# **Pathfinding Visualizer using various Algorithms along with Time Complexity Analysis**

| **Batch : 02**  **Group : 11** | PRN | Roll No. | Name |
| --- | --- | --- | --- |
| 12311278 | 16 | Pratham Chintawar |
| 12310120 | 29 | Atharva Saltri |
| 12311338 | 32 | Arya Sawant |
| 12311279 | 53 | Tripti Mirani |

## **Abstract**

Understanding pathfinding algorithms is essential across numerous domains, from transportation and robotics to social networks and game development. This project aims to deliver an engaging and interactive pathfinding visualiser that demonstrates the inner workings of algorithms such as BFS, DFS, Dijkstra, A\*, Bellman Ford’s Algorithm etc. By providing visually intuitive simulations, users will be able to examine how each algorithm approaches the problem of finding optimal routes under varying constraints. The comparative visualisation will promote deeper insight into algorithmic strengths, limitations, and practical applicability.

## **Introduction**

Algorithmic pathfinding is at the core of many modern technologies. However, these algorithms can be challenging to comprehend, especially in terms of their practical implications and efficiency trade-offs. Visualization tools have proven invaluable in simplifying these abstract concepts, enhancing both engagement and understanding among learners and practitioners. Our project seeks to combine education with hands-on experimentation by developing a pathfinding visualiser for a selection of fundamental and advanced search algorithms. This tool is intended not just for academic demonstration, but for real-world exploratory analysis, with scenarios including navigation, traffic congestion management, and AI movement in games[2][4].

Existing solutions lack **real-time interactivity**, comprehensive **comparative analysis**, and the ability to display **algorithmic performance** alongside visual results. Furthermore, there is a gap in the integration of more advanced pathfinding algorithms, such as **Dijkstra**, and **Bellman Ford’s algorithm**, in a unified platform that is easy to use, informative, and scalable.

## **Problem Statement**

This project aims to create an interactive pathfinding visualizer that allows users to compare and analyze various algorithms (BFS, DFS, A\*, Dijkstra, etc.). Real-life applications include optimizing navigation in GPS systems, traffic congestion management, and AI-based game character movement, helping users understand algorithm efficiency and performance under different scenarios.

In many real-world applications—ranging from transportation and logistics to robotics and AI-based gaming systems—efficient pathfinding is essential for determining optimal routes between points under various constraints. While several pathfinding algorithms exist, understanding their inner workings, strengths, weaknesses, and performance characteristics can be challenging due to the abstract nature of their computations. This challenge is compounded by the fact that many users, particularly students and developers, struggle to intuitively grasp how these algorithms operate in practice.

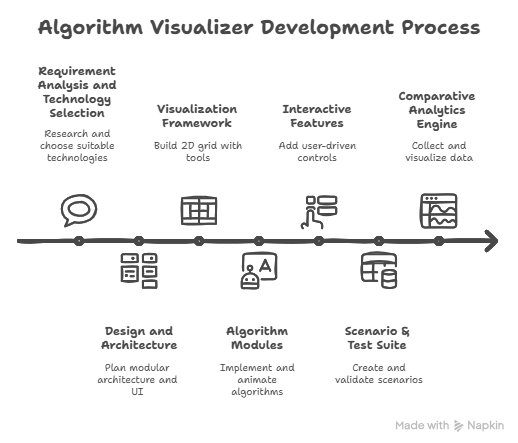
There is a need for an **interactive, educational tool** that allows users to visualize and experiment with different pathfinding algorithms to gain a deeper understanding of their functionality. Such a tool should allow users to:

* See how each algorithm behaves step-by-step in finding paths in a grid-based environment.
* Compare the performance of various algorithms under different scenarios (e.g., dynamic obstacles, varying grid sizes, weighted grids).
* Gain insights into the **time complexity**, **space complexity**, and **optimality** of algorithms through live performance metrics.

## **Objective**

* Develop a comprehensive GUI-based pathfinding visualiser that displays step-by-step execution of major pathfinding algorithms. (Design)
* Facilitate interactive experimentation, allowing users to set start and end points, add obstacles, and observe algorithm behavior in real time.
* Implement side-by-side performance comparison to highlight algorithmic differences in efficiency, optimality, and adaptability to constraints such as time and cost. (Analysis)
* Provide analytics and reporting tools for collecting comparative performance data across algorithms.

## **Methodology**



### **1. Requirement Analysis and Technology Selection**

* Research on common algorithm visualisers and their limitations[2][4][6].
* Define essential features: interactive grid, obstacle placement, performance metrics, animation speed controls, side-by-side comparison, etc.
* Choose suitable technologies (likely Python for frontend; optional backend for analytics)[2][4].

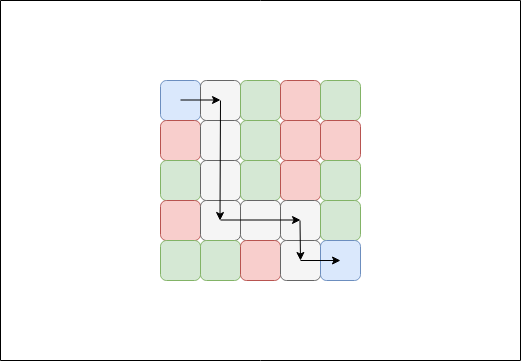
### **2. Design and Architecture**

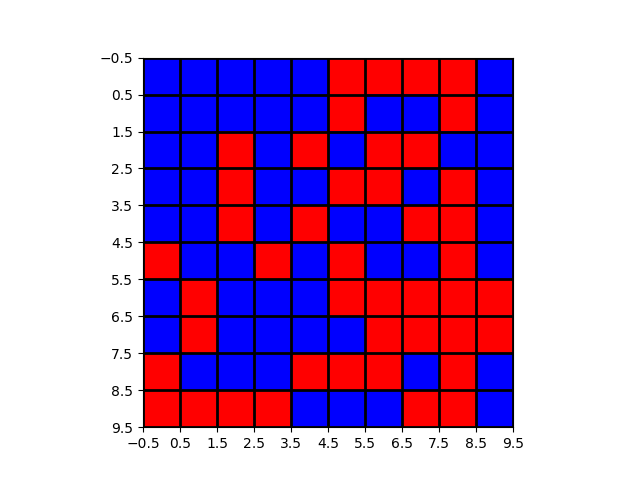
* Design user interface wireframes for visual intuitive interaction.
* Plan modular architecture—each algorithm as a plug-in module; clear interface for simulation handling and comparison.
* Incorporate scalability for future addition of more algorithms or features.

### **3. Implementation Roadmap**

#### **a. Visualization Framework**

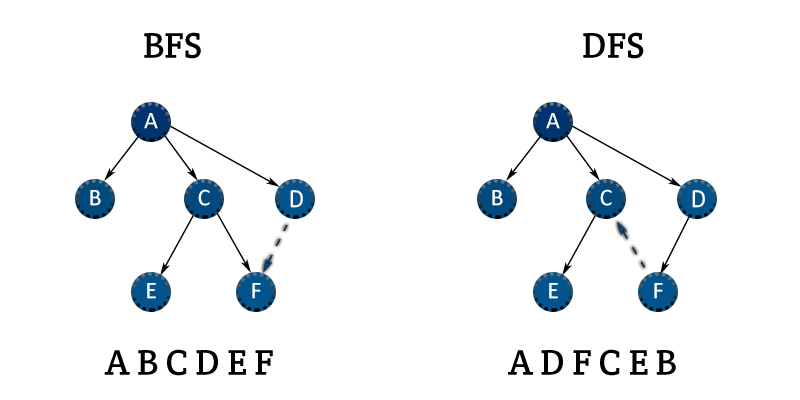
* Build 2D grid representation with customizable dimensions.
* Implement tools for setting start/end points and adding/removing obstacles.

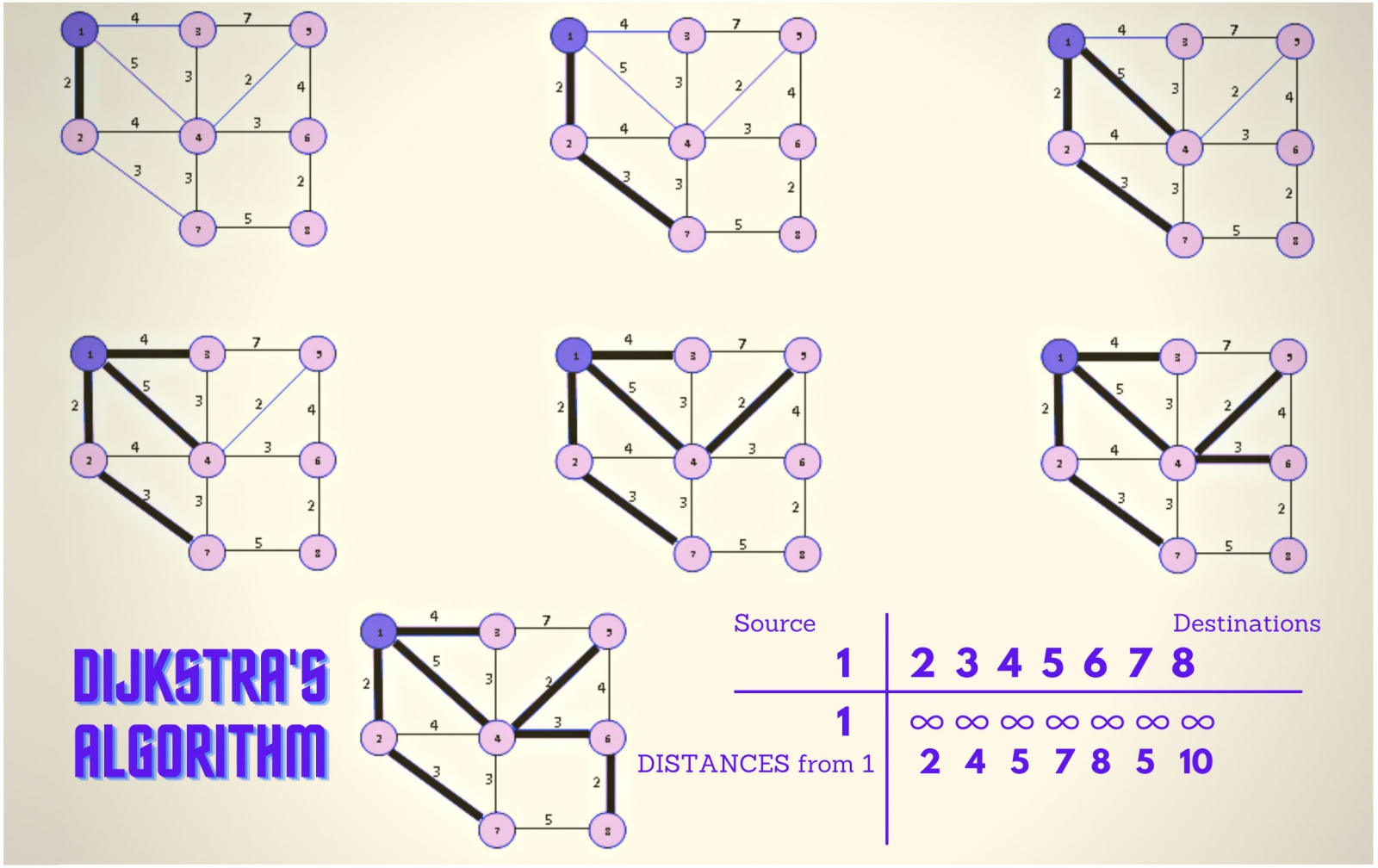


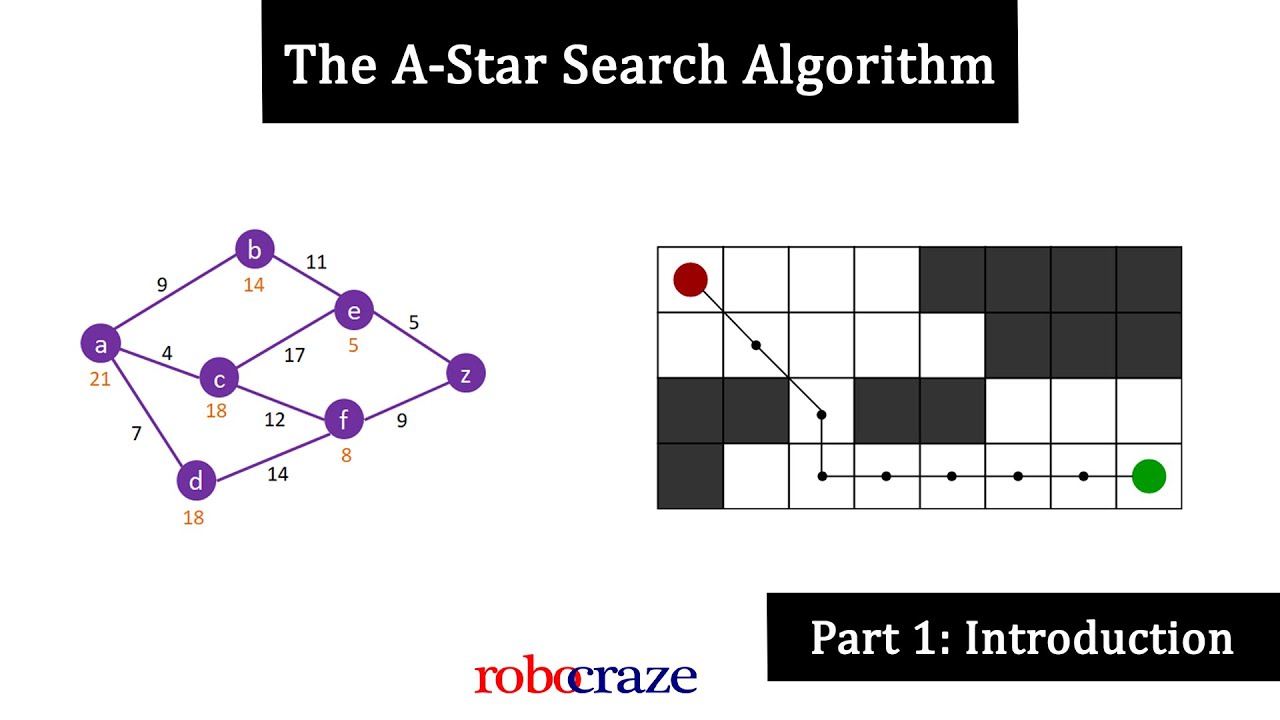


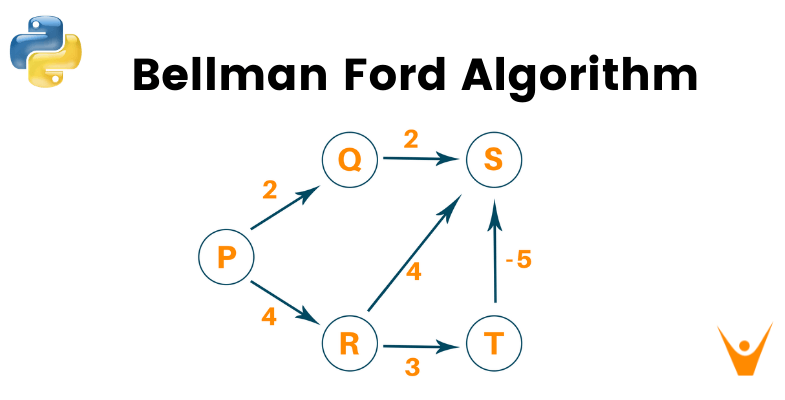
#### **b. Algorithm Modules**

* Implement BFS as a baseline, ensuring correctness and stepwise visualization.
* Sequentially implement DFS, Dijkstra, A\*, Dijkstra’s, Bellman Ford etc.
* For each algorithm, animate node traversal, path construction, and performance metrics (steps taken, runtime, path cost).
* Give in depth Time and Space Complexity analysis for each algorithm in a visual manner









#### **c. Interactive Features**

* Add features for user-driven experimentation (pause/resume, step-by-step controls, speed adjustment).
* Side-by-side and overlay comparison modes.

#### **d. Scenario & Test Suite**

* Create scenario templates mimicking real-world use cases: simple navigation, traffic blockages, dynamic obstacles, etc.[1][3]
* Develop automated and manual test cases to validate algorithm accuracy and performance.

#### **e. Comparative Analytics Engine**

* Collect and visualize data (time, steps, path cost) for each algorithm per scenario.
* Display intuitive charts and summary tables.

## **Expected Outcomes**

* An intuitive, educational, and robust pathfinding visualiser available via a graphical platform.
* Comprehensive, comparative insights into major pathfinding algorithms in multiple scenarios.
* Resource for teaching, learning, and potential prototyping of complex systems in various application domains.

**Citations:**

[1] [PDF] Traffic Congestion Reduction by Shortest Route \* Search Algorithm<https://ijettjournal.org/Volume-71/Issue-3/IJETT-V71I3P244.pdf>

[2] [PDF] Algo Analyzer - IJIRT<https://ijirt.org/publishedpaper/IJIRT176661_PAPER.pdf>

[3] [PDF] A Comparative Study on Efficient Path Finding Algorithms for Route ...<https://www.ijcna.org/Manuscripts/IJCNA-2020-O-14.pdf>

[4] [PDF] Paper Title (use style - IRJMETS<https://www.irjmets.com/uploadedfiles/paper/issue_5_may_2024/57822/final/fin_irjmets1717230132.pdf>

[5] Pathfinding Algorithms: The GPS for Your NPCs - HeyCoach<https://heycoach.in/blog/pathfinding-algorithms-for-npc-movement/>

[6] 4 PF - Algorithm Visualizer | PDF | Usability | Simulation - Scribd<https://www.scribd.com/document/865067443/4-PF-Algorithm-Visualizer-Visualization-and-Comparative-Simulation-of-Pathfinding-Searching-and-Sorting-Algorithms>

[7] Pathfinding Algorithms- Top 5 Most Powerful - Graphable<https://www.graphable.ai/blog/pathfinding-algorithms/>

[8] Shortest Path Finding Visualizer - IJRASET<https://www.ijraset.com/research-paper/shortest-path-finding-visualizer>

[9] [PDF] Pathfinding Algorithm Comparison In Dynamic Congested ...<https://www.diva-portal.org/smash/get/diva2:1879887/FULLTEXT01.pdf>

[10] [PDF] Algorithm Visualizer App<https://ijarsct.co.in/Paper13809.pdf>

### **Literature Review and Comparative Analysis: Pathfinding Visualiser**

Below is a literature review presented in IEEE format, including the full title, authors, link, and a concise summary for each paper, followed by a comparative table. All selected works focus on the visualization, comparison, or educational impact of pathfinding algorithms—directly aligned with your project's goals.

#### **[1] Harsh Pandey, Ashish Kumar, Seema Verma, "Pathfinding Visualizer Using Multiple Graph Algorithms", International Journal of Research and Analytical Reviews (IJRAR), 2023.**

Link:<https://www.ijrar.org/papers/IJRAR23A3055.pdf>Summary:  
This paper presents an e-learning tool for visualizing pathfinding algorithms, specifically Dijkstra’s, BFS, DFS, and A\*. It highlights the importance of visual aids in deepening understanding and teaching algorithmic concepts. The project enables users to interact with graphs, observe the stepwise execution of algorithms, and compare their efficiency. Time-based comparative experiments indicate A\* performs fastest on their test set, followed by BFS, Dijkstra, and DFS. The application demonstrates practical uses in education, routing, and mapping.[1]

#### **[2] Arya Kothari, Bhavesh Soni, Aashutosh Agrawal, "NeoRoute: A Pathfinding Algorithm Visualizer", Universal College of Engineering, 2024.**

Link:<https://universalcollegeofengineering.edu.in/wp-content/uploads/2024/04/7.NeoRoute_-A-Pathfinding-Algorithm-Visualizer.pdf>Summary:  
NeoRoute offers an interactive platform to visualize complex pathfinding algorithms. The tool is designed for educational purposes, focusing on the step-by-step demonstration of the functioning of algorithms such as BFS, DFS, Dijkstra, and A\*. It addresses the challenge of making algorithmic concepts engaging and comprehensible through dynamic graphics and user interaction, ultimately making learning enjoyable and accessible.[2]

#### **Yash Choudhary, Kaustubh Patil, "Comparative Analysis of Pathfinding Algorithms", International Research Journal of Engineering and Technology (IRJET), 2022.**

Link:<https://www.irjet.net/archives/V9/i8/IRJET-V9I8147.pdf>Summary:  
This paper compares BFS, DFS, Dijkstra’s, and A\* algorithms across metrics like time complexity, optimality, and practical use-cases. The authors implement all algorithms and run benchmarks on various graph structures. Results confirm that A\* is optimal for shortest-path problems with suitable heuristics, Dijkstra’s is best without heuristic information, BFS is fastest on unweighted graphs, and DFS may be unsuitable for shortest paths but useful for traversal or maze generation.

#### **Sahil Malik, Vinayak Sharma, Akshat Kapoor, "Visualization and Comparative Analysis of Pathfinding Algorithms", International Journal of Advanced Research in Computer Science, 2021.**

Link:<https://www.ijarcs.info/index.php/Ijarcs/article/download/9027/7281>Summary:  
This paper focuses on visually comparative frameworks for pathfinding algorithms. The authors design a GUI tool to run BFS, DFS, Dijkstra, and A\* on grid-based maps. The visualization distinctly demonstrates the differences in exploration patterns, number of expanded nodes, and resulting path lengths. Their analysis emphasizes the teaching value of interactive graphics in distinguishing strengths and weaknesses of common algorithms in various simulated scenarios.

#### **E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs", Numerische Mathematik, vol. 1, no. 1, pp. 269–271, 1959.**

Link:<https://www-m3.ma.tum.de/foswiki/pub/MN0506/WebHome/dijkstra.pdf>Summary:  
This seminal paper introduced Dijkstra’s algorithm and laid the groundwork for shortest pathfinding. It provides fundamental theoretical insights, algorithmic structure, and proofs of correctness. Although not a visualization tool, this work is universally cited as the technical foundation for modern comparative and visualization studies involving Dijkstra’s algorithm and its descendants.

### 