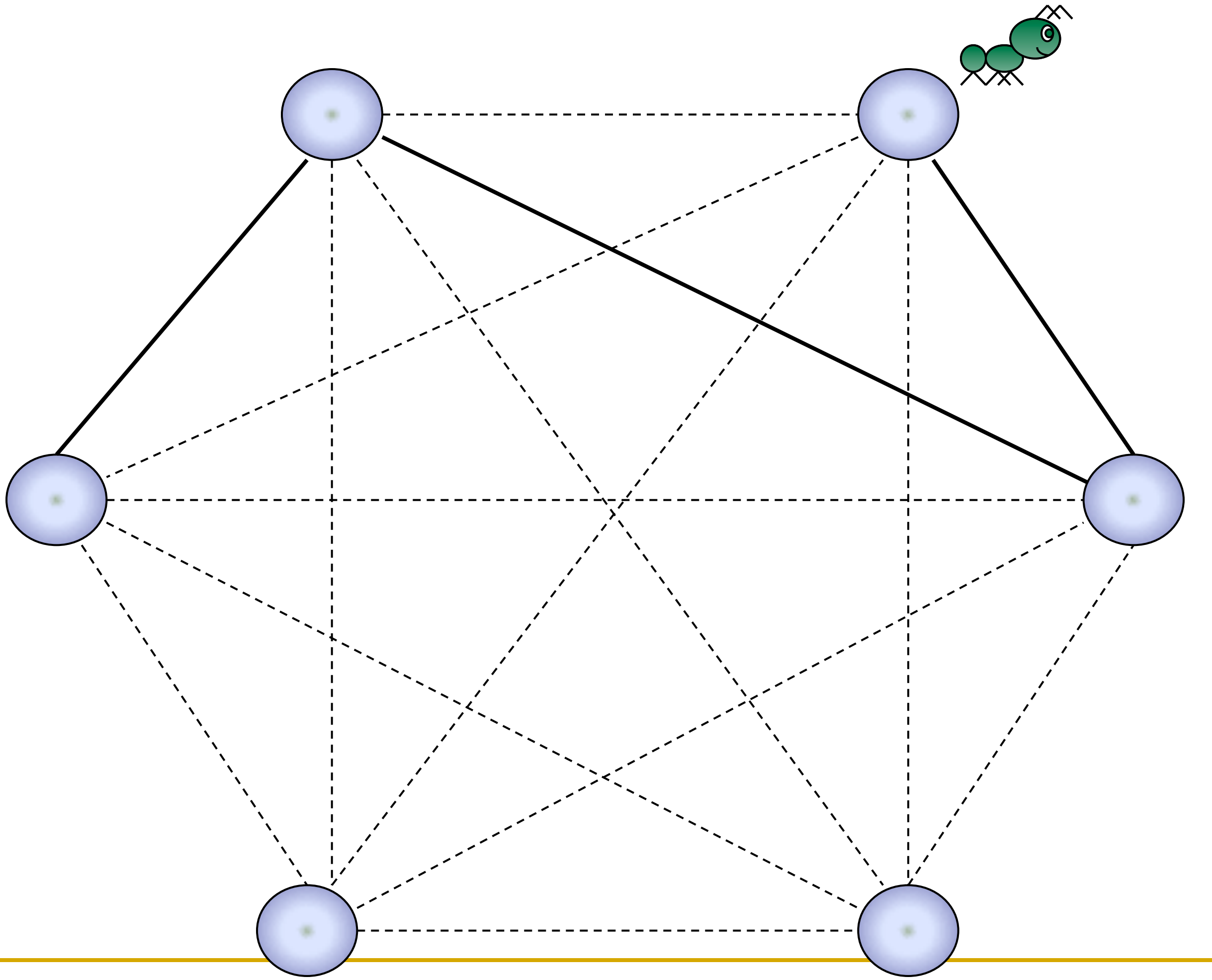
ACO-TSP



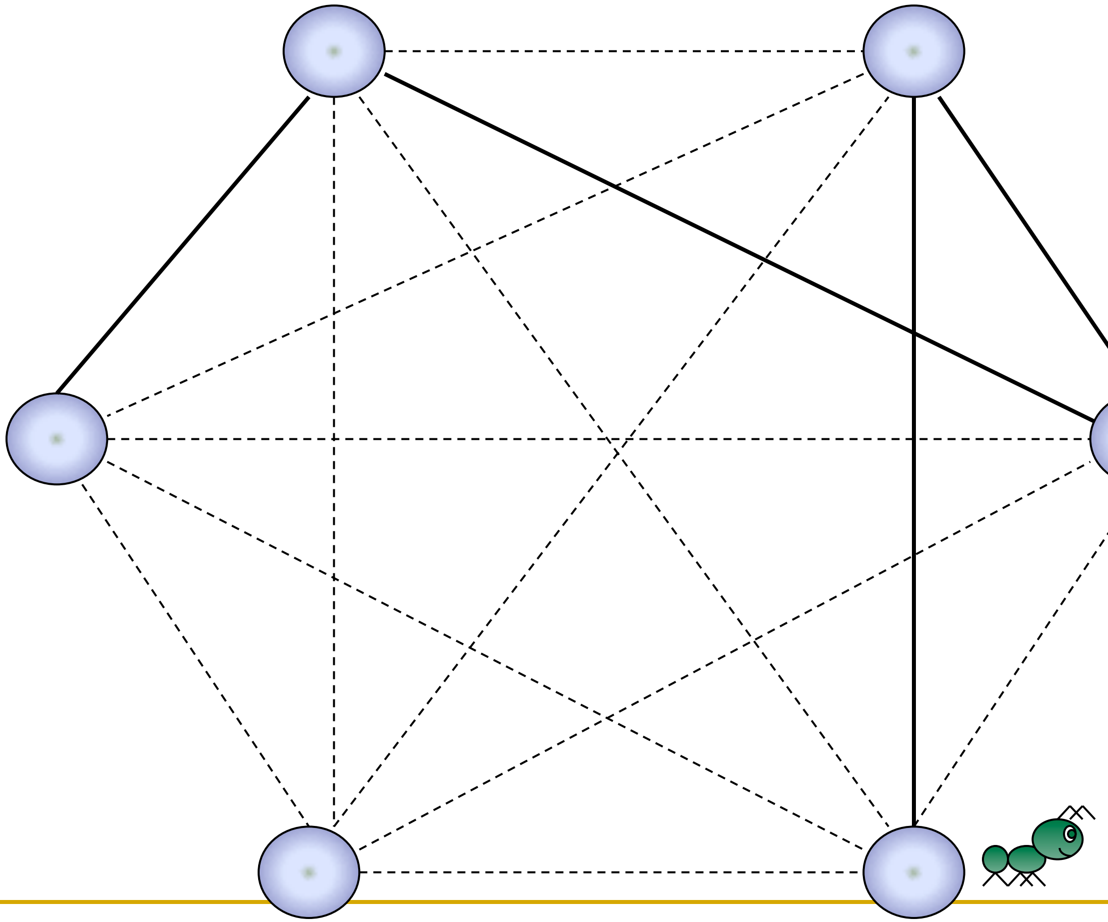
ACO-TSP



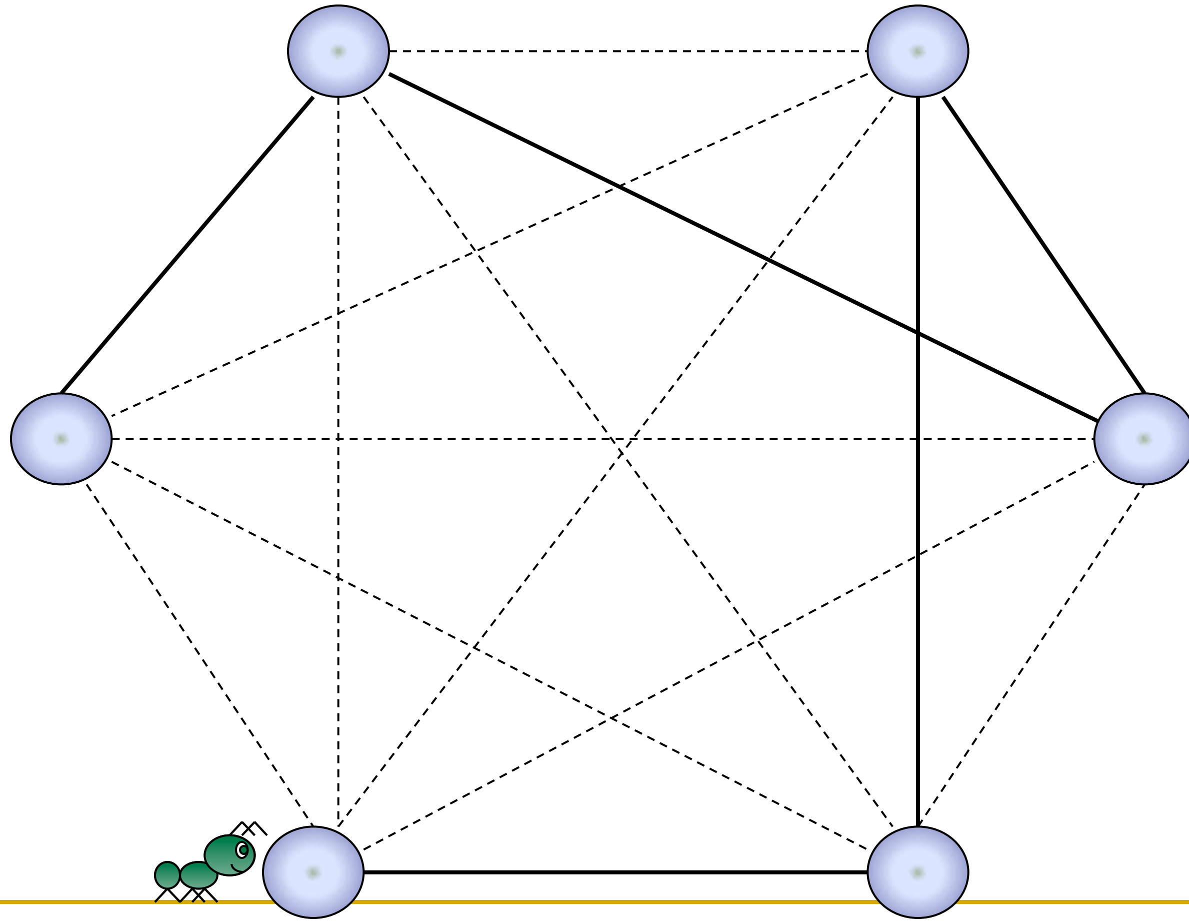
ACO-TSP



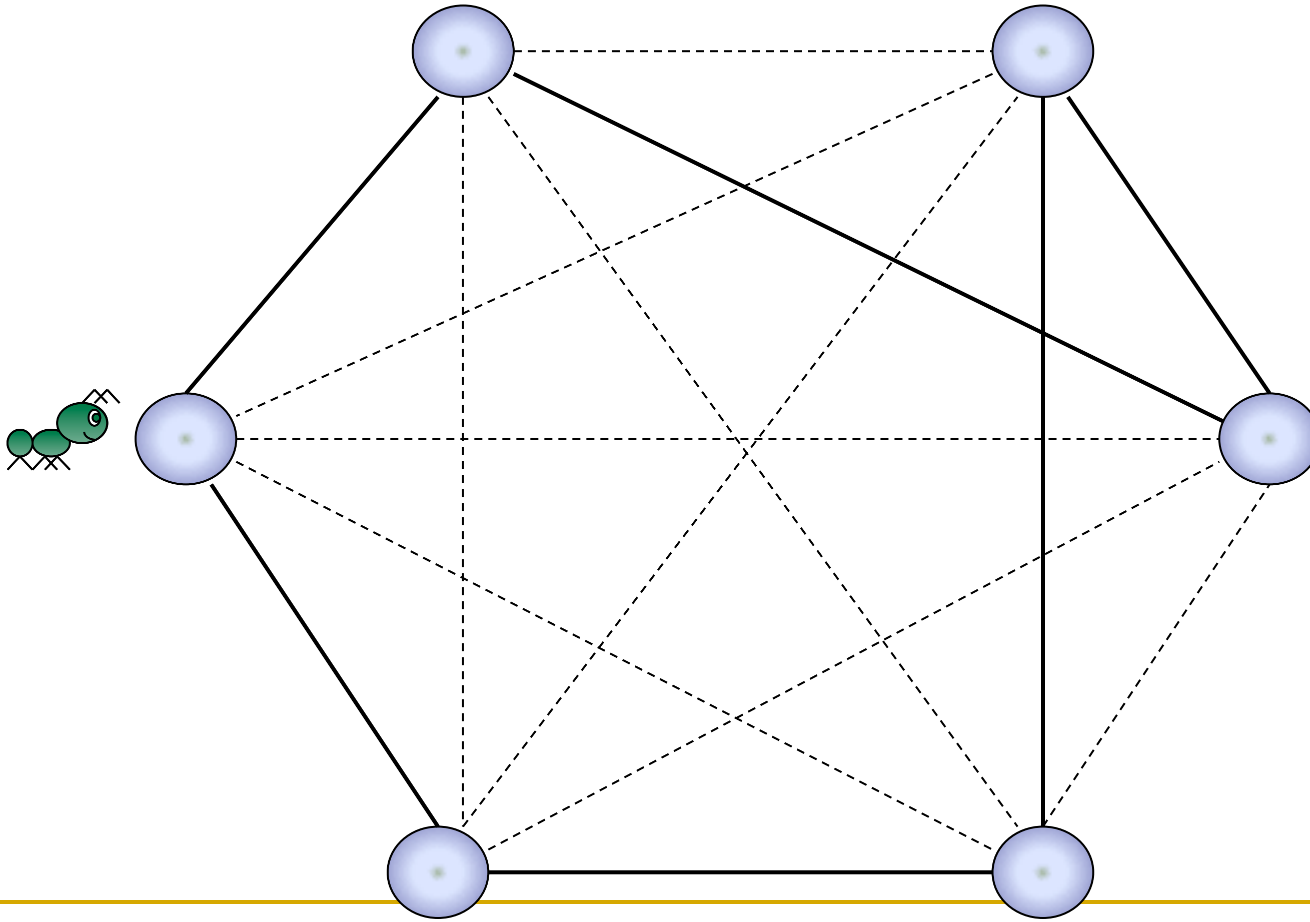
ACO-TSP



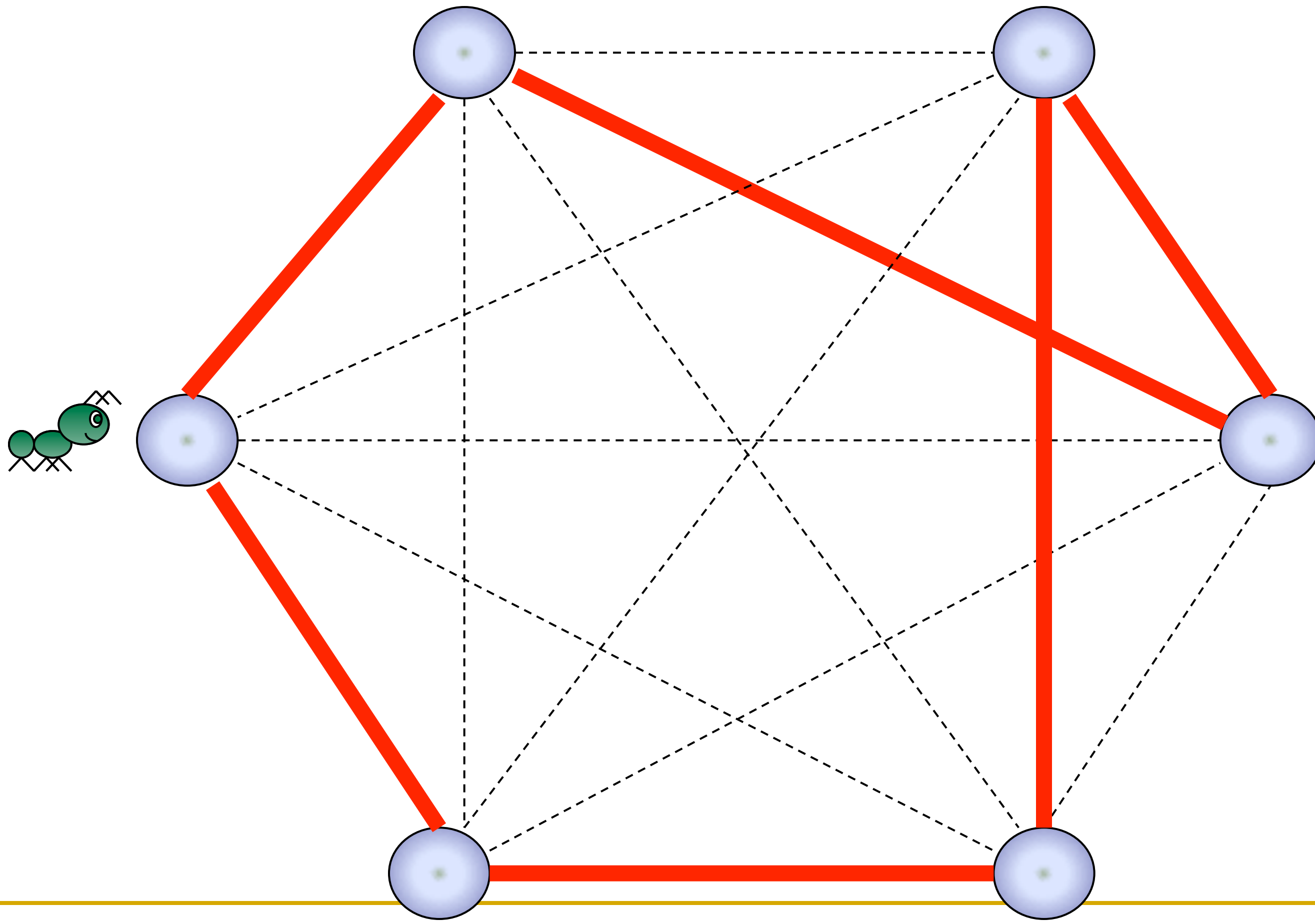
ACO-TSP



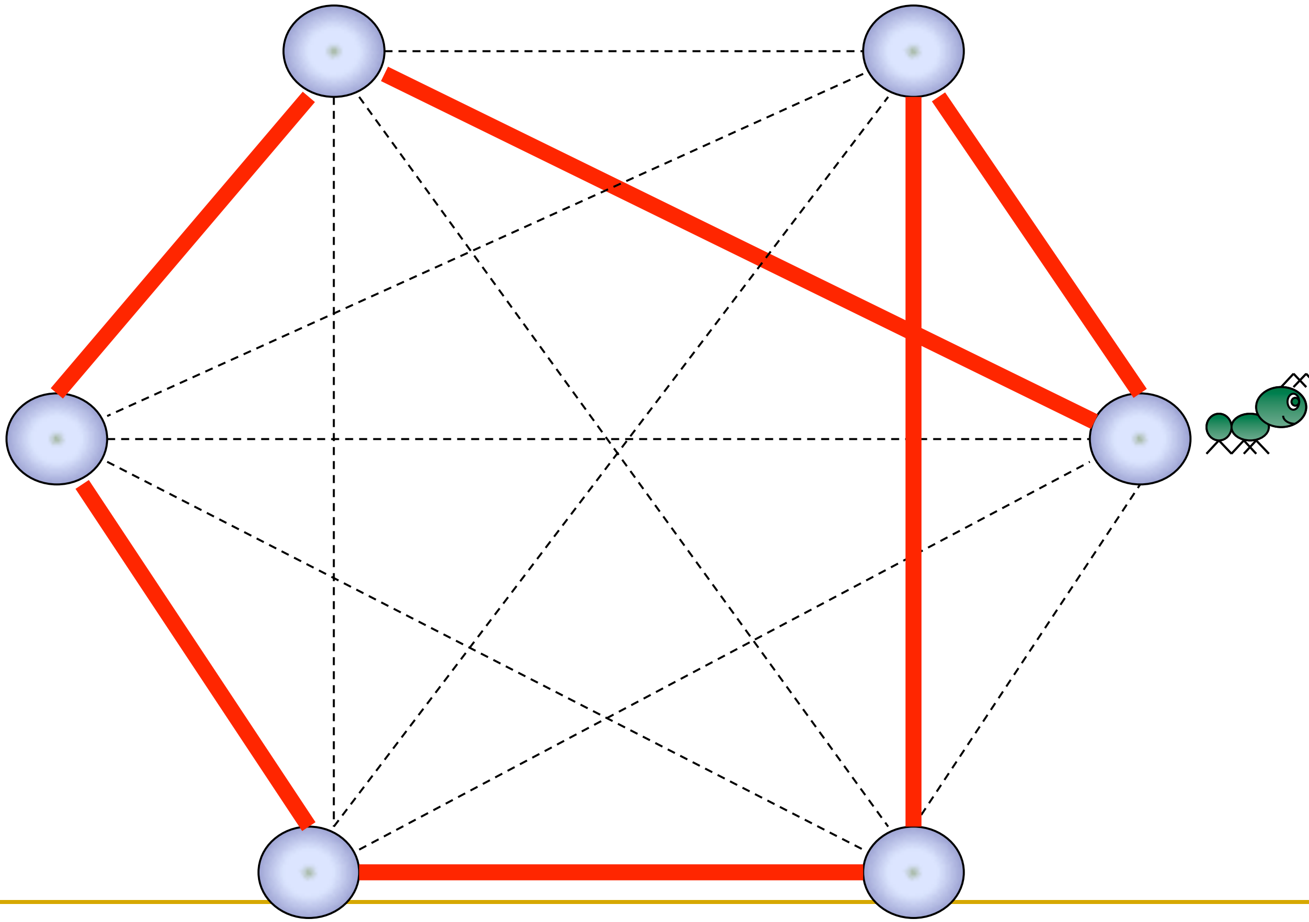
ACO-TSP



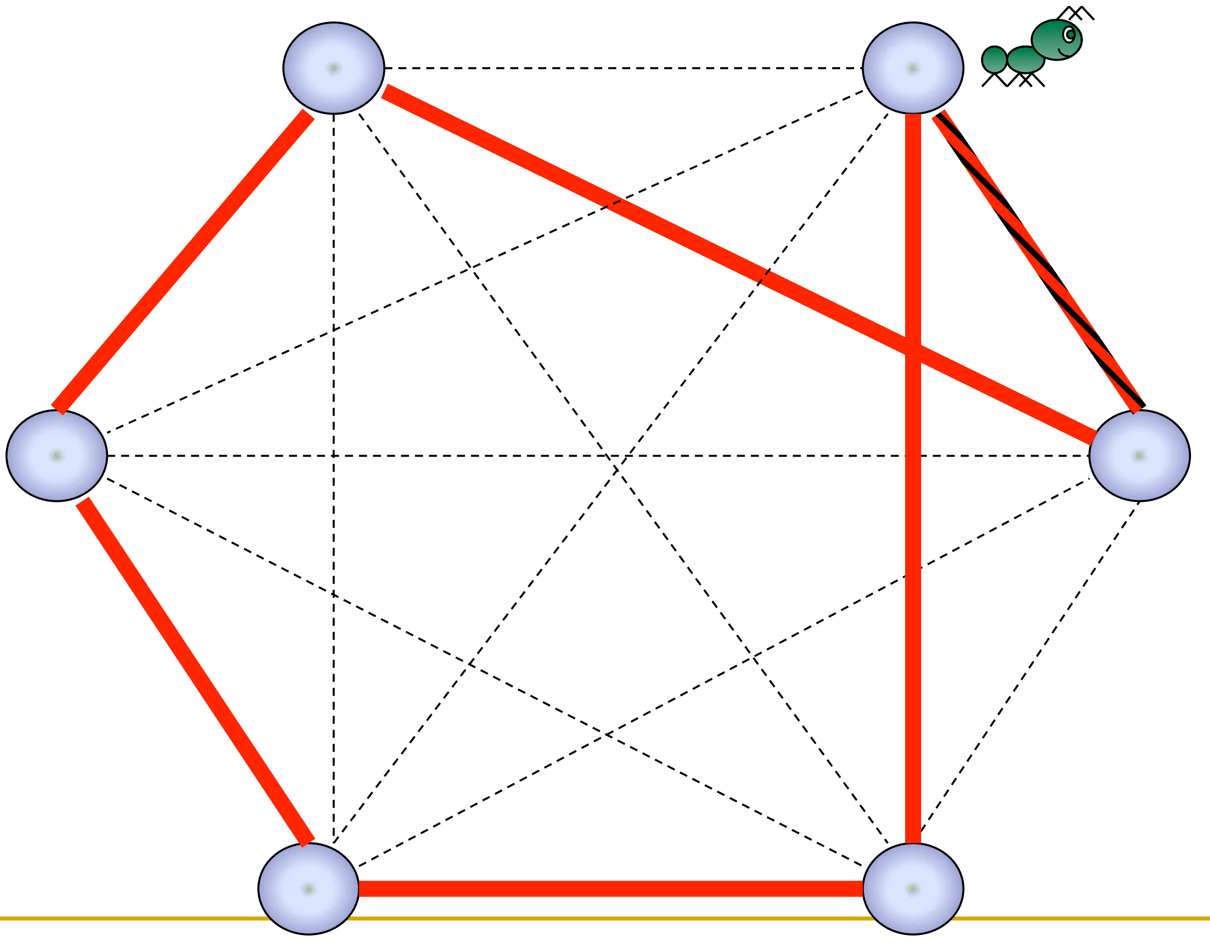
ACO-TSP



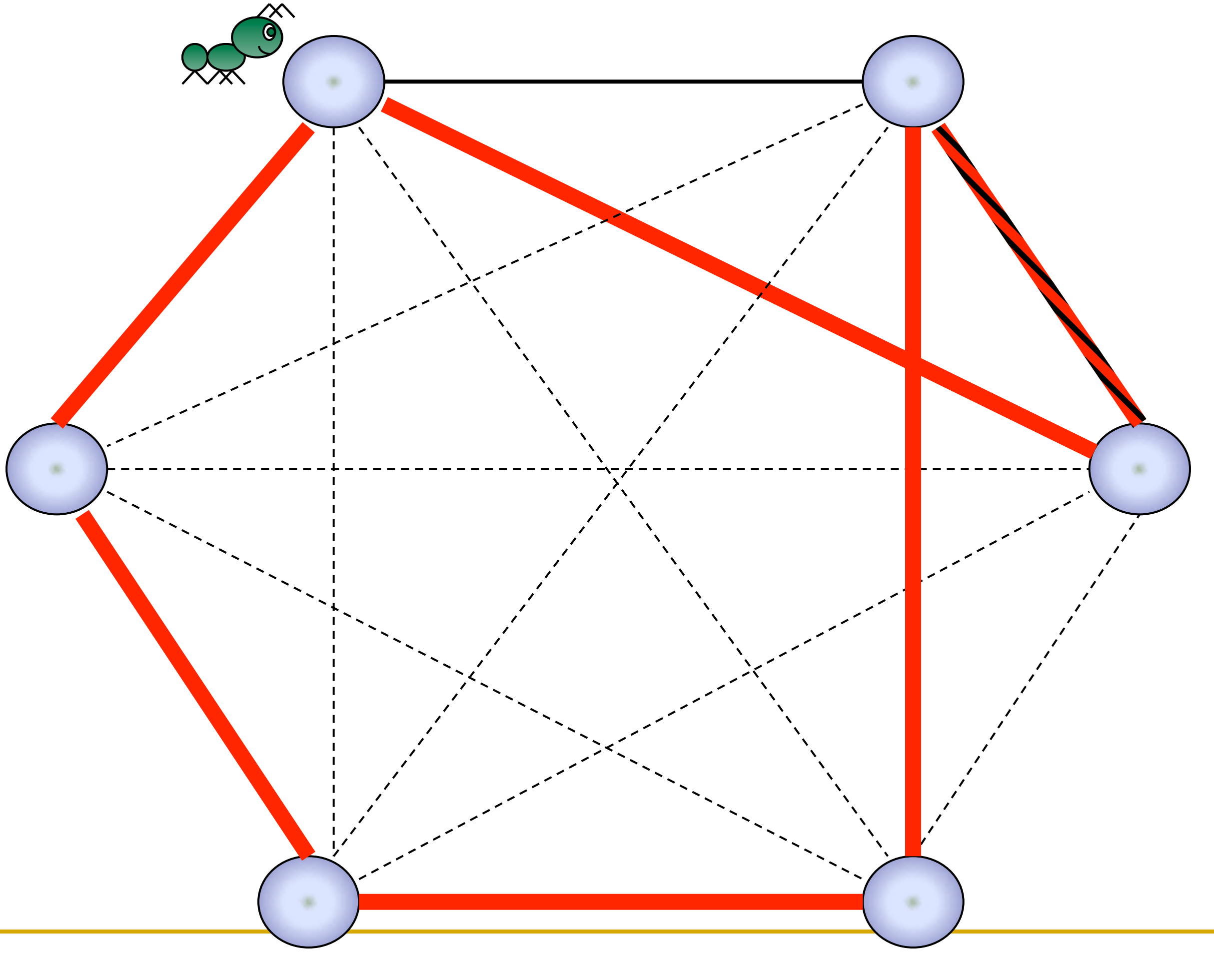
ACO-TSP



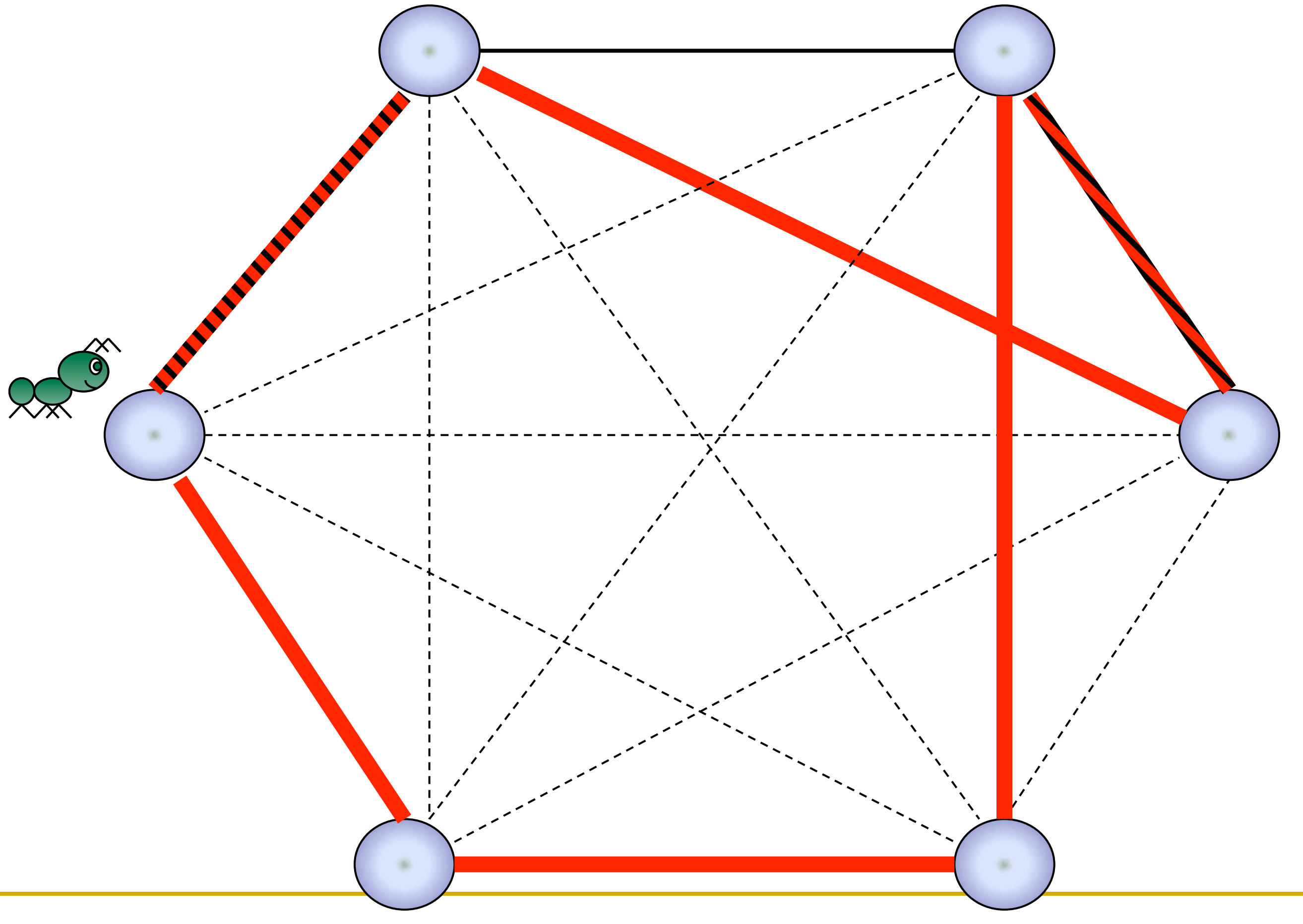
ACO-TSP



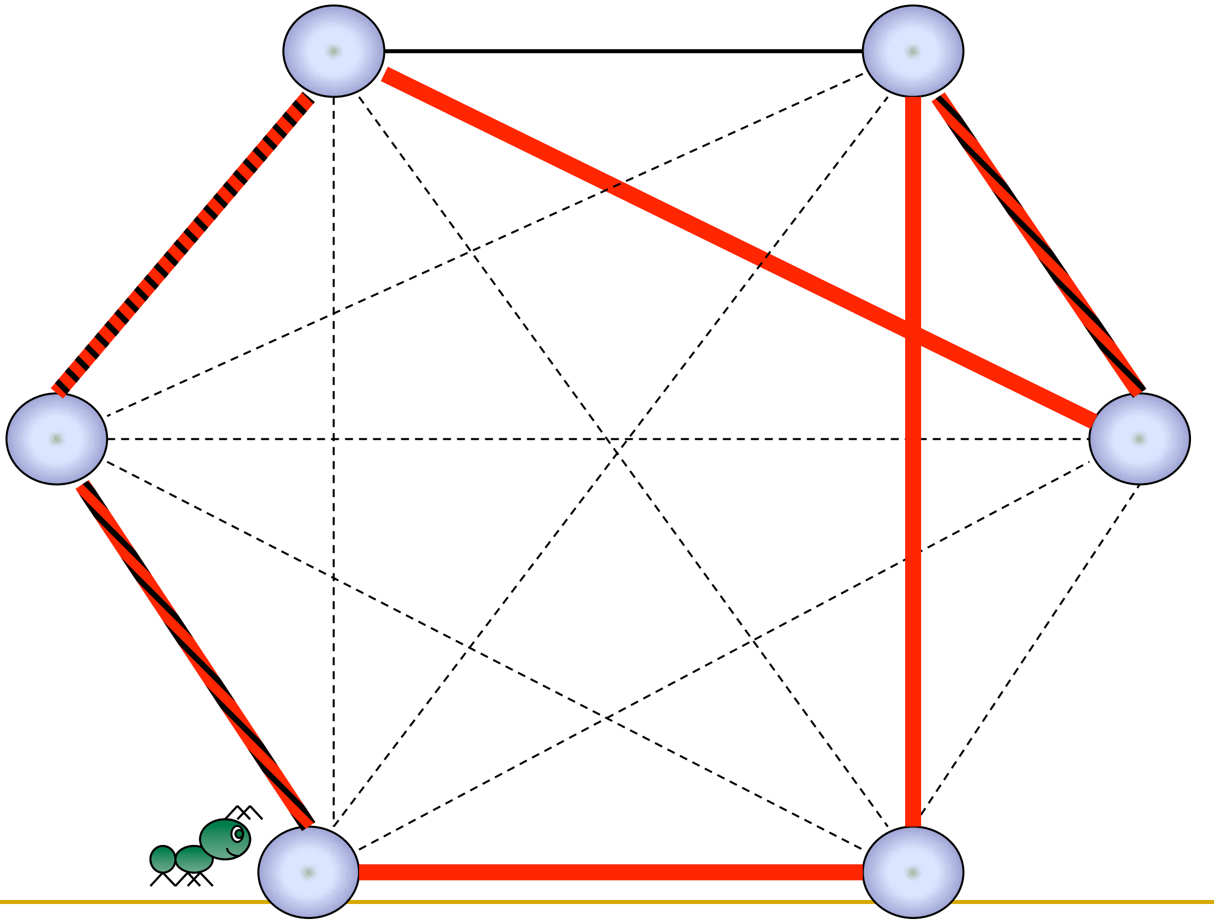
ACO-TSP



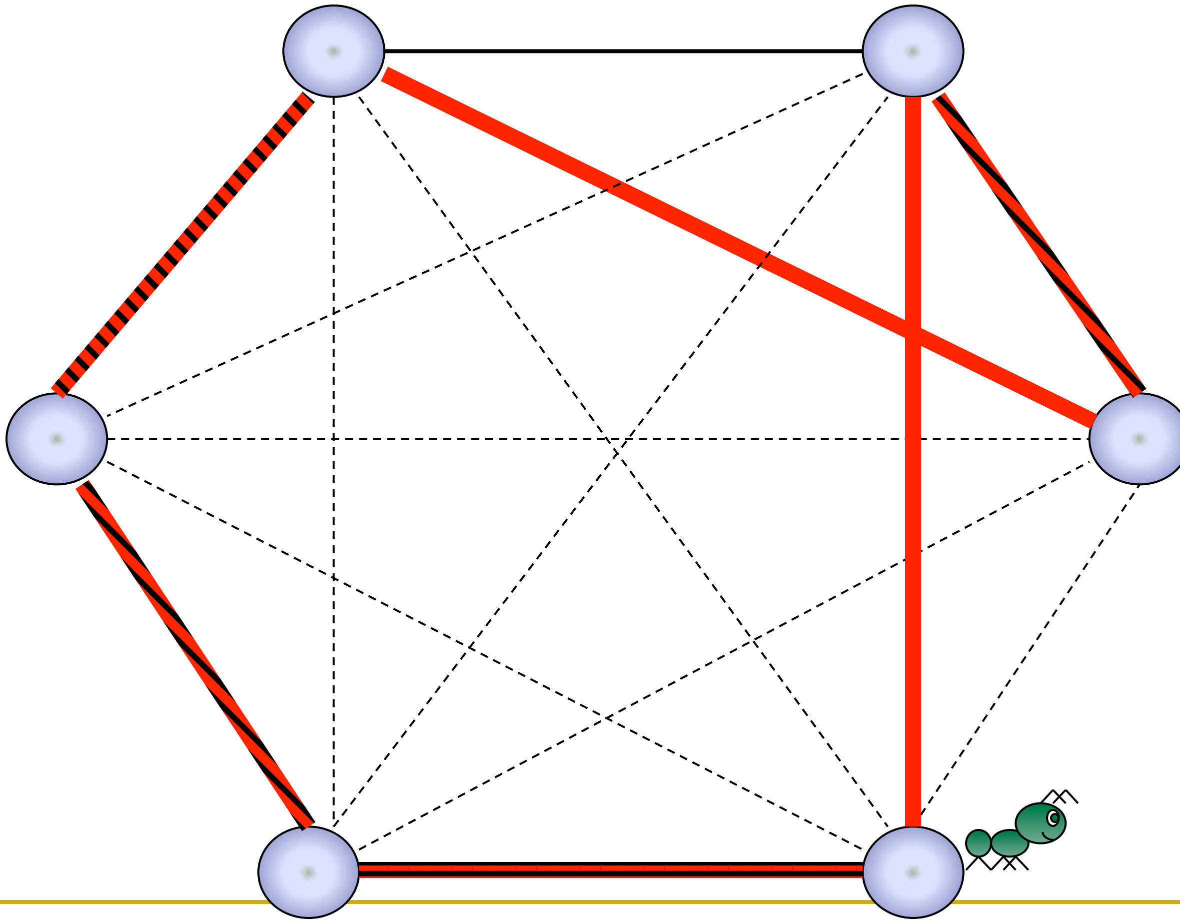
ACO-TSP



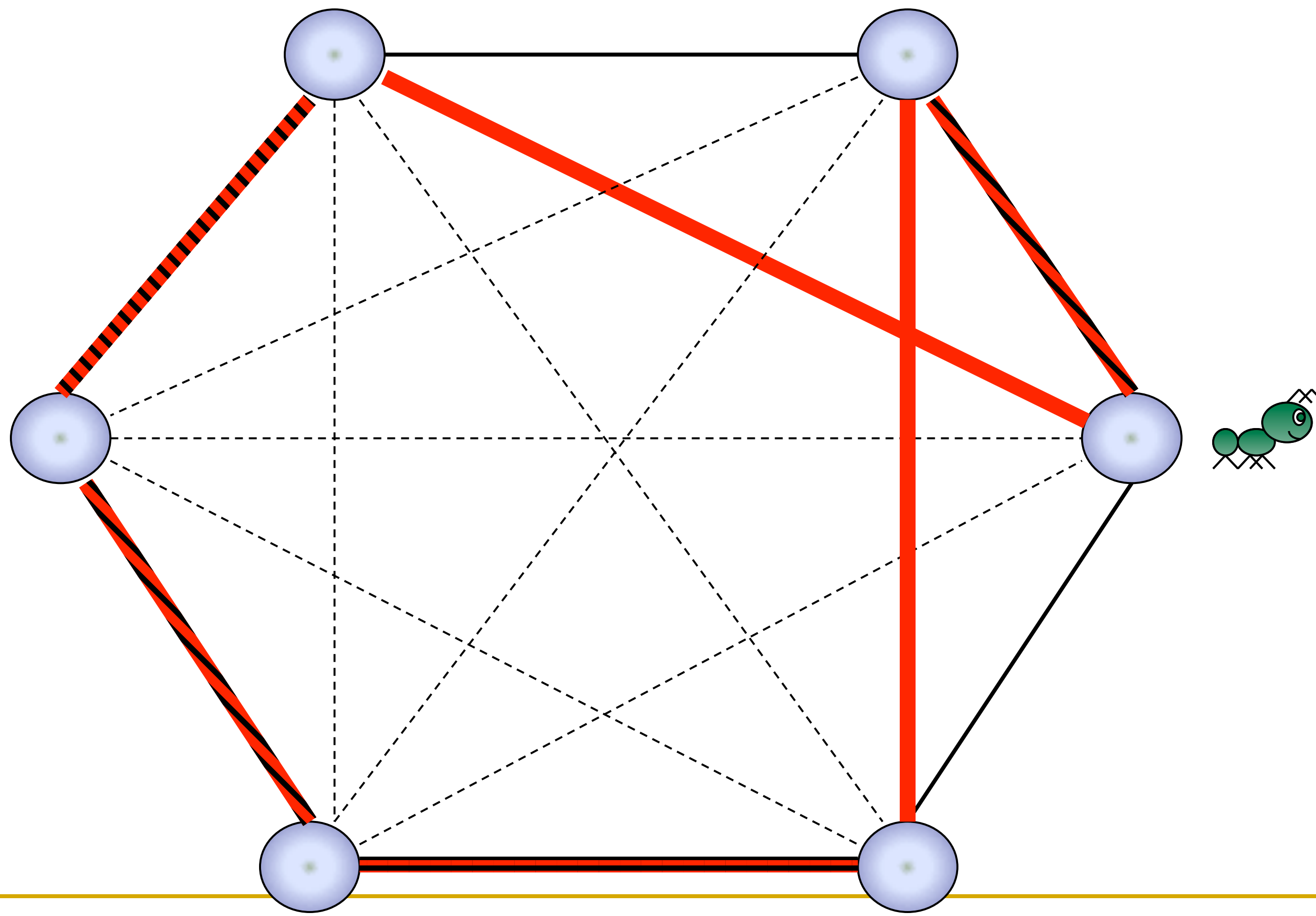
ACO-TSP



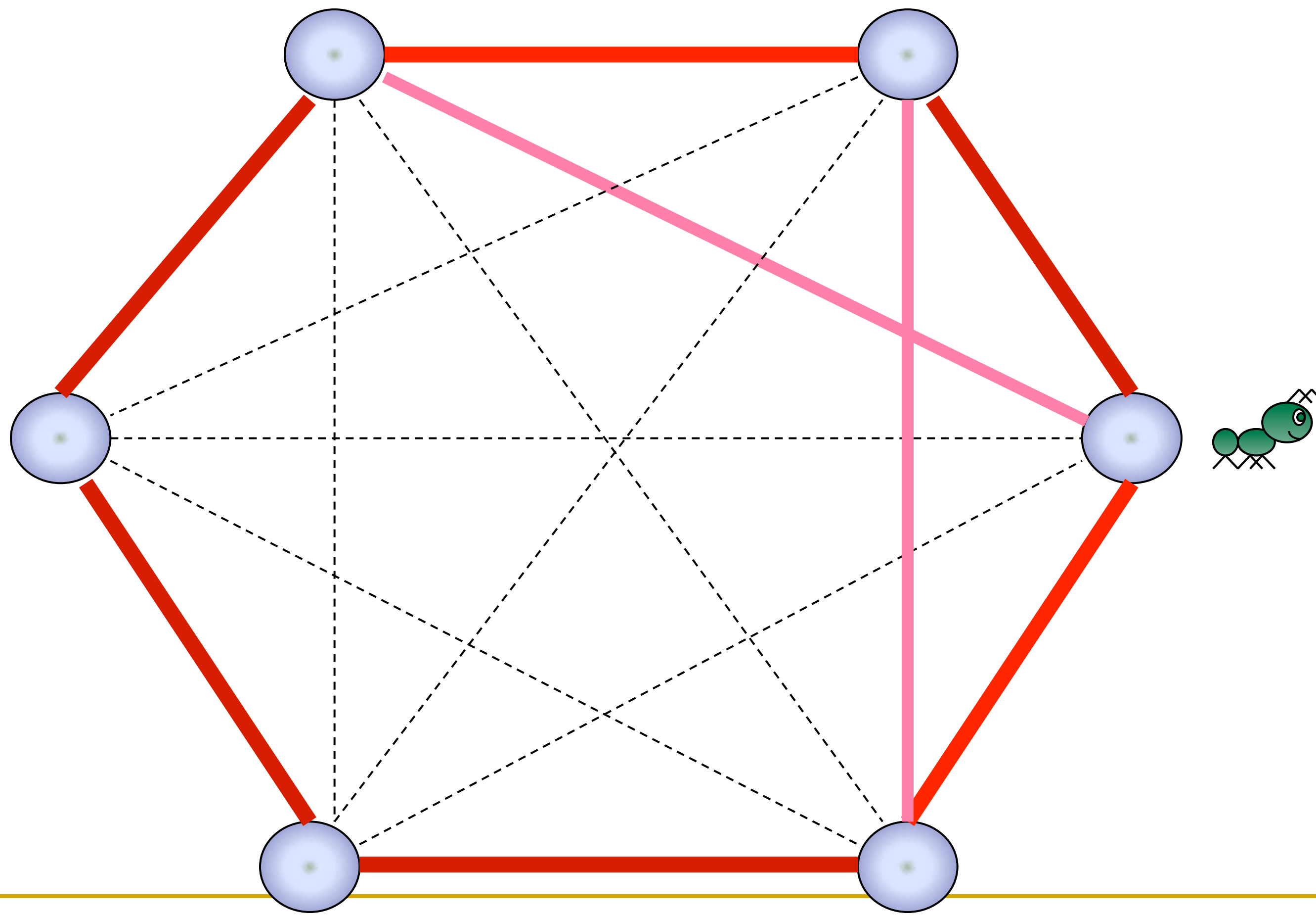
ACO-TSP



ACO-TSP



ACO-TSP



Ant Algorithms and the TSP

- At the start of the algorithm an ant is placed in each city
- Variations have been proposed by Dorigo et al.

Ant Algorithms and the TSP

- Time, t, is discrete. $t(0)$ marks the start of the algorithm. At $t+1$ every ant will have moved to a new city
- Assuming that the TSP is being represented as a fully connected graph, each edge has an *intensity of trail* on it. This represents the pheromone trail laid by the ants
- Let $T_{i,j}(t)$ represent the intensity of trail edge (i,j) at time t

Ant Algorithms and the TSP

- When an ant decides which city to move to next, it does so probabilistically, based on the distance to that city and the amount of pheromone intensity on the connecting edge

- The distance to the next city, is known as the *visibility*, n_{ij}, and is defined as $1/d_{ij}$, where, d, is the distance between cities i and j.

Ant Algorithms and the TSP

- At each time unit, i.e., constructive step, *evaporation* takes place
- The amount of evaporation, p, is a value between 0 and 1

Ant Algorithms and the TSP

- **In order to prevent ants from visiting the same city in the same tour a data structure, *Tabu list*, is maintained**

- **This prevents ants from visiting cities they have already visited**

- ***Tabu*$_k$ is defined as the list for the k^{th} ant and it holds the cities that have already been visited**

Ant Algorithms and the TSP

- After each ant tour the trail intensity on each edge is updated using the following formula

$$T_{ij}\,(t + n) = p\,.\,T_{ij}(t) + \Delta T_{ij}$$

$$\Delta T_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k} & \textit{if the kth ant uses edge}(i, j) \textit{ in its tour} \\ & (\textit{between time } t \textit{ and } t + n) \\ 0 & \textit{otherwise} \end{cases}$$

Q is a constant and L_k is the tour length of the k^{th} ant

Ant Algorithms and the TSP

- **Transition Probability**

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[T_{ij}(t)]^{\alpha} \cdot [n_{ij}]^{\beta}}{\sum k \in allowed_k \quad [T_{ik}(t)]^{\alpha} \cdot [n_{ik}]^{\beta}} & if\ j \in allowed_k \\ 0 & otherwise \end{cases}$$

where α and β are control parameters that control the relative importance of pheromone trail versus visibility

Ant Algorithms - Applications

- Marco Dorigo, who did the seminal work on ant algorithms, maintains a WWW page devoted to this subject

- **http://www.aco-metaheuristic.org**

Check thagt site for further information about ant algorithms, tutorial, software, applications and main publications.

- **and his webpage has many interesting related work**
  http://iridia.ulb.ac.be/~mdorigo/HomePageDorigo/

Ant Colony Optimization applied to Job Shop Scheduling

Mario Ventresca

and

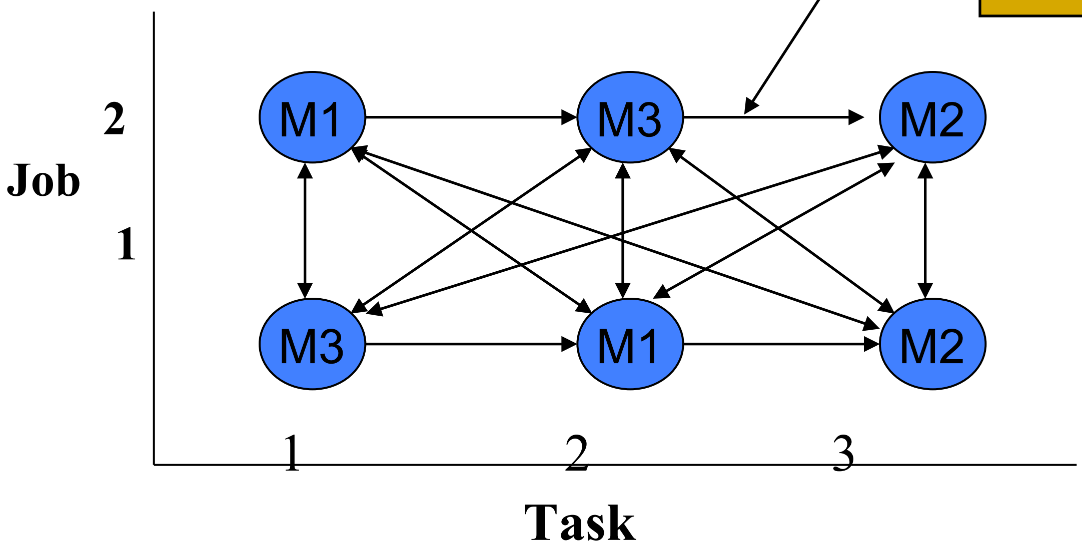
B.Ombuki-Berman

Job Shop Scheduling (JSSP)

- Difficult NP-Hard Optimization Problem
- Involves assigning jobs to machines
- Jobs broken into tasks
  - Same number of tasks per job as machines
  - Each task has processing time
- Task constraints and limitations
  - Processed according to predefined order
  - No concurrent processing
  - No-preemption

Ant System and JSSP

- Graphical Format

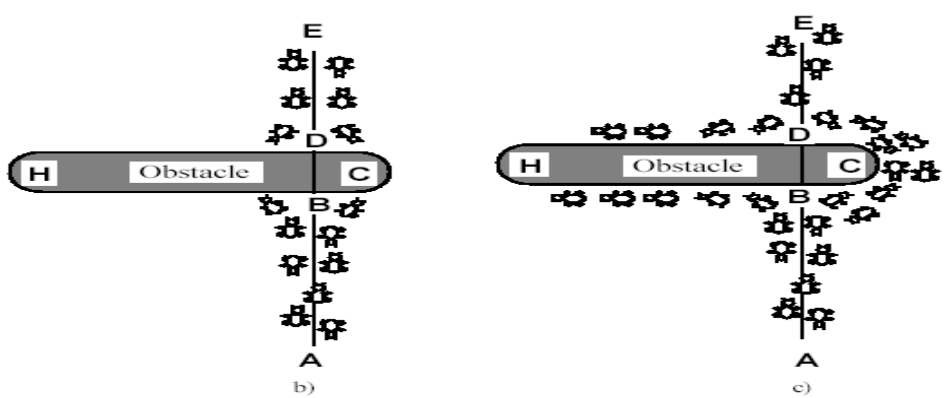


Uni-directional links maintain pre-ordering criteria, distance is processing time

NOTE!!
Once a task has been visited it cannot be returned to

Foot Stepping

- An original contribution used to enhance exploration in the Ant System
- In nature it would be like stepping on an already found ant path (being careful not to step on any)
  - Ants are forced to find a different way around, maybe (hopefully) even a better path



Foot-Stepping

- Developed in 2003 (Ventresca, Ombuki)
- Allow ants to perform further search as opposed to hybridization (via local search)
  - More like nature, uses ants to improve solutions by altering environment
- Idea
  - Alter pheromone values to force ants to search more of solution space
  - Apply when ants seem to have converged to a solution

Summary of Results

- Foot-Stepping always improved the Ant System results (unless AS found an optimal)
  - Most foot-steps caused improvements
- Foot-Stepping was compared to Max-Min
  - Both techniques seemed to yield similar results
  - Nearly all solutions were within 2-5% of each other