Investigating Solutions to the Traveling Salesman Problem

By Izaac Facundo and Steven Oh

Izaac Miguel Facundo, imf339, TACC: izaac, izaacfacundo@utexas.edu
Steven Jaehyeok Oh, jo25672, TACC: jo25672, oh.jae.steven@utexas.edu

Objective:

The objective of this project is to create a route that was considered the most efficient for a delivery truck, or multiple delivery trucks. In order to tackle this problem, the Traveling Salesman Problem (TSP) was explored, where the best route was crafted given a list of addresses. This issue is relevant today as many goods are transported on trucks, or other motorized vehicles, to the final destination. This is important in companies like Amazon, where the most optimal route would save both time and money while bringing customer satisfaction through fast deliveries. In the case of Amazon, specifically, they offer a premium subscription called Amazon Prime, where many applicable goods are delivered a day after the order has been placed. There are other considerations like multiple trucks or days, dynamism, and ethics, as well, which will all be discussed later in this report.

For the purpose of this experiment, the addresses are plotted on a cartesian coordinate plane. This is similar to a bird's eye view of a map, where locations can be defined using longitude and latitude. In this case, the x-value is the longitude and the y-value is the latitude. The depot or warehouse was determined to be the origin of the plot, (0, 0). The depots have been added at the beginning and end of every address list to represent the truck starting and ending at the warehouse.

The programming language C++ is used in this experiment. The packages of <iostream>, <vector>, and <cmath> are used to complete this experiment. The standard functions of cout (print out) and endl (next line) were used from the <iostream> library; the standard function of vector was used from the <vector> library; and sqrt (square root), abs (absolute value), and pow (power) from the <cmath> library. The graphical representations in this report are created using MATLAB with the regular data plotting functions (plot(), scatter(), etc.).

Optimization:

The three classes built for this study were the Address, AddressList, and Route classes. The Address class holds two data members: its x and y coordinates (as doubles). The AddressList contains a protected data member: a vector of Addresses. The Route class is a derived class from the AddressList class. The Route class contains methods for adding depot addresses to the beginning and end, reordering the Route via the Greedy solution, reordering the route via the Optimized solution, and splitting the route across multiple trucks.

The two solutions used for this study were the Greedy solution and the Optimized solution. The Greedy solution to the TSP worked as follows: the depot is chosen as the starting point, and each stop is the address closest to the last. This approach uses the distance-between method to find the closest address to the previous one. The Optimized solution was a little more complex. The Optimized solution took the original randomly ordered address list and perturbed it to find the shortest route. It consists of a nested for-loop where the outer loop selects a new address to start from while the inner for-loop swaps two elements in the list. After each iteration in the loops, the length of the route is found. If the length is shorter than the previously found shorter route, this route replaces it. This method has a runtime complexity of n(n-1) as the nested loop has iterations of the size of the address list on the outside and the size of the address list minus one on the inside.

A combination of the two methods was also used to get a shorter route. For some of the experiments, the Optimized method was used on a route that had already been reordered using the Greedy method.

Lastly, multiple trucks/multiple days were added as a method to the Route class. This method bisects the Route as it is and splits the addresses across multiple new routes depending on how many are requested.

Experiments:

Multiple experiments were conducted in order to determine which route optimization algorithm was most effective and whether the combination of the two would prove to be the best. The first experiment is the comparison of the Greedy route optimization versus the Optimized route optimization with 3 sets of 7 different addresses. The second experiment is combining the Greedy route and Optimized route algorithms, exactly in that order, with the same set of data as the first experiment. The third experiment is another comparison of the Greedy route versus the Optimized route, but instead with a data set of 20, 10, and 4 addresses. This touches upon the dynamism issue of this project. Lastly, the final experiment is to split up the routes of the first two experiments into two trucks or days and compare them to each other. For all of these experiments, the data was presented through the terminal.

Experiment 1:

Experiment 1 is used to compare the effectiveness of the Greedy route optimization to the Optimized route optimization. There are 3 sets of data with 7 different addresses in each set. This is so that no matter what kind of data was given, it can hopefully draw a firm conclusion based on the findings.

After conducting this first experiment, it was concluded that the Greedy route optimization was a better choice than the Optimized route optimization as it provided lower route lengths. This means that with a lower route length, the time it takes to deliver each package would also be lower, increasing efficiency and saving valuable resources like time and money.

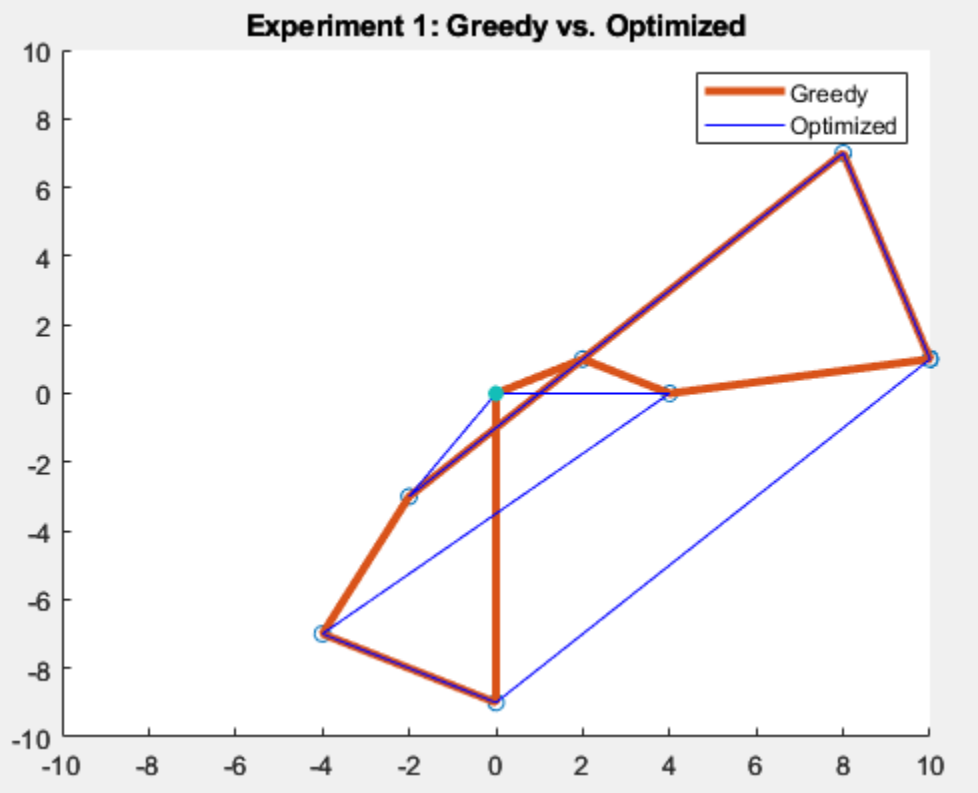

Figure 1: As shown in the graph, the Optimized route seems to be scattered, while the Greedy route is more organized. This is true for the other sets of data as well. The purple lines are showing the Optimized route lines on top of the Greedy route lines.

Ultimately, these results are reasonable as the Greedy route is simply looking for the shortest distance between each address, saving up that extra time it would've usually taken in any other route scenario.

Experiment 2:

Experiment 2 combined the Greedy route and Optimized route optimizations with the same set of data as experiment 1. The purpose of this experiment is to compare the results from this experiment with the results of experiment 1 and observe whether the combination of the route optimizations further provided better, worse, or the same routes as before.

After conducting experiment 2, it was concluded that the Greedy route optimization is still the best choice in terms of creating the shortest route. This is because the length of the sets of data either got worse or stayed the same as the Greedy route optimization. The first set of data was the only case where the length of the route was worse, which makes a lot of sense when looking at the code of the Optimized route method. It was hardcoded that the shortest path is 999999, instead of being the length of the route that it originally stored, in this case, the Greedy route. This means that the hardcoded value will cause the Optimized route algorithm to rearrange the list of addresses to find the shortest length, not accounting for what the list used to be beforehand. When the code is changed to the length of the route that it originally stored, the results should be the same as when the Greedy route optimization is run on its own.

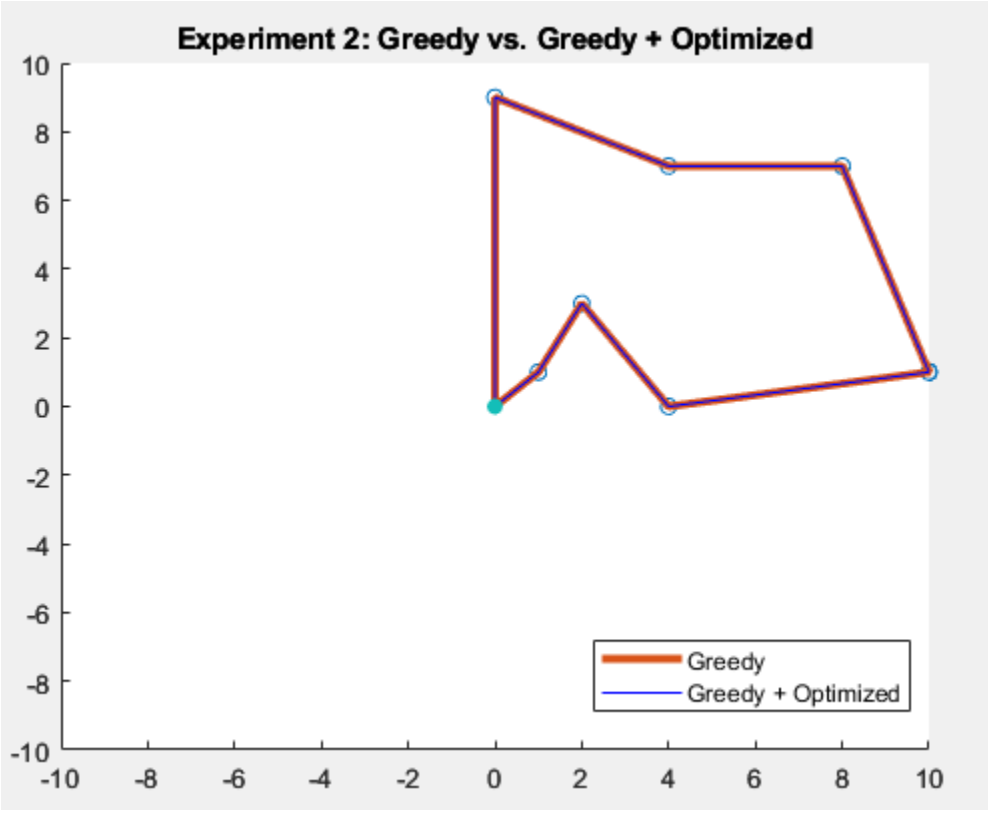

Figure 2: As shown in the graph, the Greedy + Optimized route algorithm prints the exact same result as the Greedy route optimization on its own. This shows that the Greedy route optimization is efficient on its own.

Experiment 3:

Experiment 3 is an experiment that contains 3 different sets with 3 different sizes of data: the first set contains 20 addresses, the second set contains 10 addresses, and the third set contains 4 addresses. The purpose of this experiment is to compare whether different sizes of data, also known as dynamicism, will affect the results of the Greedy route versus the Optimized route. As experiments 1-2 have shown that the Greedy route optimization is superior, this experiment further solidifies or argues that the Greedy route optimization is best.

After conducting experiment 3, set 1 and 2 of the experiment continue to support the idea that the Greedy route optimization is the best, however, set 3 with 4 addresses was an outlier. It showed that the optimization of the Optimized route was better than the Greedy route optimization.

Figure 3: As shown in the graph, the Greedy route optimization looks more scattered than the Optimized route, which is quite different from all the other results. The purple lines are showing the Optimized route lines on top of the Greedy route lines.

Ultimately, for dynamism, the strategy is to utilize the Greedy route optimization, but also to readily have data using the Optimized route optimization, which will guarantee that the best route is constructed.

Experiment 4:

Experiment 4 is used to compare the data between the Optimized route and the Greedy route when there are multiple trucks or days. While the program written can account for multiple trucks or days, this experiment was only accounting for 2 trucks or days.

After conducting experiment 4, it was found that the Greedy route optimization, once again dominated when compared to the Optimized route. This shows that even with multiple vehicles or days, Greedy route optimization is the best to rely upon.

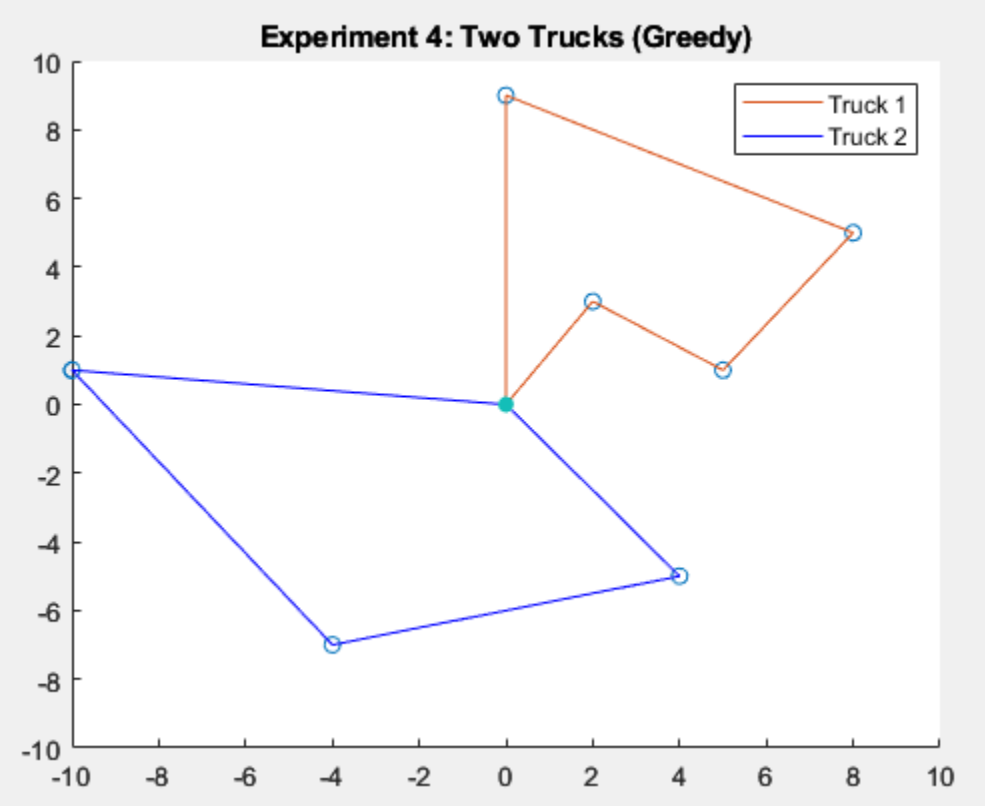

Figure 4: As shown in the graph, the two trucks or days are split quite nicely when using the greedy route optimization beforehand.

Summary of Experiments:

In conclusion, the results from the four experiments overwhelmingly show that the Greedy route optimization is far superior when compared to the Optimized route and the combined route optimizations. Even with variables like dynamicism and multiple trucks or days, it continued to show that the Greedy route optimization was effective at creating efficient routes.

Ethics:

Amazon has been condemned numerous times by numerous people for its unfair labor practices. Truck drivers are most of the time contractors and do not receive adequate benefits, and the rapid pace of work and lack of adequate break time has been detrimental to their health and well-being. A better solution to the route optimization problem would not necessarily affect this. Efficient and effective routes can be found and used while still allowing workers to have the rights they deserve. Breaks can be factored into these algorithms without noticeable differences in efficiency. From our work here, we can say that drivers can be treated well without ruining the efficiency gained from well-made route planning algorithms.

Future Work:

In the future, further exploration of the effect of address list size could prove beneficial. In the third experiment, at an address size of 4, the Optimized solution beat the Greedy solution. This is the only tested scenario where that happened. This is interesting and deserves future attention.

Another concept to explore would be real-life constraints. The capacity of each truck, fuel, employee breaks, and actual roads were not included in this study. These could provide more realistic solutions to the TSP. These things would add more complexity to the study, but they would be worth the effort.

Appendix:

To calculate the length of the entire route and the distance between two different addresses, a simple distance formula was used:

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

This formula calculates the distance between two addresses, which would later be added up to find the length of the entire route. Unlike the Manhattan distance, where the city streets are built on a grid-like structure, this experiment assumes that there is always a straight line between two addresses.