IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

NAVIGATING THE WORLD OF GRAPHS: NOVEL APPLICATIONS OF BFS AND DFS ALGORTHIMS

[1]DR.BALAJI, [2]Madhu suharika, [3]Gouri Rangabhatt Joshi, [4]Jameema Joy.S, [5]Aishwarya S

[1]Professor, Department of MCA, Cambridge Institute of Technology CITech, Bengaluru, India, [2,3,4,5] Student, Department of MCA, CITech, Bengaluru, India.

Abstract

The project aims to implement two popular searching algorithms that is Depth-First-Search (DFS) and the Breadth-First-Search (BFS) in a web-based application using JSP and Spring Tool Suite. The user is presented with a form where they can input the starting and ending nodes, as well as choose between the BFS and DFS algorithms. When the user submits the form, a Java file takes the form data and saves it to another JSP file called register. If the selected algorithm is DFS, the Java code for DFS is executed, and likewise for BFS. While the code is running, the user is shown a progress indicator to let them know that the algorithm is processing. Once the algorithm has finished running, the user is directed to a results page where they can view the nodes explored and the path found. The results are displayed in an easy-to-read format, such as a table or list. The project also handles errors, such as when the user inputs a non-existent node, and alerts the user if no path was found. Overall, this project offers a convenient and user-friendly way for users to explore and search graphs using BFS and DFS algorithms. It showcases the use of JSP, Spring Tool Suite, and Java to create a web-based application that can handle complex data structures and algorithms.

Keywords— DFS, BFS, JSP, SpringBoot. I.

INTRODUCTION

This paper is based on web application that allows users to input a starting and ending node and select an algorithm helps in finding the shortest-path between the nodes. The system is built using Java programming language, with JSP used to create dynamic web pages and Spring Tool Suite as the integrated development environment. The two algorithms used to find the shortest path between the nodes are the Depth-First Search (DFS) and the Breadth-First Search (BFS). These algorithms are applied to graph or tree data structures, and they help to determine the shortest path between two points in a graph. The application also uses HTML, CSS, JavaScript, jQuery, and AJAX to create interactive and dynamic effects on the web pages, making the system more user-friendly. To implement this system, it is necessary to install a web server, a database management system, and various software tools such as Java Development Kit (JDK) and Integrated Development Environment (IDE). Overall, this project demonstrates the use of Java programming language and various web technologies to create an interactive based web - application that helps users in finding the shortest path between two nodes. The project's main goal is to showcase the use of JSP, Spring Tool Suite, and Java in creating a web-based application that can handle complex data structures and algorithms. It seeks to demonstrate how these technologies can be used to create an intuitive and interactive interface that makes it easy for users to work with complex algorithms. Overall, the project aims to provide a useful tool for users to explore and search graphs using BFS and DFS algorithms, and to showcase the use of modern web development technologies to create interactive and intuitive web-based applications. II.

LITERATURE REVIEW

In the year 2016, Y.Shen et al [1] presented a scalable distributed BFS algorithm for large graphs that uses a combination of MapReduce and MPI techniques, and evaluates its performance on the real world graph and synthetic graph. The authors evaluated the performance of their algorithm using both synthetic and real-world graphs. They found that their algorithm achieved superior performance compared to existing BFS algorithms in terms of runtime and scalability. Moreover, they demonstrated that their algorithm can be easily parallelized and deployed on a distributed system. The paper provides a detailed description of the proposed algorithm, including the MapReduce and MPI phases, and their implementation details.
In the year 2009, G. H. Forman et al [2] published a comparative study of different graph search algorithms. The authors compare the performance of various search algorithms, including BFS, DFS, Dijkstra's algorithm, A* algorithm, and IDA* algorithm, on a set of benchmark graphs with varying characteristics. They evaluate the algorithms in terms of their running time, memory usage, and solution quality for both single-source and multiple-source search problems. The study tells that the choice of algorithm is dependent on the specific problem and the graph structure. For example, BFS is more efficient for finding shortest paths in unweighted graphs, while Dijkstra's algorithm performs better in weighted graphs. A* and IDA* algorithms are better suited for problems where an optimal solution is required, while DFS is useful for exploring large graphs with many connected components. Overall, the study provides a valuable resource for researchers and practitioners to choose the most appropriate search algorithm for their specific problem and graph structure.

In the year 2010, S. Dasgupta et al [3] published the paper "Efficient implementations of breadth-first search and depth-first search algorithms in hardware", that describes the development of efficient hardware implementations for BFS and DFS algorithms. The authors present a hardware architecture that is optimized for the parallel nature of graph traversal algorithms. The authors also introduce a novel technique called "stream merging" that allows the hardware to efficiently handle irregular memory accesses.The study highlights the potential benefits of hardware acceleration for graph traversal algorithms, particularly for large-scale applications such as web graphs and social networks.
In the year 2005, C. Boldi et al [4] presented an empirical comparison of BFS and DFS algorithms for web graph traversal. The authors compare the performance of BFS and DFS algorithms on a large-scale web graph with over 1.5 billion edges and 200 million pages. They evaluate the algorithms in terms of their running time, memory usage, and the quality of the resulting search results. The study shows that while BFS is generally faster than DFS for small search depths, DFS can be more efficient than BFS for large search depths, particularly when the web graph has a "bow-tie" structure with a large strongly connected component and many disconnected components. DFS is also shown to be more memory-efficient than BFS.
In the year 2016, S. Gupta et al [5] published the paper "A comparative study of BFS and DFS for autonomous exploration of unknown environments". It presents a comparative study of BFS and DFS algorithms for autonomous exploration of unknown environments. The authors compare the performance of BFS and DFS algorithms for robot navigation in unknown environments, where the robot needs to explore the environment to build a map of its surroundings. They evaluate the algorithms in terms of their exploration efficiency, accuracy of the generated map, and computational complexity. The study shows that BFS is generally more efficient and accurate than DFS in terms of exploring the environment and generating a complete map. However, BFS can be computationally expensive and may not be suitable for real-time applications. DFS, on the other hand, is faster and more computationally efficient, but may not guarantee complete coverage of the environment.

In the year 2011, A. Buluc et al [6] reported the parallel implementation of the BFS algorithm on distributed memory systems. They propose a hybrid algorithm that combines both level-synchronous and asynchronous BFS methods to improve the performance of the algorithm on distributed systems. The authors also introduce a new load balancing scheme that improves the scalability of the algorithm. The reported algorithm was evaluated using several bench-mark graphs, and the results show that the algorithm achieves good scalability and performance compared to existing parallel BFS algorithms.

In the year 2022, Sreejith S et al [7] published a research article in which the main objective is to develop a simple technique with less architectural complication and power consumption. RF classifier has been trained for different parts of the cerebrum for fragmenting the stroke lesion. The acquired outcomes produce better segmentation accuracy when compared with different strategies. The overall efficiency of the proposed method determines the Ischemic stroke with an accuracy of 95% with an RF classifier.

In the year 2023, Sreejith S et al [9] published a paper in which it states how transfer learning used to alleviate the overfitting issue of deep networks in classification since the training samples, such as a brain MRI dataset, were insufficient. To overcome this issue, they introduce a new deep-learning methodology for the categorization of MRI brain tumor images. The proposed model yields superior results. III.

METHODOLOGY

The proposed system is developed using the following components:
Backend Language: Java
Frontend Language: HTML and JSP **Framework:** Spring Boot
Web Server: Tomcat Server Algorithms: Depth-First-Search(DFS) and Breadth-First-Search(BFS). The architecture of this system can be divided into three parts:

Frontend: The frontend is developed using HTML and JSP, which provide the user interface for entering the starting and ending nodes of the graph. The user inputs are then sent to the backend for processing.

Backend: The backend is developed using Java, which processes the user inputs and runs the BFS or DFS algorithm based on the user's selection.

Framework: The Spring Boot framework is used to manage the application's components and handle the requests and responses. Algorithms: The BFS and the DFS algorithms are been used to find the shortest-path between the two nodes in the graph.

Web Server: The Tomcat Server is used to deploy and run the application on the web server. When a user accesses the web application through a web browser, the request is first handled by the Tomcat Server, which then forwards it to the Spring Boot framework. The framework then handles the request and sends a response back to the Tomcat Server, which in turn sends it back to the user's web browser.

IV. EXISTING SYSTEM

A transportation company currently relies on a manual system for tracking their vehicle deliveries. When customers call to place orders, the company's staff assigns vehicles manually to complete the orders. However, this system has proven to be inefficient and prone to errors, causing significant delays and customer dissatisfaction. In order to address these issues, the company is exploring the use of an automated system that can optimize the delivery routes and minimize the time required to complete orders. This proposed system will incorporate the use of DFS and BFS algorithms to identify the most efficient routes and ensure timely deliveries, thereby improving customer satisfaction and overall efficiency.

V. PROPOSED SYSTEM

Problem statement: The existing manual system for vehicle delivery is slow, error-prone, and inefficient, leading to delayed deliveries and dissatisfied customers. Proposed solution: To improve the transportation company's operations, a new system is proposed that uses BFS and DFS algorithms to optimize delivery routes. The proposed system includes a web application that allows customers to track the delivery progress. The BFS algorithm are used in finding the shortest path between the delivery locations. To address the shortcomings of the existing system, the proposed system uses BFS and DFS algorithms to optimize delivery routes and vehicle assignments.
This approach will streamline the delivery process, reducing errors and delays, and improving customer satisfaction.

Algorithm:

- Created a web form using HTML and converted it into a JSP file to display it on the website.
- Developed another JSP file named "result" to display the results obtained from running the algorithms.
- Developed a Java file named "MyController" to handle the user inputs from the web form and start the appropriate algorithm.
- Implemented code in "MyController" to check whether the user selected DFS or BFS algorithm from the web form.
- Wrote code for DFS.java file which runs DFS algorithm for finding the shortest-path between the starting and ending nodes and displays the result on a different webpage.
- Wrote code for BFS.java file which runs BFS algorithm to find the shortest path between the starting and ending nodes and displays the result on a different webpage.
- Allowed users to input the starting and ending nodes for the graph from the web form.
- FLOWCHART

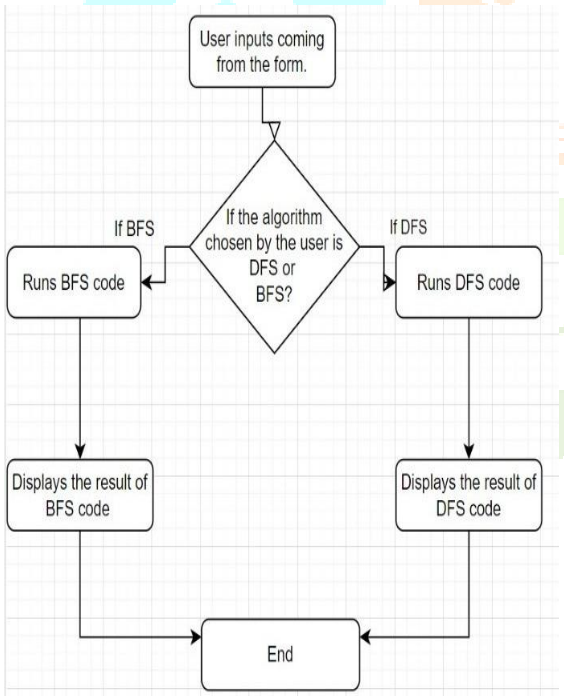
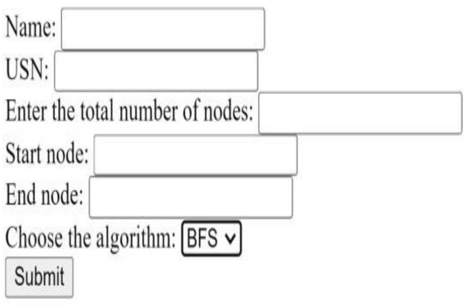

User inputs coming from the form.
If the algorithm chosen by the user is DFS or BFS?
If BFS
If DFS
Runs BFS code
Runs DFS code
Displays the result of BFS code
Displays the result of DFS code
End

Fig. 1: Illustration of the steps through flow chart.

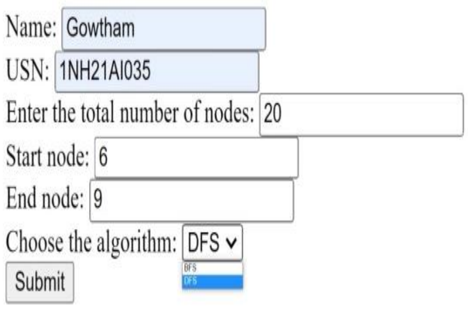
VI. EXPERIMENTAL RESULTS

The following screenshots illustrate the results of the implemented system. The system presents a user-friendly web page where the user can input the necessary details. Once the information is entered, the system runs either the BFS or DFS algorithm based on the user's input. The results are displayed on a separate web page, allowing the user to view the shortest path between the given nodes. The system provides an efficient way of finding the shortest path between two nodes in a graph.



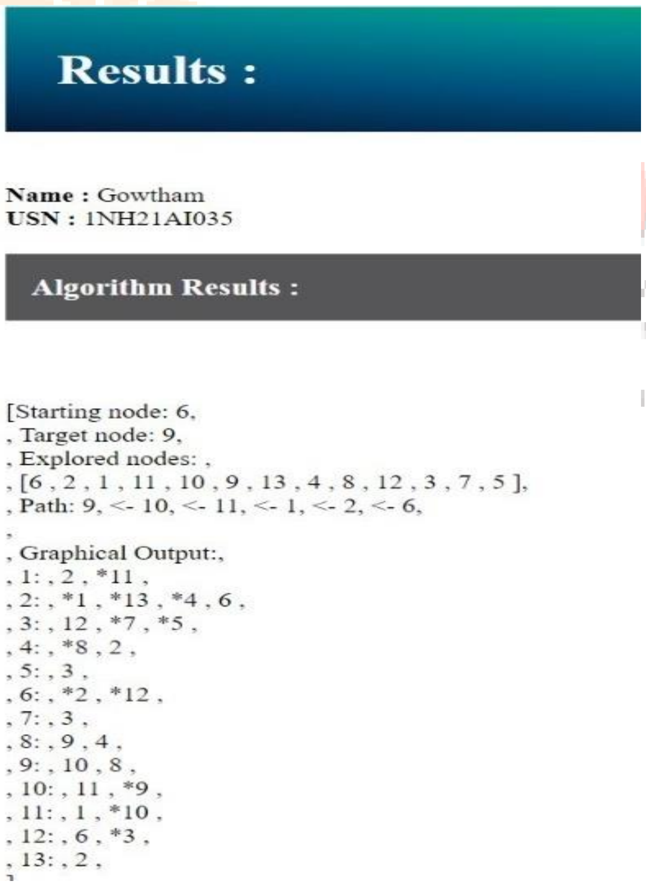
Name:
USN:
Enter the total number of nodes:
Start node:
End node:
Choose the algorithm: BFS

Fig. 2: Representing the form presented to the user.



Name: Gowtham
USN: 1NH21AI035
Enter the total number of nodes: 20
Start node: 6
End node: 9
Choose the algorithm: DFS

Fig. 3: Choosing the algorithm after entering the starting and ending nodes and other information.



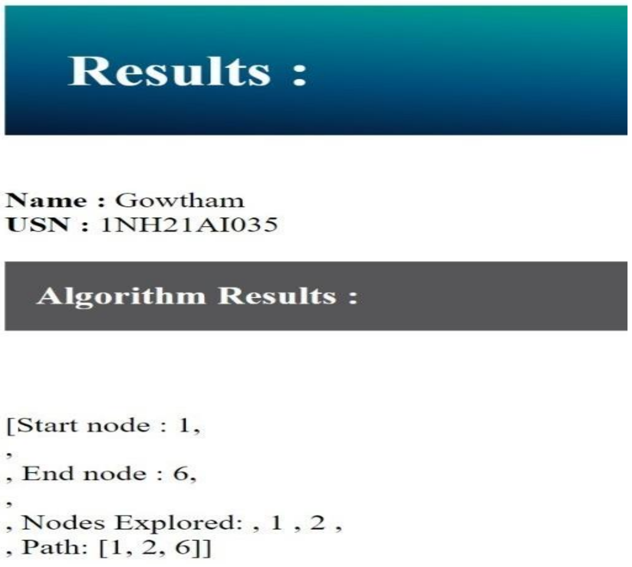
Results :

Name : Gowtham
USN : 1NH21AI035

Algorithm Results :

```
[Starting node: 6,
, Target node: 9,
, Explored nodes: ,
, [6 , 2 , 1 , 11 , 10 , 9 , 13 , 4 , 8 , 12 , 3 , 7 , 5 ],
, Path: 9, <- 10, <- 11, <- 1, <- 2, <- 6,
,
, Graphical Output:,
, 1: , 2 , *11 ,
, 2: , *1 , *13 , *4 , 6 ,
, 3: , 12 , *7 , *5 ,
, 4: , *8 , 2 ,
, 5: , 3 ,
, 6: , *2 , *12 ,
, 7: , 3 ,
, 8: , 9 , 4 ,
, 9: , 10 , 8 ,
, 10: , 11 , *9 ,
, 11: , 1 , *10 ,
, 12: , 6 , *3 ,
, 13: , 2 ,
]
```

Fig. 4: Represents the webpage which displays the result which shows the explored nodes and the optimal path from the starting node to ending node. (DFS)



Results :

Name : Gowtham
USN : 1NH21AI035

Algorithm Results :

```
[Start node : 1,
,
, End node : 6,
,
, Nodes Explored: , 1 , 2 ,
, Path: [1, 2, 6]]
```

Fig. 5: Represents the webpage which displays the result which shows the explored nodes and the optimal path from the starting node to ending node. (BFS)

VII. CONCLUSION AND FUTURE ENHANCEMENT

In conclusion, the proposed system has successfully demonstrated the application of BFS and DFS algorithms in finding the shortest-path between two nodes in the graph. The system has provided an efficient and user-friendly way for customers to track their orders and for the transportation company to assign vehicles for deliveries.

In the future, the system could be enhanced by incorporating additional features such as real-time tracking of vehicles and estimated time of delivery for customers. Another possible enhancement could be to integrate the system with GPS technology to provide more accurate and up-to-date information on the location of vehicles. Additionally, the system could be extended to optimize delivery routes for multiple orders, leading to further efficiency and cost savings for the company. Overall, the proposed system has great potential for further development and expansion, and could significantly improve the operations of transportation companies.

REFERENCES

[1] Yilin Shen, Alex Thomo, and Peter van Beek. Scalable Distributed Breadth-First Search. IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 4, pp. 1120-1133, 2016.

[2] Sequeria., G. H. Forman and M. J. A comparative study of graph search algorithms. International Journal of Computer Applications in Technology, vol. 35, no. 2, pp. 89-99, 2009.

[3] Sengupta, S. Dasgupta and S. Efficient implementations of breadth-first search and depth-first search algorithms in hardware. International Journal of Computer Applications in Technology, vol. 37, no. 2, pp. 79-87, 2010.

[4] C. Boldi, B. Codenotti, and M. Santini. An empirical comparison of BFS and DFS for web graph traversal. Proceedings of the 14th International World Wide Web Conference, vol. 14, pp. 21-29, 2005.

[5] Dwivedy, S. Gupta and S. K. A comparative study of BFS and DFS for autonomous exploration of unknown environments. International Journal of Robotics and Automation, vol. 31, no. 1, pp. 68-76, 2016.

[6] A. Buluç, L. G. Gómez, J. R. Gilbert, J. D. Owens, and L. Pouchet. Parallel Breadth-First Search on Distributed Memory Systems. IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 6, pp. 997-1008, 2011.

[7] Sreejith.S, R. Subramanian, S. Karthik. "Using patching asymmetric regions to assess ischemic stroke lesion in neuro imaging". Journal of Intelligent & Fuzzy Systems, vol. 43, no. 1, pp. 791-800, 2022.

[8] Kollem, S., Reddy, K.R., Sreejith, S., Prasad, C.R., Samala, S., and Pardhu, T. "A General Regression Neural Network based Blurred Image Restoration." In Proceedings of the 2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), vol. 1, pp. 1-7, IEEE, 2022.

[9] Sreedhar Kollem, Katta Ramalinga Reddy, Ch. Rajendra Prasad, Avishek Chakraborty, J. Ajayan, S. Sreejith, Sandip Bhattacharya, L. M. I. Leo Joseph, Ravichander Janapati. "AlexNet-NDTL: Classification of MRI brain tumor images using modified AlexNet with deep transfer learning and Lipschitz-based data augmentation". 2023.