

CS225 Final Project Result

Our CS225 final project fully accomplished our goals: process the OpenFlight dataset as a graph structure and run the following algorithms on it: DFS traversals for traversing the whole graph, Floyd-Warshall algorithm for calculating shortest path between two points, airports and airline routes visualization on a world map. This document will briefly illustrate the outcome of this project and introduce our program.

After compiling and running the program, there will be specific instructions for choosing datasets and algorithms. Our program supports reading other data files formatted the same way as the airports and routes data in OpenFlight dataset. Due to the runtime of Floyd-Warshall algorithm, running the program on OpenFlight dataset can take up to 1.5 hours, so we also provided a subset for grading and testing purpose.

After choosing the dataset, users will be able to choose which algorithm to run. By choosing BFS or DFS, the program will respectively run each traversal algorithm and save the result to the same folder as “BFS.txt” or “DFS.txt” with a list of vertices based on the traversal order; By choosing shortest path, the program will calculate the distance matrix and the path matrix using Floyd-Warshall Algorithm, then it will demonstrate the shortest path as a list between the source and destination airports chosen by users. Since Floyd-Warshall Algorithm computes shortest distance between all pairs of vertices, the program supports multiple searches; By choosing visualization, program will result two PNGs: “world_map_with_airports.png”, and “world_map_with_airports_and_routes.png”, as two pictures with airports as nodes on the world map as well as the routes as edges.

We did have discoveries while proceeding our final project. In particular, we found drawing pixel lines between two specific points surprisingly challenging. We thought this can be

solved using a simple for loop but we failed after many attempts. After some researches, we found out that there are many different cases we need to consider and computer scientists have developed many algorithms for this problem. We chose the Bresenham's Line Algorithm. Moreover, we found many disconnected airports. This is because the airport and route data we used were collected in different years. We modified several our data processing functions to avoid problems caused by this situation.

Screenshots of our outputs are included in next page

Traversal Result:

```
BFS.txt
BFS.txt
1 13803: Mitchell Municipal Airport
2 13758: Granbury Regional Airport
3 13717: Camp Pendleton MCAS (Munn Field) Airport
4 13707: Seldovia Airport
5 13694: Stennis International Airport
6 13642: Potomac Airpark
7 13635: Denali Airport
8 13607: John C Tune Airport
9 13551: Frankfort Dow Memorial Field
10 13439: Red Dog Airport
11 13354: Sandpoint Airport
12 13158: Hutchinson County Airport
13 13156: Marion County Regional Airport
14 13154: Perry Lefors Field
15 13144: Rusk County Airport
16 13137: Blue Ridge Airport
17 13131: Socorro Municipal Airport
18 13123: Stanley County Airport
19 13080: Prairie Du Chien Municipal Airport
20 13075: Gaylord Regional Airport
21 12950: Statesboro Bulloch County Airport
22 12873: Jasper County Airport
23 12856: Clearfield Lawrence Airport
24 12672: Marion County Brown Field
25 12469: Bolton Field
26 11868: Zanesville Municipal Airport
27 11865: West Woodward Airport
28 11858: Tonopah Airport
29 11138: Mount Airy Surry County Airport
30 11857: Saline County Regional Airport
31 11856: Sonora Municipal Airport
32 11852: New Richmond Regional Airport
33 11851: Dutchess County Airport
34 11847: Columbus Municipal Airport
35 11846: Orangeburg Municipal Airport
36 11844: Brunswick Executive Airport
37 11842: Southern Illinois Airport
38 11839: Derby Field
39 11838: Monticello Municipal Ellis Field
40 11837: Kibbie County Airport
41 11834: Murray Field
42 11833: Duke Field
43 12796: Danielson Airport
44 11832: Needles Airport
45 11831: Desert Rock Airport
46 11824: Northeast Iowa Regional Airport
47 11822: W K Kellogg Airport
48 11820: Wiley Memorial Field
49 11819: Baker City Municipal Airport
50 11814: St Louis Regional Airport
51 11312: Reid-Hillview Airport of Santa Clara County
52 11141: Arlington Municipal Airport
53 11134: Karl Stefan Memorial Airport
54 11110: Pocono Mountains Municipal Airport
```

```
DFS.txt
DFS.txt
1 13803: Mitchell Municipal Airport
2 13758: Granbury Regional Airport
3 13717: Camp Pendleton MCAS (Munn Field) Airport
4 13707: Seldovia Airport
5 13694: Stennis International Airport
6 13642: Potomac Airpark
7 13635: Denali Airport
8 13607: John C Tune Airport
9 13551: Frankfort Dow Memorial Field
10 13439: Red Dog Airport
11 13354: Sandpoint Airport
12 13158: Hutchinson County Airport
13 13156: Marion County Regional Airport
14 13154: Perry Lefors Field
15 13144: Rusk County Airport
16 13137: Blue Ridge Airport
17 13131: Socorro Municipal Airport
18 13123: Stanley County Airport
19 13080: Prairie Du Chien Municipal Airport
20 13075: Gaylord Regional Airport
21 12950: Statesboro Bulloch County Airport
22 12873: Jasper County Airport
23 12856: Clearfield Lawrence Airport
24 12672: Marion County Brown Field
25 12469: Bolton Field
26 11868: Zanesville Municipal Airport
27 11865: West Woodward Airport
28 11858: Tonopah Airport
29 11138: Mount Airy Surry County Airport
30 11857: Saline County Regional Airport
31 11856: Sonora Municipal Airport
32 11852: New Richmond Regional Airport
33 11851: Dutchess County Airport
34 11847: Columbus Municipal Airport
35 11846: Orangeburg Municipal Airport
36 11844: Brunswick Executive Airport
37 11842: Southern Illinois Airport
38 11839: Derby Field
39 11838: Monticello Municipal Ellis Field
40 11837: Kibbie County Airport
41 11834: Murray Field
42 11833: Duke Field
43 12796: Danielson Airport
44 11832: Needles Airport
45 11831: Desert Rock Airport
46 11824: Northeast Iowa Regional Airport
47 11822: W K Kellogg Airport
48 11820: Wiley Memorial Field
49 11819: Baker City Municipal Airport
50 11814: St Louis Regional Airport
51 11312: Reid-Hillview Airport of Santa Clara County
52 11141: Arlington Municipal Airport
53 11134: Karl Stefan Memorial Airport
54 11110: Pocono Mountains Municipal Airport
```

Shorted path example result:

```
[peiyuan3@linux-a2 peiyuan3-haoyul4-xinshuo3]$ make
clang++ .objs/graph.o .objs/main.o .objs/cs225/HSLAPixel.o .objs/cs225/PNG.o .objs/cs225/lopng/lopng.o -std=c++1y -stdlib=libc++ -lc++abi -o graph
[peiyuan3@linux-a2 peiyuan3-haoyul4-xinshuo3]$ ./grapg
bash: ./grapg: No such file or directory
[peiyuan3@linux-a2 peiyuan3-haoyul4-xinshuo3]$ ./graph

This program generates graph and runs algorithms on flight datasets

Do you want to run algorithms on the OpenFlights dataset (OF) we used or dataset of your choice (Y)
Due to the runtime of Floyd-Warshall algorithm, the program takes a long time to find the shortest path for large datasets,
including the OpenFlight dataset that we are using.
For testing and grading purpose, we provide a subset of the OpenFlight dataset, which only include US airports and routes.
You can enter US to run algorithm on this subset.
Y

Please enter the filename of the airport data (also include folder name if in a different folder)
Or enter B to choose a different dataset
data/airports_US.dat

Please enter the filename of the route data (also include folder name if in a different folder)
data/routes.dat

Please choose the algorithm you want to run
The available algorithms are:
DFS/BFS traversal (enter DFS for DFS and BFS for BFS)
Shortest path between two points (Floyd-Warshall algorithm) using distance as weight (enter SP)
Project onto map based on Openflight dataset (enter V)
SP

building distance and path matrix... (1512/1512)

Please enter the full name of the origin airport
Chicago O'Hare International Airport

Please enter the full name of the destination airport
Rick Husband Amarillo International Airport

The shortest path from Chicago O'Hare International Airport to Rick Husband Amarillo International Airport is:
Chicago O'Hare International Airport --> Dallas Fort Worth International Airport --> Rick Husband Amarillo International Airport

Enter Y if you want to find shortest path between another two airports
Enter anything else to stop the algorithm

```

Visualization result:

