# Experiment 4: Distance Vector Routing

**Aim:** To generate routing tables for a network of routers using Distance Vector Routing

**Objective:** After carrying out this experiment, students will be able to:

* Generate routing tables for a given network using Distance Vector Routing
* Analyze the reasons why Distance Vector Routing is adaptive in nature

**Problem statement:** You are required to write a program that can generate routing tables for a network of routers. Take the number of nodes and the adjacency matrix as input from user. Your program should use this adjacency matrix and create routing tables for all the nodes in the network. The routing table should consist of one entry per destination. This entry should contain the total cost and the outgoing line to reach that destination.

**Analysis:** While analyzing your program, you are required to address the following points:

* Why is Distance Vector Routing classified as an adaptive routing algorithm?
* Limitations of Distance Vector Routing

**MARKS DISTRIBUTION**

|  |  |  |
| --- | --- | --- |
| **Component** | **Maximum Marks** | **Marks Obtained** |
| Preparation of Document | 7 |  |
| Results | 7 |  |
| Viva | 6 |  |
| **Total** | **20** |  |

Submitted by: Subhendu Maji

Register No: 18ETCS002121

1. Algorithm/Flowchart

STEP 1: Start

STEP 2: declare variables

STEP 3: totalNodes, adjMat, source from user

STEP 4: for i from 0 to totalNodes, do

4.1: distances[i]

STEP 5: distances[source] 0

STEP 6: for i from 0 to totalNodes, do

6.1: for j from 0 to totalNodes, do

6.1.1: if j is source or j equal to i or adjMat[i][j] is , continue

6.1.2: else, relax node by checking if distances[j] greater than distance[i] + adjMat[i][j]. If yes, distance[j] distance[i] + adjMat[i][j]

STEP 7: display results

STEP 8: Stop

1. Program



Figure Source Code with comments

1. Results

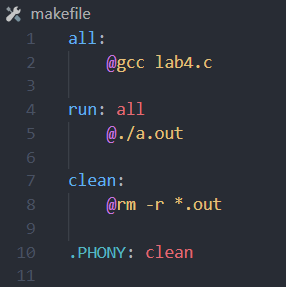


Figure makefile

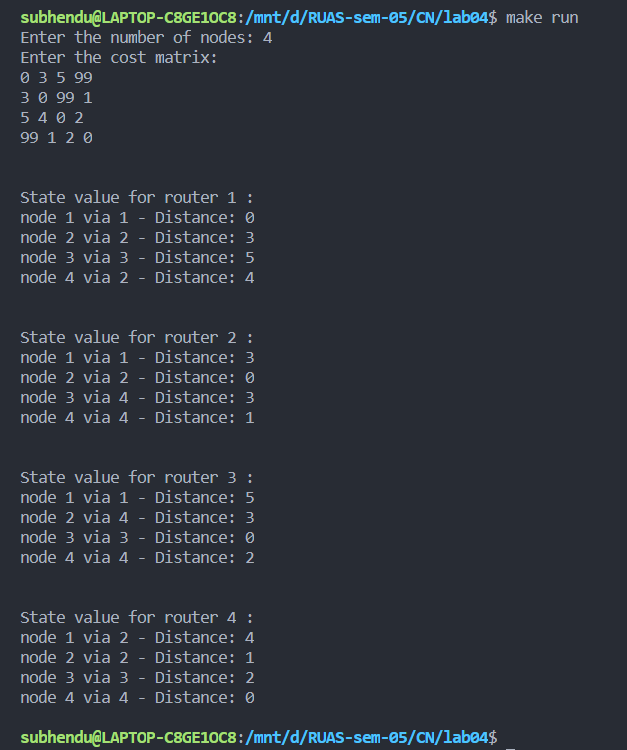


Figure execution

1. Analysis and Discussions

Bellman Ford is one of the algorithms used to calculate the least cost to travel from vertex to vertex. The uniqueness that Bellman Ford bring to the table is that it works with negative edge weights as well. The adjacency matrix consisting of the edge weights of the graph and the number of are given by the user. Using this matrix, “number of nodes minus one” operations for Bellman Ford algorithm were executed. Every vertex was compared with other vertices if there exists an edge between them and if the weight already present is lesser or greater than the new weight. If the new weight is lesser than the old weight, it is exchanged from the new weight. Bellman Ford protocol requires that a router informs its neighbours of topology changes periodically. That is why the Bellman Ford algorithm is called adaptive routing algorithm. The algorithm implementation always checks for N-1 conditions even after these are no more changes to the graph. This is inefficient and extra work for the program.

1. Conclusions

The Distance Vector Routing method using Bellman Ford algorithm was learned and implemented in C.