Routing Simulator using AODV in JavaFX

Computer Networks

Semester 2nd

Submitted to: **Mam Poma Panezai**

Group Members:

- Muzammil Mehdi – 67459
- Murtaza Hassan – 67712
- Ali Gohar – 68757
- Irfan Ali – 66529

Submission Date: **10-07-2025**

BALUCHISTAN UNIVERSITY OF IT, ENGINEERING & MANAGEMENT SCIENCES, QUETTA

Table of Contents

1. Introduction

In the modern digital era, networks form the backbone of seamless communication — from everyday browsing to critical real-time applications. With the rapid growth of mobile and ad hoc networks, efficient routing protocols have become essential to ensure that data reaches its destination accurately and quickly. One such protocol that has gained significant importance is the **Ad hoc On-Demand Distance Vector (AODV)** routing algorithm.

This project focuses on designing and building an interactive **AODV Routing Simulator** using Java and JavaFX. The goal was not only to simulate the AODV algorithm but also to provide an intuitive and visual way to understand how it functions under various conditions. Instead of just implementing the algorithm behind the scenes, the simulator enables users to **dynamically create network topologies**, assign weights to edges, simulate pathfinding, and visualize packet flow in real time.

Whether it's for educational purposes, debugging routing logic, or exploring how on-demand route discovery works, this simulator serves as a bridge between theoretical understanding and practical experience. Throughout this project, we explored both the technical and conceptual aspects of AODV routing — all wrapped into a clean, interactive desktop application.

2. Objectives

The primary goal of this project is to design and develop an **interactive graphical simulator** that models the **AODV (Ad hoc On-Demand Distance Vector)** routing protocol in a visual and engaging way. The simulator helps users understand how AODV works within dynamically created network topologies, and allows them to interact with the simulation in real-time. It's crafted to serve both educational and exploratory purposes for learners, researchers, and enthusiasts in the field of computer networks.

Why Routing Algorithms Matter

In any computer network, **routing algorithms are the invisible decision-makers** that determine how data travels from one point to another. Whether it's sending a message across a LAN or delivering video streams across continents, routing algorithms ensure that data packets take the most efficient, reliable, and sometimes even the safest path. In wireless or ad hoc networks, like the ones simulated in this project, routing becomes even more critical due to the lack of fixed infrastructure and the constant change in topology. Understanding these algorithms — especially ones like AODV that are designed for dynamic networks — is key to building resilient, intelligent communication systems.

Key Objectives

- **Design a User-Friendly Graphical Interface**
  Create an intuitive GUI that enables users to add or remove nodes and connections visually, making the topology creation feel natural and smooth.
- **Simulate AODV Routing Protocol**
  Implement the AODV algorithm to demonstrate how it discovers routes on-demand, using either hop count or weighted paths as the routing metric.
- **Enable Real-Time Interaction and Feedback**
  Allow users to modify the graph (e.g., by adding weights or deleting nodes) and immediately observe how these changes affect routing decisions.
- **Visualize Routing Behavior and Metrics**
  Provide visual feedback for routing paths, packet movement, and simulation logs so users can clearly understand each step of the algorithm.
- **Design with Modularity and Scalability in Mind**
  Write clean, extensible code that can be expanded to include other algorithms or features like congestion, routing tables, or mobility.

3. Tools & Technologies Used

- **Programming Language**: Java 21
- **GUI Framework**: JavaFX, FXML used alongside with JavaFX
- **Algorithm**: AODV (Ad hoc On-Demand Distance Vector)

- **IDE**: Visual Studio Code
- **Build & Run**: Gradle
- **Version Control**: Git & GitHub

4. System Requirements/Prerequisites

- Operating System: Windows/Linux/Mac
- Java Development Kit **Version 21**
- JavaFX latest version
- Gradle
- Visual Studio Code, or any other
- Git to clone repository

5. System Architecture

The AODV Routing Simulator is structured using a modular **Model-View-Controller (MVC)** architecture. This separation ensures clean code organization, easier debugging, and scalability as the project grows.

1. Model Layer

The **Model** is the heart of the application—it manages the core logic and data structures that simulate a computer network.

- **Node.java**: Represents a network node with properties like ID and coordinates.
- **Edge.java**: Represents a link (or connection) between two nodes with an optional weight.
- **Graph.java**: Manages nodes, edges, and implements the routing algorithms (AODV – both hop count and weighted paths).
- **LogEntry.java**: Represents each entry in the simulation log for display and tracking.

This layer is completely independent of how data is displayed—it only focuses on how data is structured and computed.

2. View Layer

The **View** defines the user interface—what the user sees and interacts with.

- **MainView.fxml**: The FXML layout file that defines buttons, tables, radio buttons, and the pane for displaying the network.
- Styled with JavaFX to provide a modern and interactive GUI.

👁 *The View does not contain any logic—it just knows how to look.*

3. Controller Layer

The **Controller** acts as the bridge between the Model and the View. It handles user input, manages events, and updates both the UI and the data.

- **MainController.java**:
  - Responds to button clicks like “Add Node”, “Remove Node”, or “Run AODV”.
  - Adds and removes visuals for nodes and edges on the canvas.
  - Calls the right algorithm (hop-based or weighted path) based on user selection.
  - Animates packet movement and updates logs accordingly.

This layer ensures the UI remains responsive while managing complex logic underneath.

Build System: Gradle

The entire project is built and run using **Gradle**, which:

- Handles JavaFX dependencies.
- Compiles source files.
- Runs the application.
- Allows easy scaling and dependency management in the future.

7. Working Flow

The AODV Routing Simulator follows an intuitive yet structured working flow, allowing users to visually understand how routing algorithms operate within a dynamic network. Here's a step-by-step breakdown of how the application works from start to finish:

1. Launch the Application

Once the application is built and executed using Gradle using the command " ./gradlew build" and then this command "./gradlew clean run", a user-friendly JavaFX interface appears with a blank canvas and control panel on the side.

2. Add Nodes

- The user clicks the **"Add Node"** button.
- Then clicks anywhere on the canvas to place the node.
- Each node is automatically labeled (e.g., *N1*, *N2*, *N3*) and positioned using its X and Y coordinates.

This simulates creating devices in a network, like routers or systems.

3. Create Connections (Edges)

- To form a connection between two nodes, the user **clicks on two nodes one after another**.
- A dialog box prompts for **edge weight input** (used in weighted routing).
- A line is drawn between them, labeled with the given weight.

These connections act as the links between devices in a real network and may carry different costs/weights.

4. Select Routing Type

- The user selects one of the two metrics via radio buttons:
    - **Hop Count (AODV)** – finds the path with the least number of hops.
    - **Weighted Path** – finds the path with the lowest total weight.

This step decides how the routing algorithm behaves.

5. Choose Route (Source & Destination)

- The user clicks **“Select Route”**.
- Then selects two nodes: the **source** and the **destination**.
- The simulator highlights the selected nodes and logs the selection.

This sets up the pathfinding request, similar to sending a packet in a real network.

6. Run AODV Simulation

- The user clicks **“Run AODV”**.
- Based on the selected metric, the corresponding algorithm runs:
    - Finds the shortest or most efficient route.
    - Logs each step of the algorithm execution.

The path is animated using a red packet dot traveling across the selected route. This shows how data flows from source to destination in real-time.

7. Clear or Edit Network

- **Clear Path** removes the animated route and allows a new simulation.
- **Remove Node** enables selecting a node for deletion—also removes its related edges and weights.

This allows the user to dynamically update the network topology.

8. Track Logs

Every user action (e.g., adding nodes, running AODV, removing nodes) is recorded in a structured **Simulation Log Table**, including:

- Timestamp
- Event Type (Add, Remove, Simulate)
- Details (e.g., Node ID, Route info)

This log helps trace back steps and observe routing behavior.

11. Conclusion

The AODV Routing Simulator project successfully demonstrates the core concepts of network routing in an interactive and visual manner. By simulating the **Ad-hoc On-Demand Distance Vector (AODV)** protocol alongside weighted routing, this tool provides users with a deeper understanding of how data is transferred through dynamic and complex networks.

Through features like node creation, edge connection with customizable weights, route selection, and animated path traversal, users can visualize real-time routing behaviors—something that is often abstract and difficult to grasp from textbooks alone.

The project not only helped in learning how AODV functions but also emphasized the importance of routing algorithms in ensuring optimal network performance. It showcases how efficient routing can minimize congestion, reduce delays, and improve overall communication in distributed systems.

Additionally, developing this simulator reinforced skills in **Java, JavaFX, object-oriented programming, graph theory, and algorithm design**, along with build automation using **Gradle**.

In conclusion, this simulator bridges the gap between theory and practice, making it a valuable educational and demonstrative tool for students, educators, and networking enthusiasts alike.

12. Links

GitHub Repository: https://github.com/Meta-Captain819/Routing-Algorithm

YouTube link: https://youtu.be/h8WKhzOcGH4?si=4NpsF4TKXltsFshC