Project Report

on

BELLMAN FORD ALGORITHM VISUALIZER

Submitted by

Vishnu Kumar P- (20BCS138)

Under the guidance of

Dr.Sunil C K,Assistant Professor

Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

14/05/2024

Contents

List of Figures

1 Abstract

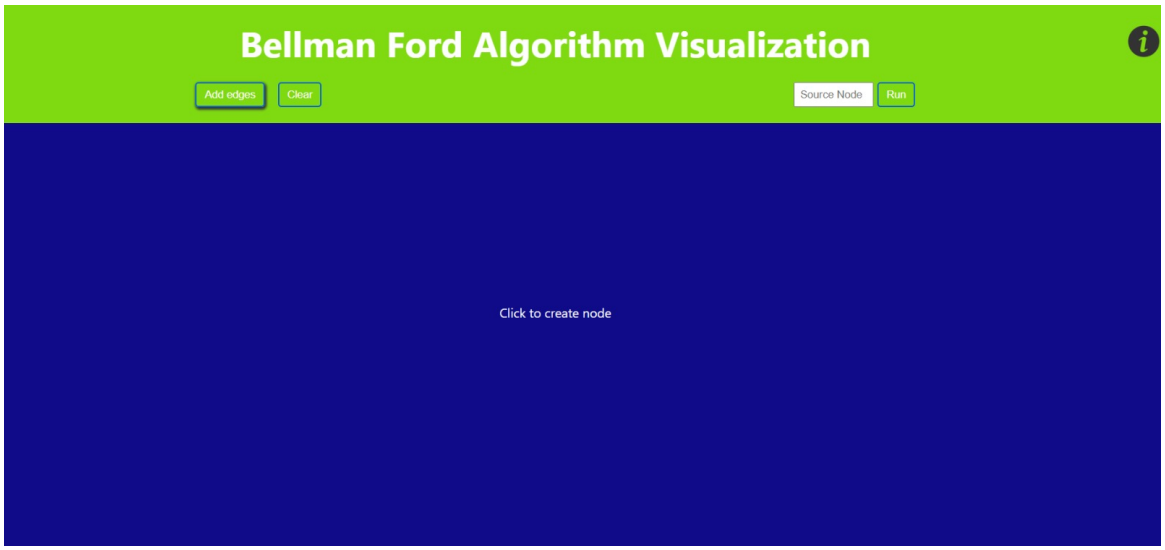
The Bellman-Ford Algorithm Visualizer is a web-based tool designed to illustrate the functionality and operation of the Bellman-Ford algorithm, a fundamental algorithm in graph theory used to find the shortest paths between vertices in a graph. This interactive visualization tool provides users with a hands-on experience to explore and understand the Bellman-Ford algorithm's inner workings and its application in solving shortest-path problems. Through an intuitive and user-friendly interface, users can create, manipulate, and analyze graphs, add nodes and edges, and initiate the shortest path computation from a specified source vertex. The visualizer employs dynamic visualization techniques to demonstrate the algorithm's iterative process of updating distance estimates and identifying the shortest paths.

Additionally, it offers features such as highlighting the shortest path on the graph and displaying detailed path information, enhancing users' comprehension and learning experience. With its educational value and practical applications, the Bellman-Ford Visualizer serves as a valuable resource for students, educators, and professionals interested in graph algorithms, network optimization, and algorithmic problem-solving.

2 Home Page (HTML CSS)

The code represents the structural foundation for a web page designed to visualize the Bellman-Ford algorithm for finding the shortest path in a graph. This algorithm is fundamental in graph theory and is ulized in various applications, including network roung protocols and distance-vector roung algorithms.

At the core of the web page lies the header section, encapsulating essential navigaon elements and opons for user interacon. The title secon within the header identifies the web page's purpose, offering a hyperlink for easy navigaon back to the homepage. Adjacent to the tle, an icon resembling a setings cog links to additional instructions or sengs related to the visualization.



Within the options secon of the header, users are empowered with funconality to manipulate the graph visualization. The "Add edges" button facilitates the addion of edges between nodes, crucial for constructing the graph representaon. Conversely. The "Clear" button allows users to reset the drawing area, wiping the canvas clean of any existing graph elements. These interacve features empower users to engage dynamically with the visualizaon, fostering a more immersive learning experience.

Further down the HTML structure lies the main content area, comprising the drawing area and path display. The drawing area serves as the canvas for visualizing the graph and its consistent nodes and edges. Accompanying this canvas is a helpful instructional prompt, guiding users on interacting with the visualization. Meanwhile, the path display secon presents the shortest path calculated by the Bellman-Ford algorithm, enhancing the user's understanding of the algorithm's operaon and efficacy.

In summary, the HTML code provides a robust framework for building a web page dedicated to visualizing the Bellman-Ford algorithm. By incorporang intuitive navigaon elements, interacve funconality, and informave content secons, the web page offers a user-friendly plaorm for exploring and comprehending this essential graph algorithm.

3 Instructions part (HTML CSS)

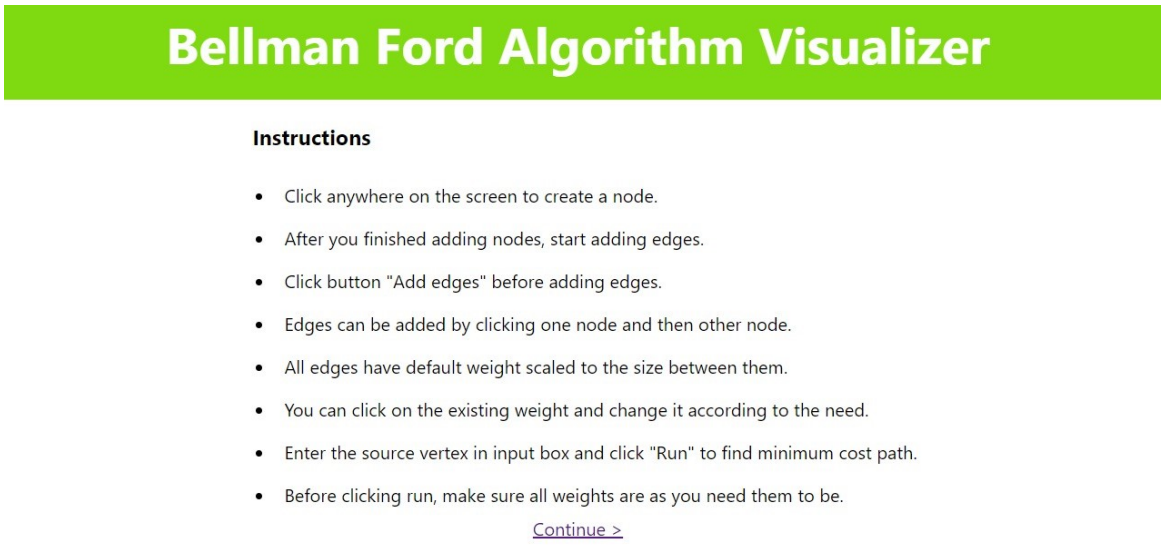
The code encapsulates a set of instructions for a web-based visualization tool designed to illustrate the Bellman-Ford algorithm. This algorithm, a fundamental component of graph theory, is employed to determine the shortest path between nodes in a weighted graph. The instructions outlined in the HTML document serve as a user guide, facilitating the effective utilization of the visualization tool to comprehend the functionality and operation of the Bellman-Ford algorithm.

The HTML structure begins with the declaraon of the document type and language, ensuring compability and accessibility across various web browsers. Metadata tags specify the character encoding, viewport sengs, and compability direcves, opmizing the rendering and display of the web page. Additionally, external style sheet links and inline CSS rules are included to define the visual appearance and layout of the content, enhancing user experience and readability.

The Header section of the HTML document features a title element, providing a succinct description of the visualization tool as the "Bellman Ford Algorithm Visualizer". This title serves as a navigational aid, enabling users to easily identify and access the visualization tool from other web pages. Furthermore, the header reinforces the navigational structure by incorporating a hyperlink to the homepage, facilitating seamless navigation between different sections of the website.

The main content area of the web page comprises a series of instructions delineated in an unordered list format. These instructions elucidate the sequenal steps required to utilize the visualization tool effectively for exploring the Bellman-Ford algorithm. Users are guided through the process of adding nodes and edges to the graph, adjusting edge weights, specifying the source vertex, and execung the algorithm to compute the shortest path. Each instrucon is succinctly arculated, ensuring clarity and comprehensibility for users of varying levels of familiarity with graph algorithms.

Concluding the HTML document is a hyperlink labeled "Continue ¿", which directs users back to the homepage of the visualization tool. This hyperlink serves as a call-to-acon, prompting users to proceed with utilizing the visualization tool after familiarizing themselves with the provided instructions. Overall, the HTML code encapsulates a comprehensive guide for leveraging the



Bellman-Ford Algorithm Visualizer, empowering users to effectively explore and understand the intricacies of the Bellman-Ford algorithm through interacve graph visualizaon.

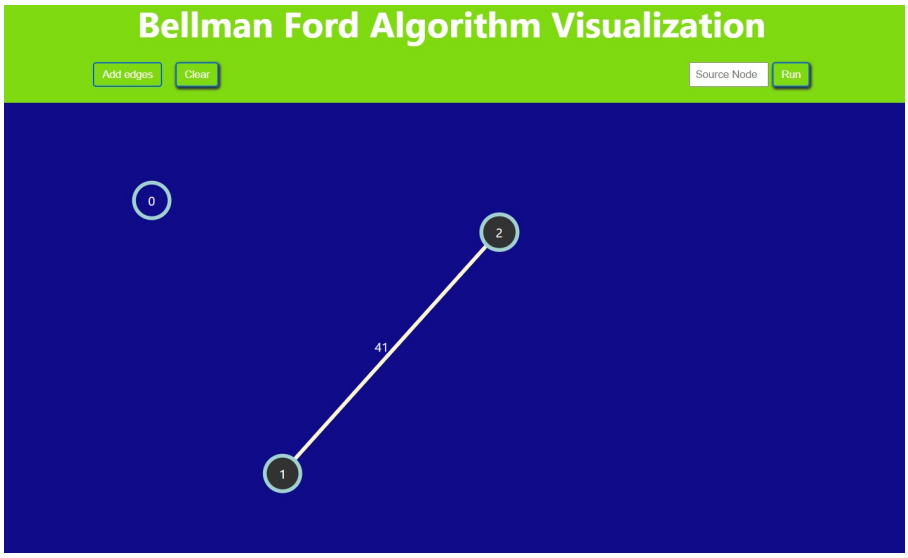
4 Algorithmic Part (Javascript)

4.1 Initialization and Alert

This section of the code initializes variables and checks whether the user has been alerted before with a local Storage flag. If not, it displays an alert message to encourage the user to read the instructions before proceeding with using the visualization tool. This alert serves as a helpful reminder for users to familiarize themselves with the tool's functions and guidelines, ensuring a smoother experience with the visualizaon.

4.2 AddEdges Function

The addEdges function allows users to start adding edges between nodes after creating at least two nodes on the screen. It handles the logic for enabling edge creaon by seng a boolean flag, disabling the "Add edges" button once edges can be added, and enabling the "Run" button to



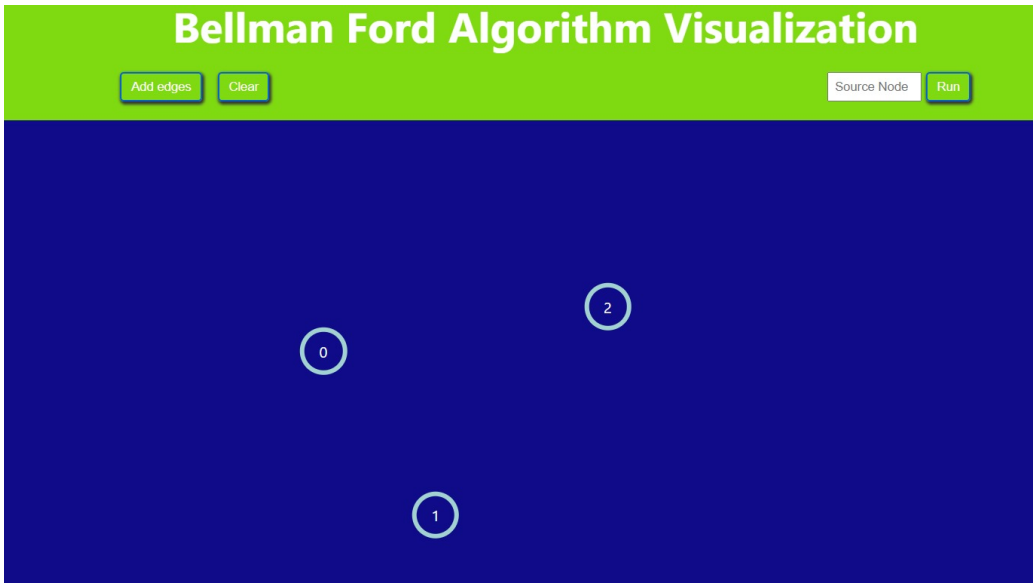
find the shortest path. Addionally, it inializes a distance matrix to store edge weights between nodes, essential for compung the shortest path using the Bellman-Ford algorithm.

4.3 Append Block Function

The append Block function is responsible for dynamically adding a new block (representing a node) to the drawing area when the user clicks on the screen.It creates a new div element with a unique ID and event listener to manage edge creation between nodes. This function enhances the interacvity of the visualization tool by allowing users to intuitively add nodes to the graph canvas, facilitating the exploration of graph structures.

4.4 Blocks Event Listener

This event listener triggers when the user clicks on the drawing area, invoking the append Block function to create a new node unless the "Add edges" mode is acve or the maximum number of nodes (12) has been reached. By monitoring user interactions with the drawing area, this event listener ensures a responsive and user-friendly experience, allowing for seamless node creation and graph exploration.

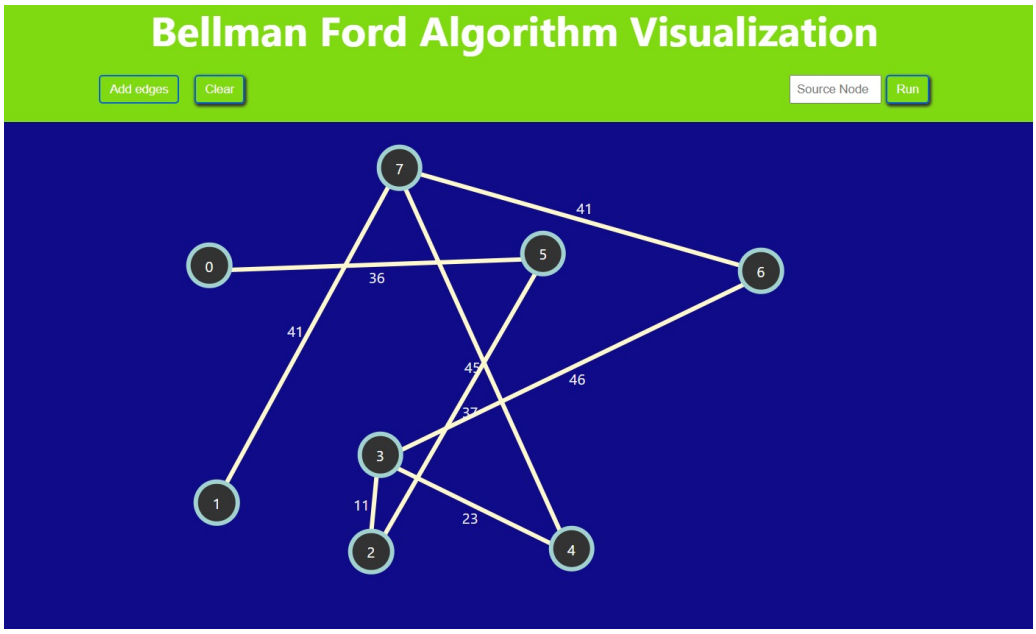


4.5 DrawLine Function

The drawLine function is responsible for drawing a line (representing an edge) between two nodes on the screen. It calculates the length and angle of the line based on the positions of the nodes and updates the distance matrix with the corresponding edge weight. By dynamically generang visual representaons of edges between nodes, this function contributes to the visualizaon of graph structures, aiding users in understanding the connecvity and relationships within the graph.

4.6 DrawUsingId Function

The drawUsingId function coordinates the drawing of a line between two nodes based on their unique IDs. It retrieves the posions of the nodes and invokes the drawLine function to create the edge between them. This function facilitates the seamless visualizaon of edges between nodes, enhancing the clarity and comprehensibility of the graph representaon for users interacng with the visualizaon tool.

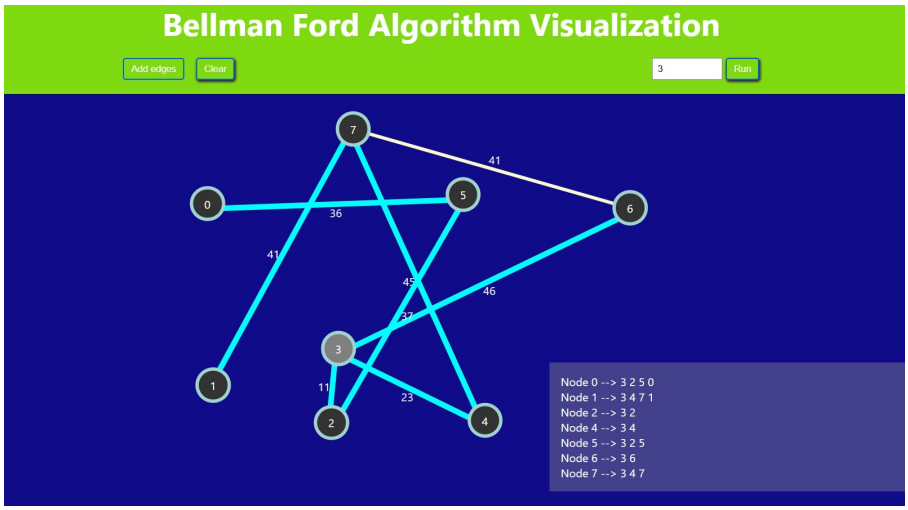


4.7 Bellman Ford Algorithm function

This function orchestrates the process of finding the shortest path from a specified source vertex to all other verces in the graph. It begins by extracng the source vertex from the input element passed as an argument, ensuring its validity. If the source vertex is valid, it marks it as visited by changing its background color to grey. Then, it inializes arrays to store visited verces, parent vertices, and distances from the source vertex. Employing the Bellman-Ford algorithm's relaxaon step, it iterative updates the distance array by considering all edges in the graph. Once the shortest path distances are computed, it calls the indicated path funcon to visualize the shortest paths.

4.8 Printpath function

This function recursively constructs the shortest path from a given vertex to the source vertex. It takes as parameters the array containing parent verces, the current vertex index, and the paragraph element to update with the shortest path information. If the current vertex is the source vertex, the function terminates. Otherwise, it recursively calls itself with the parent of the current vertex to continue construcng the shortest path. After construcng the shortest path



sequence, it updates the paragraph element with the vertex sequence and appends it to the path display area. Additionally, it calls the colorEdge function to visually highlight the edges corresponding to the shortest path.

5 Applications

The Bellman-Ford algorithm visualizer serves as a valuable educational and praccal tools with several applications

5.1 Learning Tool for Students

The visualizer provides hands-on learning experience for students studying graph theory and algorithms. By interacvely exploring the Bellman-Ford algorithm's operaon and its applicaon in finding the shortest path in a graph, students can deepen their understanding of key concepts and improving problem-solving skills.

5.2 Algorithm Visualization

Professionals and enthusiasts in computer science and related fields can use the visualizer to better comprehend the inner workings of the Bellman-Ford algorithm. By visually observing

how the algorithm iterative updates distance estimates and idenfies the shortest paths, users gain insights into its computational complexity and efficiency.

5.3 Algorithm Comparison

The visualizer enables users to compare the Bellman-Ford algorithm with other shortest path algorithms, such as Dijkstra's algorithm. By visualizing and analyzing their respecve performances on different types of graphs, users can gain a deeper understanding of the strengths, weaknesses, and trade-offs of each algorithm.

5.4 Network Routing Simulation

In networking and telecommunicaons, the Bellman-Ford algorithm visualizer can simulate roung protocols and network optimization strategies.

5.5 Network Routing Simulation

Users can model network topologies and simulate the propagaon of routing information to dynamically compute shortest paths, aiding in network design, troubleshooting, and performance analysis.

5.6 Graph Visualizaon and Analysis

Beyond shortest path computation, the visualizer can be used as a general-purpose graph visualizaon tool. Users can create, manipulate, and analyze graphs with nodes and edges, facilitating the exploration of various graph algorithms, network structures, and data representations.

5.7 Educational Demonstrations

Educators and instructors can incorporate the visualizer into classroom lectures, presentaons, and demonstraons to illustrate graph algorithms and data structures. By engaging students in interacve exercises and demonstraons, educators can enhance learning outcomes and reinforce theorecal concepts with practical applications.

Overall, the Bellman-Ford algorithm visualizer serves as a versatile tool with diverse applications in education, research, and practical problem-solving across various domains, including computer science, engineering, mathematics, and network management. Its interacve and visual nature makes it an effective tool for both learning and professional use.

6 Conclusion

In conclusion, the Bellman-Ford algorithm visualizer offers a powerful platform for understanding, analyzing, and experiencing an one of the fundamental algorithms in graph theory. By providing an intuive and interactive environment, the visualizer empowers users to explore the intricacies of the Bellman-Ford algorithm and its applications in finding the shortest path in a graph. Through hands-on experimentaon and visualizaon, users can deepen their understanding of key concepts such as dynamic an programming, graph traversal, and shortest path algorithms.

Moreover, the visualizer serves as a valuable educational tool for students, educators, and professionals alike. Students can benefit from the visualizer as a supplementary learning resource to complement theoretical studies, enhancing their grasp of complex algorithms and data structures. Educators can leverage the visualizer to engage students in interacve lessons, fostering acve learning and reinforcing conceptual understanding through practical demonstrations. Furthermore, the visualizer's versality extends beyond educaonal purposes to praccal applicaons in network roung, algorithm analysis, and graph visualizaon. Professionals in fields such as computer science, networking, and telecommunicaons can use the visualizer to simulate routing protocols, analyze network topologies, and optimize network performance.

Additionally, researchers can utilize the visualizer to explore algorithmic optimizaons, compare algorithmic approaches, and conduct experiments to advance knowledge in graph theory and related areas.Overall, the Bellman-Ford algorithm visualizer represents not only a valuable educational resource but also a versale tool for research, analysis, and problem-solving in various domains. Its intuive interface, interacve features, and practical applications make it an indispensable asset for both learning and professional use, contributing to a deeper understanding of graph algorithms and their real-world implications. As technology continues to evolve, the visualizer stands as a testament to the enduring importance of visualization tools in facilitating learning, innovation, and discovery in computational sciences.

7 References

1. https://www.geeksforgeeks.org/bellman-ford-algorithm-dp-23/
2. https://www.youtube.com/watch?v=cpETrnfMkiY
3. https://www.sciencedirect.com/topics/computer-science/bellman-ford-algorithm