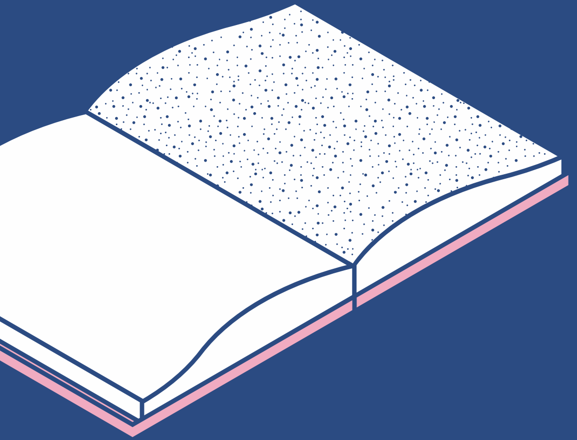
Navigating Airports Using Graph Algorithms

Course Title : – Data Structures & Algorithms

Course Code : – CSL2020

Instructor : – Suchetana Chakraborty

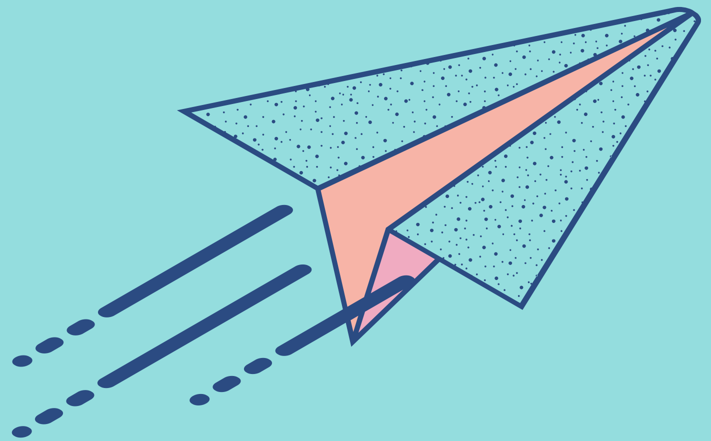
Mentor : – Jainan Nareshkumar Tandel (M23CSA010) &
Dhruv(B20EE016)



Team Members: –

- Aaditya Kamble (B22MT024)
- Jadeja Vishwjeetsinh (B22MT023)
- Sahil (B22MT038)
- Ghanshyam (B22PH009)

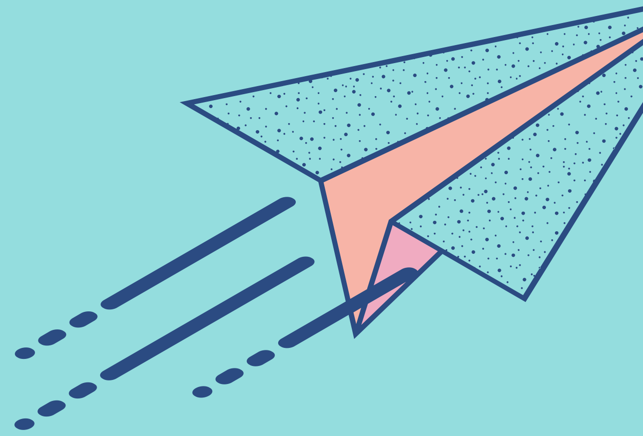
Problem Statement

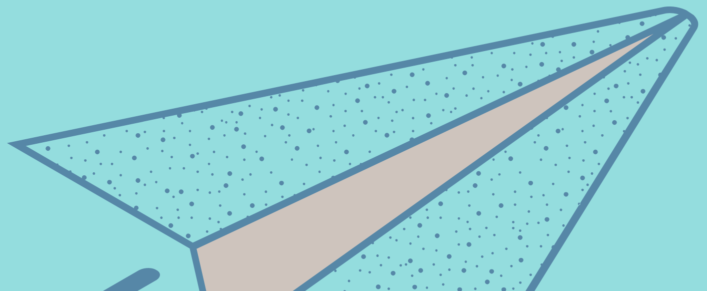


Introducing the Problem:

Finding the shortest distance between airports in a network of routes. Utilizing datasets containing airport and route information. Implementing Dijkstra's, Floyd Warshall's, and Bellman-Ford's algorithms. And aalso comparing their outputs.

- Efficient route planning is crucial for fast travel and passenger satisfaction in aviation.
- Comparing the outputs of Dijkstra's, Bellman-Ford, and Floyd-Warshall's algorithms using accurate data allows for a comprehensive evaluation of their performance and suitability for different scenarios, leading to informed decision-making in route selection.





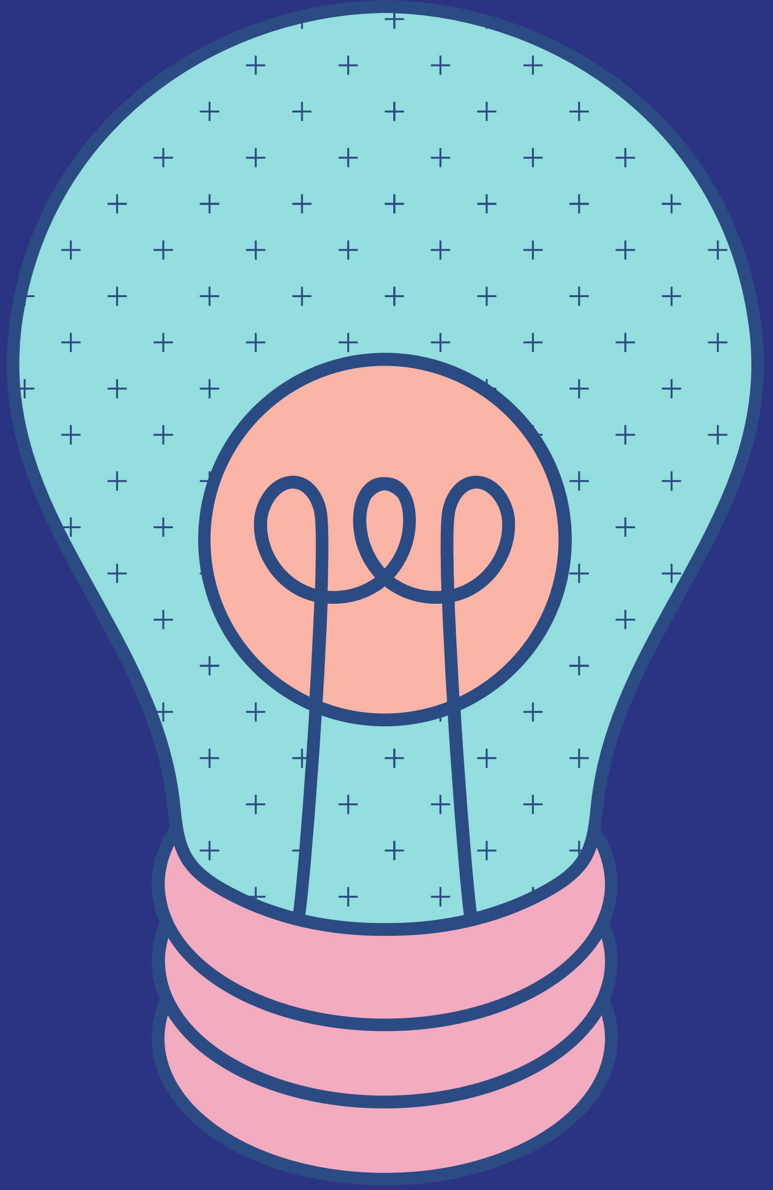
Current Status

Some Research papers have been published in this Regard.

- According to GeeksforGeeks, Bellman-Ford is the faster Algorithm when the graph is unweighted.
- As per medium.com

“The Bellman-Ford algorithm is more versatile .... Dijkstra’s Algorithm is more efficient .... weights.”

Overall, It is seen as not easy to answer, so which is the better algorithm? It is a common perception that Dijksrtra is more efficient and Bellman is better for handling negative cycles, and both of these are better than the Floyd Algorithm.



Limitations of Algorithms

Dijkstra's

- Limited to graphs with non-negative edge weights.
- Inefficient for graphs with negative edge weights or cycles.
- Requires a priority queue for efficient implementation, which can be memory-intensive and challenging to implement in certain scenarios.

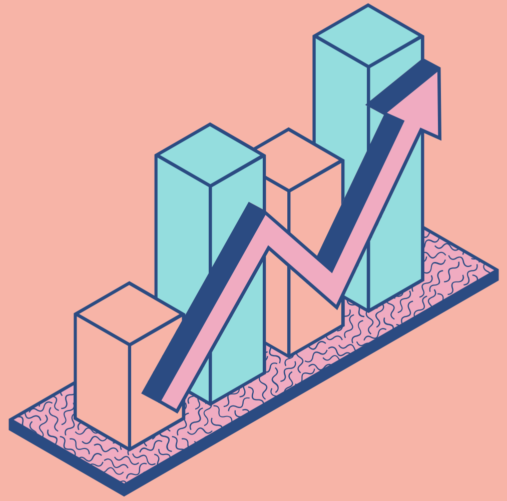
Bellman-Ford

- It may need help finding the shortest path in negative-weight cycles.

Floyd Warshall

- Memory-intensive nature due to the need to store the distance matrix.

Idea



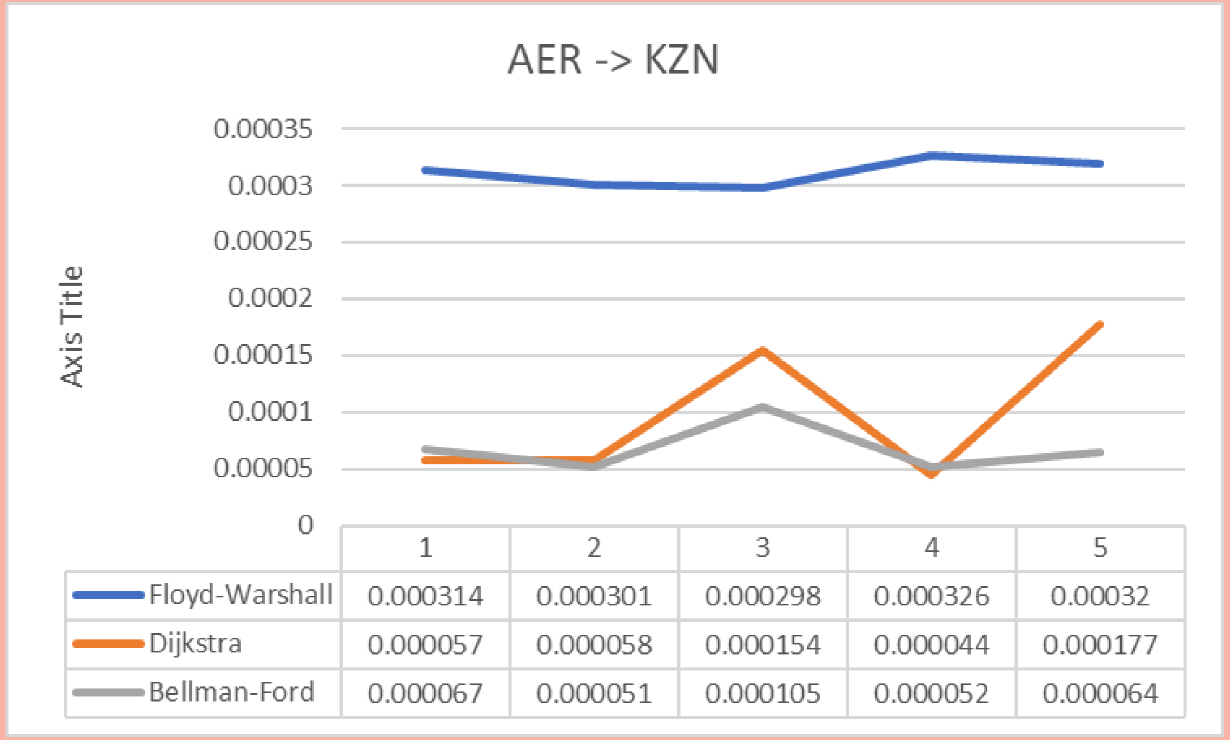
Improvements & Implementation:

Github Repo Link: ideathon-code-submission-b22mt024

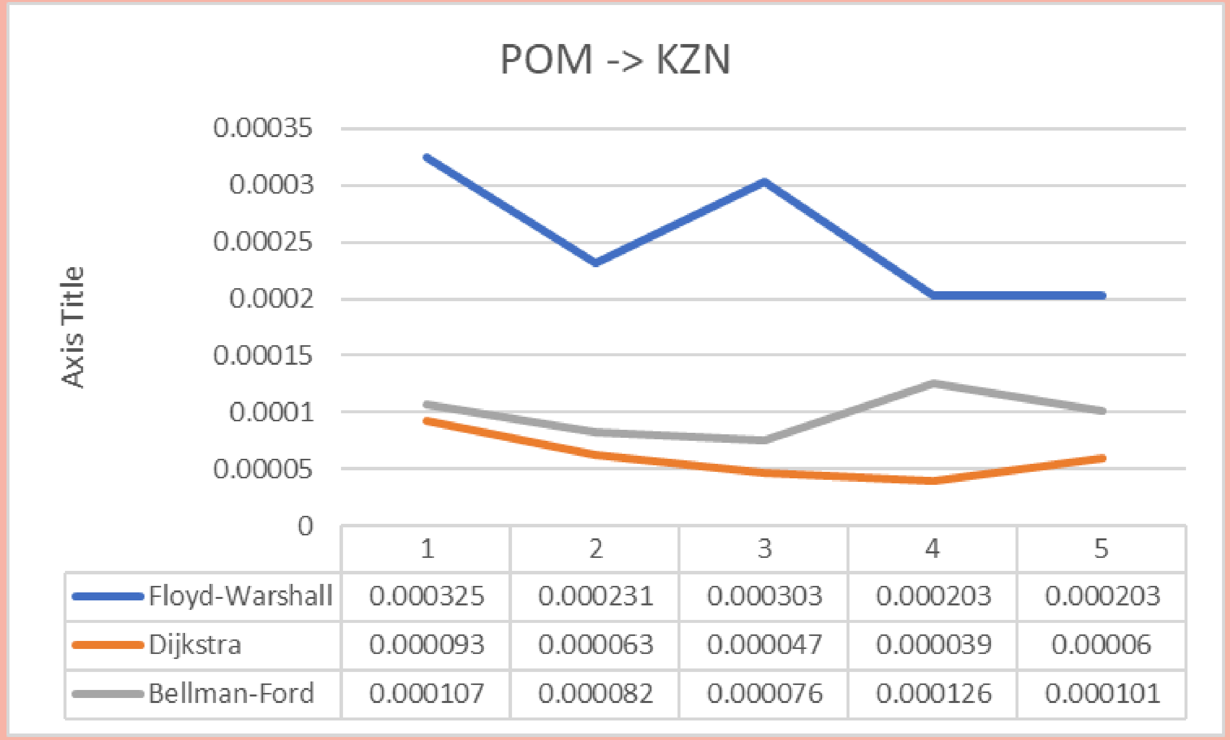
Data structures: Used a graph data structure to represent the network of airports and routes and adjacency lists and matrices to store the connections between airports.
Algorithms: Selected well-suited and common algorithms to handle the problem.
Used both types of algorithms, to find the shortest path between a single source and all other airports, used Dijkstra and to find the shortest path between all pairs of vertices, used Floyd-Warshall.

Results & Analysis

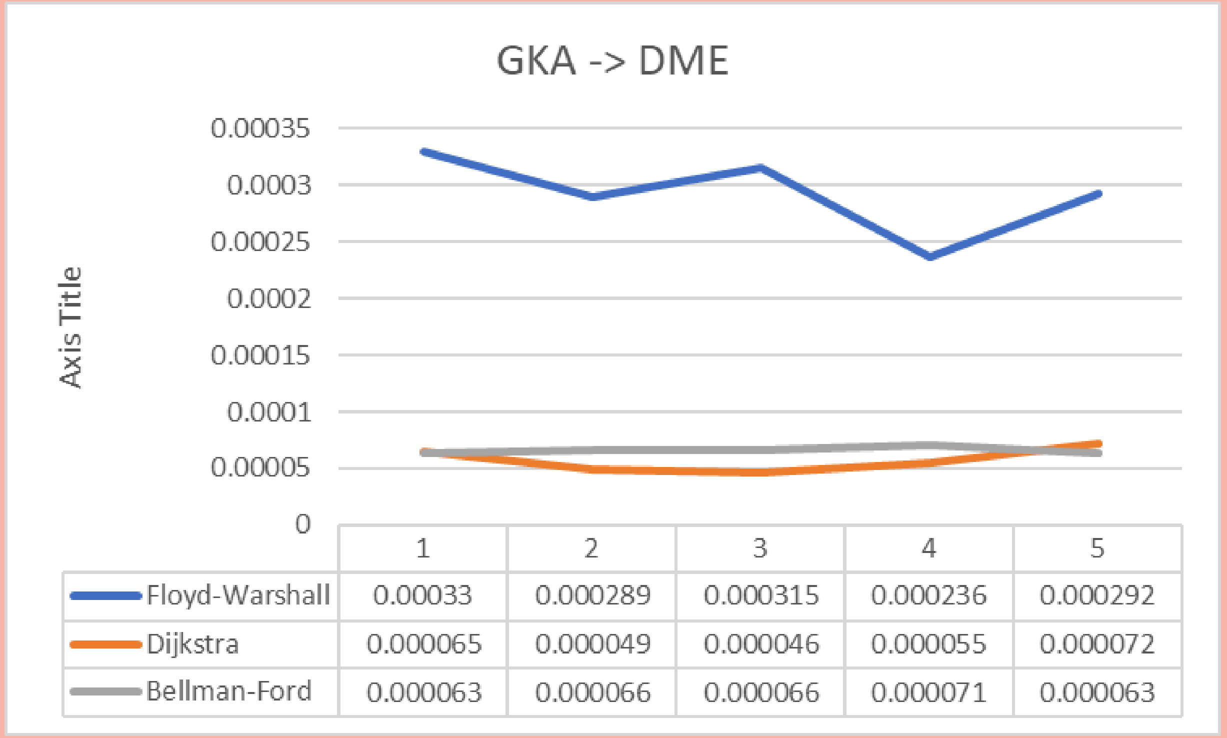
Time complexity:



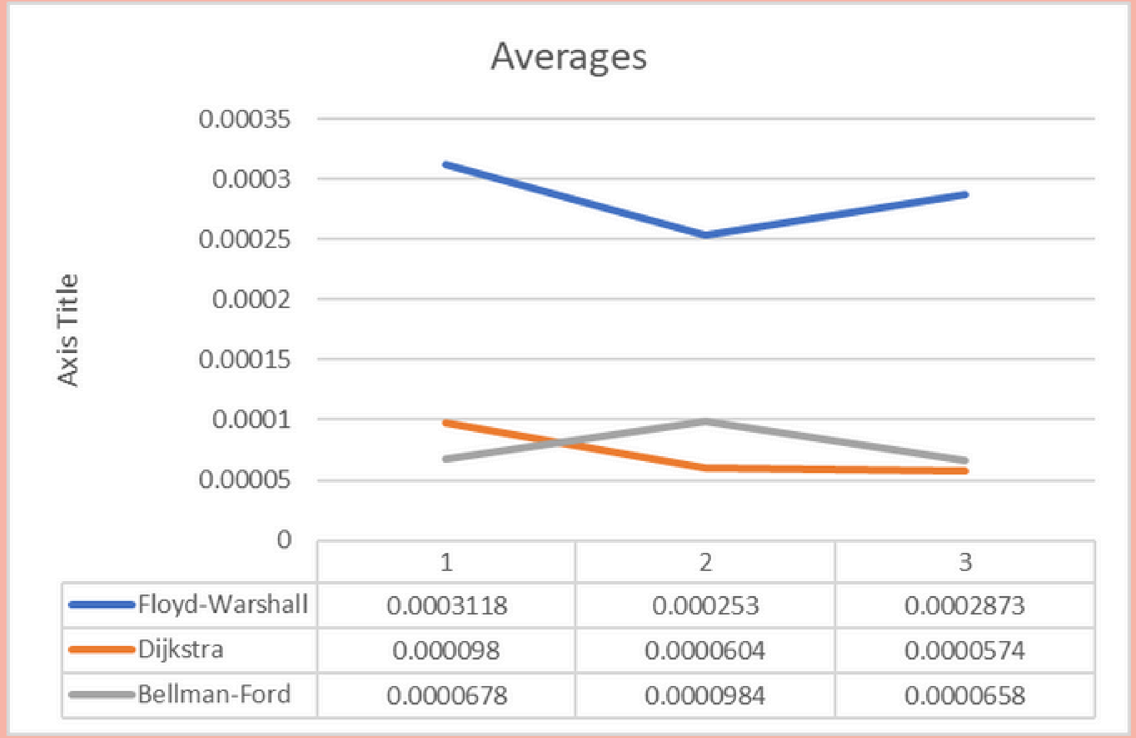
	1	2	3	4	5
Floyd-Warshall	0.000314	0.000301	0.000298	0.000326	0.00032
Dijkstra	0.000057	0.000058	0.000154	0.000044	0.000177
Bellman-Ford	0.000067	0.000051	0.000105	0.000052	0.000064



	1	2	3	4	5
Floyd-Warshall	0.000325	0.000231	0.000303	0.000203	0.000203
Dijkstra	0.000093	0.000063	0.000047	0.000039	0.00006
Bellman-Ford	0.000107	0.000082	0.000076	0.000126	0.000101



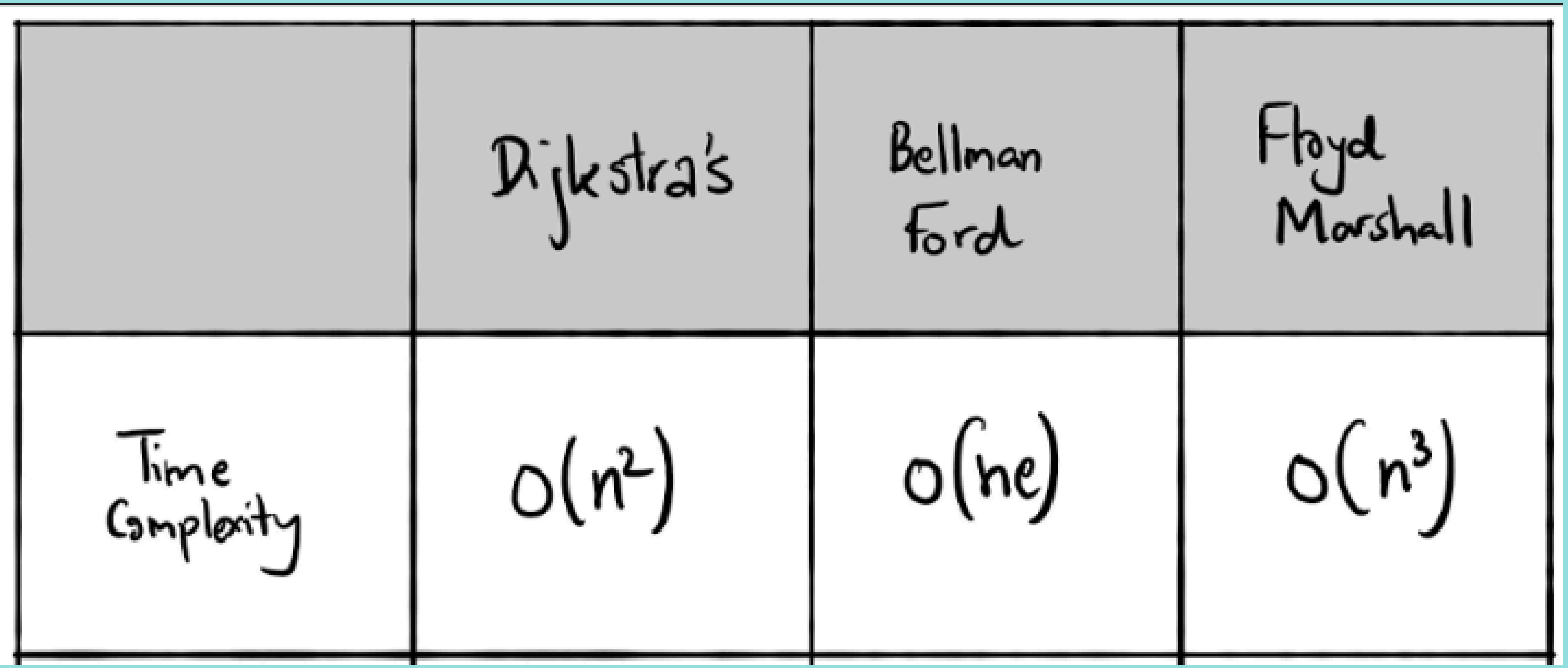
	1	2	3	4	5
Floyd-Warshall	0.00033	0.000289	0.000315	0.000236	0.000292
Dijkstra	0.000065	0.000049	0.000046	0.000055	0.000072
Bellman-Ford	0.000063	0.000066	0.000066	0.000071	0.000063



	1	2	3
Floyd-Warshall	0.0003118	0.000253	0.0002873
Dijkstra	0.000098	0.0000604	0.0000574
Bellman-Ford	0.0000678	0.0000984	0.0000658

- In terms of time complexity, Dijkstra's algorithm and Bellman-ford seems to be comparatively same and the efficient compared to Floyd-Warshall.

- Floyd-Warshall consistently performs slower than Dijkstra and Bellman-Ford algorithms in the given scenarios.



	Dijkstra's	Bellman Ford	Floyd Marshall
Time Complexity	$O(n^2)$	$O(ne)$	$O(n^3)$

- n -> number of vertices
- e -> number of connections.

Conclusion

Finally, to conclude our implementation of all three algorithms, Dijkstra, Bellman-Ford, and Floyd-Warshall, It is seen that both the Dijkstra and Bellman are very closely matched and outperform each other at several occasions, so it would be unfair to say that any one algorithm is faster than other. Still, the Floyd algorithm is the slowest among the three. This might have been because it uses three nested loops to iterate over all pairs of vertices. We have used any graph with negative cycles, which is when the bellman algorithm is asit to shine, so in comparison of Time complexities, Dijkstra has a better consists at outperforming other algorithm any graph with negative cycles, which is when the bellman algorithm is asit to shine, so in comparison of Time complexities, Dijkstra has a better consists at outperforming other algorithms, and look at the point that no one cycles were considered It could be stated that Dijkstra is a super algorithm if negative cycles are not considered.

Conclusion

Scope of future extension:

This initiative supports future expansion research in numerous areas. First, improving algorithms and data structures for massive datasets may boost efficiency. Parallelization and heuristics may save computation time and optimise resource use. The route planning system may be more resilient when finding and addressing negative-weight cycles in the Bellman-Ford algorithm. Finally, real data sources, including flight schedules, weather, and airport congestion, may improve route planning accuracy and responsiveness, improving the traveller experience.

Summary:

By applying graph algorithms, the research optimised aircraft route planning. The research implemented and compared Dijkstra's, Bellman-Ford's, and Floyd-Warshall's algorithms to assess their performance and application. Using accurate data and thorough research, route selection improved client travel experiences. The project's findings provide the groundwork for aircraft route planning enhancements.

Contribution & Acknowledgement

Each of our Team members made equal contributions to the codes and we took lot of references from other sources also.

Vishwjeetsinh Jadeja (B22MT023)

- Took a lead of Floyd-Warshall algorithm.
- Study the time complexities of all algorithms.

Aaditya Kamble(B22MT024)

- Lead the Team
- Took a lead of Bellman-Ford algorithm.
- Visulize the graphs along with minor improvements here and there

Sahil (B22MT038)

- Took a lead of Dijkstra algorithm.
- Did major research required.
- Suggested the idea of the project.

Ghanshyam (B22PH009)

- Handle all the trouble shooting issues and errors.
- Made datasets, did data processing and manage all datas of airport and routes files.

Dijkstra's Algorithm

iqcode programesecure
stackoverflow stackflow
stacksoverflow geeksforgeeks
codereview.stackexchange

Floyd Warshall Algorithm

dev
github
stackoverflow
code.pieces.app

Bellman-Ford Algorithm

geeks for geeks
Programiz