Graph Algorithm Visualizer

SARVESH ARORA[1], DR. K. C. TRIPATHI[2], DR. M. L. SHARMA[3]
[1, 2, 3] *Information Technology, MAIT, Delhi, India*

Abstract- Analysis and creation of Graph Algorithms present a significant difficulty for both computer science and information science students [1-2]. In order to help pupils and lessen their difficulties, this project was developed. Even though it has been claimed that some of these tools help people learn programming, the issue is still largely unsolved. The components of this project include Dijkstra, A* Search, Greedy Best-first Search, Bi-directional Breadth-first Search, Breadth-first Search, and Depth-first Search. To construct this web-app HTML5 (Hypertext Markup Language, Version 5), JavaScript, and CSS (Cascading Style Sheets) were used. This interface is intended to make the user feel completely engaged and focused. The objective of this project is to increase student motivation to learn Graph Algorithms by making them more enjoyable.

Indexed Terms- Visualization, Algorithms, e-learning, path finding, Depth First Search,Breadth First Search, A-star, Dijkstra’s, Greedy Best First, Bi-directional Bfs.

I. INTRODUCTION

Motivated by the saying, “Exciting graphics...pull your learners in, and meaningful interactions make the learning memorable and effective” many scholars and educators believe that by adopting algorithm visualization approaches, learners would learn an algorithm more quickly and thoroughly. As a result, we created a technique for teaching students and teachers about various pathfinding algorithms through visualization and hands-on experience. This tool can aid both inexperienced and seasoned programmers in better comprehending algorithms. This platform's major objective is to offer students a welcoming learning environment where they may compare and analyse different algorithms using graphs to learn more effectively. We may utilize visualization to better grasp these crucial conceptual processes by using the visual system of the human brain. It is a web application consisting of a (27 x 60) block grid where each block can be empty or wall i.e. that it allows the algorithm to pass through itself or not. Users then choose an algorithm and see it finding the path to the end node with different speeds. The web app consists of a single page with a navbar on top for different functionalities for visualization.

Video games, exploration, and industrial robots are the main applications of pathfinding algorithms. In video games, where the computer is tasked with guiding virtual adversaries around the map. The usage of pathfinding is by far the most complex. Exploring difficult-to-reach/dangerous regions is an emerging, but intriguing, application of pathfinding. These are locations that are difficult for humans to get, to but which robots could profitably explore if they could avoid getting lost or getting stuck on a rock.

This study aims to assess the visualizer's usefulness as a teaching and learning tool for graph algorithms. In order to achieve this, we run a number of tests in which we contrast the visualizer's performance with that of alternative approaches to teaching and learning graph algorithms. The tests' findings will shed light on the advantages and disadvantages of utilising visualisations to teach and learn graph algorithms, and they will also help shape the creation of new visualizers and other educational tools in the future.

II. PROBLEM STATEMENT

One fundamental idea in computer science that helps students become ready for more advanced ideas is the concept of algorithms. Students typically learn this idea by memorising the details of how a certain predetermined algorithm is executed, which only gives them a cursory comprehension of the fundamental process. The pupils are denied the chance to figure out problems on their own and have underdeveloped problem-solving abilities like computational thinking. Although students studying computer science must handle a lot of complex problems, it would be problematic if they lacked the necessary cognitive abilities because problem-solving abilities are an important part of computer science education. Visualization is a wonderful medium for promoting skills since it has several pedagogical advantages and encourages active learning and student-centered learning.

III. LITERATURE SURVEY

Algorithm visualizers are tools that use visualizations to help understand algorithms. They are increasingly being used in educational settings and other contexts to help students and others learn and retain algorithms more effectively. This literary survey provides a summary and review of the existing research and literature on algorithm visualizers.

Previous research has shown that visualizations can be effective for teaching and learning algorithms. For example, studies have found that visualizations can help students better understand complex algorithms, and can improve their performance on tasks such as writing and debugging code. However, there are also challenges and limitations to using visualizations for algorithm visualization. For instance, creating effective visualizations can require specialized skills and software, and it can be difficult to present complex algorithms in a way that is accessible to a broad audience.

There are various types of visualizations that have been used for algorithm visualization. These include static images, such as flowcharts and diagrams, which can provide a high-level overview of an algorithm. Interactive animations, on the other hand, can provide a step-by-step simulation of an algorithm, allowing the user to see how it works in real-time. Other approaches, such as code animations and data visualizations, can also be used to help understand algorithms.

The potential benefits of using visualizations for algorithm visualization are numerous. For example, visualizations can help students and others understand algorithms more effectively, and can improve their retention of algorithms over time. Visualizations can also help identify common mistakes and misunderstandings, and can provide a more engaging and interactive learning experience.

However, there are also challenges and limitations to using visualizations for algorithm visualization. For instance, creating effective visualizations can require specialized skills and software, which may not be readily available to all educators and learners. Additionally, presenting complex algorithms in a way that is accessible to a broad audience can be difficult, and may require significant design and development effort.

In conclusion, this literary survey has reviewed the existing research and literature on algorithm visualizers. It has highlighted the potential benefits of using visualizations to help understand algorithms, as well as the challenges and limitations of this approach. The findings of the survey suggest that algorithm visualizers can be a valuable tool for educators and learners, and can help improve understanding and retention of algorithms. However, further research is needed to better understand the best practices for using visualizations in this context, and to develop effective and accessible algorithm visualizers.

IV. METHODOLOGY USED

A pathfinding algorithm's primary goal is to identify the route between two places. In this project, multiple pathfinding techniques are visualised. The algorithms on this project have all been modified for a 2D grid.
path-finding algorithm consists of the following:

A. Algorithms-

1. Dijkstra's Algorithm: Dijkstra's algorithm enables us to identify the shortest route between any two network vertices. In order to identify the next best answer, the algorithm takes a greedy approach in the hopes that the final result will be the best solution for the entire problem [4].
2. A*Search: One of the best and most widely used methods for path-finding and graph traversals is the A* Search algorithm. It is a highly clever and effective algorithm [5].
3. Greedy Best-First Search: This algorithm always chooses the path that now seems to be the best

option. It combines methods for depth-first search and breadth-first search [6].

4. Depth First Search: The algorithm begins from the root node (choosing an arbitrary node as the root node in the case of a graph) and proceeds as far as it can along each branch before turning around [7].
5. Breadth First Search (BFS): BFS begins at the root of the tree and investigates all nodes at the current depth before going on to nodes at the next depth level [8].
6. Bidirectional Breadth First Search: This graph search algorithm finds the shortest route from the source to the goal vertex. It performs backward search from goal/target vertex toward source vertex and forward search from source/initial vertex toward goal vertex simultaneously [9].

B. Speed

The project consists of 3 speeds namely fast, medium and slow which maintains the speed of the visualization. Everyone learns at a different pace, so this function allows the user to adjust the speed of visualisation.

C. Grid

The Grid is a 2-d array of div elements which will consists of a Start Node (in red) and End Node (in green) which can be changed according to the user and the visualization takes place in this grid of 27 rows and 60 columns.

D. Walls

In order to add a wall, click the grid. W wall is impenetrable, a path cannot pass through it. The visualisation can be linked to real-world scenarios as wall will act as obstructions to the path founded by algorithms.

This project's development has been divided into 5 stages. These stages include from data gathering and processing to user output.

1. build the graph matrix.
2. Including event listeners and walls.
3. Include the algorithms for graphs.
4. Include the pathfinding feature.
5. Enhance the UI and design.

Technologies used: -

1. HTML: - HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and final major HTML version that is a World Wide Web Consortium recommendation
2. CSS: - Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML.
3. JavaScript: - JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions.

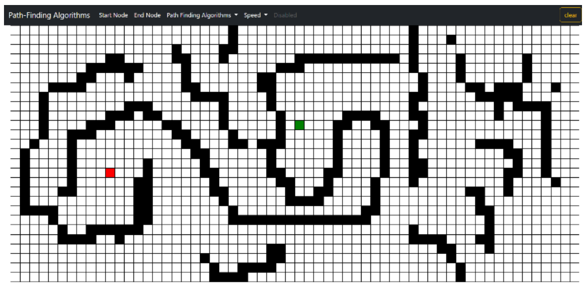

Fig1: - drawing walls on grid

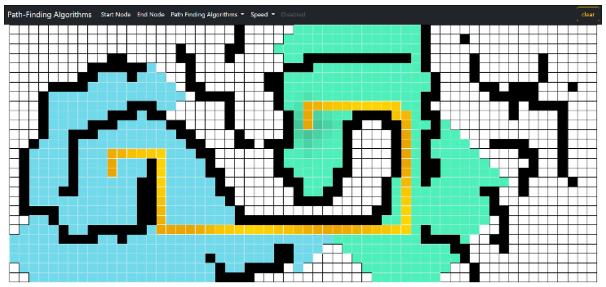

Fig2: - visualising bi directional bfs

V. RESULT AND ANALYSIS

Any type of visual aid, including images, videos, and even short clips of animation, tends to be more easily remembered by people, as has been consistently shown via several research and experiments on sizable populations.

Students can relate algorithm functionality to real-world examples in addition to the visualisation thanks to our application's maze and pattern features (likes obstructions in the form of walls) like in Fig 1.

Since using the programme only requires access to a website on the internet, our project can easily be integrated into our educational system while supporting different learning strategies from the conventional blackboard method.Moreover the visualisations are very eye-catching and fun to watch (Fig 2).

CONCLUSION

In this research, we present a pathfinding visualizer that was developed and implemented using web-technologies. Through the creation of this specific project, we made it simple to comprehend and learn about the algorithms. Humans acquire topics more easily when exposed to pictures rather than just text or spoken explanations. The application is very user friendly, allowing users of any age to become involved and begin learning new things immediately. The application would also have a variety of entertaining activities, such as visualisation through mazes and patterns. Because the web app is so user-friendly, people of any age may participate and immediately begin learning new things. The web application for path-finding algorithms will have almost all significant and well-known algorithms available, making it a one-stop shop for students studying this field. However, there is still a great desire to conduct research and produce animations like this to enhance student learning.

FUTURE WORK

We would like to add many more pathfinding methods and want to visualise them. Currently, the Web Application only contains a small number of graph pathfinding algorithms. We will be adding various maze generating algorithm such as Prims and recursive algorithm. We think of adding various algorithm forsorting as well helping students to understand more about the Data Structures and Algorithms.

REFERENCES

[1] Crescenzi, Pilu, Malizia, Alessio, Verri, M Cecilia, Díaz, Paloma, &Aedo, Ignacio. (2012). Integrating Algorithm Visualization Video into a First-Year Algorithm and Data Structure Course. Educational Technology & Society, 15(2), 115-124.

[2] Osman, Waleed Ibrahim, &Elmusharaf, Mudawi M. (2014). Effectiveness of Combining Algorithm and Program Animation: A Case Study with Data Structure Course. Issues in Informing Science and Information Technology, 11.

[3] Mayer, R. E. (2005). Cognitive theory of multimedia learning. In M. P. Simonson & M. A. McCombs (Eds.), The theory and practice of online learning (pp. 25-60). Edmonton, AB: Athabasca University Press.

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

[5] 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.

[6] Nilsson, N. J. (1971). Problem-solving methods in artificial intelligence. New York, NY: McGraw-Hill.

[7] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to algorithms (3rd ed.). Cambridge, MA: MIT Press.

[8] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to algorithms (3rd ed.). Cambridge, MA: MIT Press.

[9] Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation, 12(4), 566-580.