Assignment 4

Routing Protocols

Course: CS425: Computer Networks

Instructor: Adithya Vadapalli

TAs Incharge: Rishit and Yugul

Submission Deadline: 21.04.2025, EOD

Objective

The objective of this assignment is to implement the *Distance Vector Routing (DVR)* and *Link State Routing (LSR)* algorithms in `C++` using an adjacency matrix as input. The goal is to simulate the operation of both algorithms in a network with a given set of nodes and links. The assignment will (hopefully!) help you understand the workings of two fundamental routing algorithms and how they compute the optimal path between nodes. You can download the assignment from GitHub repository: `https://github.com/privacy-iitk/cs425-2025.git`

Background

Distance Vector Routing (DVR) and Link State Routing (LSR) are two common routing algorithms used in computer networks for finding the shortest paths between nodes in a graph. Both algorithms are based on a graph where the nodes represent routers or computers, and the edges represent links with a given cost or metric.

Distance Vector Routing (DVR)

In the DVR algorithm, each router maintains a routing table that contains the best-known cost to reach each destination node. The table is periodically updated based on the information received from its neighbors. The primary challenge in DVR is to prevent routing loops, which can occur when routers continuously update their tables with incorrect information.

Link State Routing (LSR)

LSR, on the other hand, works by having each router discover the entire network topology. Each router broadcasts its link-state information (its neighbors and their respective costs) to all other routers in the network. Using this information, each router runs Dijkstra's algorithm to compute the shortest path to every other router.

Assignment Description

In this assignment, you will be simulating both the *Distance Vector Routing (DVR)* and *Link State Routing (LSR)* algorithms for a given network. The input will be a graph represented as an adjacency matrix, and the routing metric will be provided as either *bandwidth* or *latency* (or any other user-defined metric).

Your task is to implement the simulation of both algorithms and produce the corresponding routing tables for each node.

Input Format

The program will read an adjacency matrix from a file. The adjacency matrix represents the links between nodes and the associated metric (e.g., bandwidth, latency).

- The first line of the file contains an integer n, the number of nodes in the network.

- The next n lines contain n space-separated integers, representing the adjacency matrix where the i^{th} row and j^{th} column entry corresponds to the metric (e.g., cost, bandwidth, latency) of the link between node i and node j.
- A value of 0 indicates no cost or no link between the nodes.
- A value of 9999 will be used to represent an unreachable link (or infinite cost).

Example Input File ('inputfile.txt'):

```
4
0 10 100 30
10 0 20 40
100 20 0 10
30 40 10 0
```

In this example, there are 4 nodes, and the adjacency matrix specifies the metric (cost, bandwidth, etc.) for each link.

Output Format

The output of the program will be the routing tables for each node after running the *Distance Vector Routing (DVR)* and *Link State Routing (LSR)* algorithms.

- For DVR: Print the routing table for each node at each iteration, showing the destination node, the computed metric, and the next hop node.
- For LSR: After computing the shortest paths using Dijkstra's algorithm, print the routing table for each node, showing the destination node, the computed metric, and the next hop node.

Example Output:

```
--- Distance Vector Routing Simulation ---
Node 0 Routing Table:
Dest    Metric    Next Hop
0         0         -
1         10        1
2         100       2
3         30        3

Node 1 Routing Table:
Dest    Metric    Next Hop
0         10        0
1         0         -
2         20        2
3         40        3

--- Link State Routing Simulation ---
Node 0 Routing Table:
```

```
Dest    Metric    Next Hop
1         10         1
2         30         3
3         30         3

Node 1 Routing Table:
Dest    Metric    Next Hop
0         10         0
2         20         2
3         40         3
```

Algorithm Requirements

1. Distance Vector Routing (DVR) Algorithm

In the DVR algorithm:

- Each node starts with its own routing table, which initially contains the direct link cost to each other node.
- Nodes exchange their routing table with their immediate neighbors and update their tables accordingly.
- This process repeats until there are no further updates.

2. Link State Routing (LSR) Algorithm

In the LSR algorithm:

- Each node knows the entire network topology.
- Each node uses *Dijkstra's algorithm* to compute the shortest path to each other node.
- The result is a routing table that lists the shortest path to every other node in the network.

Deliverables

Students must submit the following:

- Source code implementing the simulation.
- A README file with instructions on how to run the code.

Submission Instructions

- Submit a zip file containing the source code README
- The filename should be `A4Rollnumberofmember1Rollnumberofmember2Rollnumberofmember3`
- Upload your submission to HelloIITK before the deadline. Only one team member should submit the assignment.

Grading Rubric

- **Correctness (80%)**: The server works as expected and supports all required features.
- **Code Quality (10%)**: Comments in the code.
- **Documentation (10%)**: Clear instructions and explanation in the README.

Note: For any clarifications, post a message on Piazza. Ensure all submissions are made before the deadline. Late submissions will incur a penalty as per the course policy.

Instructor: **Adithya Vadapalli** TA Incharge: **Rishit and Yugul**