Nylogan2
Cgordon5
Ndarand2

Results

After 4 weeks of working as a group, we were able to successfully complete our core goals of this project of implementing two shortest path algorithms using BFS and Dijkstra's, and a map projection onto a mercator map. There was no problem with communicating or productivity within the group because of the fact that all of us had a clear vision of what we needed to accomplish and who was responsible for the parts. A lot of this can also be attributed to the fact that we made sure our code was well commented and would hop onto a zoom call whenever there was an issue or a part we weren't sure how to go about.

Now I will go over some of the things that we initially planned on implementing but didn't due to several reasons. First, we were planning on implementing a GIF animation of the flight paths kind of like in *mp_traversals* but we were not able due to time constraints. Second, we had planned on implementing a function that would indicate levels of traffic through each airport but we felt it wasn't as relevant with our direction of the project as a whole in the late stage. Finally, we would have liked to more accurately implement the map projection by having the function recognize if the closest path involved going through the boundaries (for example, going west through the left of the mercator PNG instead of going east when we go from Chicago to Tokyo). However, not overloading ourselves helped us to drive the main goal of this project home, allowing the visualization of BFS vs Dijkstra's. We were also able to flesh out everything and make things intuitive and clean, and finished off with an interactive ./pathFinder for an easy and pleasant experience for the user.

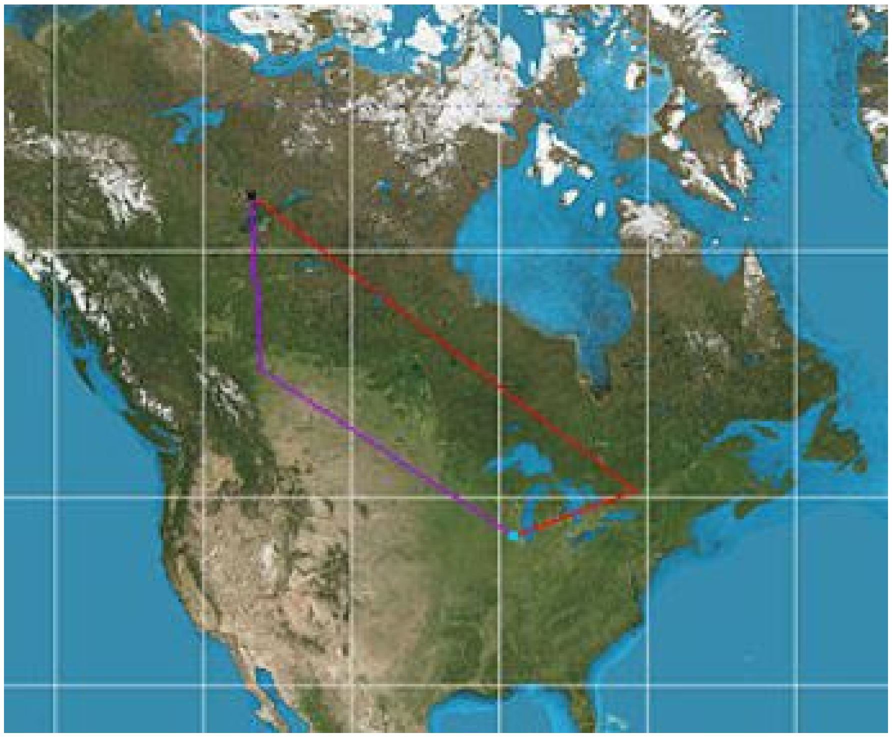
We also ended up hitting some snags that we had to adapt for. The most troublesome, centered around our Dijkstra's shortest route algorithm. During our later stages of testing, once we started to use the full data sets for routes, we started to get occasional *bad_alloc* exceptions when called. We initially attempted to find a way for Dijkstra's algorithm to use less memory, but found our algorithm was implemented in the most minimal way we could think of or find through online research; 60,000 possible edges was just too much to handle. Instead of messily allocating more space, we decided to limit the number of routes to 15,000, where we never saw a bad_alloc issue again. This also sped up the processing time of findShortestWeightedRoute drastically. If given more time, we would have liked to research memory allocation to allow us to use our full data set.

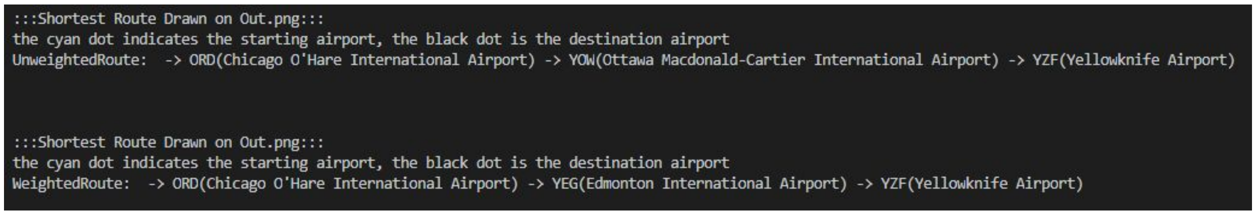
Our code also yielded some interesting behavior -- that while not bad or inaccurate -- is interesting and deserves addressing. That is, when both shortest route by *stops* and *distance* is called, they often have the same number of stops -- as shown below in *Figure 1,* but different paths. This behavior is caused by the nature of the BFS traversal, stopping at the first route with the least amount of nodes. On the other hand, Dijkstra's distance based search will go through all routes, effectively finding the shortest route.

We have also placed several other routes below that clearly show different behavior patterns below, with their Out.png output and the iostream output after finished running. Feel free to look through them to get a feel for our interface!

Throughout the duration of the project, there are some areas where we learned a great deal through our struggles and research. For our data extraction, we all gained a strong handle on csv fixing our constructors many bugs, whereas our research into mercator projection exposed us to many theories on the algebra converting 3d points to 2d and vice versa. In creating a basic terminal interface, we also experimented with command line parameters and their uses, and -- not forgetting the intention of the project -- are now confident in our ability to navigate graphs through our code.

All in all, our project finished successfully. We are happy with our result and the progress we were able to achieve as a group.



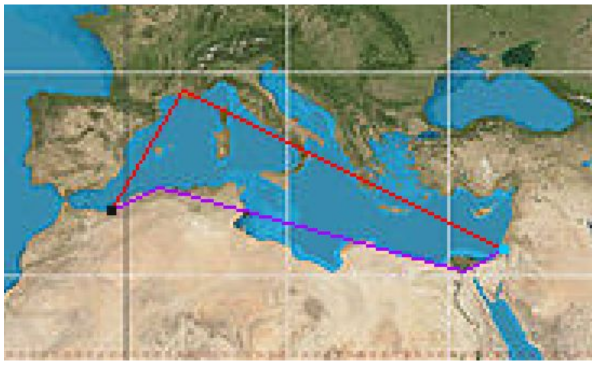


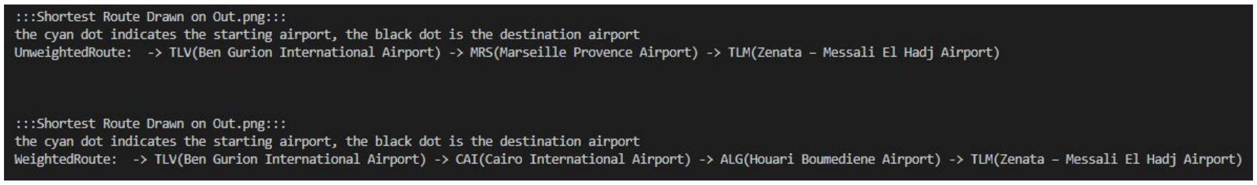
```
:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
UnweightedRoute:  -> ORD(Chicago O'Hare International Airport) -> YOW(Ottawa Macdonald-Cartier International Airport) -> YZF(Yellowknife Airport)



:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
WeightedRoute:  -> ORD(Chicago O'Hare International Airport) -> YEG(Edmonton International Airport) -> YZF(Yellowknife Airport)
```

Figure 1: ORD(Chicago O'hare International Airport) to YZF(Yellowknife Airport) - BFS(red) and Dijkstra's(violet)

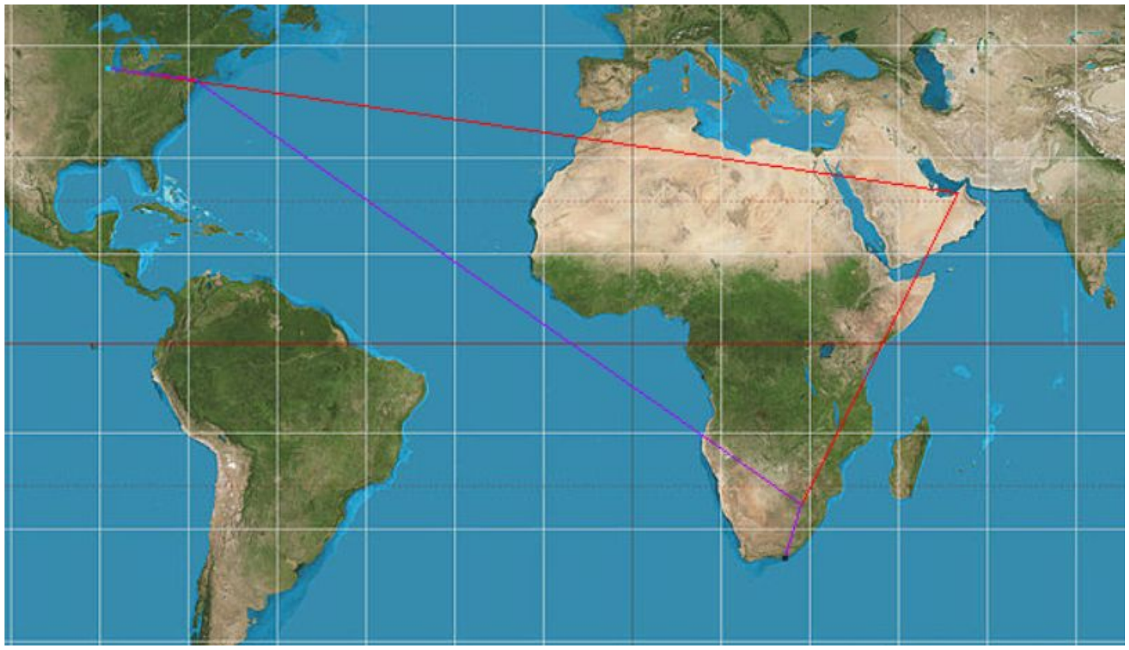


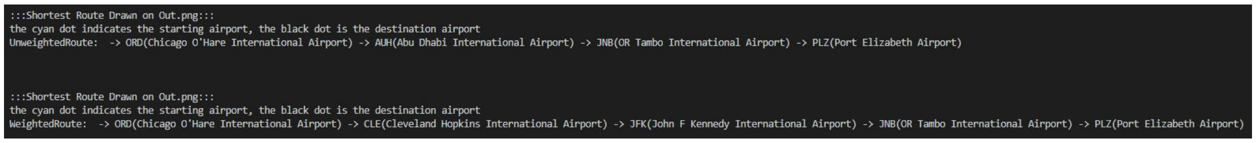


```
:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
UnweightedRoute:  -> TLV(Ben Gurion International Airport) -> MRS(Marseille Provence Airport) -> TLM(Zenata – Messali El Hadj Airport)


:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
WeightedRoute:  -> TLV(Ben Gurion International Airport) -> CAI(Cairo International Airport) -> ALG(Houari Boumediene Airport) -> TLM(Zenata – Messali El Hadj Airport)
```

Figure 2: TLV(Ben Gurion International Airport) to TLM(Zenata - Messali El Hadj Airport) - BFS(red) and Dijkstra's(violet)

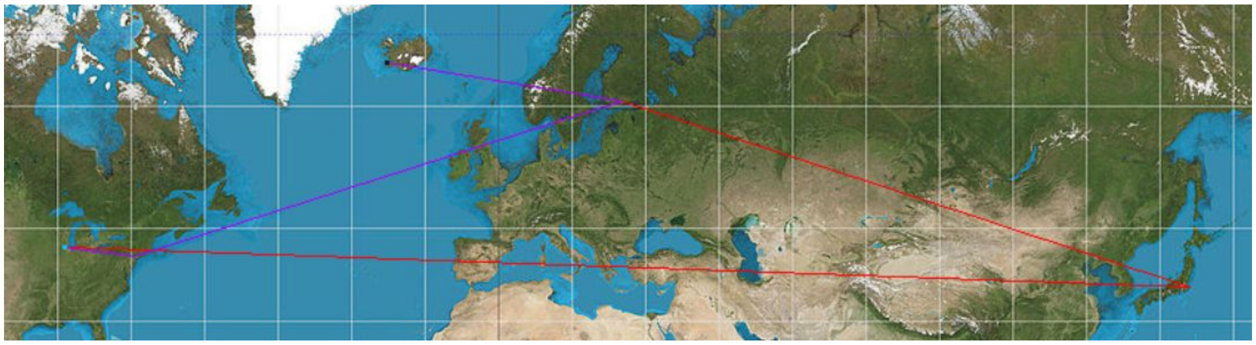


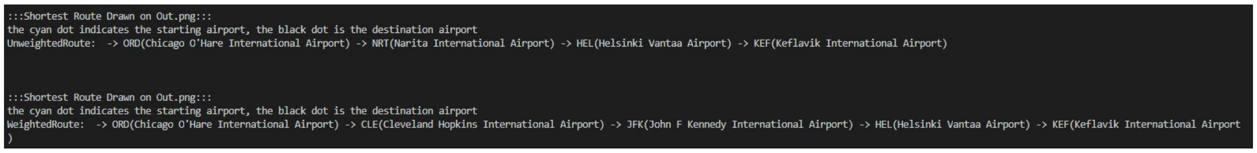


```
:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
UnweightedRoute:  -> ORD(Chicago O'Hare International Airport) -> AUH(Abu Dhabi International Airport) -> JNB(OR Tambo International Airport) -> PLZ(Port Elizabeth Airport)


:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
WeightedRoute:  -> ORD(Chicago O'Hare International Airport) -> CLE(Cleveland Hopkins International Airport) -> JFK(John F Kennedy International Airport) -> JNB(OR Tambo International Airport) -> PLZ(Port Elizabeth Airport)
```

Figure 3: ORD(Chicago O'hare International Airport) to PLZ(Port Elizabeth Airport) - BFS(red) and Dijkstra's(violet)





```
:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
UnweightedRoute:  -> ORD(Chicago O'Hare International Airport) -> NRT(Narita International Airport) -> HEL(Helsinki Vantaa Airport) -> KEF(Keflavik International Airport)


:::Shortest Route Drawn on Out.png:::
the cyan dot indicates the starting airport, the black dot is the destination airport
WeightedRoute:  -> ORD(Chicago O'Hare International Airport) -> CLE(Cleveland Hopkins International Airport) -> JFK(John F Kennedy International Airport) -> HEL(Helsinki Vantaa Airport) -> KEF(Keflavik International Airport
)
```

Figure 4: ORD(Chicago O'Hare International Airport) to KEF(Keflavik International Airport) - BFS(red) and Dijkstra's(violet)