ECE 232E Project 4

Yifan Shu, Chengshun Zhang, Xuan Yang, Yifan Zhang

Introduction

In this project we explored graph theory theorems and algorithms, by applying them on real data. In the first part of the project, we considered a particular graph which models correlations between stock price time series. In the second part, we analysed traffic data on a dataset provided by Uber.

Part 1: Stock Market

In this part, we will study data from stock market. The goal of this part is to study correlation structures among fluctuation patterns of stock prices using tools from graph theory.

Question 1.

In this project, we firstly read data from Name_sector excel file, and output their names of those companies as an array. Then we read the closing stock price of each company at different dates. Based on those prices, we calculate the log-normalized return and find its correlation.

Theoretically, the lower bound and upper bound for the correlation should be -1 and 1, which means nothing in common for value -1, and all in common for value 1. In our case, the calculated lower bound is -0.95536 and the upper bound is 0.99909.

Instead of the regular return, the log-normalized return makes the dataset closer and denser, which offers better result in the plot. The return data sometimes contains zero values, which cannot be calculated by log, and thus we add a normalized value 1 to all of them.

Question 2.

For this part, we first construct a correlation graph as Fig.1.1, whose node is the sectors' name and weighted edge calculated by: $w_{ij} = \sqrt{2(1-p_{ij})}$.

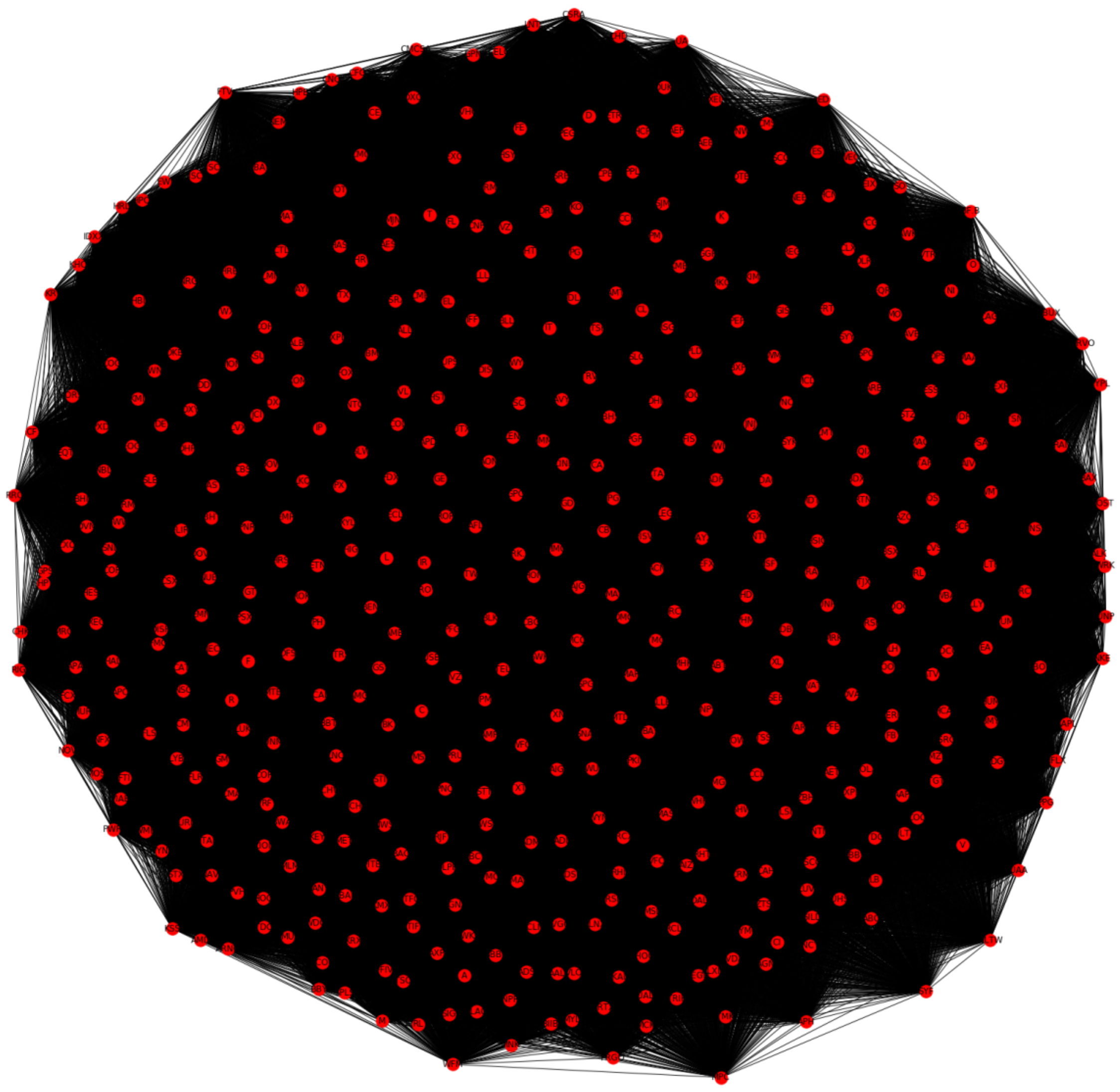

Figure 1.1. The Correlation Graph of the Stock Market

Then we plot a histogram showing the un-normalized distribution of edge weights as follows Fig.1.2.

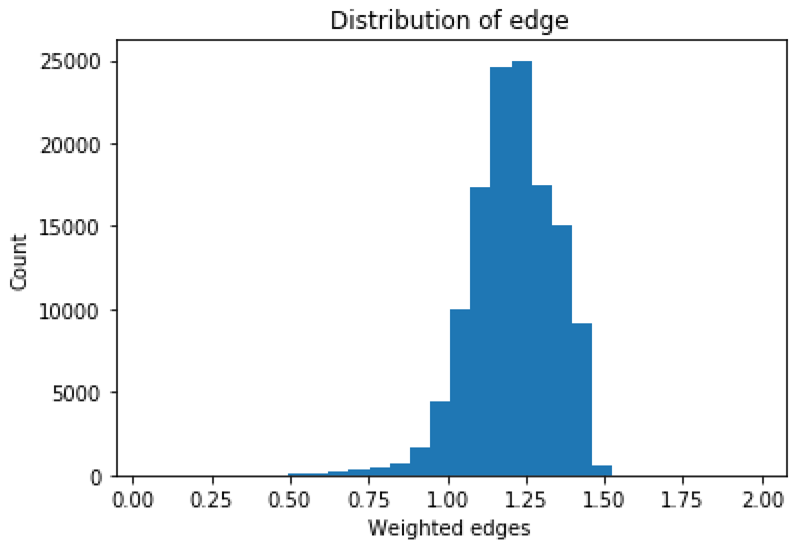

Figure 1.2. The Un-Normalized Distribution of Edge Weights for the Stock Market Graph

Question 3.

In this part, we extract the minimum spanning tree attributes from the correlation graph. Then we plot the minimum spanning tree and randomly color-code the nodes based on sectors' name as follows.

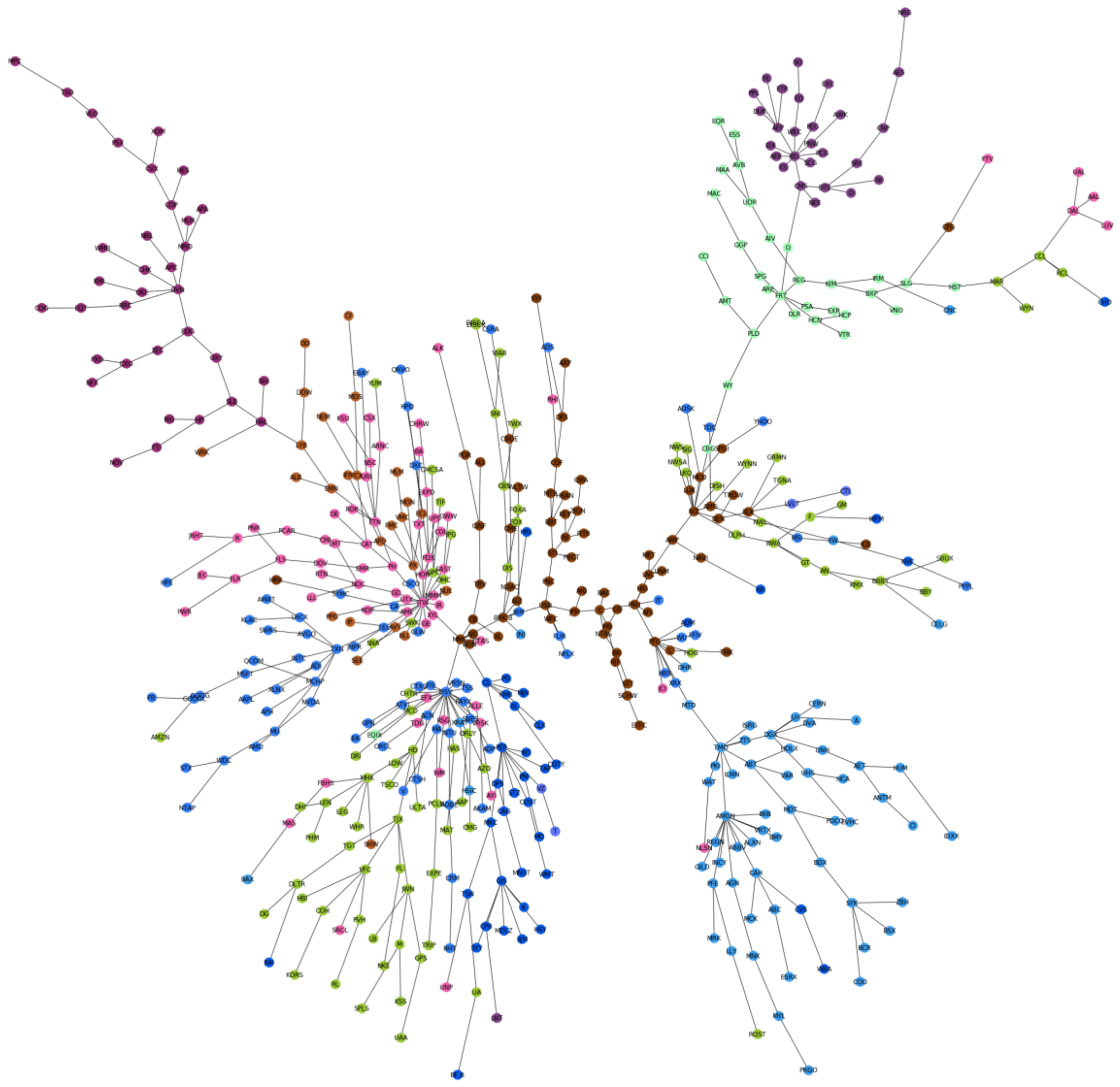

Figure 1.3. Minimum Spanning Tree of the Correlation Graph

From the plot, we can find that nodes with the same color and same sector tends to connect together, which is called vine cluster, and only have two nodes connected to other clusters. Those nodes in the same sector have more attributes in common, and thus they have larger correlation and less weight. To interpret this fact in our stock predicting project, those companies in the same sector might be influenced at the same time by the same event on the same level. This is how we can predict the future of the stock. In minimum spanning tree algorithm, we tend to pick less weight of edges, and thus MST plot a tree that has larger correlation and less weight.

Question 4.

In this part, we are required to predict the market sector of an unknown stock, and we define the metric as follows: $\alpha = \frac{1}{|V|} \sum_{v_i \in V} P(v_i \in S_i)$., where S_i is the sector of node i.

The first method is $P(v_i \in S_i) = \frac{|Q_i|}{|N_i|}$, where Q_i is the set of neighbors of node i that belong to the same sector as node i, and N_i is the set of neighbors of node i. The second method is $P(v_i \in S_i) = \frac{|Q_i|}{|V|}$. For our project, we get alpha equaling to 0.78879 for the first method, and 0.04554 for the second method. If we name them alpha1 and alpha2, we can easily see that the alpha1 is much larger than alpha2, which means the method 1 is much better, because the MST gives us clusters of nodes and one node in the tree tends to have the same sector with the majority of its neighbors. This phenomenon is mentioned above in the question 3. Meanwhile, the second method predict the unknown stock based on the whole market instead of its neighbors, and thus the result contains too many data which are less relevant to the node we want to explore.

Question 5.

In this question, all the process are the same compared to the previous questions, except extracting the Mondays' data instead of the whole dataset. Then based on the new dataset, we interpret them, and plot the minimum spanning tree as shown in Fig.1.4. At this time, we can find that the vine cluster of the MST is less obvious compared to the previous one. We still can find clusters, but they are not as obvious as the previous MST, which means clusters crossing each other. This result might be caused by that the correlation of weekly data is less stronger than the daily data, and thus decrease the size of clusters.

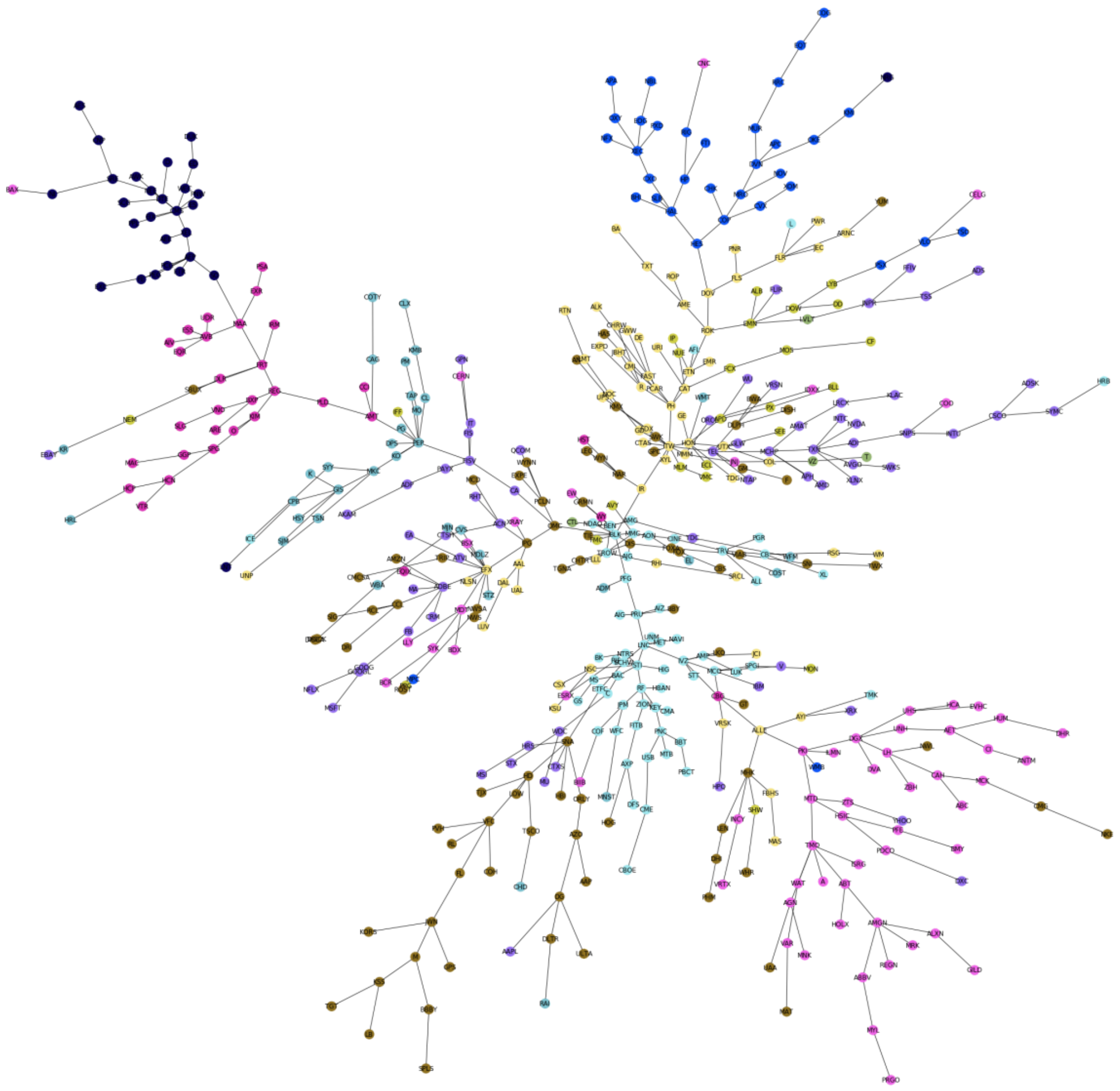

Figure.1.4. The Minimum Spanning Tree from the Correlation Graph Based on Weekly Data

Part 2: Let’s Help Santa

In this part we are using the network theory to facilitate gift delivery problem for the next christmas, given the statistics about transportation dynamics provided by Uber

Question 6.

We read the data with pandas read_csv and read_json and built the graph in which nodes corresponding to locations and undirected weighted edges corresponding to mean travelling times between each pair of locations. We added the street address as attribute Display name and the mean of the coordinates of the polygon's corners as attribute Location to the nodes in the graph. We also removed isolated nodes and kept only the giant connected components of the graph G. The number of nodes and edges in G is:

Number of nodes: 1898

Number of edges: 321703

Question 7.

We called minimum_spanning_tree function in networkx to build the minimum spanning tree of G. Some of the street addresses of two endpoints of edges are shown below.

400 Northumberland Avenue, Redwood Oaks, Redwood City ---- 1500 Oxford Street, Palm Park, Redwood City
400 Northumberland Avenue, Redwood Oaks, Redwood City ---- 100 Fifth Avenue, South Fair Oaks, Redwood City
18300 Sutter Boulevard, Morgan Hill ---- 17300 Lotus Way, Morgan Hill
18300 Sutter Boulevard, Morgan Hill ---- 1900 Alpet Drive, Morgan Hill
3200 Huntsman Drive, Rosemont Park, Sacramento ---- 8900 Cal Center Drive, Sacramento
3200 Huntsman Drive, Rosemont Park, Sacramento ---- 9500 River Rose Way, Premier Garden, Sacramento
100 Carlsbad Circle, Vacaville ---- Interstate 505, Vacaville
100 Carlsbad Circle, Vacaville ---- 500 Morningstar Way, Vacaville
Unnamed Road, Vacaville ---- 400 Bowline Drive, Vacaville
Unnamed Road, Vacaville ---- 100 Monte Verde Drive, Vacaville
700 Carlsbad Court, Petaluma ---- 0 Maria Drive, Petaluma
700 Carlsbad Court, Petaluma ---- 200 Ely Road North, Petaluma
700 Carlsbad Court, Petaluma ---- 900 Telford Lane, Petaluma
500 Hyde Street, Tenderloin, San Francisco ---- 200 Myrtle Street, Tenderloin, San Francisco
500 Hyde Street, Tenderloin, San Francisco ---- 200 Hyde Street, Tenderloin, San Francisco
3200 Nightingale Drive, Modesto ---- 2300 Pamela Lane, Modesto
3200 Nightingale Drive, Modesto ---- 5400 Stoddard Road, Modesto
3200 Nightingale Drive, Modesto ---- 2200 Northridge Drive, Modesto

By looking up the addresses in google map, the results are quite intuitive. We found that two addresses are indeed close to each other, which has very short mean travelling time and thus small edge weight. This is exactly the property that a minimum spanning tree has.

Question 8.

We randomly picked 1000 different triangles, each time three points. If some edge does not exist between two points, we just re-picked new points. After we successfully picked three points and each edge exists, we calculated whether they satisfy triangle inequality. The percentage of triangles that satisfy the triangle inequality is 97.6%.

Question 9.

We created a multi graph from the original graph G and called eulerian_circuit function in networkx to get the euler cycle of the multi graph. Since we need the upper bound of the approximate algorithm and we have the following relation:

$$\begin{aligned} \textit{Length of the tour returned by the algorithm} &\le \textit{Distance covered by Euler Cycle} \\ &\le 2 \times \textit{weight of minimum spanning tree} \\ &\le 2 \times \textit{weight of minimum tour length} \end{aligned}$$

Thus, we have

$$\rho = \frac{\textit{Approximate TSP Cost}}{\textit{Optimal TSP Cost}} \le \frac{\textit{Distance covered by Euler Cycle}}{\textit{Optimal TSP Cost}} \le \frac{\textit{Distance covered by Euler Cycle}}{\textit{weight of minimum spanning tree}}$$

So we used the sum of the weights in euler cycle as numerator and the sum of the weights of minimum spanning tree as denominator and calculated

$$\rho = 1.5217849499512983$$

Question 10.

The trajectory of Santa’s travel is shown in Fig.2.1.

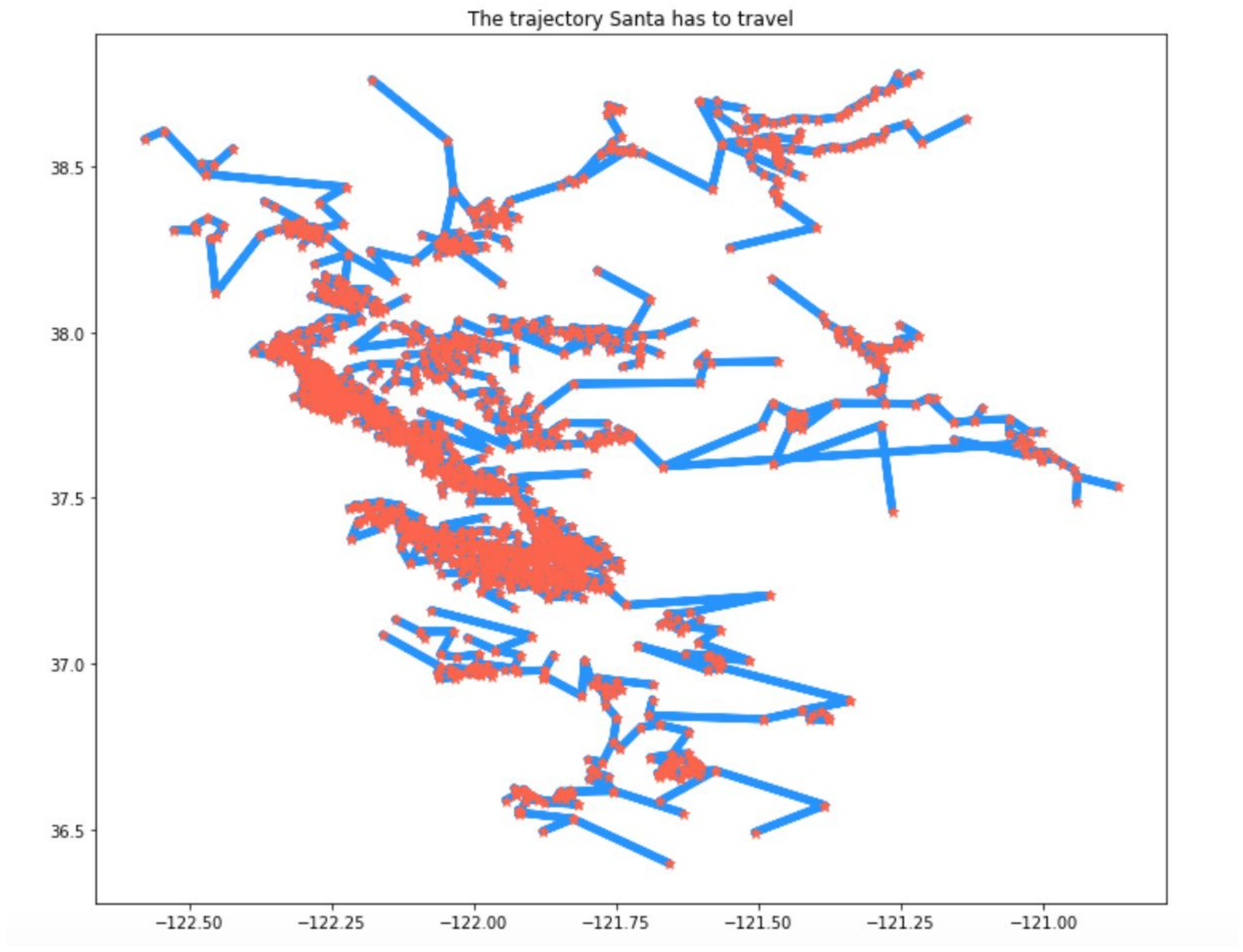

Figure 2.1. The Trajectory that Santa has to Travel

Question 11.

Here, to estimate the map of roads without using actual road data, we applied Delaunay Triangulation Algorithm to the nodes coordinates and plotted the road mesh. We also created the graph G_{Δ}, where the nodes are different locations and edges are produced by triangulation. The plot is as below:

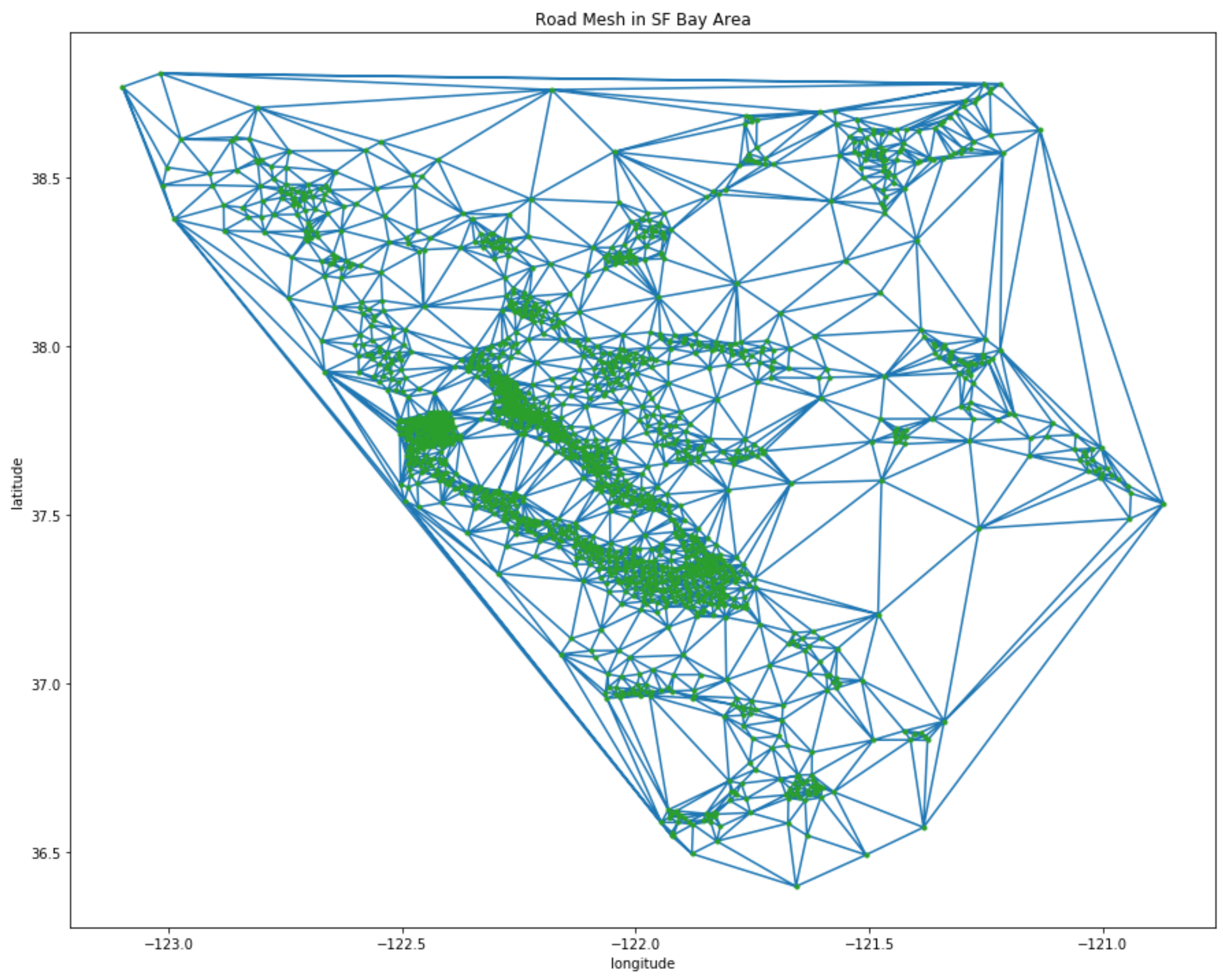

Figure 2.2. Road Mesh using Delaunay Triangulation Algorithm

In graph G_{Δ}, there are 1898 nodes in total with 5680 edges. From Figure 2.2, we could see that the profile looks similar to Figure 10-1 but with more triangular edges. The green points are the nodes in the graph, which show similarity with Figure 10-1, where Santa needs to visit. And the blue triangles show the paths where the triangle inequality gets satisfied.

Question 12.

Here we used the following equations to calculate the traffic flow for each road in terms of cars per hour:

Here we assume:

- Each degree of latitude and longitude ≈ 69 miles

- Car length ≈ 5m = 0.003mile
- Cars maintain a safety distance of 2 seconds to the next car
- Each road has 2 lanes in each direction

So here we set the constants as:

- miles_per_degree = 69 (miles)
- car_len = 0.003 (miles)
- safe_time = 2/3600 (hour)
- lanes = 2

If there is an edge between node u and node v in graph G_{Δ}, we let mean_time equals the mean travel time between the two nodes (the unit is seconds). But if not, we will set the mean_time = 1e8. Then we used the following equations to calculate the traffic flow for each road:

- Distance_between_nodes = $\sqrt{(u(x) - v(x))^2 + (u(y) - v(y))^2}$
- Car_speed = Distance_between_nodes / (mean_time/3600)
- Safety_distance = Car_speed * safe_time
- Car_flow = Car_speed / (car_len + Safety_distance) * lanes

Then we added the Car_flow for each road we calculated to the edge attribute of capacity in graph G_{Δ}.

Question 13.

Here we calculated the maximum number of cars that can commute per hour from Stanford to UCSC and the number edge-disjoint paths between the two spots. The results are as below:

(1) Maximum number of cars that can commute per hour from Stanford to UCSC is: 14866.477294089982

(2) There are 5 edge-disjoint paths:

- 2607 -> 1363 -> 1869 -> 744 -> 2242 -> 1980 -> 1968
- 2607 -> 1736 -> 1762 -> 1210 -> 2458 -> 1955 -> 1980 -> 1431 -> 1968
- 2607 -> 1726 -> 1736 - > 1737 -> 744 -> 938 -> 1989 -> 1431 -> 1424 -> 1968
- 2607 -> 1725 -> 1733 -> 1209 -> 1762 -> 1763 -> 1980 -> 2241 -> 1968
- 2607 -> 2240 -> 1363 -> 1737 -> 1763 -> 1955 -> 1171 -> 2241 -> 748 -> 1968

(3) The zoomed figures are as following:

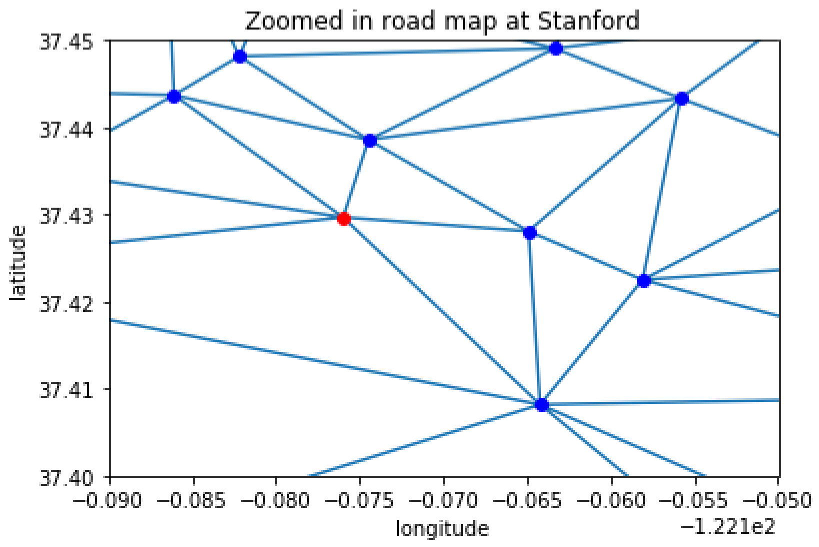

Figure 2.3. Zoomed road map at Stanford

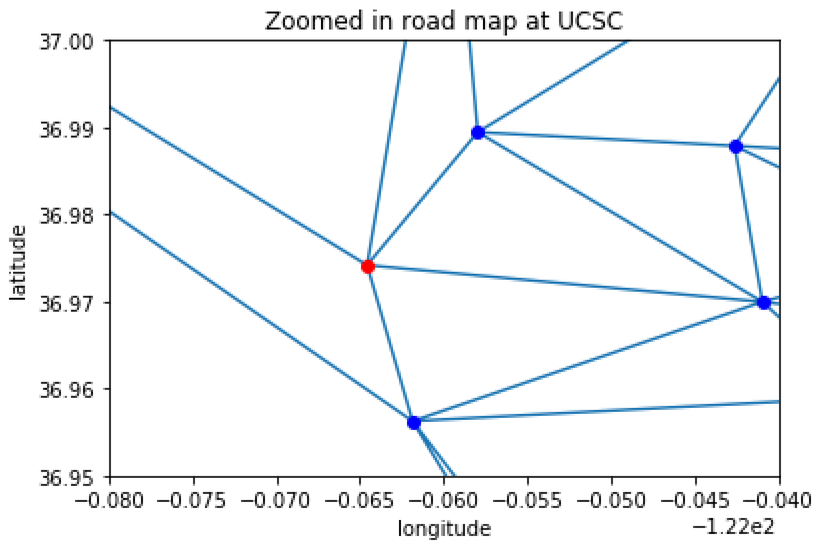

Figure 2.4. Zoomed road map at UCSC

Here we could see from the results and figures that there are actually also 5 paths on the road map.

Question 14.

Here we created $\overline{G}_{\Delta}$, which we applied a threshold on travel time of the roads in G_{Δ} to remove the fake edges and the resulting graph is $\overline{G}_{\Delta}$. We found that the maximum traffic flow for fake edges is about 3.5, so here we set the threshold to be 5. The graph of $\overline{G}_{\Delta}$ is as below:

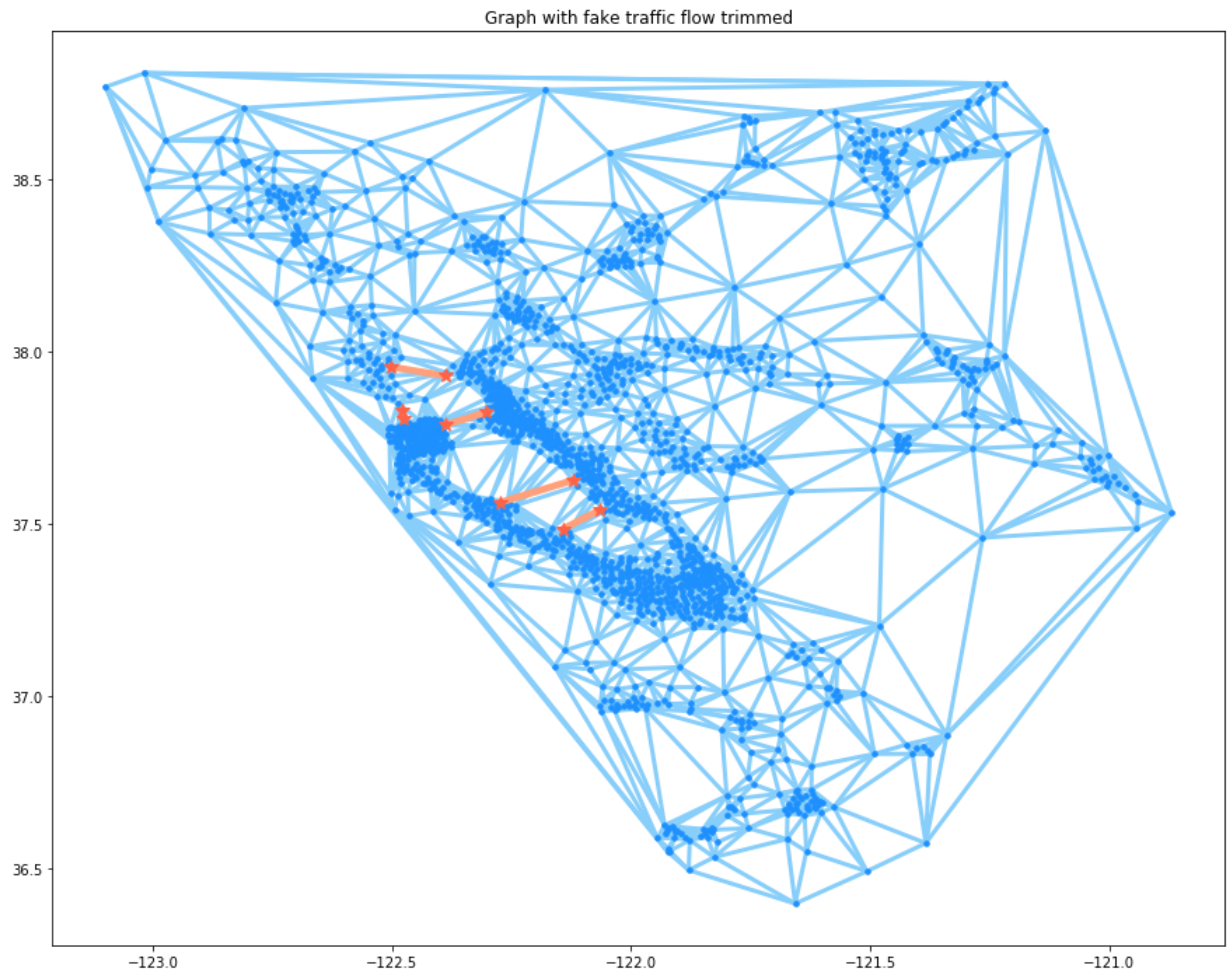

Figure 2.5. $\overline{G}_{\Delta}$ with fake edges trimmed

In this graph, the blue points and triangular edges are from graph $\overline{G}_{\Delta}$, and the red points with red edges are real bridges:

- Golden Gate Bridge: [[-122.475, 37.806], [-122.479, 37.83]]
- Richmond, San Rafael Bridge: [[-122.501, 37.956], [-122.387, 37.93]]
- San Mateo Bridge: [[-122.273, 37.563], [-122.122, 37.627]]
- Dambarton Bridge: [[-122.142, 37.486], [-122.067, 37.54]]
- San Francisco - Oakland Bay Bridge: [[-122.388, 37.788], [-122.302, 37.825]]

We could see that those bridges are almost preserved in the new trimmed graph. Since there is little error between the coordinates given from those real bridges and the nodes set in graph, we could see the lines and nodes do not overlap perfectly.

Question 15.

Here we repeated what we did in question 13 on $\overline{G}_{\Delta}$ with fake edges trimmed on G_{Δ}. The results are as below:

(1) Maximum number of cars that can commute per hour from Stanford to UCSC is: 14866.477294089984

(2) There are also 5 edge-disjoint paths.

(3) The zoomed figures are as following:

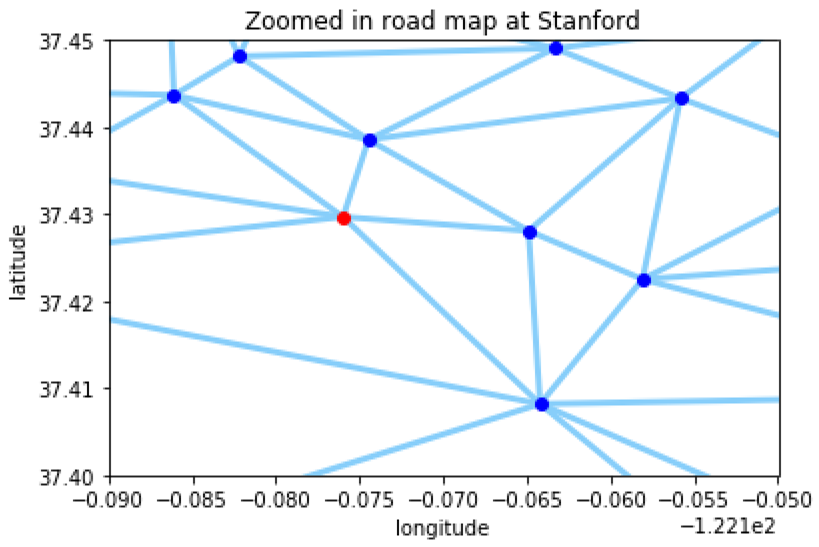

Figure 2.6. Zoomed road map at Stanford

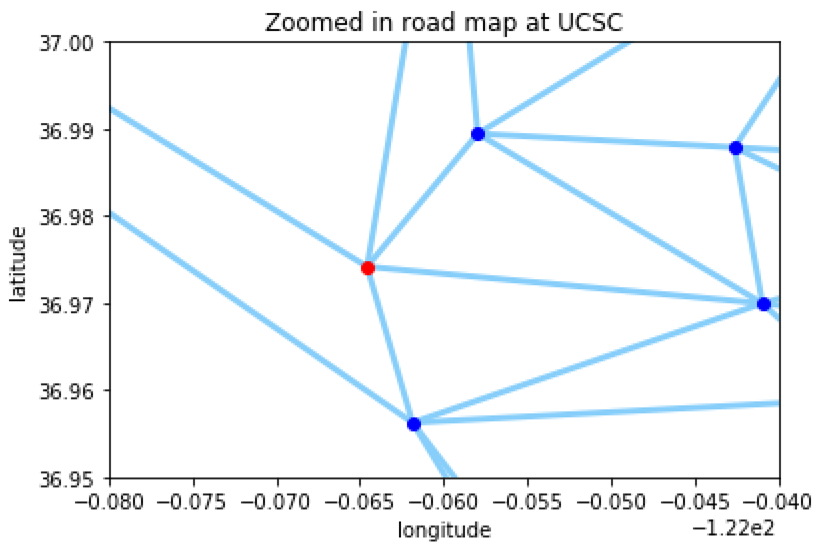

Figure 2.7. Zoomed road map at UCSC

We could see from the results that there are not significant changes in $\overline{G}_{\Delta}$ with fake edges trimmed on G_{Δ}.