ECE 232E Project 4 Report

Yingbo (Max) Wang, 604-593-537

Question 6.

As explained in the specification, we read the San Francisco Monthly Aggregated Traffic data file line by line and select entries that has “month” attribute equals 12. One edge is created between each pair of source and destination node IDs with weight equals the average traveling time between the two nodes.

This graph has 2710 vertices and 321713 edges excluding the extra nodes in the Geolocation metadata file. Then we find the Giant Connected Component of this graph; it has 1898 vertices and 321703 edges. Thus, about 70% of the nodes form the biggest cluster with about 99.997% of the edges in the original graph.

Question 7.

The Minimum Spanning Tree has 1898 vertices and 1897 edges. 15 edges in the MST are randomly sampled and presented in the following table. We include the edge ID as a tuple of (source node ID, target node ID), the average traveling time between the two nodes, and the address and coordinate of both the source location and target location.

The results make sense as they correspond to the travel time given on Google Maps. Locations pairs that have higher travel times are either geographically farther away or with frequent heavy traffic in between. For example, it’s a relatively short drive from “3600 Parsons Court, Castro Valley” to “13700 Westboro Drive, Alum Rock, San Jose” of 39.2 miles – Google Maps estimates it to take 55 mins with medium traffic and our dataset shows 67.05 mins, which is very close. On the other hand, a drive from “200 North 5th Street, Central San Jose, San Jose” to “5700 Wildwood Drive, Marysville” is 154 miles and takes much longer with about 3 hour and 16 mins estimated by Google Maps. That’s close to the average traveling time in our dataset of about 225.455 minutes.

Table. Street Addresses of Two Endpoints on a few Edges

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Edge | Average Traveling Time | From Address | From Coordinate | To Address | To Coordinate |
| (833, 1433) | 244.71 | 200 El Cajon Avenue, Covell Park, Davis | (-121.752, 38.565) | 8100 Dunes Court, Sacramento | (-121.402, 38.459) |
| (165, 1694) | 163.215 | 2200 Franklin Street, Pacific Heights, San Francisco | (-122.424, 37.794) | 700 Folsom Street, South of Market, San Francisco | (-122.402, 37.782) |
| (850, 1166) | 82.685 | 1300 Greenborough Drive, Cirby Ranch, Roseville | (-121.275, 38.723) | 600 Cinnabar Court, Diamond Oaks, Roseville | (-121.28, 38.764) |
| (927, 1375) | 135.85 | 800 Aspen Way, Palo Verde, Palo Alto | (-122.109, 37.426) | 700 Golden Avenue, Lodi | (-121.259, 38.146) |
| (978, 1383) | 226.095 | 7600 Wilbur Way, Sacramento | (-121.398, 38.477) | 100 Tomlinson Drive, Folsom | (-121.176, 38.702) |
| (424, 1703) | 149.23 | 5100 61st Street, Southeastern Sacramento, Sacramento | (-121.437, 38.529) | 2600 21st Street, Mission District, San Francisco | (-122.406, 37.759) |
| (506, 923) | 175.005 | 2700 Marty Way, Southwestern Sacramento, Sacramento | (-121.494, 38.557) | 1300 North Carpenter Road, Modesto | (-121.033, 37.657) |
| (545, 807) | 225.455 | 200 North 5th Street, Central San Jose, San Jose | (-121.889, 37.342) | 5700 Wildwood Drive, Marysville | (-121.55, 39.124) |
| (423, 1105) | 152.64 | 200 Locust Avenue, San Rafael | (-122.508, 37.982) | Unnamed Road, Castro Valley | (-121.987, 37.621) |
| (364, 1171) | 114.755 | 5300 Laguna Park Drive, Elk Grove | (-121.433, 38.43) | 3100 Forest Lake Road, Pebble Beach, Del Monte Forest | (-121.945, 36.588) |
| (1266, 1269) | 243.805 | Spring Creek Trail, Santa Rosa | (-122.666, 38.451) | 500 Yosemite Drive, Milpitas | (-121.899, 37.421) |
| (1658, 1820) | 154.725 | 6800 Fairmount Avenue, El Cerrito | (-122.298, 37.902) | 2900 Glacier Street, Bohemian Village, Sacramento | (-121.408, 38.62) |
| (601, 1274) | 67.05 | 3600 Parsons Court, Castro Valley | (-122.084, 37.706) | 13700 Westboro Drive, Alum Rock, San Jose | (-121.827, 37.359) |

Question 8.

We first find all triangles in the Giant Connected Component graph by getting cliques with size 3. Then 1000 triangles are randomly selected and weights of its three edges are tested with the triangle inequality. We find that the percentage of those triangles satisfying the triangle inequality equals about 92.60%.

The result makes sense as most of the times going from place A to place B directly is faster than adding an intermediate stop (going from place A to place C and then from place C to place B). The few exemptions can be caused by traffic conditions: the direct route between the two locations may usually have heavy traffic, so going a longer detour with less traffic could actually save time.

Question 9.

To calculate the cost of the 1-Approximate TSP algorithm, we first implement the Tree Algorithm to find the optimal path that Santa travels: it equals an Eulerian circuit of the multigraph of the graph’s MST, and then only keeps its unique nodes (with a shortest path connecting the pairs without an edge connecting the two nodes). The Eulerian path can be calculated by first finding a random cycle on the multigraph from a starting node and then repeatedly adding cyclic paths starting from nodes already in the path, until no node in the path is incident to a remaining edge.

The generated Eulerian path travels 3795 steps and is presented as a list of node IDs EULERIAN\_PATH.py. The calculated Santa’s tour path travels 2250 steps and the list of node IDs is presented in TOUR\_PATH.py. The result makes sense according to the following visualization in the next question, because the traveler thoroughly visits every location in the Northern California area while adjacent locations are reasonably close to each other. A list of coordinates in this tour is shown in tour\_path\_coordinates.csv.

Then we sum up the weights of all traversed edges to get the cost of this tour to be 504399.755. Also, the cost of the optimal TSP path is lower bounded by the cost of MST that equals 289315.675 (the summed weights of all edges in the graph’s MST). Thus, in this problem, the upper bound on the empirical performance of the approximate algorithm equals:

The result makes sense because it’s less than two times the cost of MST.