NATIONAL INSTITUTE OF TECHNOLOGY DELHI

CSBB 202 Design and Analysis of Algorithms

SYNOPSIS ON

"**Optimising Pathfinding by Comparison of Different Algorithms"**

by

Parth Singh (241210086)

Vaishali Yadav (241210123)

Vibhuti Dureja (241210129)

B.Tech CSE

Submitted to

Dr. Gunjan

Assistant Professor

Department of Computer Science and Engineering

INTRODUCTION:

In modern navigation systems and location-based services, determining the shortest route between two geographical points is a vital problem. This project aims to design and develop a web-based application that integrates with the Google Maps API to allow users to visually select start and end points. The application calculates and compares the shortest path between these points using different pathfinding algorithms such as Dijkstra's Algorithm and ultimately provides the most optimized route with relevant details. The system uses Python (Flask) for backend logic and HTML, JavaScript for an interactive and responsive front-end interface.

LITERATURE REVIEW:

- The most seminal work in this area is Dijkstra's Algorithm, introduced by Edsger W. Dijkstra in 1959 [1]. This greedy algorithm efficiently solves the single-source shortest path problem for graphs with non-negative edge weights. It serves as the fundamental building block for many routing applications.
- A significant improvement for goal-directed searching is the A* search algorithm, first proposed by Peter Hart, Nils Nilsson, and Bertram Raphael in 1968 [2]. A* enhances Dijkstra's by incorporating a heuristic function, h(n), which estimates the cost from the current node n to the goal.
- By prioritizing exploration based on f(n) = g(n) + h(n) (where g(n) is the path cost from the start), A* effectively prunes the search space, making it substantially more efficient than Dijkstra's for large, known graphs like geographic maps [3].
- The application of these algorithms evolved from abstract networks to real-world geographical mapping. Geographic Information Systems (GIS) and the integration of the Google Maps API utilize these shortest path concepts for large-scale outdoor routing [4].
- The success of these systems relies on modelling the road network as a massive graph and employing optimization techniques like Contraction Hierarchies or A* with precise distance heuristics to provide real-time results [5].

OBJECTIVE:

- To implement multiple pathfinding algorithms for comparison of their performance and results.
- To integrate Google Maps API for realistic map visualization and route plotting.
- To create a user-friendly web interface that allows users to input locations visually and see computed paths dynamically.
- To enable real-time route computation with appropriate error handling for invalid or unreachable inputs.
- To compare algorithm performance in terms of time efficiency, path optimality, and computational cost using real map data.

TOOLS AND TECHNOLOGIES:

- Programming Language: Python, JavaScript
- Backend Framework: Flask
- Frontend Technologies: HTML5, CSS3, JavaScript (jQuery, AJAX)
- API: Google Maps API
- IDE: Visual Studio Code / PyCharm
- Version Control: Git and GitHub

PROJECT STEPS:

- Study pathfinding algorithms, review API documentation, and define user requirements.
- Configure Flask environment, generate Google Maps API key, and integrate the API for real-time data.
- Implement and optimize Dijkstra and A* algorithms for shortest path computation using actual map data.
- Build interactive web pages using HTML, CSS, and JavaScript for input and map visualization.
- Use AJAX for asynchronous data exchange between client-side JS and Flask backend.
- Perform algorithm performance tests under various route conditions and handle API edge cases.
- Finalize the web interface and deploy on an online server.

EXPECTED OUTCOME:

- A functional web-based application capable of finding and visualizing shortest routes between user-selected points.
- Comparison insights between Dijkstra's and A* algorithms in terms of speed, accuracy, and resource usage.
- A responsive and user-friendly interface integrated with Google Maps API.
- Enhanced understanding of pathfinding algorithm behaviour on real-world map data.
- A deployable web application accessible through a local or hosted server.

TIMELINE:

- Week 1: Understanding algorithms, exploring Google Maps API and implementing Flask project setup and focusing on API key integration.
- Week 2 : Implementing Dijkstra and A* algorithms in Flask and to build UI, map visualization, and real-time data handling.
- Week 3: Evaluate both algorithms and optimize performance using various resources.

REFERENCES:

[1] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269-271.

[2] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100-107.

[3] Pearl, J. (1984). Heuristics: Intelligent Search Strategies for Computer Problem Solving. *Addison-Wesley*. (A foundational text discussing A* and its optimality properties).

[4] Winter, S. (2002). Modeling costs of turns in route planning. In *International Conference on Geographic Information Science (GIScience 2002)*, 321–335. (Discusses how pathfinding algorithms are adapted for real-world geographic constraints).

[5] Sanders, P., & Schultes, D. (2007). Engineering Highway Hierarchies. In *European Symposium on Algorithms (ESA 2007)*, 28-40. (Covers advanced graph acceleration techniques used in modern routing engines).