# New England Route Planner

CS566 A2 – Analysis of Algorithms

By Gowtham Saravanan and Sarvesh Krishnan Rajendran

Abstract

The focus of this research project is on the design and analysis of algorithms, particularly exploring the applications and intricacies of Breadth-First Search (BFS), Depth-First Search (DFS) and Dijkstra's algorithm in the context of graph theory. This project aims to provide a comprehensive understanding of these algorithms' functionalities and their practical implications in real-world scenarios, particularly in route planning applications. The research will contribute insights into algorithm design, analysis, and practical implementation, fostering a deeper appreciation for the role of algorithms in solving complex problems.

Index

## I – Introduction:

In the realm of computer science, the study and application of algorithms play a pivotal role in solving complex problems and optimizing various processes. Algorithms are the backbone of computational solutions, and their design and analysis constitute a thriving field of research. This project embarks on a journey into the heart of algorithmic design, focusing on the exploration of two fundamental graph search algorithms— Breadth-First Search (BFS), Depth-First Search (DFS) and Dijkstra's algorithm.

## II – Graph Data Structure and Algorithm:

Graphs, indeed, represent a universal and indispensable data structure, providing a means of representing ordered sets, where some pairs of elements are linked by a link. The inter-connected nodes are usually displayed as vertices or nodes, and the association between them indicated by the edges. A graph shows vertices as cities in a transportation map or as people in a social network. Vertex is used instead of entities at this point. Edges represent relationships or routes between two vertices. They may be either directed or undirected too. The direction of edges classifies directed graphs, and the opposite case, no direction, occurs in undirected graphs, suggesting bidirectional relationships in both.

## III – Theory behind Breadth First Search Algorithm:

The BFS (Breadth First Search) algorithm for exploring graphs is one of the main ones, being widely applicable in disparate areas such as network exploration, game development as a part of pathfinding, and artificial intelligence. It comes down to checking all vertices in the level and then moving to the next before passing on up, therefore covering the level by level of investigation. This systematic approach is based on a queue data structure which facilitates the use of a first-in, first-out principle that helps in traversing the nodes in an orderly level-by-level manner.

## IV – Theory behind Depth First Search Algorithm:

DFS (Depth first search) is the most basic algorithm that is usually employed for traversing tree or graph structures. It places great emphasis on, one should plunge as deep to the root as possible and then, vice versa, finally return to the branch i.e. explore the depth-wise along branches before nodes. This approach is especially applicable in circumstances when these thorough searches are in need, for example, when solving a puzzle, finding a way in a labyrinth, and analyzing the connectivity of a network.

## V – Theory behind Dijkstra Algorithm:

Dijkstra Algorithm is a well-known algorithm that has been applied in computing and operations research. It is used to discover a shortest-paths tree for a source vertex that connects to all other vertices in the graph which is weighted with non-negative weights. Dijkstra's algorithm first appeared back in 1956, and it plays an essential role in many applications such as geographic mapping services, network routing protocols, and as a subroutine in other algorithms including the ones that can find a minimum spanning tree.

## VI – Application:

We are creating an API to help tourists travel around the capital cities of New England states. From Boston to cities like Hartford and Providence, we are leveraging a graph-like structure to showcase how these different city locations are connected and how it is possible to travel between places. As shown in Figure 1, we decided on the list of New England State Capitals include:

1. Hartford
2. Augusta
3. Boston
4. Concord
5. Providence
6. Montpelier

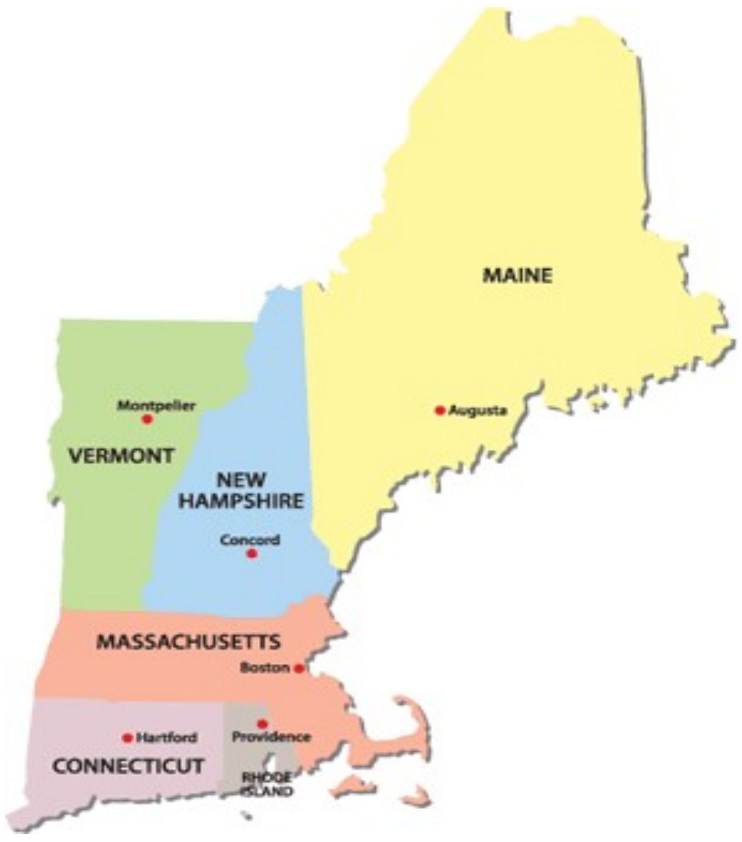
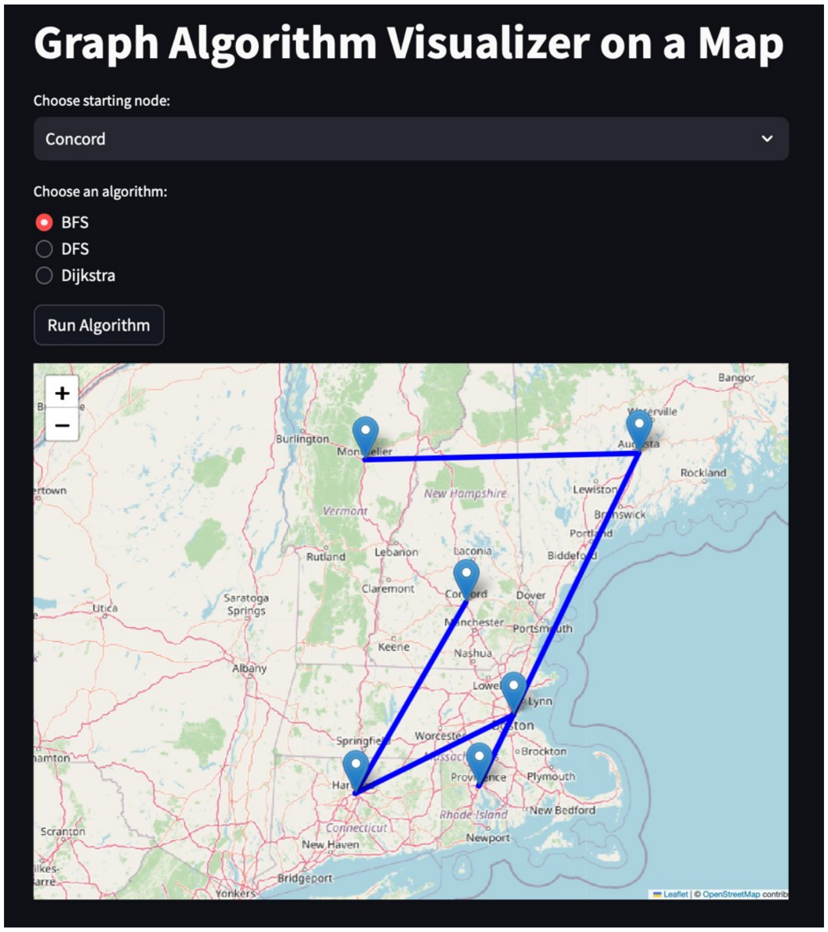


Figure 1: Mapping of all New England State Capitals

## VII – Implementation and Conclusion:

We have developed a graph algorithm visualizer using Python, incorporating the Streamlit framework for the user interface, folium for mapping, and NetworkX for managing graph operations. Our script introduces a custom `Graph` class that handles nodes, edges, and their respective distances, enabling the addition of both nodes and edges with geographical coordinates and specified distances.



Graph Algorithm Visualizer on a Map

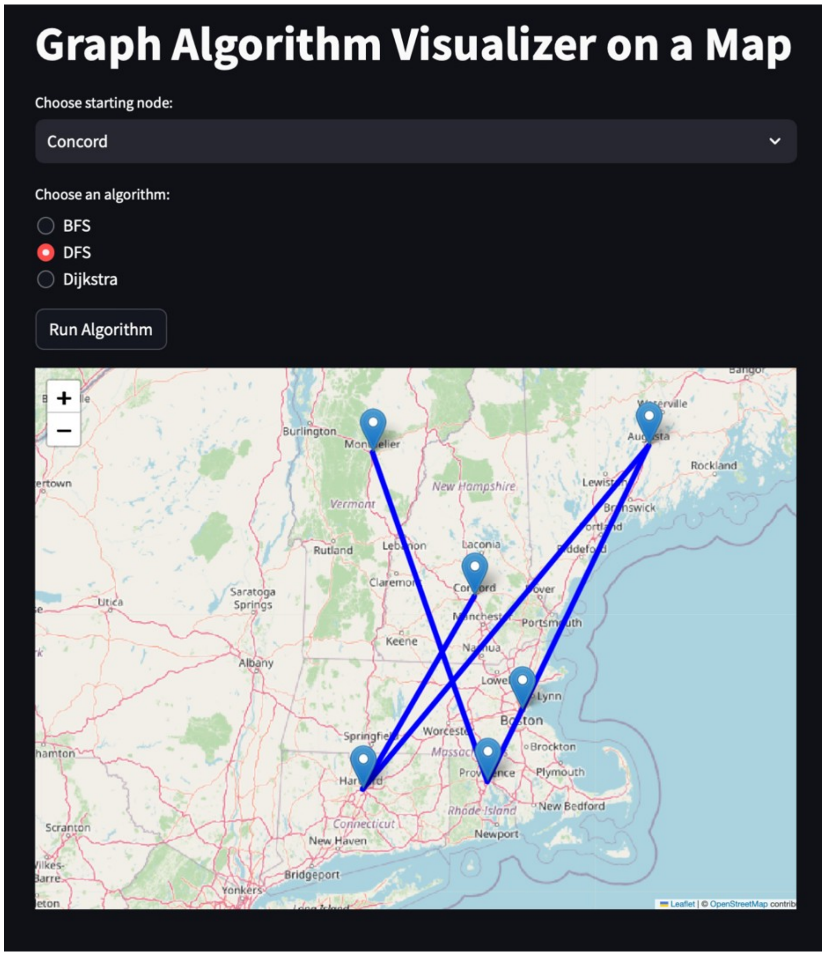
Choose starting node: Concord

Choose an algorithm:
[✓ BFS]
DFS
Dijkstra

Run Algorithm

Figure 2.1: BFS Algorithm

At the heart of our application are three fundamental graph algorithms: Breadth-First Search (BFS), Depth-First Search (DFS), and Dijkstra's algorithm. We have implemented these to facilitate the exploration of paths within the graph—BFS (Fig2.1) and DFS (Fig2.2) for general traversal and Dijkstra (Fig2.3) for identifying the shortest path from a given starting node to all other nodes.

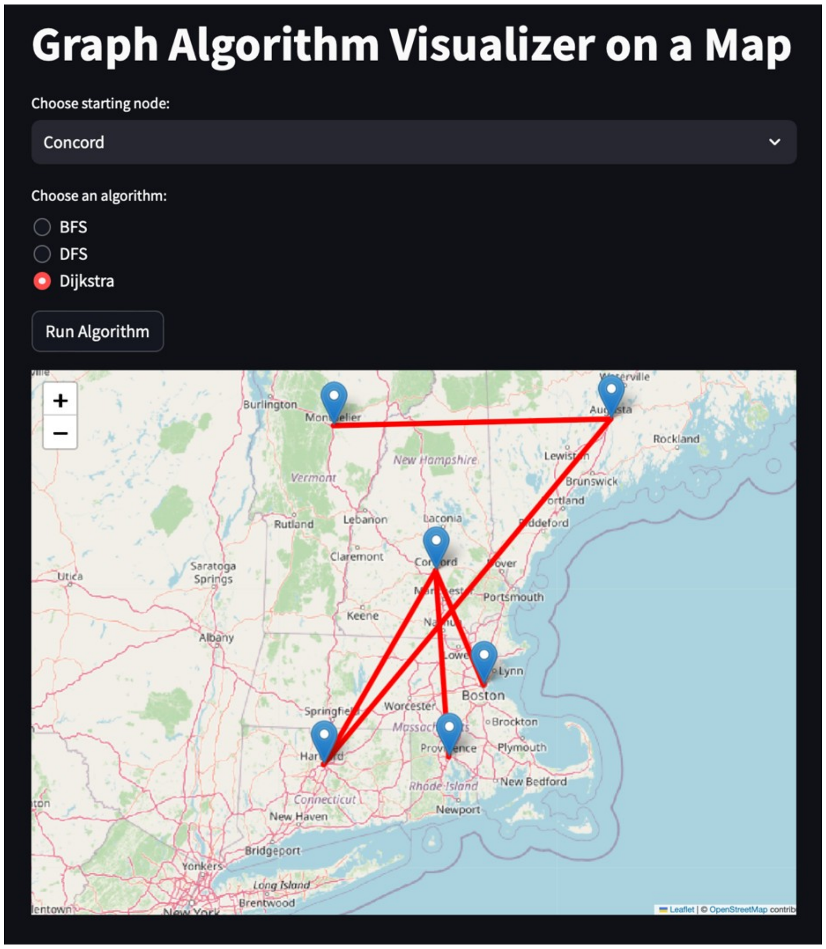

Graph Algorithm Visualizer on a Map

Choose starting node: Concord

Choose an algorithm: BFS [✓ DFS] Dijkstra

Figure 2.2: DFS Algorithm

Users can interactively choose a starting node and the algorithm they want to run through a user-friendly interface. Once an algorithm is executed, the results are visually displayed on a map, utilizing the folium library to enhance user engagement and provide a geographical context to the data.



# Graph Algorithm Visualizer on a Map

Choose starting node: Concord

Choose an algorithm:
BFS [✓ Dijkstra] DFS

Figure 2.3: Dijkstra Algorithm

We have integrated various Streamlit features, including select boxes, radio buttons, and action buttons, to create a dynamic and interactive user experience. This setup is especially useful for analyses in areas where geographical data is crucial, such as in routing and network optimizations. By combining graph theory with geographic mapping, my tool offers a practical and sophisticated means for visualizing and deciphering complex relationships in a spatial format. The Web Interface that we created is http://10.0.0.77:8501/

References:

https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/

https://medium.com/@chaitanyasirivuri/building-your-first-streamlit-app-a-step-by-step-tutorial-e058d5dfe5f4

https://www.scaler.com/topics/data-structures/dijkstra-algorithm/

https://www.sciencedirect.com/topics/computer-science/dijkstra-algorithms