Travelling Salesman Problem

Pratham Aggarwal

Abstract--- Traveling Salesman Problem (TSP) is a classic optimization problem with numerous practical applications. In this project, we analyzed three different metaheuristic algorithms – Simulated Annealing, Threshold Accepting, and Hill Climbing – to solve the problem.

1 INTRODUCTION

The Travelling Salesman Problem (TSP) is a well-known optimization problem. In this problem we are given a set of 100 cities and we used and analyzed different metaheuristic algorithms to find the shortest tour and the shortest distance.

1.1 Motivation

Optimization problems are popular and finding effective solutions is crucial for obtaining the best resource allocation and efficiency. However, many optimization techniques might become caught in local optima. In this problem, we aim to solve this issue by carefully selecting the parameters of algorithms and finding a balance between exploration and exploitation.

1.2 Problem Statement

Given set of 100 cities, implement Simulated Annealing, Threshold Accepting and Hill Climbing to find the best route and the shortest distance. Compare and analyze how these metaheuristic algorithms performs and what are different parameters and termination conditions used to avoid getting stuck in local optima.

1.3 Related Work

TSP Problem can be solved using direct methods as well. Some of them include:
Brute Force: This method involves evaluating every possibility of visiting a city to find the best tour.
Nearest Neighbor: This method involves visiting the nearest neighbor that hasn't been explored yet.
The problem can also be solved using metaheuristic algorithms like genetic algorithm.

1.4 Contributions

We are solving the problem using three metaheuristic algorithms. We are using different parameters and termination conditions for all the three algorithms to avoid one of the most major setbacks by an optimization problem, the problem of getting stuck in local optima. These conditions help us find a perfect balance between exploration and exploitation.

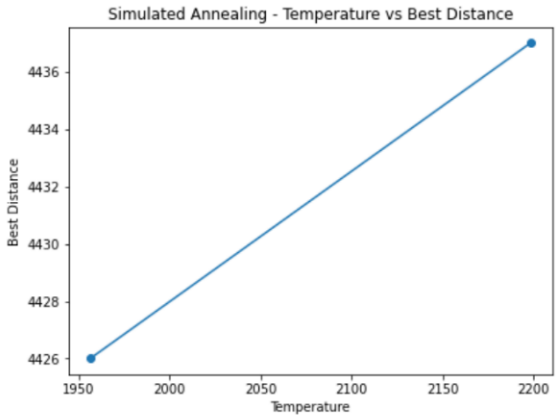

Figure 1: Temperature vs Best Distance

2 Background Material

Simulated Annealing: An algorithm inspired by the process of annealing in metallurgy. It explores the solution space by accepting best and worst solutions with a certain probability that decreases over time.
Threshold Accepting: Similar to Simulated Annealing, it used threshold values to check if the new solution is within the threshold of the current solution.
Hill Climbing: A greedy algorithm that moves to the best neighbor solutions found, iteratively, until found a local optimum.

3 Approach

We started by randomly generating possible tour and distances between each pair of cities. We next used the 2-opt swap method to generate possible tour neighbors. Different parameters were then used to determine whether the new neighbor tour was better than the present tour, and the best tour was determined accordingly. This process continues until a certain termination condition is reached and the final best tour is considered. The following discusses the way we used various parameters and termination conditions for each algorithm:

Simulated Annealing

In simulated annealing, we used an exponential function to decide which solution to accept. This

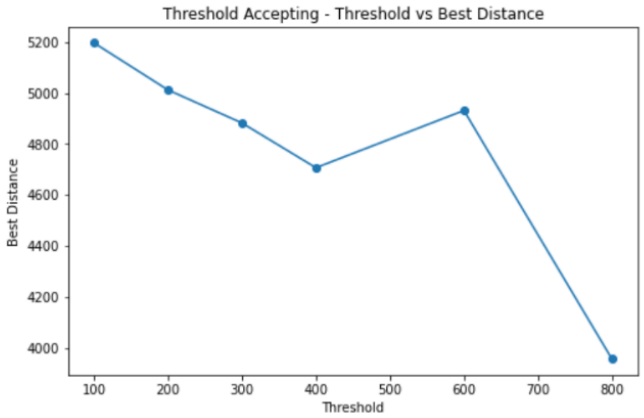

Figure 2: Threshold Vs Best Distance

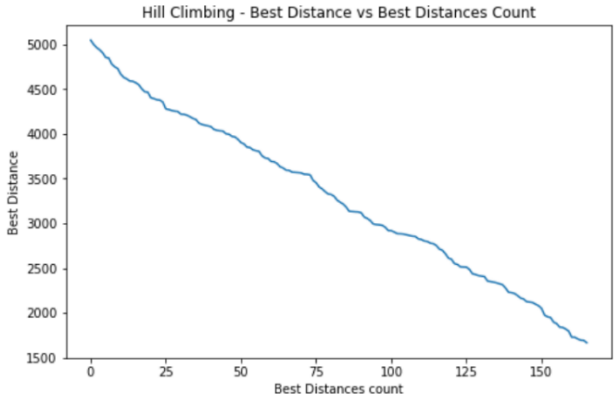

Figure 3: Best Distance graph Vs Best Distances count

ensures that at greater temperatures, we accept even the worst solutions, which encourages exploration. However, when the temperature reduces over iterations, the probability of accepting the worse solutions decreases, pointing closer towards exploitation. Starting with a higher temperature allows us to escape local optima at an early stage. Finally, our termination condition was the minimum temperature, ensuring that exploitation was preferred in the end to achieve a global optimum.

Threshold Accepting

In threshold accepting, we used various threshold values that decreased over iteration. These threshold values determined whether the new solution met the threshold of the current solution. Higher threshold values accepted more of the worst solutions, prompting exploration of the solution space. As the threshold value lowered over time, the algorithm accepted fewer worst solutions and focused on improving the present solutions, i.e. exploitation. For each threshold value, we ran the algorithm for the maximum amount of time. This time constraint acted as the algorithm's termination condition.

Hill Climbing

Hill climbing, as a greedy algorithm, has the highest probability of becoming stuck in local optima. To deal with this issue, we used minimum improvement as the parameter. The new solution was not accepted until it exceeded the minimum improvement. This allows us to explore the solution space to a considerable amount without finding a solution too early in the algorithm. We ran the algorithm till we reached a certain number of consecutive iterations without finding a better solution. This number of consecutive iterations functioned as the algorithm's termination condition.

Experiment set-up

The number of cities we evaluated was 100, and the tour was a randomly produced list of 100 integers ranging from 0 to 99. The distance between each pair of cities was also a randomly generated integer between 1 and 100. We collected matrices such as the best tour and distance for each algorithm. Then, for simulated annealing, we collected temperature, iterations, and temperature for each best distance found. For threshold accepting, we collected the best solution for each threshold value. Finally, for hill climbing, we stored the best distance found each time.

Parameters and termination values:

Simulated Annealing:

Initial temp (10000), *cooling rate* (0.89) and *min temperature* (0.01)

Exponential Function:

$$e^{\frac{(current\ solution - new\ solution)}{temperature}}$$

Threshold Accepting:

Threshold values- (800,600,400,300,200,100)

max time - 800 sec

Hill Climbing:

Minimum Improvement - 0.0001

consecutive iterations – 1000

3.1 **Results and Discussion**

Best Distance:

Simulated Annealing: 4426

Threshold Accepting: 3955

Hill Climbing: 1666

The results show that the three algorithms are effective at solving TSP. Simulated Annealing found a tour, and the temperature vs. best distance graph [Fig1] demonstrate the algorithm's ability to balance exploration and exploitation by obtaining best distance every time the temperature decreases. Threshold Accepting produced slightly better answer, and the threshold vs. best distance graph [Fig 2] showed that

larger threshold values resulted in better results. Hill Climbing outperformed the other two algorithms in identifying the best tour, as demonstrated by the gradual decline in the best distance graph [Fig 3].

4 Conclusions

In conclusion, Hill Climbing gave the best solution for this problem. The parameter and the termination condition, available computer resources and computational time, determine the solution. Simulated Annealing and Threshold Accepting, due to its random nature, are better suited to escaping local optima, but may demand more computer resources. Each algorithm's performance was heavily influenced by the parameters used, such as initial temperature, cooling rate, threshold values, and minimum improvement thresholds. These findings emphasize the significance of selecting parameters and the effectiveness of metaheuristic algorithms in addressing difficult optimization problems such as the Traveling Salesman Problem. So, our future work includes to try different parameters to find the best solutions.

5 References

https://medium.com/@francis.allanah/travelling-salesman-problem-using-simulated-annealing-f547a71ab3c6

https://www.cs.ubc.ca/~hutter/previous-earg/EmpAlgReadingGroup/TSP-JohMcg97.pdf

https://www.psychicorigami.com/2007/05/12/tackling-the-travelling-salesman-problem-hill-climbing/