Group Members—

- Manpreet Singh (300206509)
- Sahibjeet Singh (300206168)
- Shreyas Dutt (300206165)

IEEE Project Report

MP10 – Routing Table Visualizer

Static Routing Decision Visualization Using Python & Web Technologies

Abstract—

Routing is a fundamental operation in computer networks, enabling packets to move efficiently from source to destination. While dynamic routing protocols compute paths reactively, static routing remains a core concept for understanding deterministic forwarding behavior. This project implements a *Routing Table Visualizer*—a Python-based and web-enabled tool that demonstrates path selection using static routing, Dijkstra, Bellman-Ford, and hop-count (BFS). The tool provides real-time visualization of routing decisions, topology traversal, algorithm steps, and next-hop behaviors. Additionally, three recent research papers were analyzed to understand routing visualization trends, performance considerations, and educational uses in network simulation. The resulting system offers an interactive learning aid for understanding routing fundamentals through visual exploration.

Keywords—

Static Routing, Routing Table Visualization, Dijkstra, Bellman-Ford, BFS, Flask, Networking, Path Selection, Visualization.

I. INTRODUCTION

Routing determines how data travels from one node to another within a network. Static routing, while simple, offers deep insight into deterministic next-hop decisions. Understanding how routers compute shortest paths is essential for building a strong foundation in networking.

This mini-project implements a **Routing Table Visualizer**—a browser-based tool developed using Python (Flask) and web technologies. It allows users to select source/destination nodes and visualize the selected routing algorithm's behavior, including:

- Static routing (predefined next hops)
- Dijkstra's shortest-path algorithm

- Bellman-Ford algorithm
- Hop-count routing (BFS)
- Link failures
- Path animation and step-by-step explanation

This project supports hands-on learning of routing concepts, algorithm behavior, and network decision processes.

The report also critically examines **three recent research papers** related to routing visualization and network simulation.

II. PROJECT OVERVIEW

This project aligns with course objectives aimed at improving understanding of data communication through:

- **Hands-on implementation**
- **Algorithm behavior visualization**
- **Understanding static vs dynamic routing**
- **Exploring research literature**
- **Teamwork and documentation**

The final output includes the working tool, code, visualization features, and a research-backed explanation of routing models.

III. BACKGROUND

A. Static Routing

Static routing manually configures next-hop directions. It is deterministic, predictable, and used in small, stable networks.

B. Dynamic Routing

Algorithms like **Dijkstra** and **Bellman-Ford** compute shortest paths automatically. Dynamic routing is essential for scalable networks.

C. Algorithms Used

1. Static Routing Table

Uses a predefined routing table mapping destination → next hop. Implemented from the uploaded Flask code.

2. Dijkstra's Algorithm

- Greedy shortest-path algorithm
- Uses priority queue
- Tracks distance, previous hops, nodes explored

3. Bellman-Ford Algorithm

- Iterative relaxation technique
- Works with negative weights
- Detects negative cycles

4. Hop Count (BFS)

Selects the path with the fewest hops, ignoring link cost.

IV. LITERATURE REVIEW (3 Papers)

Three research papers related to routing visualization, network simulation, and path representation were studied.

Paper 1:

Interactive OSPF Visualization (2023)

Valentin Jenny, ETH Zurich

How it improves previous work

- Builds on earlier routing simulators by adding realistic OSPF message exchange.
- Provides accurate modeling of LSAs and flooding behavior.

Technical Approach

- Simulates adjacency formation, LSDB synchronization, flooding, retransmission lists.
- Abstracts unnecessary details for performance.

Metrics Used

- Convergence time
- Simulation runtime
- Scaling behavior

Strengths

- High accuracy of OSPF internal operations
- Excellent teaching tool
- Real simulation of message flows

Weaknesses

- No visualization fully implemented
- No support for OSPF areas
- Complexity grows with topology size

Relevance to Our Project

- Shows how route computation and database exchange happen inside routers.
- Our static routing tool is simpler but uses the idea of showing *why a router chooses a next hop*.

Paper 2:

Visualization of Real-time Solutions of Routing Problems on Dynamic Networks (IEOM 2024)

Osman & Saha

How it improves previous work

- Visualizes real evacuation routing using GIS.
- Provides real-time updates based on changing conditions.

Technical Approach

- Uses ArcGIS, VBA, shape files
- Converts GIS networks into node-arc matrices
- Supports 2D and partial 3D visualization

Metrics Used

- Evacuation time
- Flow density
- Route length

Strengths

- Real-world application (emergency routing)

- Handles dynamically changing environments

Weaknesses

- Limited to 2D
- Requires heavy GIS datasets
- Not suitable for beginner-level routing education

Relevance to Our Project

- Inspires the idea of showing node transitions and flow visualization.
- Our tool uses a simpler canvas-based drawing but fulfills the same educational purpose.

Paper 3:

Multi-view Routing Visualization for Identifying BGP Issues (Elsevier 2020)

Candela et al.

How it improves previous work

- Introduces multi-view layered visualization (global, local, animated).
- Helps network operators detect BGP anomalies quickly.

Technical Approach

- Uses stacked-area charts
- Graph animations
- Local/Global timeline views

Metrics Used

- Reachability
- Visibility
- Routing stability

Strengths

- Very intuitive dashboards
- Detects hijacks, outages, leaks

Weaknesses

- Primarily focused on BGP-level macro routing

- High data volume leads to performance constraints

Relevance to Our Project

- Inspired the **multi-view architecture** idea.
- Our UI also uses:
    - Node-by-node animation
    - Highlighted active paths
    - Step-by-step explanations

V. CRITICAL EVALUATION OF PAPERS

Criteria	Paper 1 (OSPF Viz)	Paper 2 (GIS Routing)	Paper 3 (BGP Multi-View)
Improvement on past work	Accurate OSPF modeling	Real-world dynamic routing	Multi-layer visualization
Technical approach	Simulated LSAs, flooding	GIS + VBA + path optimization	Stacked charts + animations
Metrics	Convergence, runtime	Flow, evacuation time	Visibility, reachability
Strengths	High protocol accuracy	Real emergency use	Easy anomaly detection
Weaknesses	No GUI	Heavy GIS dependency	Complex data
Relevance to project	Internal routing logic	Idea of path flow	Visualization inspiration

VI. SYSTEM DESIGN

A. Network Topology

Nodes: **A, B, C, D, E**
Bidirectional links:

Link Cost

A–B 1

A–C 4

B–C 2

B–D 5

Link Cost

C–D 1

C–E 3

D–E 2

(This is extracted from your Flask topology dictionary.)

B. Architecture

Frontend

- HTML + Tailwind CSS
- Canvas API for drawing nodes/links
- JavaScript fetch API for sending routing requests

Backend

- Python Flask server
- /compute endpoint handles Dijkstra/Bellman-Ford/static/BFS
- JSON responses

Visualization Features

- Active path highlighted in blue
- Animated packet icon
- Step-by-step description of algorithm logic
- Dynamic result panel

VII. IMPLEMENTATION DETAILS

A. Static Routing Logic

Pulled from your uploaded STATIC_ROUTING_TABLE.

Each router maps a destination to its next hop.

Example: Router A

A → B (to reach B)

A → B (to reach C)

A → C (to reach D or E)

B. Dijkstra Implementation

- Uses priority queue (heapq)
- Computes shortest path dynamically
- Logs:
    - node visits
    - distance updates
    - failed-link handling

C. Bellman-Ford Implementation

- Iterative relaxation
- Tracks iterations
- Detects negative cycles
- Slower but robust

D. Hop Count (BFS)

- Queue-based exploration
- Optimal for equal-cost networks
- Returns minimal-hop path

E. Web Visualization

Implemented in **index.html**
Includes:

- Scalable node placement
- Highlighted edges
- Canvas re-rendering on resize
- Packet animation
- Result summaries

VIII. RESULTS

A. Correct Path Selection

The system correctly identifies paths for all algorithms:

Example: A → E

- Static routing: A → C → E
- Dijkstra: A → B → C → E
- Bellman-Ford: Same as Dijkstra
- Hop-count: A → C → E

B. Link Failure Handling

- Static routing fails (cannot adapt)
- Dijkstra/Bellman-Ford reroute based on remaining topology

C. Algorithm Comparison

The /compare endpoint provides timing and explored node counts.

D. Visualization Output

- Paths drawn in blue
- Packet travels animation
- Router nodes colored based on role
- Costs displayed at midpoints

IX. DISCUSSION

The project successfully demonstrates how path selection differs across routing algorithms. Static routing is simple but not fault tolerant. Dijkstra and Bellman-Ford handle dynamic changes effectively. Visualization significantly enhances understanding.

X. LIMITATIONS

- Topology is small (5 nodes)
- No real-time dynamic topology updates
- Static routing table must be manually defined
- No distributed routing emulation (BGP/OSPF)

XI. FUTURE WORK

- Add support for dynamic link state updates
- Implement distance-vector protocol simulation
- Integrate timeline-based animations

- Larger network import from JSON
- Multi-view interface inspired by BGP visualization frameworks

XII. CONCLUSION

This mini-project successfully implements a **Routing Table Visualizer** that demonstrates static routing alongside Dijkstra, Bellman-Ford, and hop-count algorithms. The visual approach makes routing concepts easier to understand, matching the objective of improving learning through hands-on simulation. The research papers analyzed provide context on routing visualization trends, from protocol-level modeling to large-scale BGP anomaly detection. Overall, the work deepened conceptual understanding and delivered a functional educational tool.

REFERENCES

[1]

M. Candela, G. Di Battista, and L. Marzialetti, "Multi-view routing visualization for the identification of BGP issues," *Journal of Computer Languages*, vol. 58, p. 100966, Apr. 2020, doi:

https://doi.org/10.1016/j.cola.2020.100966.

[2]

V. Jenny, T. Schneider, and L. Vanbever, "Interactive OSPF Visualization," 2023. Accessed: Nov. 18, 2025. [Online]. Available: https://nsg.ee.ethz.ch/files/public/theses/2023-ospf-simulation-for-teaching/thesis-1.pdf

[3]

M. Saeed Osman and R. Saha, "Visualization of Real-time Solutions of Routing Problems on Dynamic Networks," *Proceedings of the International Conference on Industrial Engineering and Operations Management*, Jun. 2024, doi: https://doi.org/10.46254/na09.20240220.