Travelling Salesman Problem
CS562 - Artificial Intelligence

Shreyas Bapat (B16145) and Neeraj Yadav (B16024)

October 4, 2018

Introduction

Known to be NP-hard, the traveling salesman problem (TSP) was first formulated in 1930 and is one of the most studied optimization problems to date. The problem is as follows: given a list of cities and a distance between each pair of cities, find the shortest possible path that visits every city exactly once and returns to the starting city. The TSP has broad applications including: shortest-path for lasers to sculpt microprocessors and delivery logistics for mail services, to name a few.

The TSP is an area of active research. In fact, several variants have been derived from the original TSP. In this, we have to solve the TSP for points with euclidean as well as non-euclidean geometry.

We have tried to approach this difficult problem using two different techniques. 2-opt Swap, referred to as 2-Swap in further report and Simulated Annealing Algorithm.

Basic Maths: Let F be the cost function and D is the distance between two nodes and A(i) is an array of paths by salesman. We have to minimise the same.

$$F = \Sigma D(A[i])$$

Algorithms

Greedy Algorithm

1. Starts at an arbitrary city
2. Go to the next closest unvisited city
3. Repeat (2) until all cities have been visited, save path length
4. Start again at (1) with a diferent initial city.
5. Repeat (1-4) until all possibilities have been exausted and return the shortest path

While this algorithm is relatively fast O(N^3), it does not come up with very good solutions.

2-Opt Swap (2-Swap) Algorithm

In optimization, 2 Swap is a simple local search algorithm proposed by Croes in 1958 for solving the Traveling Salesman Problem. The main idea behind it is to take a route that crosses over itself and reorder it so that it does not. A complete 2 swap local search will compare every possible valid combination of the swapping mechanism. This technique can be applied to the travelling salesman problem as well as many related problems. These include the vehicle routing problem (VRP) as well as the capacitated VRP, which require minor modification of the algorithm. This is the mechanism by which the 2-opt swap manipulates a given route:

1. Take route[1] to route[i-1] and add them in order to new route.
2. Take route[i] to route[k] and add them in reverse order to new route.
3. Take route[k+1] to end and add them in order to new route.
4. Return new route.

Simulated Annealing Algorithm

SA algorithm is commonly said to be the oldest among the metaheuristics and surely one of the few algorithms that have explicit strategies to avoid local minima. The origins of SA are in statistical mechanics and it was first presented for optimization problems. The fundamental idea is to accept moves resulting in solutions of worse quality than the current solution in order to escape from local minima. The probability of accepting such a move is decreased during the search through parameter temperature

$$P = 1/(1 + \exp(\delta(F)/T)$$

Here 'P' is the probability of the move and 'T' is our temperature.

Modified Simulated Annealing Algorithm

For better results we have used a combination of Simulated Annealing Algorithm and 2 Swap Algorithm.
In this algorithm:

1. Start with a city and randomly select the next city from the remaining not visited cities until all cities are visited.
2. Next city the unvisited city is selected such that it is closest to the current city.
3. Then the obtained path is optimized using Simulated Annealing Algorithm in combination with 2 Swap.

Results

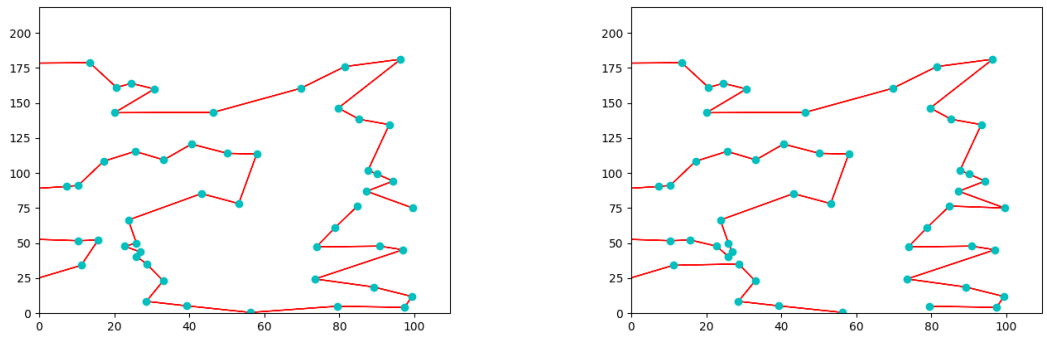
DATASET	LENGTH	TIME(sec)
euc_100	1561.62	27.13
euc_250	2589.13	44.5
euc_500	3528.55	77.52
noneuc_100	5264.62	30.81
noneuc_250	12888.47	52.41
noneuc_500	25574.26	107.07

Table 1: Observation table for Modified Simulated Annealing Algorithm

DATASET	LENGTH	TIME(sec)
euc_100	1623.69	8.15
euc_250	2646.55	67.38
euc_500	3695.24	161.48
noneuc_100	5316.41	13.32
noneuc_250	13002.95	94.21
noneuc_500	25643.98	200.32

Table 2: Observation table for 2-Swap Algorithm

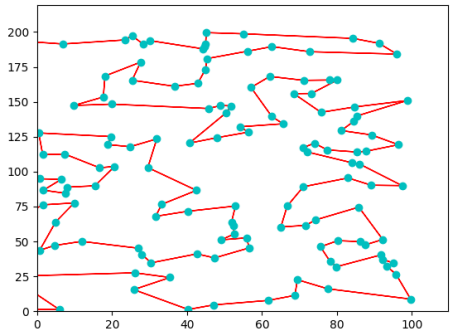
Now, we tried to plot all the paths to observe what changes are there in both the algorithms:



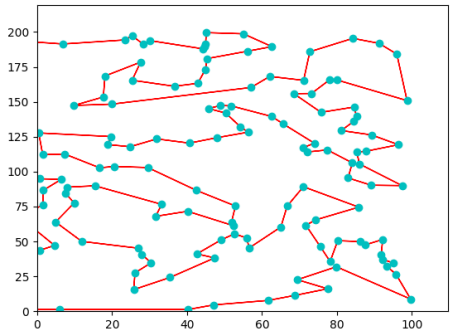
(a) Plot for Modified SA Algorithm

(b) Plot for 2-Swap Algorithm

Figure 1: Dataset euc_100

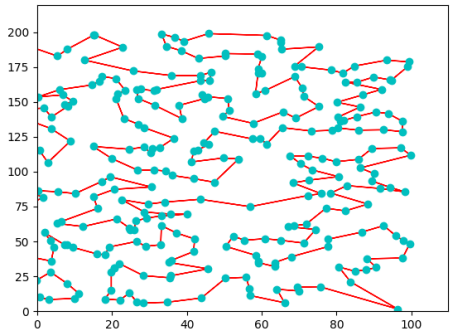


(a) Plot for Modified SA Algorithm

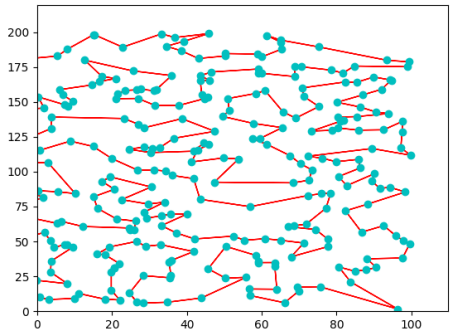


(b) Plot for 2-Swap Algorithm

Figure 2: Dataset euc_250

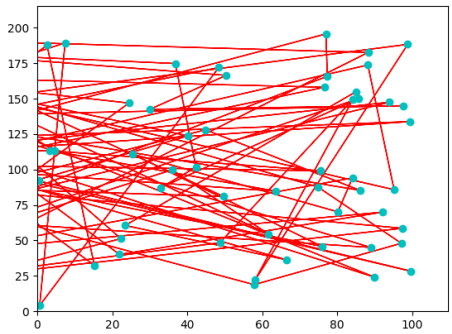


(a) Plot for Modified SA Algorithm

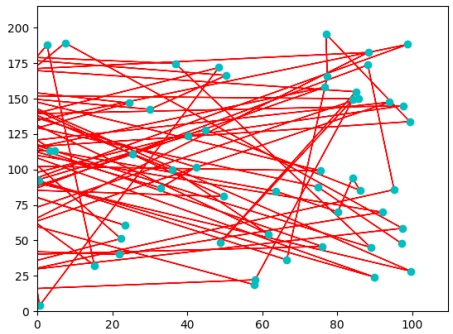


(b) Plot for 2-Swap Algorithm

Figure 3: Dataset euc_5000

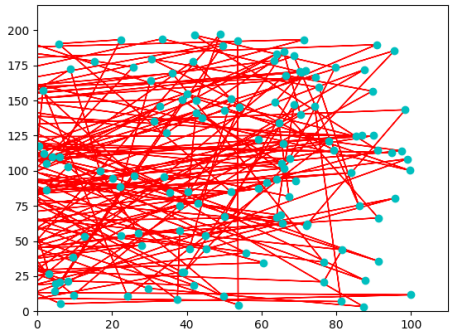


(a) Plot for Modified SA Algorithm

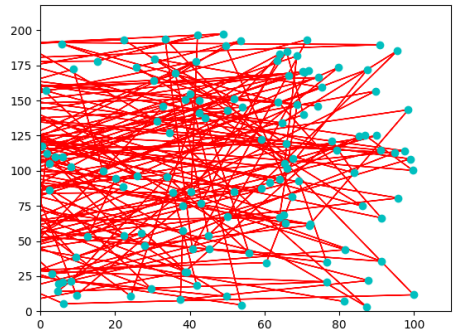


(b) Plot for 2-Swap Algorithm

Figure 4: Dataset noneuc_100

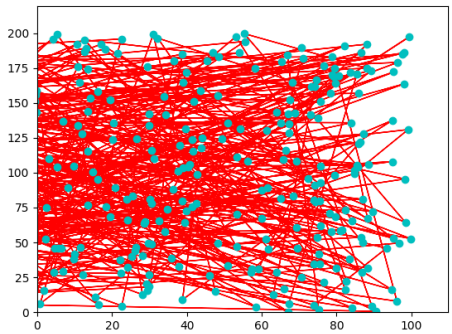


(a) Plot for Modified SA Algorithm

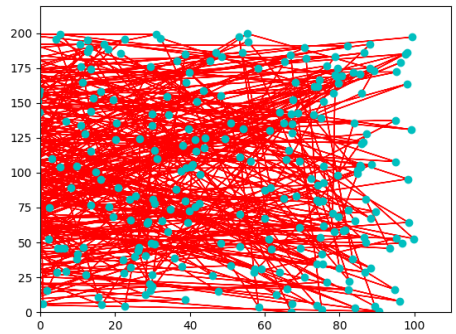


(b) Plot for 2-Swap Algorithm

Figure 5: Dataset noneuc_250



(a) Plot for Modified SA Algorithm



(b) Plot for 2-Swap Algorithm

Figure 6: Dataset noneuc_500

Conclusions

When we first tried to implement the 2-Swap algorithm, we got good results. But the results were not competitive. The 2-Swap Algorithm more or less gives a path which a human would give. As observing from the lot, it is evident that it performs really well. But the problem comes here, we can't optimize it's performance any further. After finding a local minima, it always remains there.

Therefore, we came up with a combination of 2-Swap and Simulated Annealing Algorithm. We first randomly generated a sequence and then applied simulated annealing to optimise the path. We got good results after using this and hence fine tuned some parameters and plotted the graphs for the same.