PROJECT REPORT

Title: DESIGN AND DEVELOPMENTOF ROUTING ALGORITHM USING BREADTH FIRST SEARCH(BFS)

GROUP MEMBERS: Zara Bizenjo, Boomika Kataria and Muhammad Ahmer

COURSE NAME: Computer Network

DATE: 09.07.2025

Abstract:

This project is about design and development of routing algorithm, its implementation and visualization in Java. These routing algorithms are tree based, it is a trade off between distance vector algorithm and link state algorithm. Routing algorithms ensure that a router system can determine where to forward traffic to reach a particular destination. In this project we have used Breadth First Search (BFS) algorithm that enable us to find the shortest path from source to destination.

This application that we have created enables us to add nodes and edges and connect them. It also allows us to move the nodes so we can shape the graph however we want to. It has an option that add weights to the edges which are shown in black color over the yellow edges. Once we select our source node and destination node and click run BFS option a red path will appear which represents the shortest path. We have created this project using JavaFX and JavaFX canvas for graph and visualization.

Problem statement:

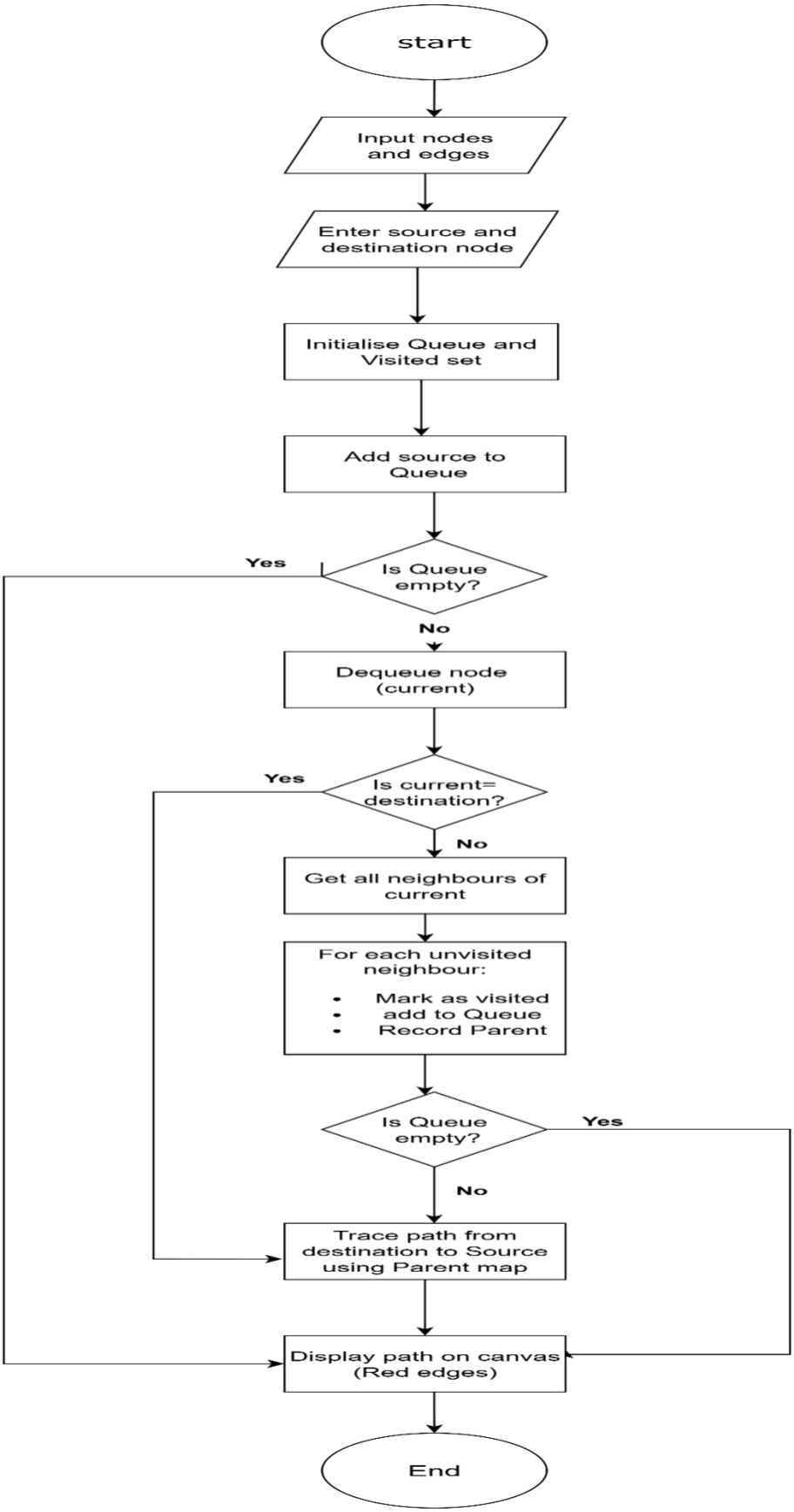
The function of a routing algorithm within computer networks is to find the best path for data packets to route from their source to destination over a network of connected devices (routers). The algorithm considers several factors including network topology, link weights, and traffic levels to make the routing decisions which optimize network congestion, latency, and resource usage while guaranteeing reliable and efficient transfer of data packets.

Algorithm chosen:

In this project we have selected the Breadth First Search (BFS) routing algorithm, it is a fundamental graph traversal algorithm. It starts from a source node and then visits all the adjacent nodes. It visits closest vertices before others, traverse vertices level by level. BFS uses queue data structure to store the node and mark them as visited. It makes a list of nodes that are visited and then dequeue the nodes once they are visited and we repeat the same steps until the queue is empty. BFS shows us the shortest path irrespective of the sum of edge weights.

BFS is a simple and reliable algorithm in comparison to other algorithms. BFS algorithm's results has a high level of accuracy. It is used world wide like in social networking websites, GPS navigation system, broadcasting networks and identifying routes and many more.

Flowchart:


start
Input nodes and edges
Enter source and destination node
Initialise Queue and Visited set
Add source to Queue
Is Queue empty?
Yes
No
Dequeue node (current)
Is current= destination?
Yes
No
Get all neighbours of current
For each unvisited neighbour:
Mark as visited
add to Queue
Record Parent
Is Queue empty?
Yes
No
Trace path from destination to Source using Parent map
Display path on canvas (Red edges)
End

Implementation details:

I have used Java language in this project. The Java coding of BFS is implemented in IntelliJ IDEA using JavaFX as our GUI tool kit, and for the graph Java canvas is used. I have implemented the BFS using queue and parent map.

The following features are used:

- The add Edge feature is implemented to allow the user to input two nodes and weight as well.
- Drag nodes allows user to position the nodes according to their will.
- Delete node option allows user to delete any unwanted node.
- BFS search lets the user find the shortest path according to fewest number of edges which will appear in red color.

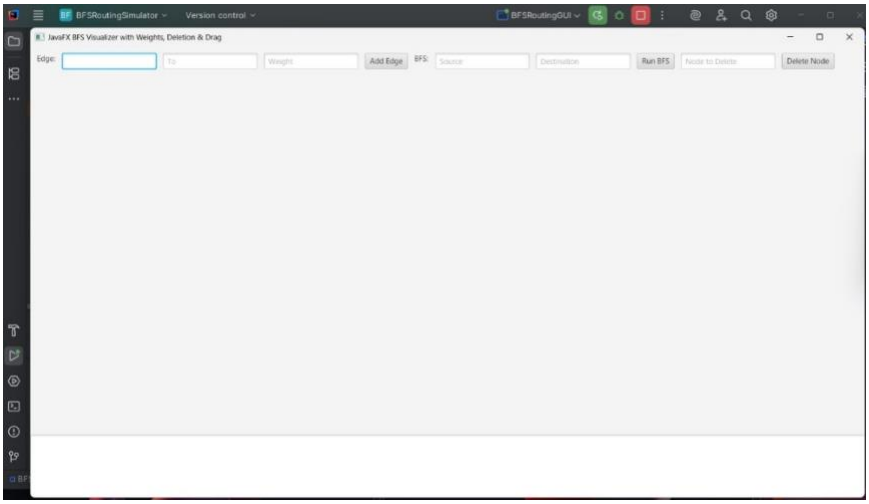
Visualization approach:

When we run the program, you can see a white screen which shows options like Add node, weight, source, destination and delete node, all of these in rectangular boxes.

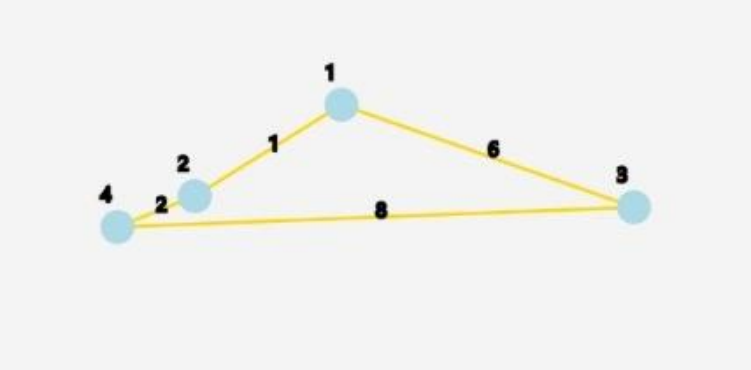
Once you enter your both nodes and click on add edge, Nodes will appear in circles and a yellow line connecting them as edge. You can add as many nodes as you wish to, after that you can select your source and destination node and once you click on Run BFS a red path will appear on the screen representing the shortest path. And at the bottom of the screen, it shows all the visiting nodes and the path found.

Results:

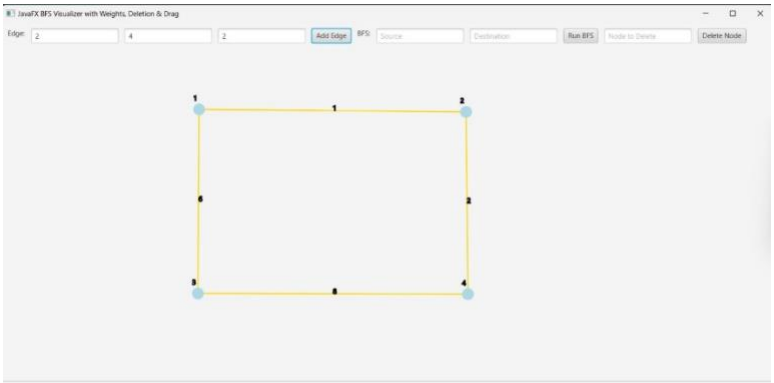
Once we run the program, the following screen will appear.



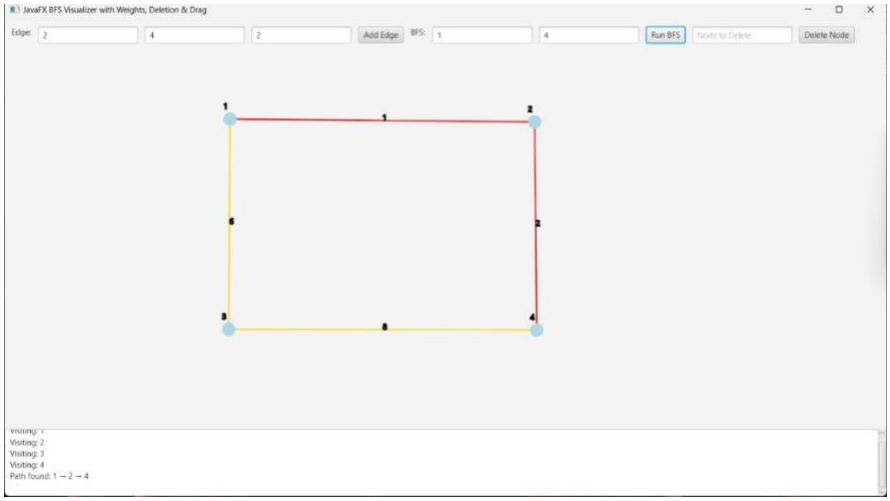
As we enter the number within the edge rectangular box and click on ‘add edge’ the nodes will appear on screen.



We can drag the nodes and reposition them.



Once we declare the source and destination node and click on ‘Run BFS’ the red path will appear.



At the bottom of screen we can see the visited nodes and path found.

Conclusion:

This project helped us build understanding about Breadth First Search algorithm, where it is used and its working. It introduced us to different applications that we were unaware of like IntelliJ IDEA. During this project we learned about Java coding and implementation of Java FX and JavaFX SDK and also Java Canvas in making of graph. Apart from these we learned to cooperate and work as a team.

In the process we faced many challenges especially in making the code run as it was showing error repeatedly. At first it was quiet a challenge as we did not know how to use IntelliJ IDEA but later things went smoothly.

References:

- https://www.baeldung.com/java-breadth-first-search
- https://docs.oracle.com/javase/8/javafx/api/overview-summary.html
- https://www.geeksforgeeks.org/dsa/breadth-first-search-or-bfs-for-a-graph/
- https://app.diagrams.net/