Optimizing EV Transportation using the Tesla Supercharger Network

Nikhil Arora,Gaurav Bhandari, Utsav Majumdar, and Selva Subramaniam

Department of Industrial and Enterprise Systems Engineering, University of Illinois

IE 532: Analysis of Network Data

Dr. Chrysafis Vogiatzis

December 15, 2023

Introduction

In the contemporary landscape of transportation, the integration of electric vehicles (EVs) into our daily lives is becoming increasingly prevalent. This rise in EV usage brings forth novel challenges in the realm of route optimization, particularly within intricate transportation networks. Traditional routing algorithms have predominantly focused on finding the shortest path between two points. However, the unique constraints associated with electric vehicles, such as their limited range and the necessity to recharge, add layers of complexity to this problem.

The paradigm of route optimization in a transportation network, where nodes represent junctions or points of interest and edges signify the distances between these nodes, must be reevaluated in the context of electric vehicles. This optimization problem becomes even more intricate considering the strategic placement of charging stations at fixed nodes within the network and the associated cost or delay incurred during the charging process.

Motivation

The significance of optimizing transportation routes for electric vehicles extends beyond mere convenience; it is a critical component in the broader objective of fostering sustainable transportation practices. As the world gravitates towards more environmentally friendly solutions, the adoption of electric vehicles is a key step in reducing our carbon footprint. However, the limited range of EVs compared to traditional fossil fuel vehicles poses a significant barrier to their widespread adoption.

Addressing the route optimization problem for EVs is essential for several reasons:

1. Range Anxiety Mitigation: One of the primary obstacles deterring individuals from switching to EVs is range anxiety—the fear of running out of power before reaching the destination or a charging station. Efficient route planning that incorporates charging stops can alleviate this concern.

2. Cost and Time Efficiency: While stopping for a charge incurs an additional penalty (in terms of time or distance), an optimized route that judiciously integrates these stops can lead to overall time and cost savings, making EVs more appealing for longer journeys.

3. Infrastructure Utilization: As EV charging infrastructure is still in its nascent stages, optimizing routes to make effective use of existing charging stations is crucial. This ensures that EV drivers have reliable access to charging facilities, thereby enhancing the practicality of EVs.

4. Environmental Impact: By optimizing routes for efficiency, EVs can operate with minimized energy usage, thus contributing to lower emissions and a reduced environmental footprint.

Problem Formulation:

In the proposed scenario, we explore a simplified yet insightful version of the electric vehicle (EV) route optimization problem in a transportation network. This network is represented as a graph where nodes signify specific locations (intersections, points of interest) and edges denote the distances between these nodes. Some nodes are equipped with EV charging stations. The goal is to determine the most time/distance-efficient route for an EV to travel from a start point or origin node(s) to a target point or destination node (t) within this network.

Basic Setup

Network: The network is a graph G(V,E) where V are the nodes or important locations and E are the edges/roads between these locations.

Charging Stations: these nodes have charging stations.

Vehicle's Range: The EV has a limited range of 150 units (distance)*.

Charging Penalty: Time spent at each charger equates to a distance penalty of 80 units*.

*These penalties and ranges change in the application to the network, details discussed later.

Objective

The primary objective is to find the optimal/quickest route from point s to point t (Origin Destination pairs), considering the EV's range limitations and the time penalty incurred at each charging station.

Preliminary Analysis with a Simplified Toy Network

Before delving into the complexities of real-world data, it was crucial to establish a foundational understanding of the problem using a simplified or "toy" network (Refer results section for more on this). This approach served several key purposes:

1. Concept Validation: The toy network acts as a controlled environment where we can validate the underlying concepts of our route optimization algorithms. It allowed us to test the basic principles of EV routing, including range limitations and charging penalties, in a more manageable context.

2. Algorithm Development and Testing: Developing and refining algorithms in a smaller, simpler network reduced computational complexity and sped up the iteration process. This environment was ideal for debugging and fine-tuning our algorithm before scaling up to our larger dataset.

Incorporation of Heatmap Visualization

To complement our analysis, we created a heatmap to visualize the distance distribution between origin and destination nodes in both the toy and real-world networks. The heatmap serves several important functions:

1. Data Insight: It provides an intuitive visual representation of the network, highlighting areas with high density of nodes and potential bottlenecks or areas with sparse charging options.

2. Algorithm Performance Visualization: By illustrating the frequency and distribution of routes taken in the network, the heatmap can help identify patterns and efficiencies in the routing algorithm.

   Although, in our analysis, we had fixed the location of the charging stations both in the toy network and the real world data (as we were using existing Tesla Supercharger locations) and wanted to just focus on the optimal routes to take with and without charging for an origin-destination pair, based on a specific range of the vehicle.

Application to Real Network

To extend this problem to real-world applications, to provide practical route suggestions for EV drivers, we utilized two datasets:

Tesla Supercharger Dataset: An open-source dataset providing the locations (latitude and longitude) of Tesla superchargers in California.

California Road Network: A comprehensive dataset of road networks in California, including nodes and edges representing distances between key locations.

Final Network for Tesla Supercharger Network (Data Preprocessing):

In our quest to devise the most efficient routes for electric vehicles (EVs) between specified origin-destination pairs, we strategically chose a particular region within California. This area, spanning from central Los Angeles to Santa Maria, offers a diverse landscape for our analysis as it encompasses both densely and sparsely populated road networks, presenting a realistic and varied testing ground for our route optimization algorithms. The distance between these two points is approximately 160-170 miles, notably exceeding the range of our EV model. This deliberate choice ensures that the utilization of a charging station becomes an integral part of the journey, thereby adding real-world complexity to our route optimization task.

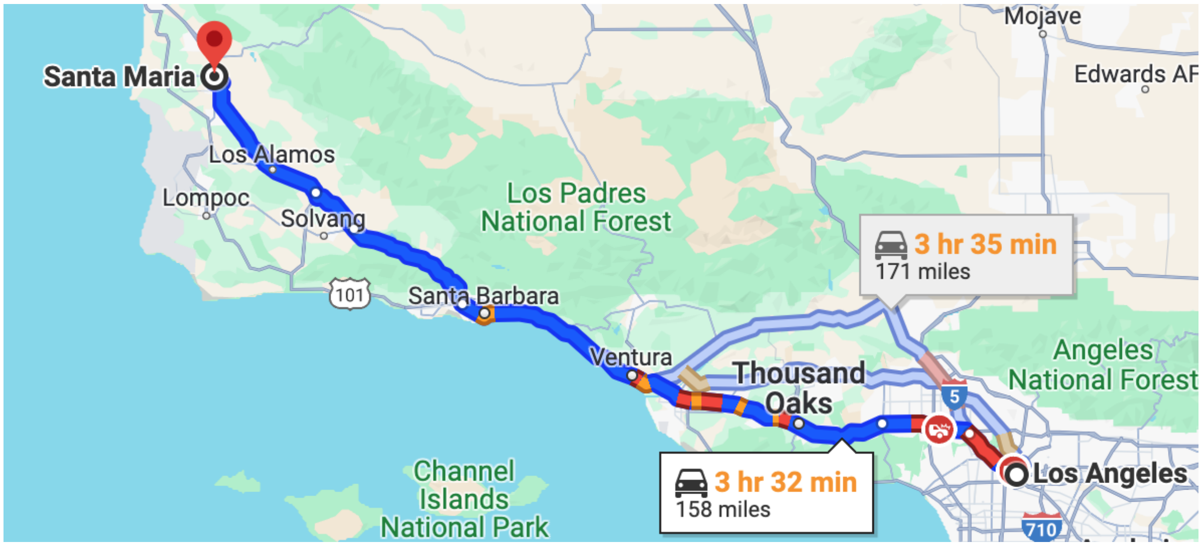

Figure 1: Google Maps route which using same application of shortest path (distance- 158 miles)

To enhance the real world nature of our model, we incorporated data from the Tesla Supercharger network. Within the selected region, we identified approximately 43/44 charging stations along the envisaged route. The geographical coordinates of these superchargers played a pivotal role in delineating the scope of our study area. For instance, the southernmost charging station in our dataset is located in Culver City, with coordinates (33.986779, -118.390213). This particular location set the southern boundary of the road network region under consideration for our study.

range of latitude: 33.986779 to 34.952928 (South most point of our area to North most point of our area);

range of longitude: -118.078417 to -120.434555 (East most point of our area to West most point of our area)

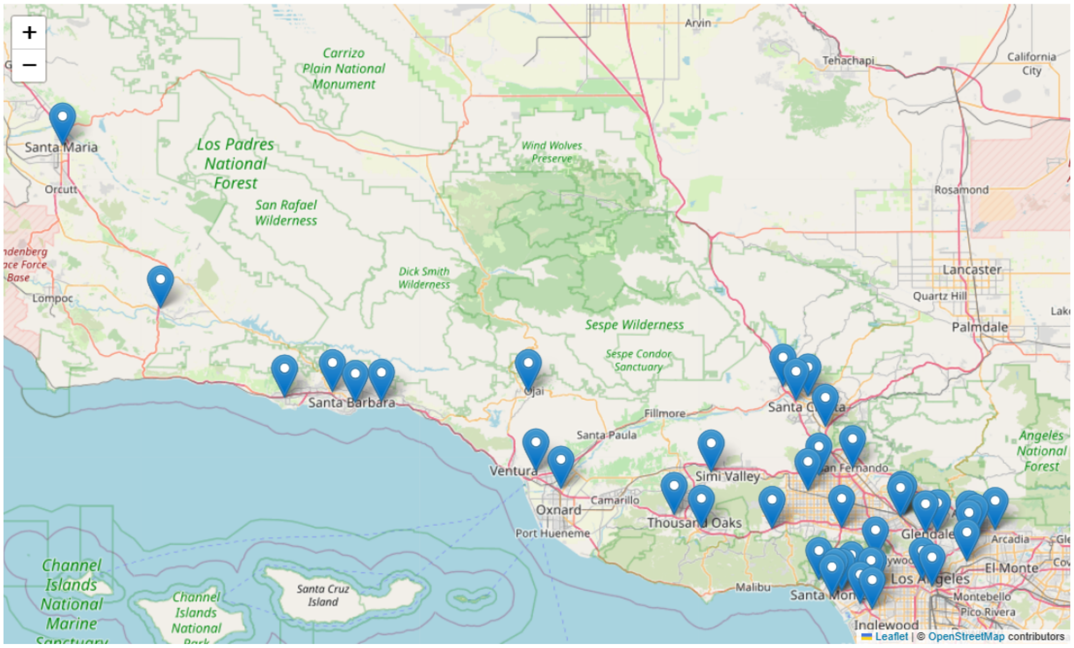

Figure 2: The Network region used for analysis with the 43 supercharger stations picked where cars can stop in their paths from any origin-destination pair.

Tailoring the Dataset to a Specific Geographic Area

Our approach to creating a working dataset for EV route optimization began with the refinement of the California road network data.

The initial step involved pruning the Cal_road_node_network dataset to align with our predetermined geographic area of interest, defined by specific latitude and longitude ranges as shown above. This process effectively created a focused working area for our network analysis, ensuring relevance and manageability.

Integrating Road Network and Edge Data

Once the geographic scope was defined, we performed a matching process. Looking at the Cal_road_network_edge dataset, we identified pairs of nodes from our trimmed node network that were interconnected. These pairs, along with the distances between them, were compiled into a new dataset. This dataset now represented a complete network of our target area, detailing all nodes and the connecting edges where applicable.

Incorporating Supercharger Locations

Addressing the challenge of integrating supercharger data required a two-step proximity analysis. Since the coordinates of the superchargers didn't directly correspond with the nodes in our network, we first identified the nearest supercharger for each node in our network. Subsequently, we located the nearest node to each supercharger. This reciprocal approach allowed us to accurately assign supercharger locations within our network.

Geographic Positioning and Visualization Using Geopandas

To bring geographic precision to our network, we utilized the Geopandas library in Python. This powerful tool enabled us to plot each node's position in geographic space, using latitude and longitude coordinates. The geodesic distance function within Geopandas was employed to calculate the distances between nodes, providing accurate edge weights in kilometers.

Finally, we created a visualization of this refined dataset. This visualization included the nodes and edges of our transportation network and overlaid the supercharger locations based on its nearest node. The resulting map offers an intuitive and detailed overview of the network within our area of interest.

The plot is shown below:

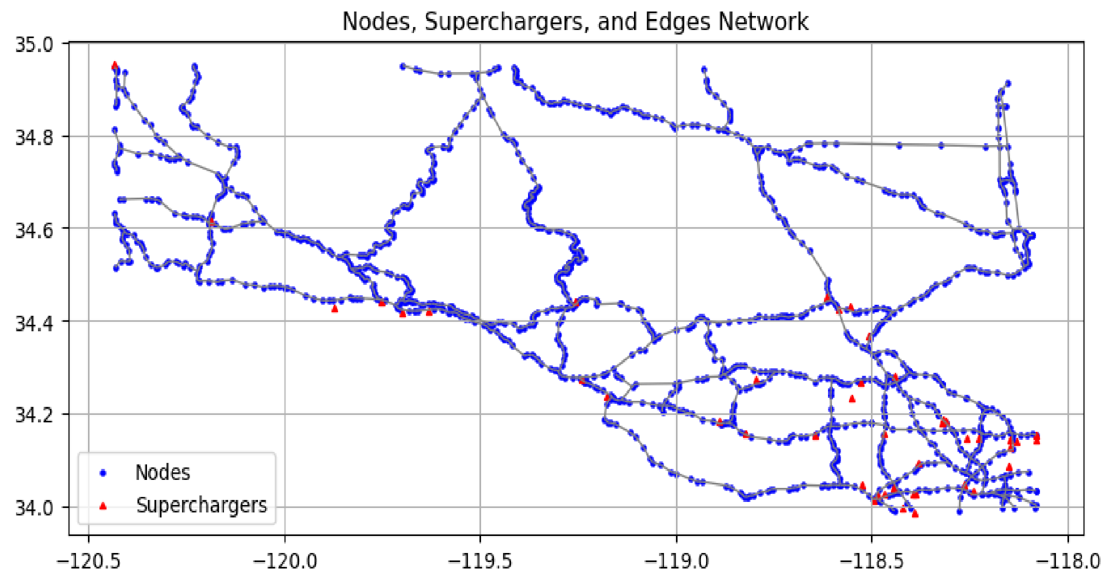

Figure 3: Nodes, Edges and Superchargers in our trimmed network/dataset (Area for analysis)

We then use the output from the data cleaning script to use as input for our graph G(V,E) for our script that calculates the shortest paths between each origin-destination pair.

Model Formulation and Code

The idea behind formulating the code is rooted in the theorem that "*All sub-paths in a shortest path are shortest paths themselves"* . We leverage this idea for the code.

1. Consider nodes that are of concern. This includes the Origin, Destination nodes as well as charging point nodes.
2. Find all shortest paths between each pair of concerned nodes. This is unfortunately the most time consuming part of the algorithm since it runs on O(2^n). However we reduce the time complexity by cutting branches on shortest paths that exceed the maximum distance range of the vehicle, since that is not of any concern for us.
3. In order to reduce space complexity, we remove any subpaths. For example if we know that the shortest path from 0 to 10 is 0-3-2-6-8-10 , then there is no point in saving separate subpaths like 3-2-6 or 2-6-8-10 because, thanks to the theorem for shortest subpaths, these paths will automatically be taken into account.
4. Added a recursive function that can be used by the vehicle limited by its distance range to jump from nodes of concern without jeopardizing the max distance it can cover. There is an additional branch cutting technique which discards all merged paths that don't start with the concerned origin node and ends in the concerned destination node.

   For example in the 1st toy example comprising 11 nodes, if the origin-destination pair is [0,10] , the charger nodes are [2,4,7,9]

   Consider if the shortest paths that are within the range of the vehicle are [[0, 1, 2], [0, 4], [2, 6, 7, 10], [4, 5, 9, 10]] , then the recursive merging function would output only the 2 possible shortest path merged combinations [[0, 1, 2, 6, 7, 10], [0, 4, 5, 9, 10]] . The recursion keeps the time complexity manageable due to memoization.
5. Once we have the shortest path merged combinations, we select the combination that corresponds with least overall distance which is our final shortest path with the constraint of a limited distance range vehicle with charging nodes.

To increase the complexity of the code :

1. We have considered that for an EV, in spite of the fact charging is an absolute necessity, there is a long time spent in charging an average EV (compared to a gasoline vehicle). To incorporate this, we have added a penalty that the EV is subjected to on the occasions that it uses a charging port.
2. For the 2nd toy example, which is a slightly larger graph, we have added a penalty factor of 80. This essentially means that if the EV were not to charge itself, it could have traversed an additional 80 units distance during the time it takes for charging.
3. The logic of penalizing is as follows :
    a. If the distance covered while traveling from origin to destination <= distance range, then there was no need to visit a charger. So the number of chargers visited is floor(shortest distance / distance range) = 0
    b. If the distance covered while traveling from origin to destination > distance range, the EV inevitably had to visit chargers on the way. The number of chargers the EV visited is again floor(shortest distance / distance range).
    c. This can be multiplied with the penalty factor for each charging station and added to the final result.

We decided to benchmark our results for all possible origin-destination pairs for a network and visualize what the shortest distance is between every origin-destination pair. We decided to build a heat map for the shortest distances for O-D Pairs. This also serves another important purpose : For impossible paths, we have built an exception case that outputs the shortest distance as -1. This sticks out in the visualization as the brightest tile and sends the point across that the driver needs to be cautioned against taking this Origin Destination pair to travel, It also alludes to a need for facility location problem, where it can give insights on paths that would need additional charging points to have network where all O-D pairs are accessible. For example in the heat map generated using the 2nd toy network, we notice that traveling from node 4 to node 13 is impossible for a vehicle of limited range. This would either prompt the driver to select a vehicle with improved range, or to a facility planner to build an additional charging point on the path from 4 to 13.

Since the spring layout in networkx was not plotting our network correctly as it randomly picks positions in the space, our shortest paths could not be visualized correctly (see Figure 5 for details). For this reason, we had to fit the model we had created to find the shortest path between origin-destination pairs into the geopandas framework. We then plot the nodes and edges like we did earlier in Figure (3) and to plot the shortest path between an origin-destination pair, we find the coordinates of all the nodes that are in the shortest path (these nodes it passes through in its shortest path is retrieved from the evchargepath function). A sample plot between 1 particular origin-destination pair is shown below:

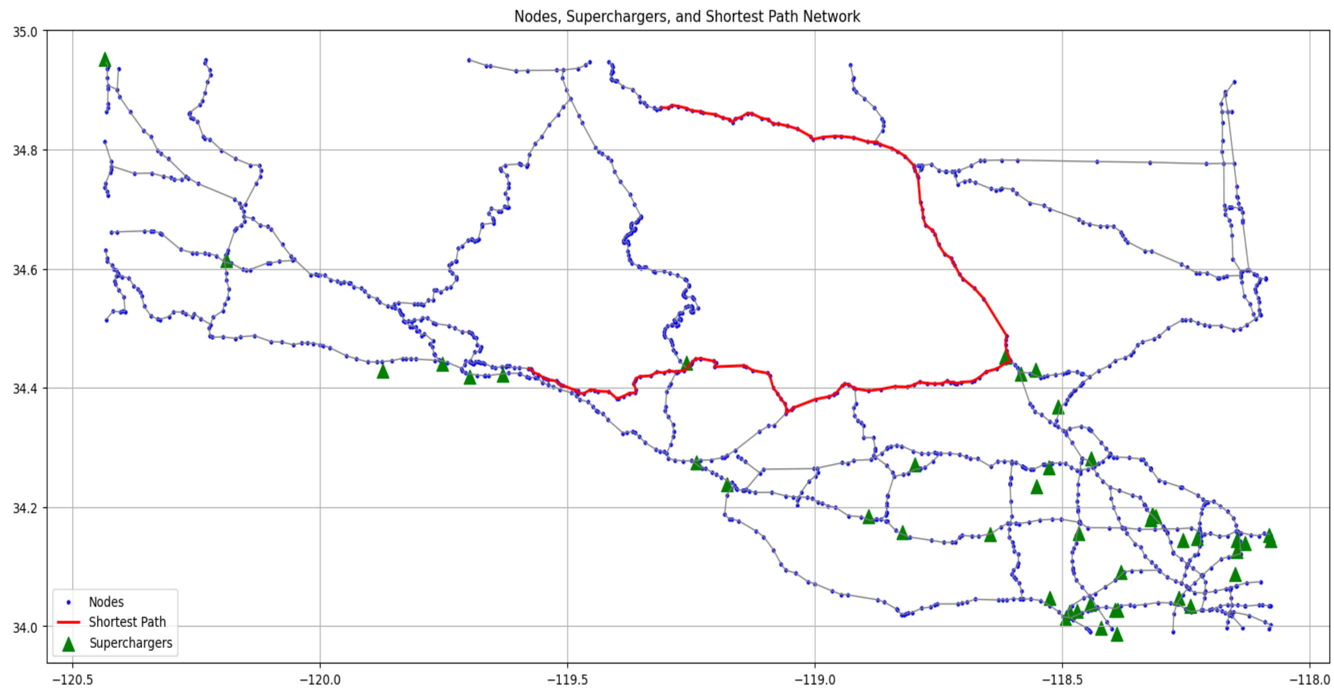

Figure 4: Shortest path between 1 origin-destination pair plotted on geographic space

The rest will be shown in the results section below.

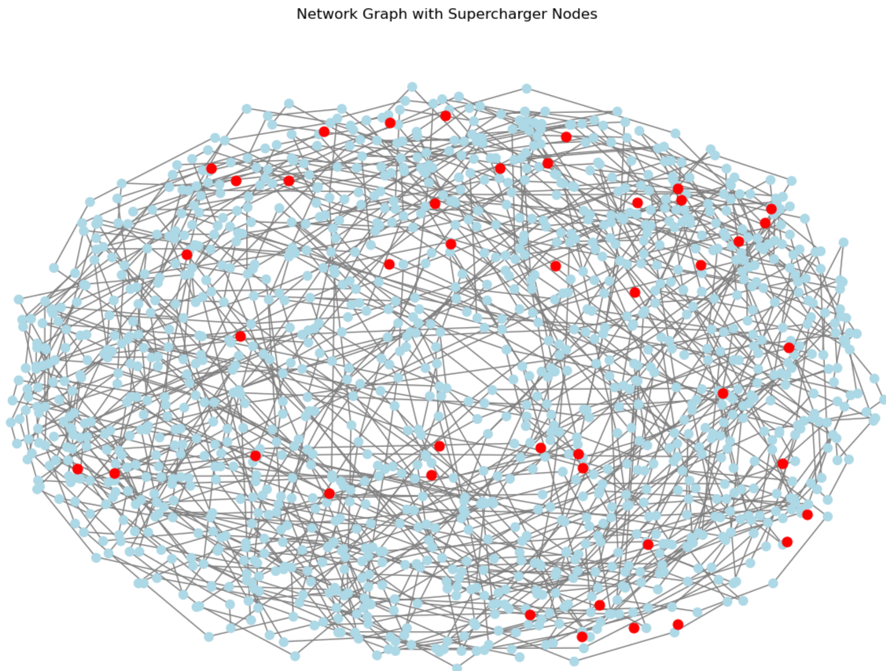

Figure 5: A visualization of the shortest path using spring layout

Results

Toy example:

2 cases:

1. Going through a charger location but not using it as we have enough range.

```
Origin :  0
Destination :  5
Shortest Path :  [0, 4, 5]
Shortest Distance :  110.0  including the distance of  0.0  that could have been travelled instead of charging
```

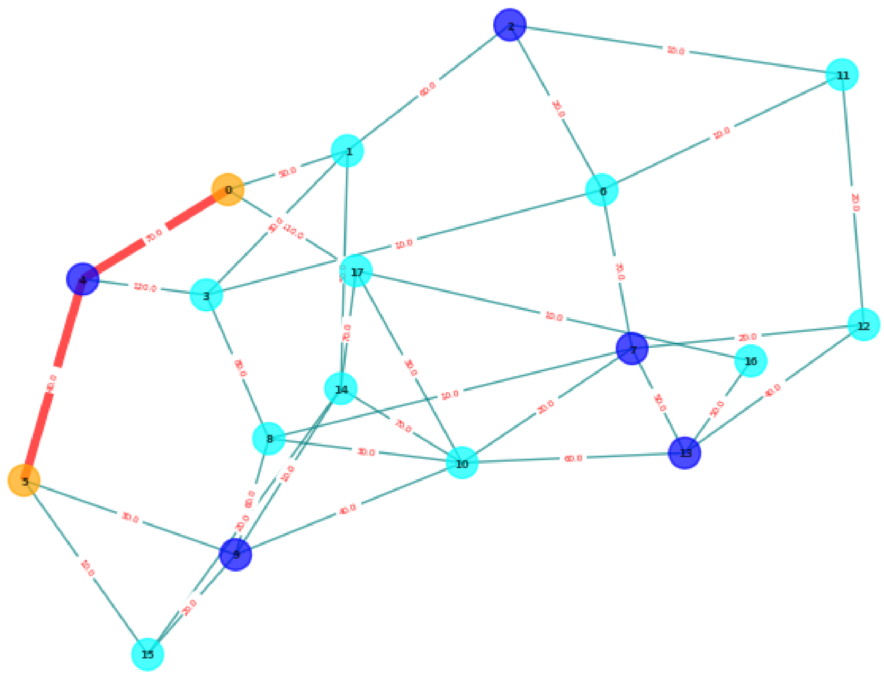

Figure 6: Shortest path from Origin 0 to Destination 5 on Toy Network

2. Going through a charger location and using it as we do not have enough range, incurring a distance penalty of 80 units.

```
Origin :  0
Destination :  8
Shortest Path :  [0, 1, 3, 6, 11, 12, 7, 8]
Shortest Distance :  230.0  including the distance of  80.0  that could have been travelled instead of charging
```

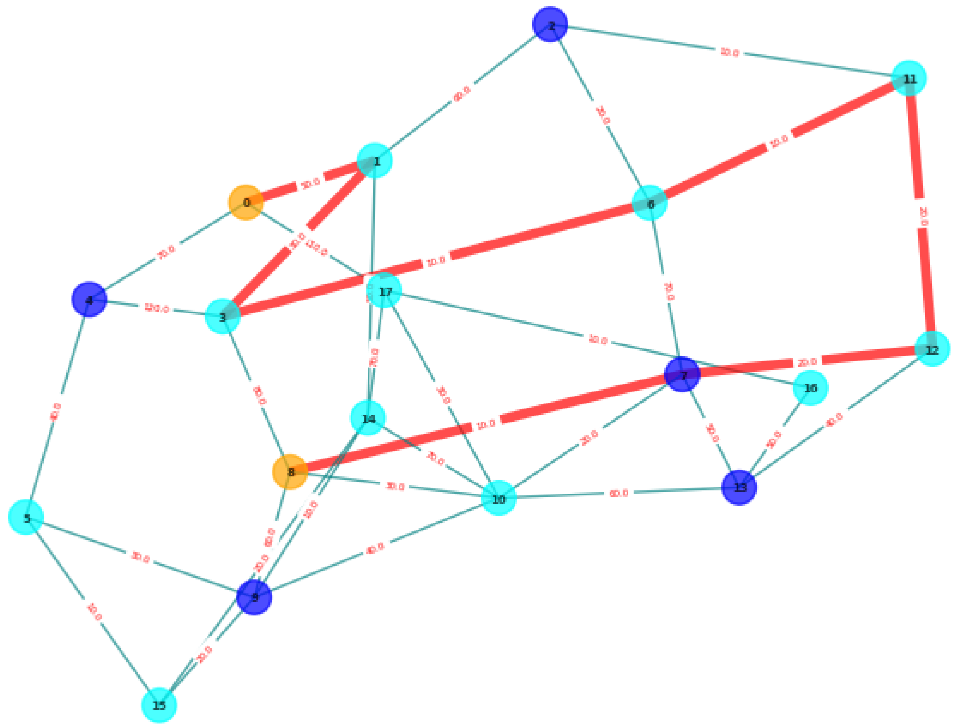

Figure 7: Shortest path from Origin 0 to Destination 8 on Toy Network

Supercharger Network :

As the number of Origin -Destination(OD) Pairs is large, we decided to sample the data, to generate random 5 OD pairs. This will generate 5C2 cases. Our function generates the graph for all the 10 cases. We have shared the largest and smallest graph for the purpose of the report, along with a case with the negative penalty, where there was no valid ride path.

```
Origin :  15788
Destination :  17207
Shortest Path :  []
Shortest Distance :  -6  including the distance of  -5  that could have been travelled instead of charging
```

```
C:\Users\nikhi\anaconda3\Lib\site-packages\geopandas\plotting.py:979: UserWarning: The GeoSeries you are attempting to plot is
composed of empty geometries. Nothing has been displayed.
  return plot_dataframe(data, *args, **kwargs)
```

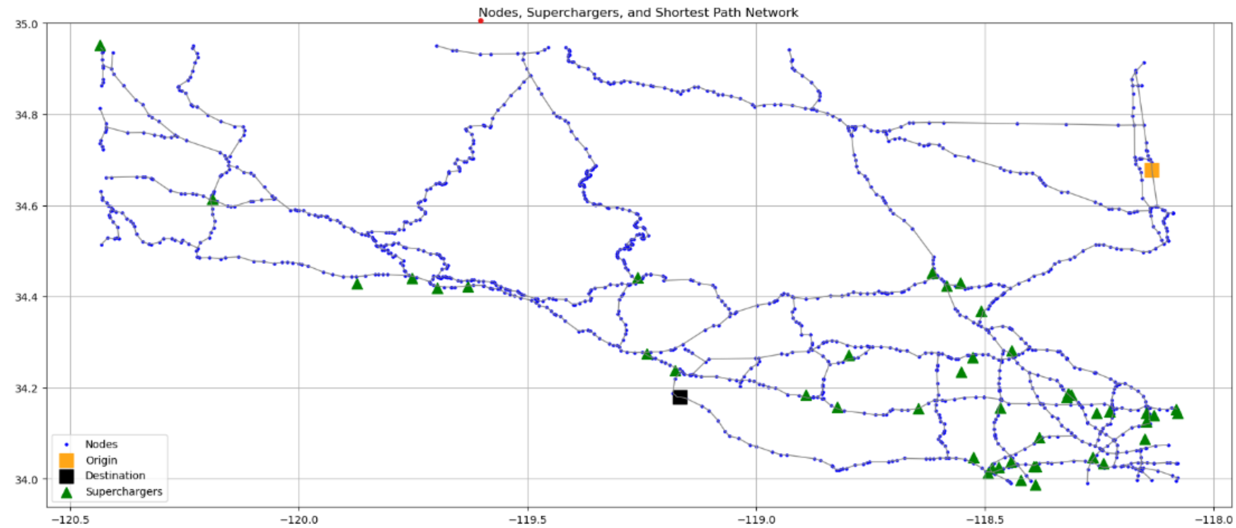

Figure 8: Network Visualization of the Network with no Valid Path

On completion of the analysis our function generates a visualization, which gives the shortest distance path and enumerates the nodes that are traversed through the shortest path. The penalty and the shortest distance is calculated.

* Note on Penalty and Vehicle Range: Our penalty is 5(for using the charger) and range was chosen to be an arbitrary value of 1.5. These values were arbitrarily selected to fit the data that was available.

In cases where the vehicle range was less than the distance to the next available supercharger, there was no valid path.

```
Origin :  16392
Destination :  16700
Shortest Path :  [16392, 16391.0, 16390.0, 16389.0, 16388.0, 16387.0, 16386.0, 16385.0, 16384.0, 16383.0, 16382.0, 16381.0, 163
80.0, 16379.0, 16378.0, 16375.0, 16376.0, 16377.0, 16264.0, 16263.0, 16673.0, 16672.0, 16671.0, 16670.0, 16669.0, 16668.0, 1666
7.0, 16666.0, 16665.0, 16664.0, 16663.0, 16680.0, 16681.0, 16679.0, 16674.0, 16675.0, 16676.0, 16677.0, 16678.0, 16583.0, 1658
4.0, 16585.0, 16586.0, 16587.0, 16588.0, 16582.0, 16581.0, 16580.0, 16579.0, 16578.0, 16577.0, 16576.0, 16575.0, 16574.0, 1657
3.0, 16572.0, 16571.0, 16570.0, 16569.0, 16568.0, 16567.0, 16566.0, 16565.0, 16564.0, 16563.0, 16562.0, 16561.0, 16319.0, 1649
6.0, 16495.0, 16494.0, 16702.0, 16701.0, 16700]
Shortest Distance :  0.9336389999999997  including the distance of  0  that could have been travelled instead of charging
```

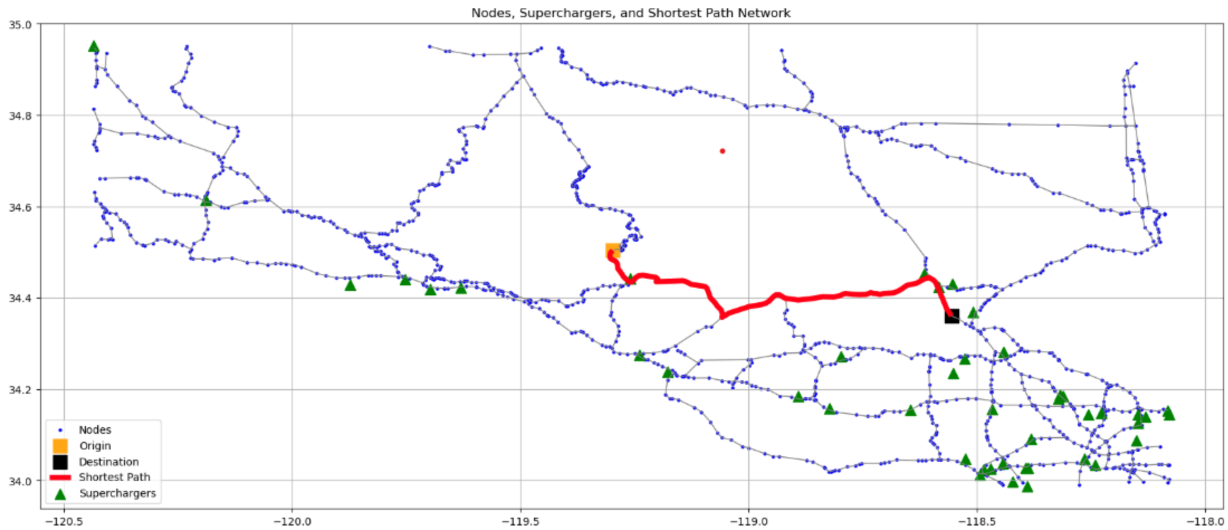

Figure 9: Network Visualization of the path with the maximum shortest path in the sample

```
Origin :  16392
Destination :  16665
Shortest Path :  [16392, 16391.0, 16390.0, 16389.0, 16388.0, 16387.0, 16386.0, 16385.0, 16384.0, 16383.0, 16382.0, 16381.0, 163
80.0, 16379.0, 16378.0, 16375.0, 16376.0, 16377.0, 16264.0, 16263.0, 16673.0, 16672.0, 16671.0, 16670.0, 16669.0, 16668.0, 1666
7.0, 16666.0, 16665]
Shortest Distance :  0.30232000000000003  including the distance of  0  that could have been travelled instead of charging
```

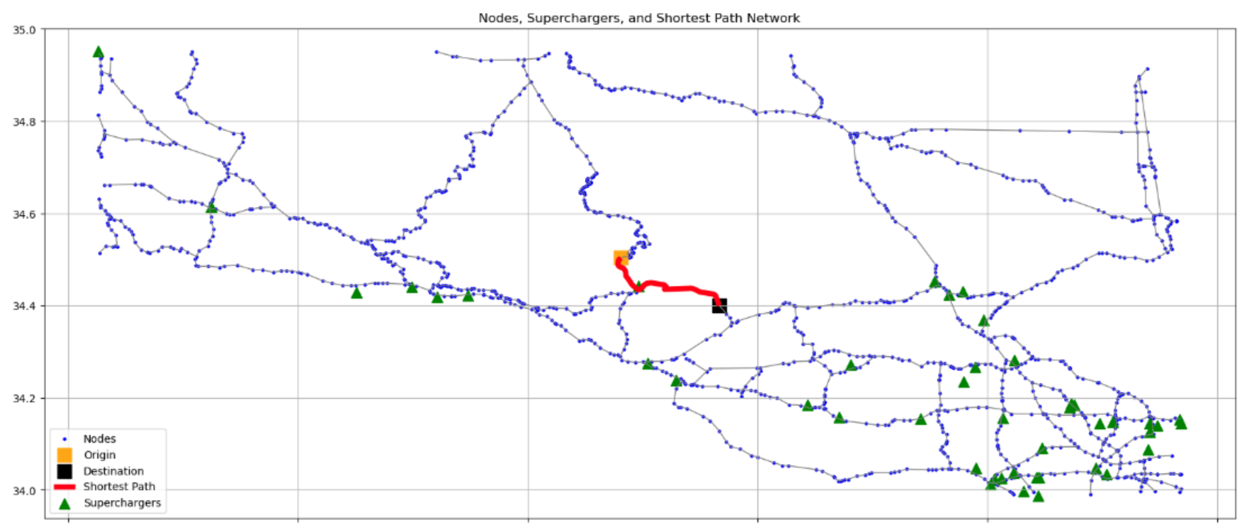

Figure 10: Network Visualization of the path with the shortest distance in the sample

HeatMap

1. Toy example 2

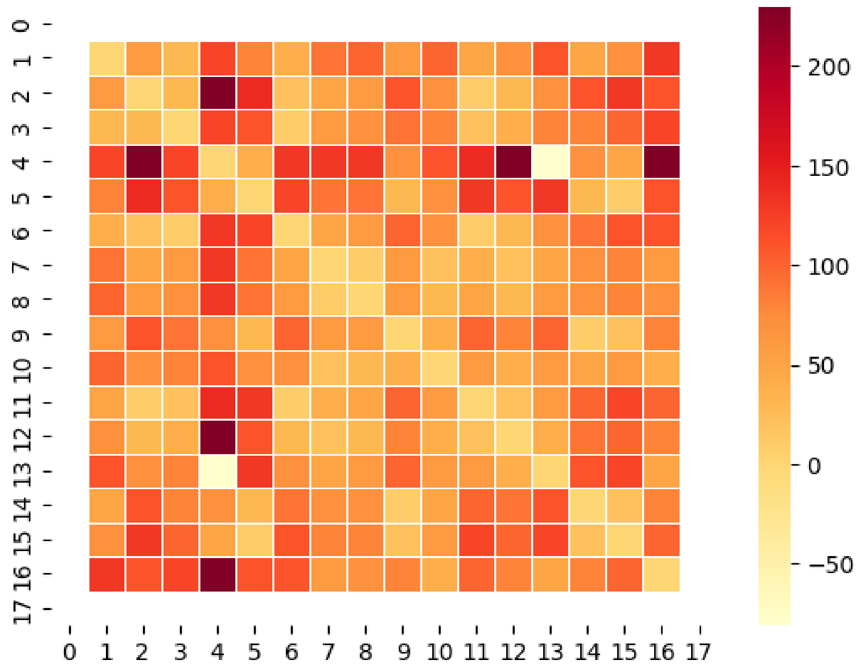

Figure 11: Heat Map for the Toy Network

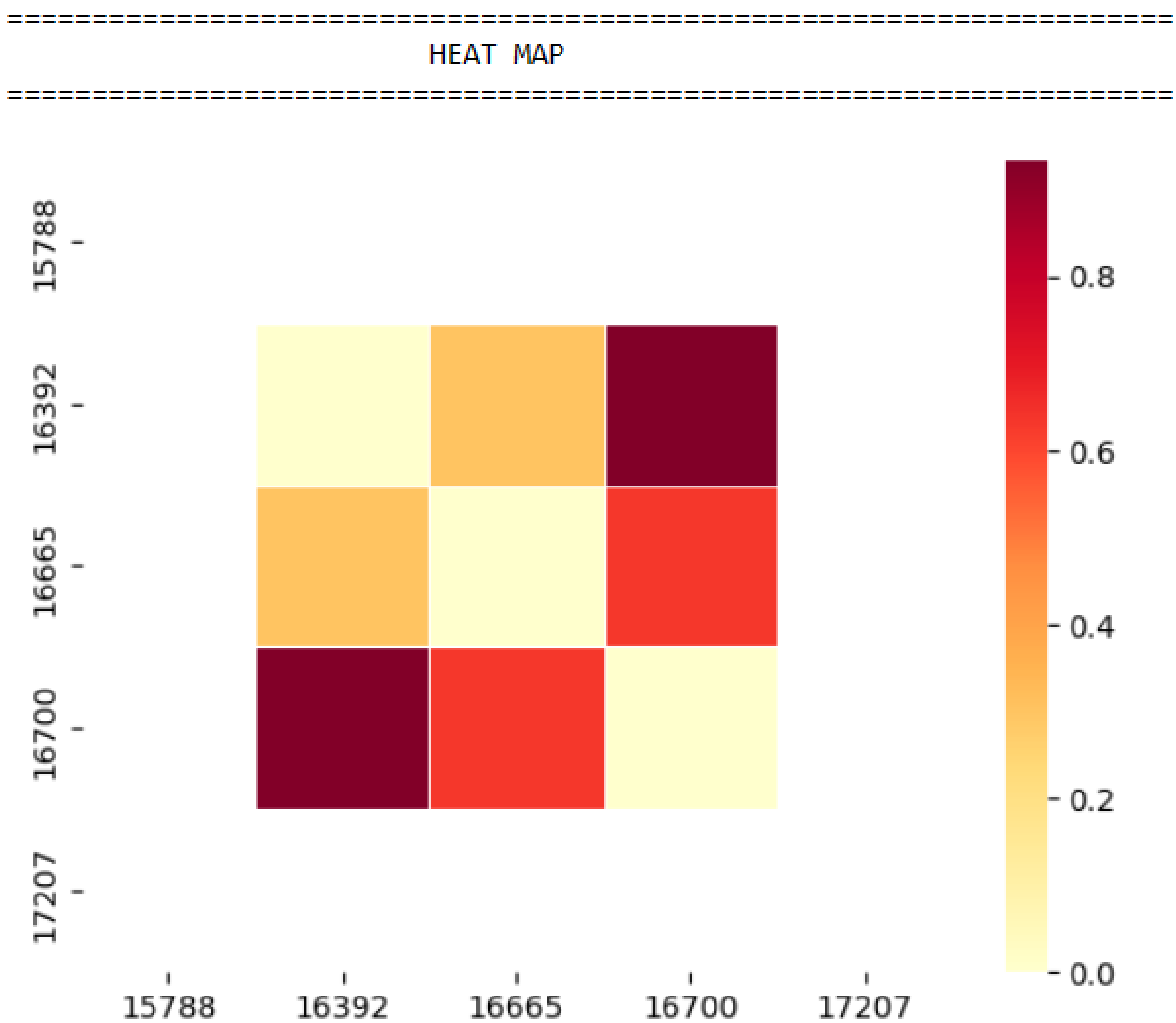

Figure 12: Heat Map for the Sampled Nodes

Future Work

Variable Charging Rates: Chargers with different wattages will recharge the vehicle at varying speeds, affecting the overall travel time.

Interactive Route Selection: Implementation of a Python widget that allows users to select the start and end locations and choose the type of EV, thus customizing the route planning experience.

References

Datasets :

https://supercharge.info/data

https://users.cs.utah.edu/~lifeifei/SpatialDataset.htm

https://github.com/utsavm2/EV_Charging_Optimization/tree/main/data