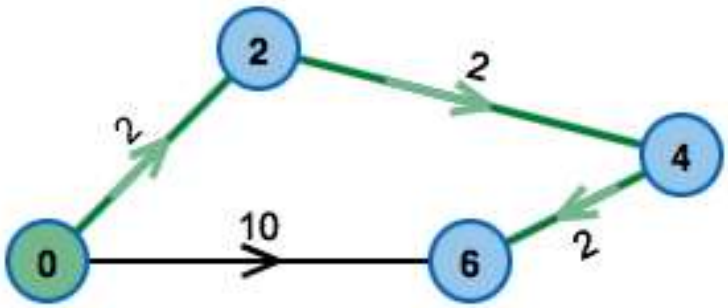
CSE-2204



Bellman-Ford Algorithm Visualization

User Manual

May 13,2023

North Western University
Khulna,Bangladesh

Table of Contents

CSE-2204

Bellman-Ford Algorithm Visualization User Manual

1.Introduction

This code is an implementation of the Bellman-Ford algorithm for finding the shortest path in a weighted graph. The code receives the graph data from the user through three input elements and displays the resulting graph and shortest path. The software is developed by using HTML,CSS and Javascript.

Overall, this code provides a functional implementation of the Bellman-Ford algorithm and a simple visualization of the resulting shortest path. However, it does not include any error handling or input validation, which could cause problems if the user enters invalid input. Additionally, the visualization could be improved by adding labels to the vertices and edges, and by allowing the user to interact with the graph.

What is Bellman-Ford Algorithm?
-Bellman-Ford algorithm is a shortest path algorithm used to find the shortest path between a source vertex and all other vertices in a weighted graph with negative edge weights. It was proposed by Richard Bellman and Lester Ford Jr. in 1958.

The algorithm works by iterating over all edges in the graph, relaxing them by comparing the distances of their endpoints, and updating the distance to the destination vertex if a shorter path is found. This process is repeated for all vertices in the graph, ensuring that the algorithm finds the shortest path from the source to all other vertices.

The Bellman-Ford algorithm can detect negative cycles in a graph, which are cycles that have a total weight less than zero. If a negative cycle is detected during the algorithm's execution, the algorithm will terminate and report that the graph contains a negative cycle.

The time complexity of the Bellman-Ford algorithm is O(|V| * |E|), where |V| is the number of vertices in the graph and |E| is the number of edges. This makes it less efficient than some other shortest path algorithms, such as Dijkstra's algorithm, which has a time complexity of O(|E| + |V|log|V|) for graphs with non-negative edge weights. However, the Bellman-Ford algorithm is the only algorithm that can handle negative edge weights.

2.Objectives

The Bellman-Ford visualization software has the objective of helping users understand the Bellman-Ford algorithm, which is a well-known algorithm used to find the shortest path in a graph from a single source vertex. Here are some points regarding its objectives:

1. Visualize the Bellman-Ford Algorithm: The software aims to provide a visual representation of how the Bellman-Ford algorithm works step by step.

2. Understand the shortest path concept: The software helps users understand the concept of the shortest path and how it is calculated using the Bellman-Ford algorithm.

3. Debugging: The software provides an interactive environment for debugging the algorithm to identify any errors in the implementation.

4. Efficiency of Algorithm: The software aims to demonstrate the efficiency of the Bellman-Ford algorithm by showing how it finds the shortest path in a graph in polynomial time.

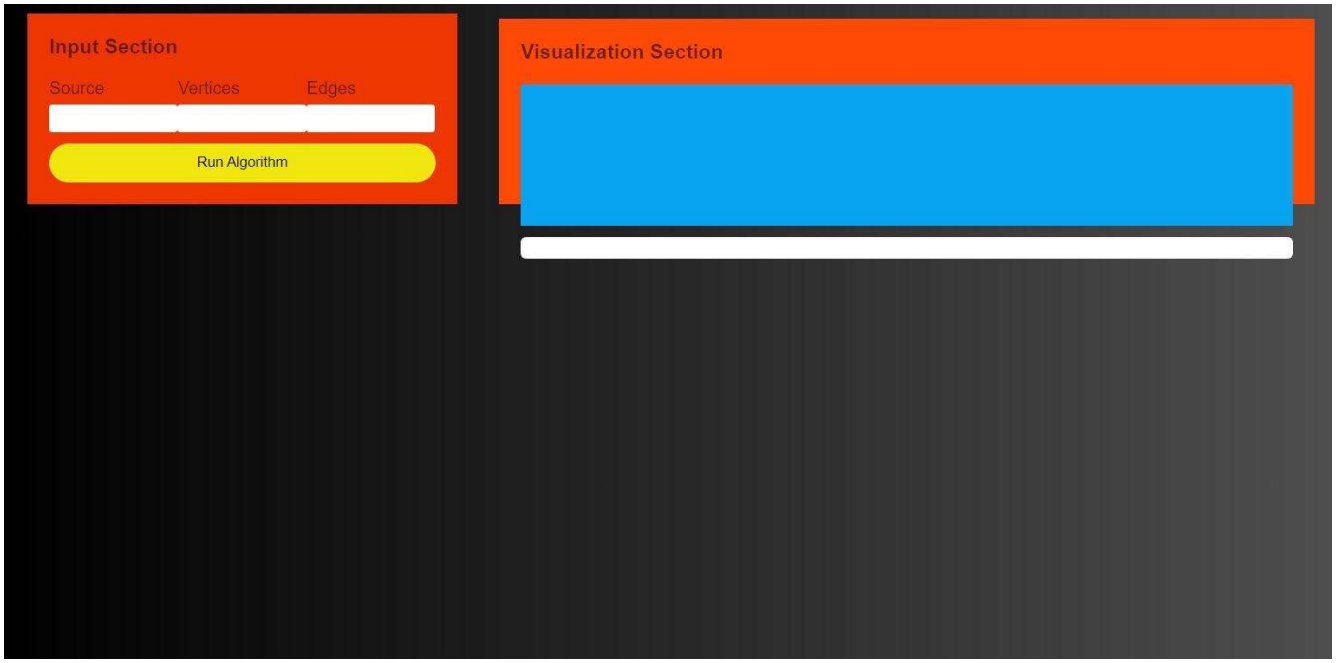
5. Customization of Graph: The software allows users to customize the graph, including the number of vertices, the weight of the edges, and the source vertex, to see how the algorithm works in different scenarios.

6. Education: The software aims to be an educational tool for students and anyone interested in learning about graph algorithms and their applications.

3.Description

Home Page

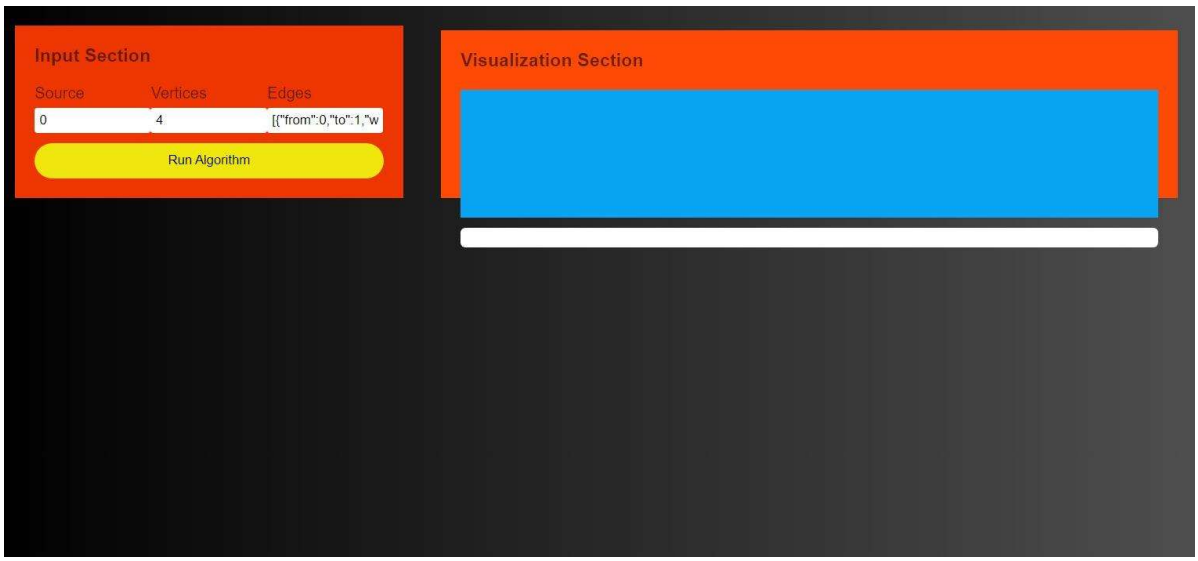
This the home page of **Bellman-Ford Algorithm Visualization Software**



The first part is to get the input elements and the visualization elements from the user. The input values are retrieved and the Bellman-Ford algorithm is executed when the user clicks the "run" button. The shortest path is then displayed, which generates a string containing the shortest path from the source vertex to each vertex in the graph.

Test Case Input:

When receiving the source vertex, the number of vertices, and the list of edges as inputs. The algorithm initializes an array of distances with a maximum value and sets the distance of the source vertex to zero. The function then iterates through all edges and updates the distance of each vertex if a shorter path is found. Finally, the function checks for negative cycles and throws an error if one is found.

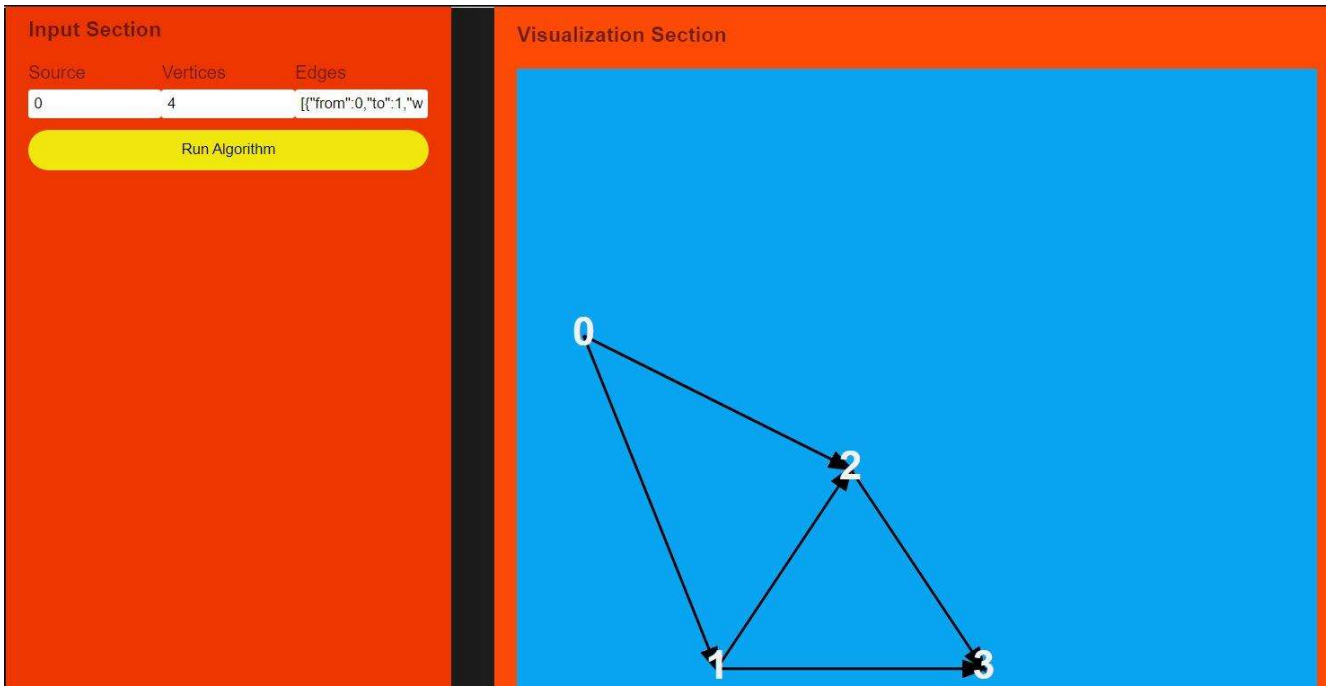


Example of Test Case:
source = 0
vertices = 4
edges =
[{"from":0,"to":1,"weight":5},{"from":0,"to":2,"weight":2},{"from":1,"to":2,"weight":1},{"from":2,"to":3,"weight":3},{"from":1,"to":3,"weight":6}]

N.B.: Make sure that the edges input is in the correct JSON format, which should look like this:
[{"from":0,"to":1,"weight":5},{"from":0,"to":2,"weight":2},{"from":1,"to":2,"weight":1},{"from":2,"to":3,"weight":3},{"from":1,"to":3,"weight":6}]
Each edge should be an object with from, to, and weight properties. Make sure that the keys and values are enclosed in double quotes, and that the entire array is enclosed in square brackets.

Algorithm Visualization

This is how **Bellman-Ford Algorithm Visualization Software** visualizes the algorithm.



After accurately entering the test case into the appropriate input fields and clicking the "Run Algorithm" button, the algorithm will begin to process and display to show how the Bellman-Ford method works and discover the shortest path step by step.Source vertices and edges will be displayed in the animation.

You must run the software in order to view the animation.

3.Dependencies

1.HTML: The HTML code is used to create the user interface for the Bellman-Ford algorithm visualization software. The code creates a web page with two sections: the input section and the visualization section.

In the input section, there are three input boxes for the user to enter the source vertex, the number of vertices, and the edges of the graph. The user can then click the "Run Algorithm" button to execute the Bellman-Ford algorithm.

In the visualization section, the graph visualization is displayed using SVG elements generated by the JavaScript code. The result of the algorithm is also displayed in a div element with the id "result".

The HTML code is linked to a CSS file for styling and a JavaScript file for implementing the Bellman-Ford algorithm and the visualization.

2.CSS: The CSS code in the provided snippet is used to style and animate the elements of a visualization software that displays the Bellman-Ford algorithm on a graph.

The code styles the background color, font family, and color of the text. It also defines the layout of the container that holds the input and visualization sections, as well as the styles for the input section and the graph visualization section.

The animation code is used to animate the edges, arrowheads, vertices, and the number of edges as the algorithm progresses. The animation also highlights the connections between the vertices and edges, and displays the weight of each edge during the animation.

Overall, the CSS code is used to enhance the visual representation of the Bellman-Ford algorithm on a graph, making it easier to understand and follow.

3.JavaScript: The JavaScript code that implements the Bellman-Ford algorithm for finding the shortest path in a graph. It takes input values from HTML input elements, runs the algorithm, and displays the graph and results in SVG format.