# Ant colony optimization for traveling salesman

Ahmed Ashraf Mohamed
2103134

## 1 Introduction

### 1.1 Definition of ACO

Ant algorithms are population-based approach to several problems, ant algorithms have been inspired by the behavior of real ant colonies, specifically their foraging behavior. The main idea of ant algorithms is the indirect communication of a colony of agents, artificial ants, based on pheromone trails.

### 1.2 Advantages

- ACO has relatively simple implementation
- Efficient for TSP and similar problems
- Guaranteed convergence

#### 1.2.1 Disadvantages

- The time of convergence is uncertain.

### 1.3 ACO applications

1. Traveling salesman
2. Quadratic assignment
3. Scheduling problems
4. Vehicle routing
5. Connection-oriented network routing
6. Connection-less network routing
7. Graph Coloring

We have chosen **Traveling salesman problem** for this report.

### 1.4 Definition of traveling salesman problem

TSP is the problem of a salesman who wants to find, starting from his home town, a shortest possible trip through a given set of customer cities and to return to his home town. More formally, it can be represented by a completed weighted graph.

## 2 The basic ACO for TSP

In the ACO algorithms ants are simple agents which, in the TSP case, construct tours by moving from city to city on the problem graph. The ants' solution construction is guided the pheromone trails. When applying ACO algorithm to the TSP, a pheromone strength $\tau_{ij}(t)$ is associated to each arc $(i, j)$, where $\tau_{ij}(t)$ is a numerical information which is modified during the run of the algorithm and $t$ is the iteration counter.

Initially, each of the $n$ ants is placed on a randomly chosen city and then iteratively apples at each city a state transition rule. An ant constructs a tour as follows. At a city $i$, the ant chooses a still unvisited city $j$ probabilistically, biased by the pheromone trail strength $\tau_{ij}(t)$ on the arc between city $i$ and city $j$ and a locally available heuristic information, which is a function of the arc length. Ants probabilistically prefer cities which are close and are connected by arcs with a high pheromone trail strength.

After all ants have constructed a tour, the pheromones are updated. This is done by first lowering the pheromone trail strengths by a constant factor and then the ants are allowed to deposit pheromone on the arcs they have visited. The trail update is done in such a form that arcs contained in shorter tours and/or visited by many ants receive a higher amount of pheromone and are therefore chosen with a higher probability in the following iteration of the algorithm. This way the amount of pheromone $\tau_{ij}(t)$ represents the learned desirability of choosing next city $j$ when an ant is at city $i$

```
1: procedure ACO ALGORITHM FOR TSPs
2:     Set parameters, initialize pheromone trails
3:     while termination condition not met do
4:         Construct Solutions
5:         Apply Local Search                                  ▷ optional
6:         Update Trails
7:     end while
8: end procedure
```

## 3 Ant system

### 3.1 Introduction

Initially, three variants of Ant System were proposed; these were called *ant-density*, *ant-quantity*, and *ant-cycle*. In *ant-density* and *ant-quantity* the ants update the pheromone directly after a move from a city to an adjacent one, in *ant-cycle* the pheromone update was only done after all the ants had constructed the tours and the amount of pheromone deposited by each ant was set to be a function of the tour quality. ant-cycle performed much better than the other two variants, hence it's the main method that is going to be used, being referred to as Ant system.

### 3.2 Steps

In AS each $n$ (artificial) ants builds a solution, i.e. tour, of the TSP, In AS no local search is applied

*Tour construction.* Initially, each ant is put on some random city. At each construction step, an ant $k$ applies a probabilistic action choice rule. particularly, the probability with which ank $k$, currently at city $i$, chooses to go to city $j$ at the $t$th iteration of the algorithm is:

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)^{\alpha}] \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}(t)]^{\alpha} \cdot [\eta_{il}]^{\beta}} \qquad \text{if } j \in \mathcal{N}_i^k \tag{1}$$

Where $\eta_{ij} = \frac{1}{d_{ij}}$ is a heuristic value, $\alpha$ and $\beta$ are two parameters which determine the relative influence of the pheromone trail and the heuristic information, and $\mathcal{N}_i^k$ is the feasible neighborhoods of ant $k$, that is the set of cities which ant $k$ has not yet visited.

*Pheromone update* AFter all ants have constructed their tours, the pheromone trails are updated. This is done by lowering the pheromone strength on **all** arcs by a constant factor and the the allow each ant to add pheromone on the arcs it has visited:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^k(t) \tag{2}$$

where $0 < \rho \leq 1$ is the pheromone trail evaporation parameter. The parameter $\rho$ is used to avoid unlimited accumulation of the pheromone trails and it enables the algorithm to "forget" previously done bad decisions. $\Delta_{ij}^k(t)$ is the amount of pheromone ant $k$ puts on teh arcs it has visited; it is defined as follows:

$$\Delta\tau_{ij}^k(t) = \begin{cases} \dfrac{1}{L^k(t)} & \text{if arc } (i,j) \text{ is used by ant } k \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Where $L^k(t)$ is the length of the $k$th ant's tour, By 3, the better the ant's tour is, the more pheromone is received by arcs belonging to the tour. In general, arcs which are used by many ants and which are contained in shorter tours will receive more pheromone and therefore are slo more likely to be chosen in future iteration of the algorithm.