Algorithm Engineering
WS 22/23

Sanket Datta
Toru Tsukamoto

Heuristics for Traveling Salesman Problem and Smoothed Analysis

Abstract

In this report, the traveling salesman problem is solved by using the nearest neighbor method, which is a type of greedy method, and the 2-Opt heuristic, which is a local search method that uses the 2-Opt neighborhood. By modifying the local search method so that the nearest neighbor method's solution was used as the starting point, a better solution than the nearest neighbor method alone was found in a reasonable amount of time, as the smoothing analysis guarantees.time.

1 Introduction

The travelling salesman problem (TSP) is the problem of finding the shortest route that visits a series of cities in sequence once and returns to the starting point. It is known that brute force is not a good way to solve this problem because it takes a long time (on the order of the factorial of the number of cities-1) and that even smarter methods can't solve it in polynomial time. Many computer scientists are interested in the TSP problem because it is easy to set up but hard to solve. Because of this, it has become one of the most important problems in combinatorial optimization.

2 Preliminaries

As approximate solutions for TSP, this report talks about the nearest neighbor method, which is a type of greedy method, and the 2-Opt heuristic, which is a local search method that uses the 2-Opt neighborhood. The simple nearest neighbor method has a small computational complexity of $O(n^2)$ but doesn't improve accuracy well. On the other hand, the local search method with the 2-Opt neighborhood can often find better solutions than the nearest neighbor method in a reasonable amount of time, but in the worst case, the computational complexity is exponential and there's no guarantee that accuracy would then improve. It is known that the neighborhood search method can be used to improve the solution after the nearest neighbor method has been optimized. Smoothed analysis has shown that there is a very high possibility that this happens. These were demonstrated by putting these two TSP solution methods and their combination to use on the TSP benchmark and solving it.

3 Algorithm & Implementation

Nearest neighbor heuristic and the 2-Opt neighborhood heuristic were implemented in this assignment.

3.1 Nearest Neighbor Heuristic

The nearest neighbour heuristic is one of the methods for solving the TSP by greedy methods: as shown in Algorithm 1, the algorithm moves from the starting city to the nearest city among those not yet visited, and returns to the starting city when all cities have been visited. In each city, the time computational complexity is $O(n^2)$, which is quite small, since the candidate city to visit next is determined in each city. The nearest-neighbour heuristic gives good approximate solutions. However, it is known that they often contain extremely long edges due to movements that tour cities within a small area and then move to other city clusters when they have completed their rounds of nearby city clusters.

Algorithm 1 Nearest Neighbor Heuristic

1: Select an arbitrary node j (City of Departure)
2: $l = j$and $W = \{1, 2, ..., n\}$
3: **while** $W \neq \emptyset$ **do**
4: Let $j \in W$ such that $C_{lj} = \min\{C_{li} | j \in W\}$ (Select nearest neighbor. C is distance between cities)
5: Connect l to j and set $W = W\ j$ and $l = j$.
6: **end while**
7: Connect l to the City of Departure.

3.2 2-Opt Neighborhood Heuristic

The local search method with 2-Opt neighborhood is a more advanced approach to the task. The local search method takes an approximate solution (the initial solution) of a TSP and makes a candidate solution by changing it slightly. Each candidate solution's total distance traveled is figured out, and the one with the shortest distance traveled is chosen. This is done over and over again until there is no solution with a shorter travel distance. A neighborhood operation is a slight change to the original solution, and the area where that solution is looked for is called the neighborhood. In particular, 2-Opt is a neighborhood operation that exchanges any two edges in a solution, and the 2-Opt neighborhood local search method uses the 2-Opt neighborhood to move the solution search forward. Algorithm 2 provides further details about the algorithm. In Note that the tours are in reverse order on the reconnected edges.
The 2-Opt heuristic works well in many cases, but in the worst case, depending on the size of the problem and the initial solution, the computation time can be extremely long and the accuracy of the solution particularly poor. However, smoothing analysis guarantees that for a perturbation ϕ, it can be computed with a computational complexity of $O(1/\phi * n^6 \log n)$. It is known that if the 2-Opt heuristic is performed using the initial solution as the result of solving by the nearest neighbour approximation, which is fast and has a reasonable accuracy, a solution equal to or better than the nearest neighbour approximation can be obtained with only a search of the same order of time computational complexity as the nearest neighbour approximation.
Additionally, in the worst situation, the computational complexity is O(n^3) if one neighborhood operation is the focus. This is due to the fact that there are $n(n-1)/2$ possible edge combinations to swap, and there are a maximum of $n-2$ cities that can be switched in a single swap.

Algorithm 2 2-Opt Neighborhood Heuristic

1: Given Initial Tour T of N cities
2: **while** $ImprovementFound$ **do**
3: $ImprovementFound \leftarrow$ False
4: **for** $i = 1, 2, ..., N-1$ **do**
5: **for** $j = 1, 2, ..., N$ **do**
6: **if** $Distance(Swaped(i,j)T) < Distance(T))$ **then**
7: $T \leftarrow Swaped(i,j)T$
8: $ImprovementFound \leftarrow$ True
9: break(2)
10: **end if**
11: **end for**
12: **end for**
13: **end while**

4 Experimental Evaluation

4.1 Hardware

Table 1 shows the specifications of the computer.

Table 1: Computer Specifications

Chip	Apple M1 Pro
Core	8 CPU, 14GPU, 16 Neural Engine
Memory	LPDDR5 16GB
Storage	APPLE SSD

4.2 Data

As input data, 27 sets of data from national cities are used. These datasets come from the TSPLIB, which was put together by Gerhard Reinelt in 1990. They can be found on the website (http://www.math.uwaterloo.ca/tsp/world/countries.html),

as the 25 National TSP instances, which are available to the public.

4.3 Results

4.3.1 Nearest Neighbor Heuristic

First, the Nearest Neighbour Heuristic was used to calculate approximate solutions. Even in China, which has the largest number of cities, Nearest Neighbour Heuristic was able to find an approximate solution within a few minutes. Measuring the accuracy of the approximate solution in terms of the total distance travelled, it increased by 19% from 46% of the optimal solution, as shown in Figure 1. The accuracy of the initial solution ranged from 89% to 8900%.

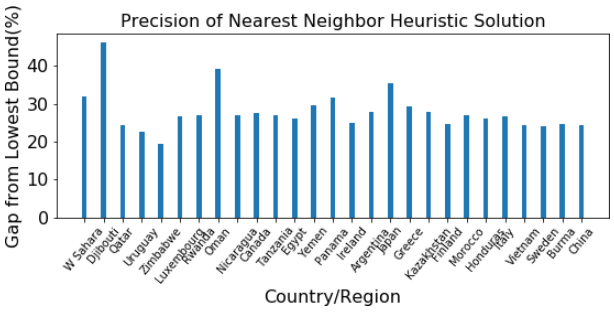

Figure 1: Precision of Nearest Neighbor Heuristic Solution,Compering Tour Total Distance Gap to Lowest Boundary.

4.4 2-Opt Neighborhood Heuristic

The 2-Opt Neighbourhood Heuristic was then used to obtain an approximate solution for the TSP.As shown in Figure 2, the computation time increased rapidly as the number of cities increased. Therefore, an upper limit of 2 minutes was set for the computation time. The computation time exceeded 2 minutes at the third Qatar (194 Cities) when the initial solution was started from Original and at the fourth Uruguay (734 Cities) when the approximate solution of the Nearest neighbour heuristic was started as the initial solution. When the upper limit of computation time was removed, Qatar could be computed in about 8 minutes when the initial solution was started from Original.

The number of swaps in the 2-Opt Neighbourhood Heuristic is shown in Figure 3. The number of swaps indicates the number of times up to the upper limit of the search time. When starting from the initial solution of Original, the number of swaps amounts to thousands of times in some national datasets, while in other datasets it ranges from a few to zero times. On the other hand, when the search was started from the Nearest neighbour heuristic results, the number of swaps was at most a few dozen times, and there were no datasets with extremely high swaps.

Figure 4 illustrates how the 2-Opt Neighbourhood Heuristic's solution improves on the Nearest Neighbor Heuristic's approximate solution. In West Sahara and Djibouti, where there are the fewest cities, better solutions than the Nearest Neighbor heuristic were produced when the initial solution was begun from Original. After Qatar, when the allotted search time was up, no better solution than the Nearest Neighbour heuristic was found. Additionally, certain nations and areas exhibited little progress from the original solution as the number of cities increased. On the other hand, improvements were seen up to the third-smallest; West Sahara, Djibouti and Qatar when the approximate answer of the Nearest Neighbor heuristic was applied as the initial solution. For larger cities, the final solution showed either no improvement at all or a very minor improvement equating to only a few Swaps. Since the Nearest Neighbor heuristic was utilized as the starting value and no deterioration could take place, the outcomes were significantly better than when the initial solution was begun from Original.

5 Discussion and Conclusion

First of all, Nearest Neighbour Heuristic stably obtained good approximate solutions that exceeded the total distance of the optimal solution by at most 46%. In addition, the 2-Opt Neighbourhood Heuristic could only solve the TSP problem up to about

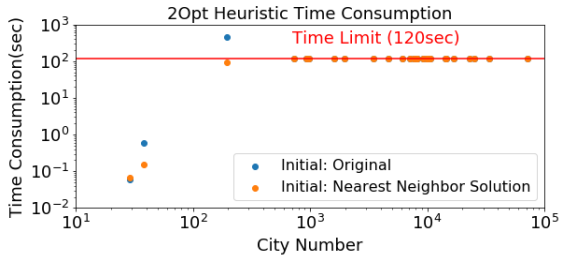

Figure 2: Time Consumption of 2-Opt Heuristic, Start from Original and Nearest Neighbor Heuristic Solution.

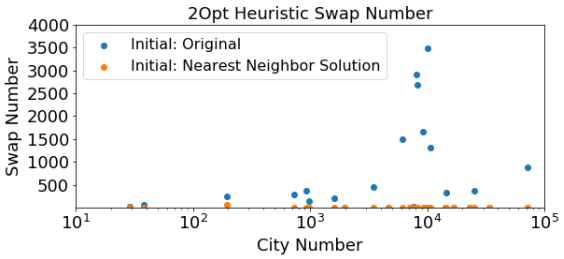

Figure 3: Swap Number of 2-Opt Heuristic, Start from Original and Nearest Neighbor Heuristic Solution.

100 cities in realistic computation time, while Nearest Neibour was able to solve the TSP problem for all regions up to China, where the maximum number of cities is 71009. This shows that Nearest Neighbour Heuristic is a valid practical option when accuracy is not pursued or when the number of cities is large.
In contrast, the 2-Opt Neighborhood Heuristic was able to find a solution that was closer to the optimal than the Nearest Neighbor Heuristic in West Sahara, Djibouti, and Qatar, where the calculation could be done in a reasonable amount of time. Because a strict upper limit on the computation time was set to make this experiment go well realistic amount of time, the theoretical analysis could not be used to confirm the expected change in computation time.
But when the calculations began with "Original"

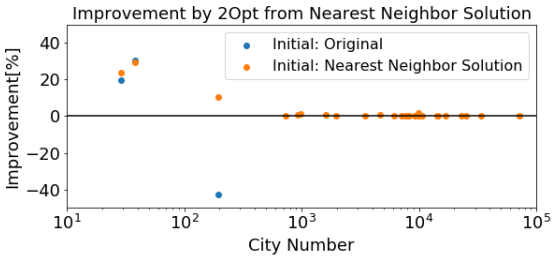

Figure 4: Improvement by 2Opt from Nearest Neighbor Solution, Start from Original and Nearest Neighbor Heuristic Solution.

as the initial value, there were regions where there wasn't much swapping between regions and regions where there were more than a few thousand swaps. It was seen that there were a lot of swap regions, which suggests that the original initial value of 2-Opt Neighborhood Heuristic is hard to solve and makes computation times very long. When the initial value of Nearest Neighbor Heuristic was used, on the other hand, the number of swaps was much lower in all cities. This is consistent with the solution being close to the optimal solution already at the initial solution stage. And it is thought that this is why such extreme variations in computation time can be avoided when starting from the initial values of original.