Ant Colony Optimization

IT468 Natural Computing

Prof. Manish Gupta

Project Report

Mihir Raj (201101057)

Achint Mehta (201101093)

Abstract

Ant colony optimization (ACO) is a heuristic algorithm which has been proven a successful technique and applied to a number of combinatorial optimization problems and is taken as one of the high performance computing methods for Traveling salesman problem (TSP). TSP is one of the most famous combinatorial optimization (CO) problems and which has wide application background.

ACO has very good search capability for optimization problems, but it still remains a computational bottleneck that the ACO algorithm costs too much time to convergence and traps in local optima in order to find an optimal solution for TSP problems. This paper proposes an improved ant colony optimization algorithm with two highlights.

Introduction

ACO inspired by the foraging behaviour of real ant was first introduced by Dorigo and his colleagues in early 1990s and has become one of the most efficient algorithms for TSP. ACO is based on the pheromone trail laying and following behaviour of some ant species, a behaviour that was shown to allow real ant colonies to find shortest paths between their colony and food sources. These ants deposit pheromone on the ground in order to mark some favourable path that should be followed by other members of the colony.

The ants move according to the amount of pheromones, the richer the pheromone trail on a path is, the more likely it would be followed by other ants. So a shorter path has a higher amount of pheromone in probability, ants will tend to choose a shorter path.

- **The behaviour of each ant in nature**
    - Wander randomly at first, laying down a pheromone trail
    - If food is found, return to the nest laying down a pheromone trail

- If pheromone is found, with some increased probability follow the pheromone trail
- Once back at the nest, go out again in search of food

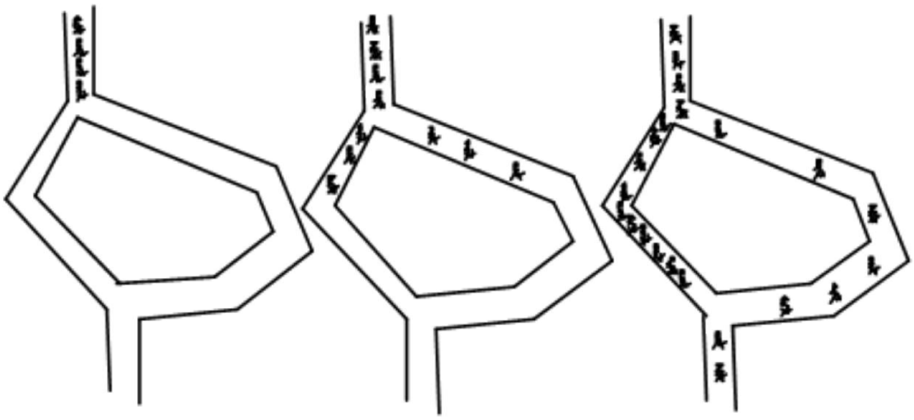
Ant Colony Optimization

ACO algorithms are stochastic search procedures. Their central component is the pheromone model, which is used to probabilistically sample the search space. Pheromone model can be derived from a model of the tackled CO problem:

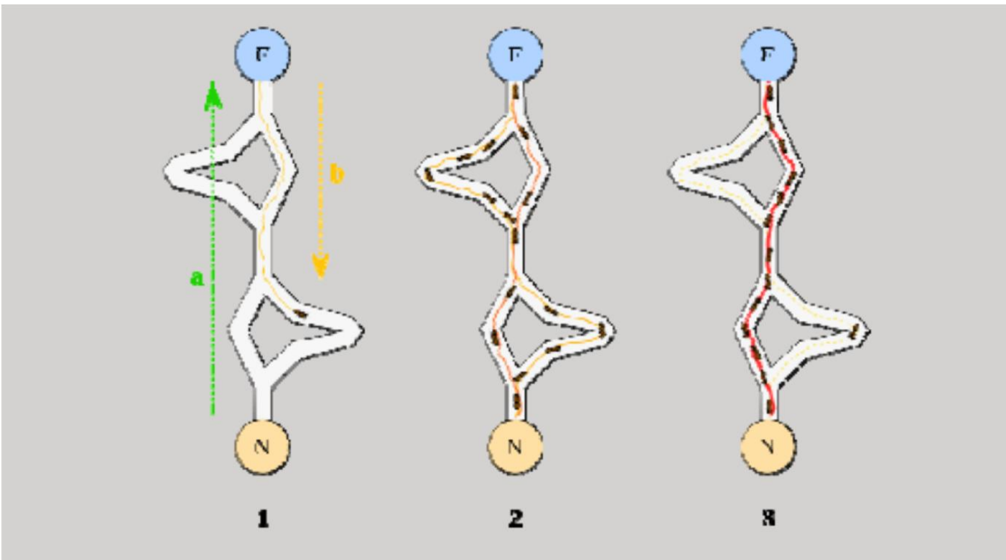
Definition: A model P = (S, Ω, f) of a CO problem consists of:

- a search (or solution) space S defined over a finite set of discrete decision variables and a set Ω of constraints among the variables;
- An objective function $f : S \rightarrow R+$ to be minimized.

Below figure is showing us how the ants find the shortest path.



Sketch Map of the ant theory



- The first ant wanders randomly until it finds the food source (F), then it returns to the nest (N), laying a pheromone trail
- Other ants follow one of the paths at random, also laying pheromone trails. Since the ants on the shortest path lay pheromone trails faster, this path gets reinforced with more pheromone, making it more appealing to future ants.

- The ants become increasingly likely to follow the shortest path since it is constantly reinforced with a larger amount of pheromones. The pheromone trails of the longer paths evaporate.

Initially, each ant is placed on some randomly chosen place. An ant k currently at place 'i' chooses to move to place 'j' by applying the following probabilistic transition rule:

$$p_k(i,j) = \begin{cases} \dfrac{\tau_{ij}^{\alpha} * d_{ij}^{\beta}}{\sum_{g \in J_k(i)} \tau_{ig}^{\alpha} * d_{ig}^{\beta}} \\ 0 \end{cases}$$

Where

- $p_k(i,j)$ is the probability that ant k in city i will go to city j.
- $j \in J_k(i)$ is the set of cities that have not yet been visited by ant k in city i.
- α is the relative importance of the pheromone trail.
- β is the relative importance of the distance between cities.

Therefore the probability that a city is chosen is a function of how close the city is and how much pheromone already exists on that trail. It is further possible to determine which of these has a larger weight by tweaking with the α and β parameters. Once a tour has been completed (i.e. each city has been visited exactly once by the ant), pheromone evaporation the edges is calculated. Then each ant deposits pheromone on the complete tour by a quantity which is calculated from the following formula:

$\tau(i,j)$ =p*$\tau(i,j) + \sum_{k-1}^{m} \Delta\tau k(i,j)$

$$\Delta\tau k = \frac{1}{Lk} \quad i,j \in tour$$

= 0 otherwise

- p* $\tau(i,j)$ multiplies the pheromone concentration on the edge between cities i and j by p(ρ), which is called the "evaporation constant." This value can be set between 0 and 1. The pheromone evaporates more rapidly for lower values.

- $\sum_{k-1}^{m} \Delta\tau k(i,j)$ is the amount of pheromone an ant k deposits on an edge, as defined by L_k which is the length of the tour created by this ant. Intuitively, short tours will result in higher levels of pheromone deposited on the edges.

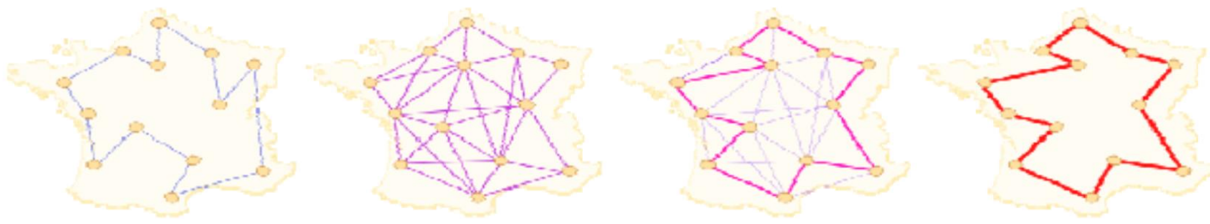
General Ant Colony Pseudo Code

```
Initialize the base attractiveness( τ) and visibility( η) for each edge;
for i < IterationMax do:
for each ant do:
        choose probabilistically the next  state to move into;
        add that move to the tabu list for each ant;
        repeat until each ant completed a solution;
 end;
 for each ant that completed a solution do:
        update attractiveness τ for each edge that the ant traversed;
 end;
 if (local best solution better than global solution)
        save local best solution as global solution;
 end;
end;
```

Traveling Salesman Problem(TSP)

In the Traveling Salesman Problem (TSP) a salesman visits n cities once.

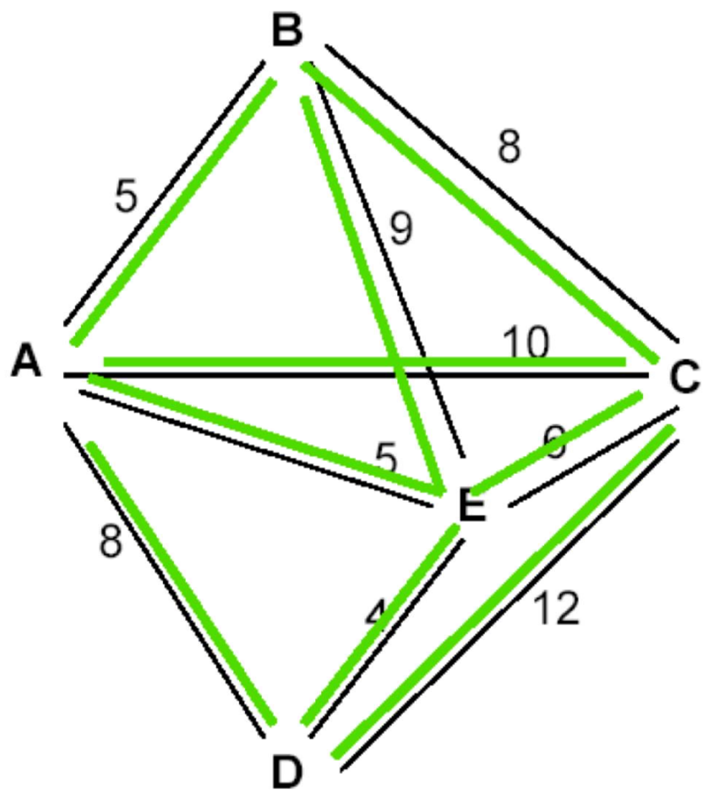
Problem: What is the shortest possible route?



1 2 3 4

Solution : Greedy Algorithm:

Searches for locally optimum solution.

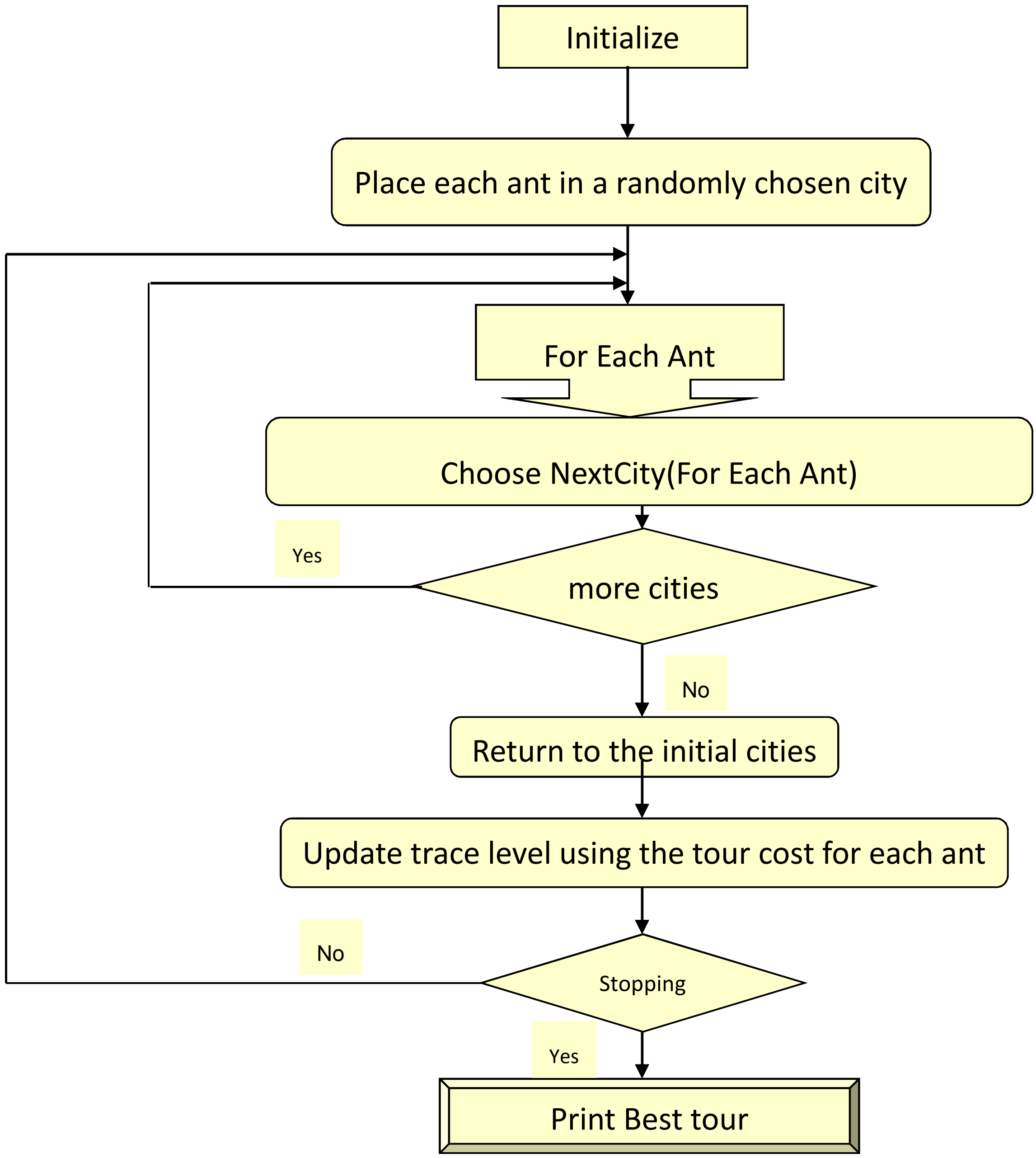


Ant System Algorithm for TSP

Pseudocode:

```
initialize all edges to (small) initial pheromone level τ0;

place each ant on a randomly chosen city;

for each iteration do:

        do while each ant has not completed its tour:

                for each ant do:

                                move ant to next city by the probability function

                end;

        end;

        for each ant with a complete tour do:

        evaporate pheromones;

        apply pheromone update;

                if (ant k's tour is shorter than the global solution)

                        update global solution to ant k's tour

        end;

end;
```

This algorithm found best solutions on small problems (75 city)


Initialize
Place each ant in a randomly chosen city
For Each Ant
Choose NextCity(For Each Ant)
Yes
more cities
No
Return to the initial cities
Update trace level using the tour cost for each ant
No
Stopping
Yes
Print Best tour

Benefits of Ant Colony Optimization

- Can solve certain NP-Hard problems in Polynomial time.
- Directed-Random Search.
- Allows a balance between using previous knowledge and exploring new solutions.
- Positive feedback for good solutions/Negative feedback for bad solutions.
- Approximately convergent.
- Optimal if not absolutely correct solutions.
- In certain examples of ACO, no one “ant” is required to actually complete an accurate solution.

	Best Parameter Set	Average Result	Best Result
ant-density	$\alpha = 1, \beta = 1,$ $\rho = .99$	426.740	424.635
ant-quantity	$\alpha = 1, \beta = 1,$ $\rho = .99$	427.315	426.250
ant-cycle	$\alpha = 1, \beta = 1,$ $\rho = .5$	424.250	423.741

Comparison among three strategies, averages over 10 trials.

Advantage

- Distributed computation avoids premature convergence.
- The greedy heuristic helps find acceptable solution in the early solution in the early stages of the search process.
- The collective interaction of a population of agents.

Disadvantage

- Slower convergence than other Heuristics.
- Performes poorly for TSP problems larger than 75 cities.
- No centralized processor to guide the Ant System towards good solutions.

• Applications

- Applications
    1. Routing problems
    2. Urban transportation systems
    3. Facility placement
    4. Scheduling problems
- How can we modify the algorithm?
    1. Vary the importance of pheromone
    2. Play around with evaporation rate
    3. Add time constraint
    4. Add obstacles

Conclusion

- ACO is a relatively new metaheuristic approach for solving hard combinatorial optimization problems.
- The cumulated search experience is taken into account by the adaptation of the pheromone trail.

- ACO shows great performance with the “ill-structured” problems like network routing.
- In ACO local search is important to obtain good results.