

- **Administrative**
  - Jason's fantastic team
  - Jason Nguyen (github - jasonneyugn, Joseph Cardillo, Diksha Kushwa
  - <https://github.com/jasonneyugn/project2cop3530.git>
  - <https://youtu.be/HQejVx-3P8o>
- Jason
- **Extended and Refined Proposal [Suggested 2 Pages]**
  - Problem: The goal of this project was to compare and visualize the shortest path problem on a very large dataset, in this case the global network of airports. The problem we originally had was to just find the shortest distance from point A to point B, but that was changed to connected airports. We also originally had just Dijkstra's algorithm, which is very costly on larger datasets, so we contrasted this with the A\* algorithm that uses heuristics.
  - Motivation: This is a very modern and important problem in computer science with things focused around GPS and routing. This is where shortest path algorithms shine because the real world is much more convoluted than any simulated data. By using the dataset that consists of every airport it helps us visualize the algorithms and the way they work.
  - Features implemented -
    - Interactive world map using SFML
    - Mercator projection of parsed data of map lat/lon into 2D coordinates.
    - Randomized edge generation to connect graph
    - Zooming and panning
    - Path visualization (green lines for shortest path)
    - Dynamic runtime execution and analysis
    - Clickable airports with displayed routes
  - Description of data
  - Tools/Languages/APIs/Libraries used
    - Language: C++ (20)
    - Graphics Library: SFML
  - Algorithms implemented
    - Implemented Dijkstra's Algorithm and A\* search algorithm.
  - Distribution of Responsibility and Roles: Diksha - zilch, Jason - coding, video, github, report (extended proposal, analysis) , idea,
  - Jason
- **Analysis [Suggested 1.5 Pages]**
  - Any changes the group made after the proposal? The rationale behind the changes.

There was a lot of changes that were required to make the idea work without fully changing the base idea and breaking the rules. If we didn't generate any new edges, the graph would be sparse and unconnected, not allowing us to test any paths out. I also did not have any other algorithm planned out with Dijkstra's, only a rough idea of testing some others. However, with the A\* search it is a good contrast to the other algorithm because it is much more efficient in the way it travels. I also had to add features such as zooming and dragging, which were not originally planned to make interacting with the map much easier in general.

- Big O worst case time complexity analysis of the major functions/features you implemented

The major functions include the routeload function, the Dijkstra algorithm, the A\* search algorithm, and the haversine formula. The haversine formula is a constant formula with trig so it's just  $O(1)$ . To parse the routing files, we just had to go through every single part, so  $O(N)$  time complexity where  $n$  is every piece of data. For the main algorithms, the Dijkstra and A\* algorithms are actually both  $O((V+E)\log V)$  time complexity.

Jason

- **Reflection [Suggested 1-1.5 Page]**

- As a group, how was the overall experience for the project?

As a group our experience was engaging and extremely beneficial for learning the process of version control and group oriented programming. This project required the application of several different areas of computer science like data structures, algorithms, and graphics, all combined into a single practical application. Seeing the algorithms in use and working together like Dijkstra's and A\* in real time gave us a satisfying sense of accomplishment. While we had many issues with debugging and version control, we felt our skills have been strengthened and that we are leaving this project more confident for future collaborative efforts.

One of the largest challenges our group faced was regarding the complexity of the OpenFlights datasets. We had initially assumed them to require less logic for parsing and did not notice how careful we needed to be to avoid incorrect behaviors or results.

- Did you have any challenges? If so, describe.
  - If you were to start once again as a group, any changes you would make to the project and/or workflow?
  - Comment on what each of the members learned through this process.

- **References**

<https://www.geeksforgeeks.org/cpp/c-program-for-dijkstras-shortest-path-algorithm-greedy-algo-7/>

<https://www.geeksforgeeks.org/dsa/haversine-formula-to-find-distance-between-two-points-on-a-sphere/>