CS225 Final Project Report

Preston Chao, Dennis McCann, Dom Quigly, Ben Civjan

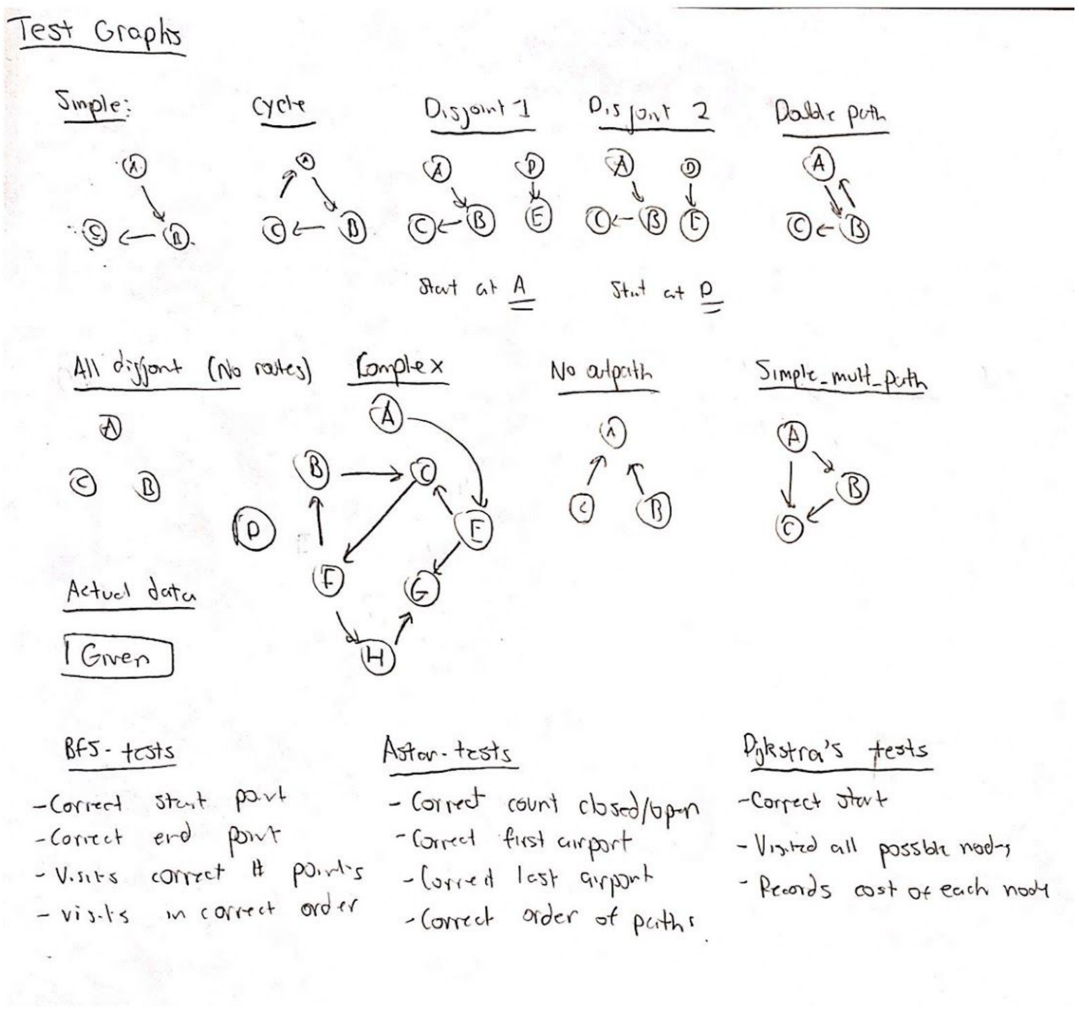
For the CS225 Final Project we used Open Flights Route Database to run the following algorithms: Breadth First Search Traversal, Dijkstra's algorithm for shortest path, and the A* search algorithm. We decided to use these algorithms because we wanted to simulate how a pandemic would spread between countries through airports and airline routes. We also wanted to see different ways to find the shorted path between different airports using both Dijkstras and A* algorithm. In this report we will show the outcomes to each of these algorithms and analyze each.

The first algorithm is a Breadth First Search Traversal. This algorithm is the key to simulating a pandemic using our Routes Dataset. For this algorithm we followed the pseudocode that was given and created a class and function to run the set algorithm. Our function took in an already populated graph as well as a specified starting point and then returns the given path that it takes. The algorithm utilizes a queue in which all of the adjacent routes are pushed onto that queue and then each vertex or airport in our case is then popped off once it is visited. It continues to do this until it visits all possible destinations. The output vector simulates the way a pandemic would spread from one root airport and the order it would spread in. To test this and all of the other algorithms, we created smaller datasets that represent different graphs which we attached at the bottom of the report.

The next algorithm is Dijkstra's algorithm for finding the shortest path. This algorithm takes a certain graph Data Structure as well as a startpoint on that graph, and outputs an ordered map that maps each node in the graph to its least cost of travel from the starting vertex. In our algorithm we decided to specify the cost of travelling to each vertex as the distance between it and its predecessor. We then iterated through this map to output each airport and its corresponding cost to the terminal. Doing this can show a consumer an ordered route and relative cost for getting to any place from a specified starting point.

The final algorithm that we built was the A* search algorithm. This algorithm finds the shortest path between a starting airport and an ending airport. It achieves this by keeping track of the distance from the start and the distance to the end for each airport (these are known as the g and h costs). It outputs a vector of airports that is the shortest path from start to finish. We iterate through this vector and output each airport name to the terminal to provide visual feedback. This algorithm can be useful because it finds the most efficient route that a consumer can take. It is important to note that this is the shortest distance traveled, not cheapest cost monetarily. However this can be changed by setting the weight of the route to be the cost as opposed to distance.

Our main objective of this final project was to simulate a pandemic and compare two shortest path algorithms. After completing BFS and looking at the output we could see how a pandemic might spread throughout the world via airports and airlines, and based on a certain starting point, which airports and eventually countries would be affected first. By building both the Dijkstra's shortest path algorithm and the A* search algorithm we can see that both are viable ways to finding the least costly path between two airports, with Dijkstra's showing the shortest path between a starting point and any airport within a graph, and A* showing a specified shortest path between two specified nodes, one start and one end.



BFS - tests
- Correct start point
- Correct end point
- Visits correct # points
- visits in correct order

Astar - tests
- Correct count closed/open
- Correct first airport
- Correct last airport
- Correct order of paths

Dijkstra's tests
- Correct start
- Visited all possible nodes
- Records cost of each node