STAT 454 Final Project

Quickest Route To/From Work:

A Shortest Path Problem

By: Bryce Landry

C00025175

Abstract:

In this report, I will be discussing and solving an instance of the shortest route problem; specifically, the quickest route that I can travel to and from work every day. In solving this problem, I will use some theory of network models from chapter 8 of "Operations Research: Applications and Algorithms, 4e", some online resources for guidance/theory, as well as MATLAB to simulate the problem.

In order to properly calculate the quickest route, three variables must be kept in mind:

- Speed limit of the roads
- Traffic level at the time of transport
- Length of each road

However, for the sake of simplicity and time, we will only use the travel times of each road produced by google maps as the weights of the graph. Google maps uses these variables, along with a few others, in order to calculate the times for us, making our job exponentially easier.

Introduction:

By simulating and solving this problem, I will be able to optimize the amount of time it takes for me to get to and from work every day. I will use Google maps in order to collect the travel times of each road. Because travel may vary at different parts of the day, I will collect them both in the morning and in the evening in order to accurately calculate both the quickest way to work and from work. Each intersection between roads will be represented as a node, and each road between these intersections will be an edge. Only the roads that have the potential to be the quickest route will be considered.

Based on my research, the most efficient algorithm to use in solving this problem would be Dijkstra's algorithm, as only a bi-directional weighted graph will be used to find the solution. However, due to included packages in MATLAB, solving the algorithm by hand would be negligent. Instead, built in functions included in MATLAB will be used., specifically the "graph", "plot", and "shortestpath" functions.

No straightforward solution to generating an accurate weight value using the variables mentioned above was found, so we will only use the travel times of each road as the weights.

Analysis:

In order to properly represent our graph with correct weights, the travel times of each road in between intersections had be collected. This was done using google maps. Below is a graph of the road network (Figure A), with the weights being the travel times of that respective road (in minutes). Note that node A represents home, node H represents work, and all other nodes represent the intersections between them.

Because google maps calculates the travel times of each road for us and there is no open-source mathematical function that we could use to do so, no calculations needed to be made.

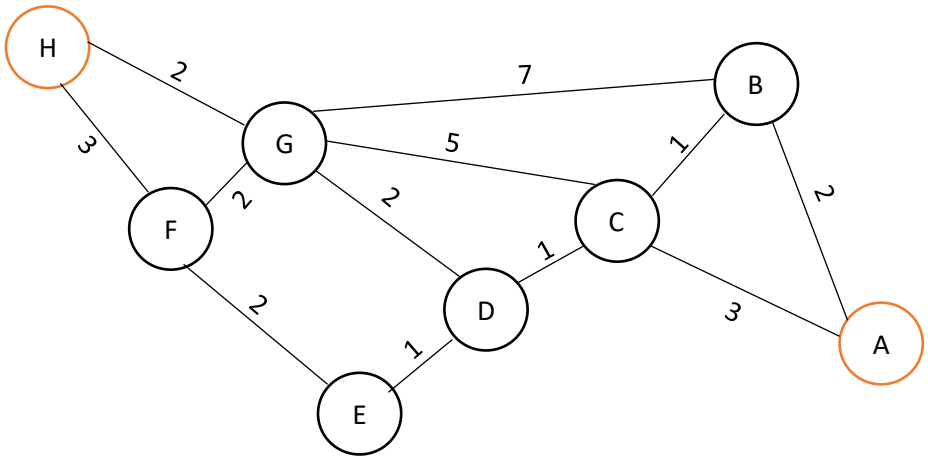


Figure A: *Road Network of roads between home (200 Oakcrest Dr., node A) and work (538 Cajundome Blvd., node H)*

Coding:

The simulation for this project was done using a free trial version of MATLAB. The code contained in the .m file is included below. Note that in the MATLAB code, specifically in vectors "s" and "t", 1 corresponds to node A in figure A, 2 to node B in figure A, 3 to node C in figure A, and so on.

Code:

```
%represent the nodes and edges to those nodes as vectors. In order to
%visualize this, the first value of s (1) and the first value of t (2) will
%form an edge. Likewise, the last value of s (7) and the last value of t(8)
%will form an edge.
s = [1 1 2 2 3 3 4 4 5 6 6 7];
t = [2 3 3 7 4 7 7 5 6 7 8 8];


%represent the weights of each edge as a vector. The first weight will
%correspond to the first edge formed by s and t above (between nodes 1 and
%2), the last weight will correspond to the last edge formed by s and t
%(between nodes 7 and 8), and so forth.
weights = [2 3 1 7 1 5 2 1 2 2 3 2];


%generate a graph based on the vectors above using the graph function
G = graph(s,t,weights);
```

Code (cont.):

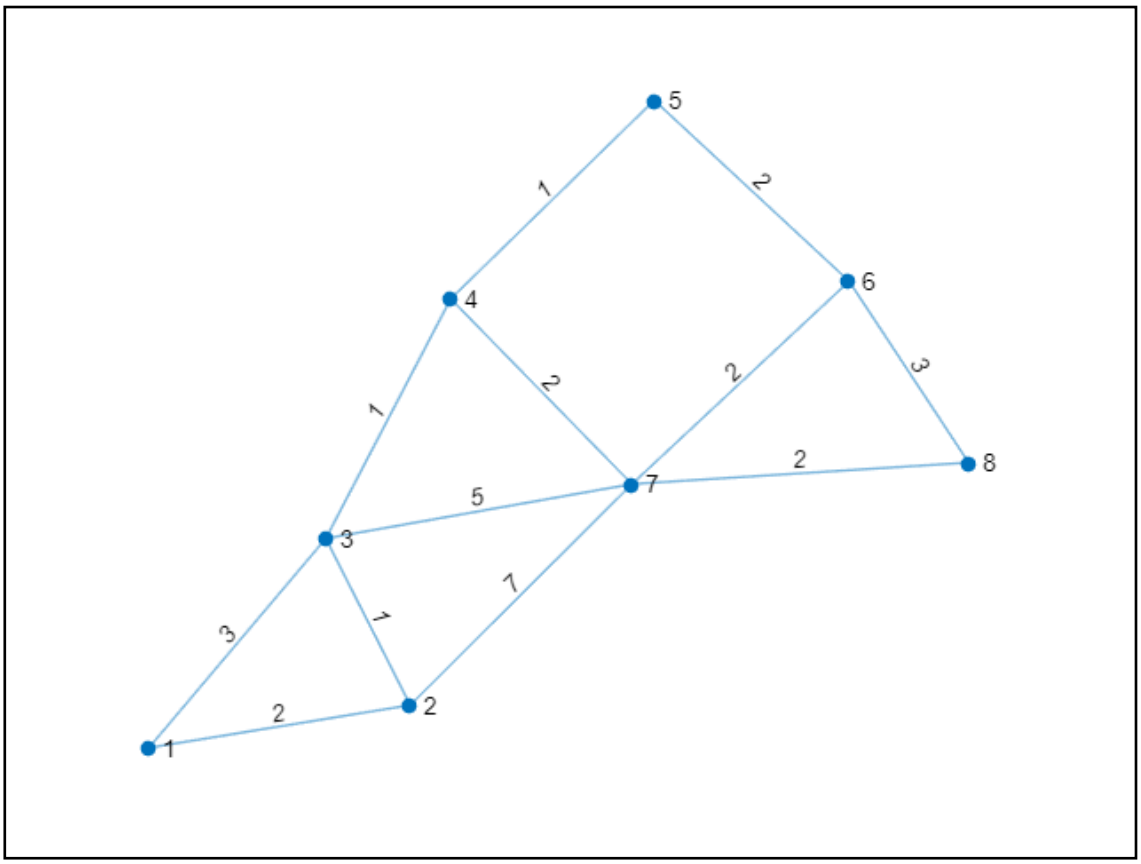
```
%plot the graph using the plot function
plot(G,'EdgeLabel',G.Edges.Weight)


%display the results of the shortest path from nodes 1 to 8,
%using the shortestpath function
[Path, length] = shortestpath(G,1,8)
```

Results:

Below are the output results of the MATLAB code above. Note that, as before, node 1 in the plot corresponds to node A in figure 1, node 2 to node B, and so on.

Output Plot:



Output Values:

```
Path =

     1     3     4     7     8


length =

     8
```

Conclusions:

As shown in the "Results" section above, the shortest path that I could take is from 1/A to 3/C, 3/C to 4/D, 4/D to 7/G, and 7/G to 8/H. The length of that path will be 8. Applying this to the real route that it corresponds to, the shortest route I could take is as follows:

1. From home, continue up Oak Crest Dr.
2. From Oak Crest Dr., turn left onto Johnston St.
3. From Johnston St., turn right onto Cajundome Blvd.
4. Continue on Cajundome Blvd. until you reach CGI Technologies and Solutions

As mentioned before, the weights of the edges in the graph correspond to the time it takes to travel between each node. As such, the length of the path, 8, means that it will take roughly 8 minutes for me to travel from home to work by taking the above route.

In order to improve this project in the future, the reliance on Google Maps could be mitigated by deducing a way to use the afore mentioned variables (speed limit, traffic congestion, length of road) to calculate a value that accurately represents the weight of each edge. Of course, a lot of mathematical theory would need to be applied in order to do this. If done, however, we could apply this concept in order to solve a huge set of networking problems far beyond the scope of this instance. These include internet routing problems, GPS simulations, integrated circuit designs, social networking problems, and many more. The same skeleton of the MATLAB code above could be used, of course, replacing the "s", "t", and "weights" vectors to accurately represent the graph being used.

Bibliography:

Shortest Path between Two Single Nodes - MATLAB, www.mathworks.com/help/matlab/ref/graph.shortestpath.html.

Joshi, Vaidehi. "Finding The Shortest Path, With A Little Help From Dijkstra." *Medium*, Basecs, 17 Oct. 2017, medium.com/basecs/finding-the-shortest-path-with-a-little-help-from-dijkstra-613149fbdc8e.

"The Shortest Route Problem." *The Shortest Route Problem | Introduction to Management Science (10th Edition)*, https://flylib.com/books/en/3.287.1.97/1/.