Comparing Ant Colony Optimization (ACO) and Simulated Annealing (SA) for CVRP

HAOYAN CHEN
Department of Computer Science, University of Bristol
hc16191@my.bristol.ac.uk

1. Introduction

It is an important logistics problem to find an efficient vehicle route. Ant Colony Optimizations (ACO) are new optimization methods proposed by *Marco Dorigo* in 1996[2]. ACO simulates the behavior of ant colonies when they forage for food and find the most efficient routes from their nests to food sources[1].

Since the vehicle routing problem is considered as NP-hard. For this kind of problems, using heuristic methods is regarded as a reasonable way in seeking solutions[1]. *Marco* compares the ACO with other heuristic approaches, such as Hill Climbing, Simulated Annealing (SA) and etc., and the experiment result shows that the performance of ACO was always very good[2].

Although many improvement strategies, like candidate list, multiple ant colonies and route mutation, are applied to ACO in order to enhance the performance of the algorithm, **SA shows a better result (cost) than ACO in FruityBun Challenge**. In this poster, both of them will be introduced to solve Capacitated Vehicle Routing Problem (CVRP), and only SA code is submitted.

2. ACO heuristic for CVRP

2.1 Route construction

Using ACO, an individual ant is regarded as a vehicle / truck, and the ant backs to the depot when the capacity maximum of each vehicle is met or when all customers are visited. Each ant has to construct a route begins with depot and ends with the same depot. To select the next customer j, the ant uses the following formula[2]

$$j = arg\max(\tau_{iu})(\eta_{iu})^{\beta} \; for \; u \notin M_k \, , \; if \; q \leqslant q_0 \tag{1}$$

where τ_{iu} is the value of pheromone on the path between the current location i and possible locations u. The η_{iu} equals the inverse of the distance between the two customer locations. β is a parameter in ACO, which shows the importance of distance comparing with pheromone value. q is a random number between 0 and 1, and the value of q_0 is a parameter. M_k stores the locations were visited. If q is greater than q_0, the algorithm uses roulette to select next customer to visit based on the probability distribution of p_{ij}[2]:

$$p_{ij} = \frac{(\tau_{ij})(\eta_{ij})^{\beta}}{\sum_{u \notin M_k}} \; if \; j \notin M_k \; otherwise \; 0 \tag{2}$$

2.2 Trail updating

The pheromone trails of ants have to be updated to reflect the ant's performance[2]. This updating is definitely an important element of ACO algorithm. The updating will be divided into two parts: local updating and global updating.

First, local updating simulates the process of evaporating pheromone in nature in order to ensure that no one path becomes too dominant.

$$\tau_{ij} = (1-\rho)\tau_{ij} + (\rho)\tau_0 \tag{3}$$

where ρ is a parameter that controls the speed of evaporation and τ_0 is equal to the initial pheromone.

After m ants construct a feasible route, global trail updating means that adding the pheromone of all arcs included the best route found by one of m ants[2]. Global trail updating is completed by the following formula:

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho L^{-1} \tag{4}$$

where L is the length of each route.

3. Improvement strategies of ACO

3.1 Candidate list

Each individual location is allocated a candidate list based on the distance to all the other locations[1].The size of each candidate list equals one forth of the total number of customers. In terms of *fruitybun250*, the size of candidate list is around 63. For instance, the candidate list of the l th location includes around 63 locations surrounding the l th point.

3.2 Mutation operation

The mutation operation is to randomly mutate tour and produce a new solution not far from the original one[3]. The process of mutation operation is as follows:

If $(q \leq qMutation)$

- Step 1:Select two tours from the current solution
- Step 2:Select the mutating points from each tour
- Step 3:Exchange the customers and hence generate a new solution
- Step 4:Ensure the new solution local optimally

where q is a random number between 0 and 1, and the value of $qMutation$ is a parameter. However, this kind of mutation operation may encounter a problem that the new solution cannot satisfy the capacity limit in *Step 2*. Hence a maximum number of attempts is very useful otherwise the algorithm may call an endless loop.

4. SA for CVRP

In order to improve the performance of the model, different algorithms are implemented, for example hill climbing and ACO. The result is that SA is not only the best solution but also very simple. The concept is explained in the lecture slides, and randomly two locations swap is applied to SA.

5. Flowchart and parameter settings for SA

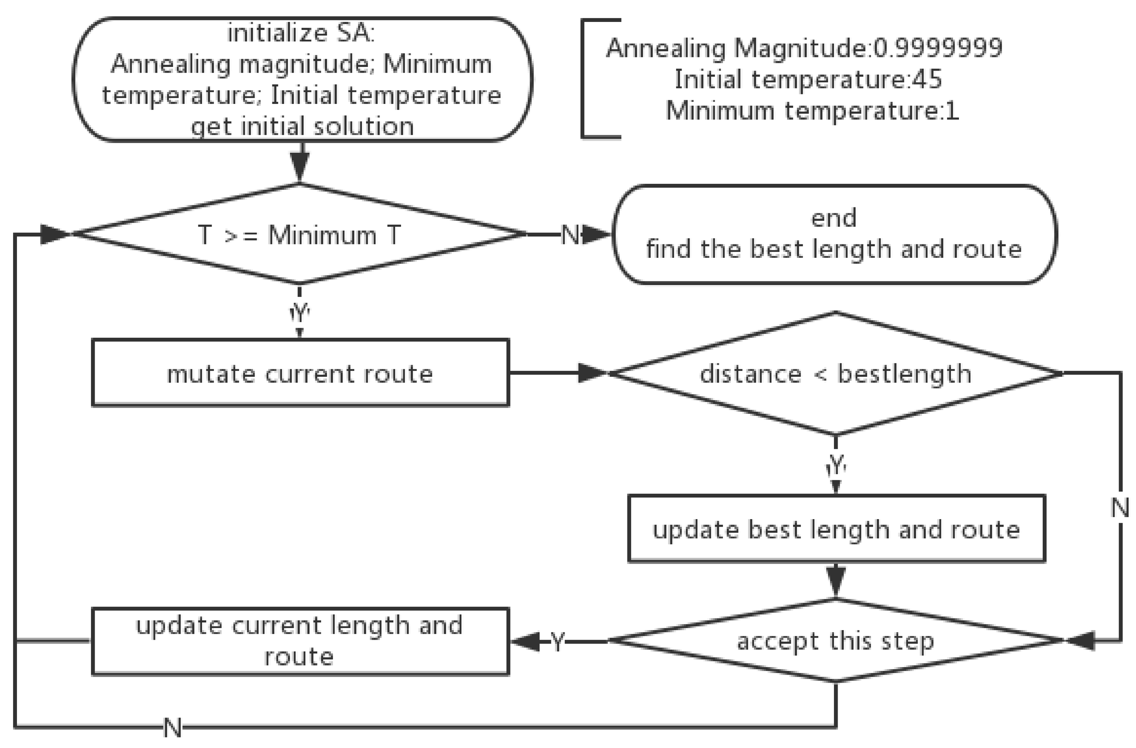

Figure 1: SA flowchart and parameter settings

6. Experiment Results

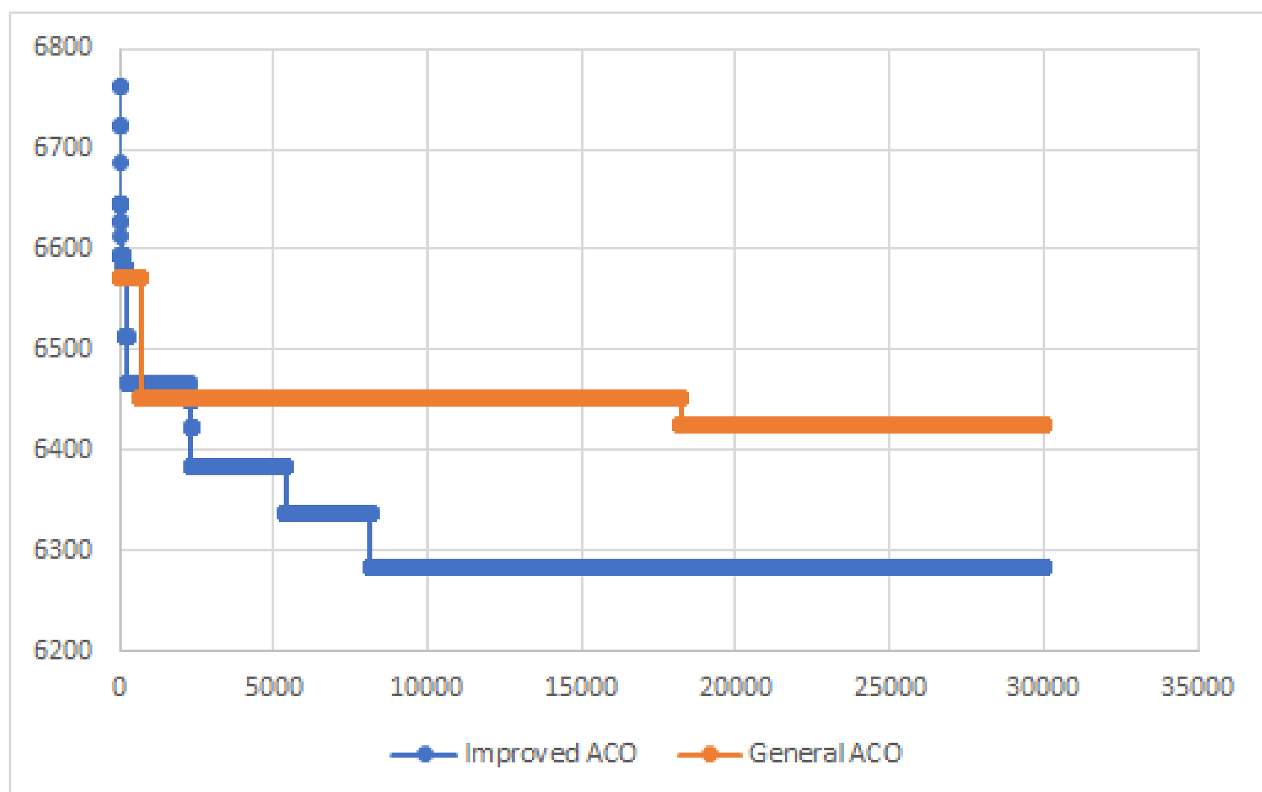

Figure 2: ACO Best length in each iteration

The Fig. 2 presents traces of general ACO and improved ACO applied to the FruityBun. As shown in Fig. 2, improvement strategies really play an important role in enhancing the performance of ACO, for the blue fat line finds a better solution (short length) than the other one.

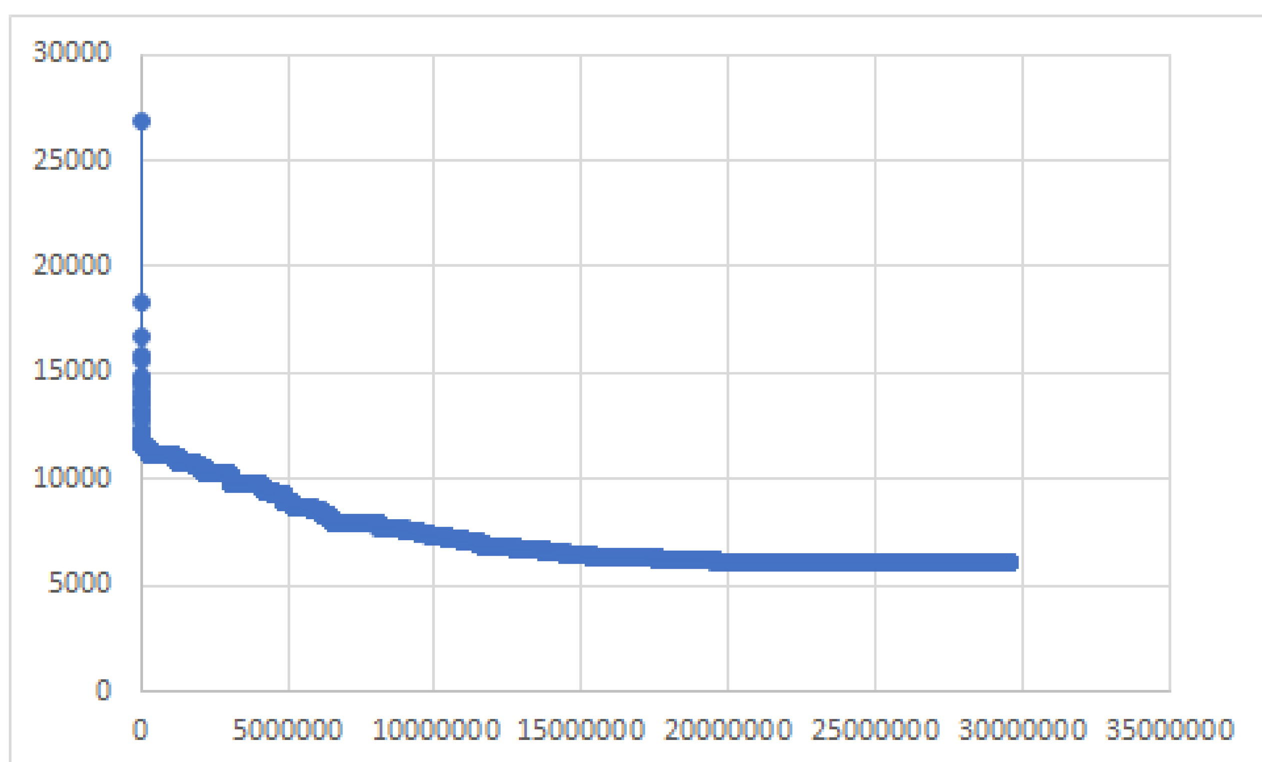

Figure 3: SA Best length in each iteration

From these two graphs, the SA model performs better than the general ACO and improved ACO models in the FruityBun Challenge, because SA model's final cost is lower than the others'.

References

[1] John E Bell and Patrick R McMullen. Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics*, 18(1):41–48, 2004.

[2] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1):29–41, 1996.

[3] Bin Yu, Zhong-Zhen Yang, and Baozhen Yao. An improved ant colony optimization for vehicle routing problem. *European journal of operational research*, 196(1):171–176, 2009.