Madan Bhurtel

1001752499  
  
**How to run this program:**   
 - Open a folder Distance Routing folder through IDE.

- Run the code RouterSimulator.java (main file) by right click in the file and run in the terminal.

- Select the file ‘test.txt’ in the directory and click the button accordingly. Also, can check Distance Vector Routing.docx to see after selecting the file.

**Note:** All the file inside the Distance Routing Vector should be opened to run the code as it contains some class file (inside bin) other than code well.

**test.txt is the input file**

**Files in the directory named 1001752499:**

* 1. **src**
     1. **app**
        1. **NodeTextRegion.java:**

Create GUI class.

Set up GUI window and then display the GUI window.

* + - 1. **Node.java:**

Create each route node class.

Set the distance vector table for route node.

Send route’s distance vector change to neighbor routes.

Re-calculate the distance vector

Process for neighbor’s distance vector change received and send it to neighbor.

Print its GUI Text Area

Update the link Cost and send to neighbor.

* + - 1. **RouterData.java:**

Create a class when route’s link cost change (process class)

This class will have id of sending router and id of destination router. The router must be an intermediate neighbor.

* + - 1. **RouterSimulator.java:**

**Main file**

Function: Distance Vector Router Simulator.

Function that creates main GUI window

Function to notice when the file open button clicks

Get the simulation time and cycle count

Contains function of simulation without intervention step by step

Function for Init Link Cost Button click event

* + - 1. **test.txt:**

**-Input file**

Text file that contains the number of two nodes followed by the link cost. (6 of them)

* + - 1. **Distance Vector Routing.docx:**   
         This file contains steps for Distance Vector Routing after the code is executed. It has some examples that shows the GUI which tells the simulation time and count as well.
  1. **Bin**
     1. **app**
        1. **All the .class file of .java file above that are needed for the code.**
  2. **Writeup**

**-**Tells about location of the files inside the directory.

**-** Explain in detail about the output.

**Observations:**

Some of the observations are discussed below:

1. Total Simulation time: 120-190 ms   
    Cycle count: 3

Total Time after resetting the Initial ink Cost: 50-80ms

1. Number of cycle that the system takes to reach a stable state: 3
2. Also, if the user wants to see all the simulation step by step, they can do that as well, just by clicking step simulator when the GUI window pops up.
3. Also, when the neighboring link is used, it is updated in the table as well with distance vector and destination link cost and routes.
4. **All the other values like link cost, distance vector. of each router with others (From 1 to router 6) will be displayed in separate GUI window.**

**Some output screenshots are here as follow: (These snapshots are when the infinity is setup as 999 (test run cases), but when actual code is executed, infinity is as 16 as per instruction)**

Here screenshot of my GUI window running simulation without intervention that display time and cycle count.

Calendar

Description automatically generated

Simulation Time= 221 ms   
Cycle Count: 3

Calendar

Description automatically generated

**Screenshot after clicking initial link cost button and step simulator.**

Calendar

Description automatically generated

**Also, the simulation can be done Step wise:**The steps tells total cycle count that the node is stable.

Here,

**notice Step 1.**

Graphical user interface, application, table, Excel

Description automatically generated

Here below **notice step 2:**

Calendar

Description automatically generated

Step 3:

Calendar

Description automatically generated

Also, you can notice that the link cost changes when node detects local link cost change and updates routing info and recalculates the distance vector. And if the distance vector changes, it will notify to the neighbor

Also, you can change the cost of the link and expect different results.

**Other observations:**

1

7

8

2

1

2

As follow is distance vector table in first cycle. At first router 1 has only distance values about its own.

In distance Table rows and columns show router number, ex: 1 2 3 4 5 6. App set router count as 6 as project requirement. For example, 0 means distance value between router 1 and router 1, 7 means distance value between 1 and 2(2 and 1), 1 means between 1 and 5 routers (5 and 1), remain values means infinite: not link directly.

Outputs:

Current state for router 1 at Cycle 1

Distance table:

1 2 3 4 5 6

------------------------------------------------------------------------------------

1 0 7 16 16 1 16

2 16 16 16 16 16 16

3 16 16 16 16 16 16

4 16 16 16 16 16 16

5 16 16 16 16 16 16

6 16 16 16 16 16 16

Our distance vector and routes:

destination 1 2 3 4 5 6

------------------------------------------------------------------------------------

Link 1 1 1 1 1 1

cost 0 7 16 16 1 16

Current state for router 1 at Cycle 2

At cycle 2, router 1 received distance values from other routers and update its distance values.

The distance value 5 is calculated by minCost(7(cost 1-2)+1(cost 2-3), 1(cost 1-5)+4(2(cost 5-4)+2(cost(4-3)))=> minCost(8,5)=>5.

In Out distance vector and routers describes neighbor route and cost to destination route. ex: when destination router is 3, distance vector route path goes through neighbor router 2 and link cost is 5.

Distance table:

1 2 3 4 5 6

------------------------------------------------------------------------------------

1 0 7 5 3 1 16

2 7 0 1 16 8 16

3 16 1 0 2 16 16

4 16 16 2 0 2 16

5 1 8 16 2 0 16

6 16 16 16 16 16 0

Our distance vector and routes:

destination 1 2 3 4 5 6

------------------------------------------------------------------------------------

Link 1 2 2 5 5 1

cost 0 7 5 3 1 16

Current state for router 1 at Cycle 3

Distance table:

1 2 3 4 5 6

------------------------------------------------------------------------------------

1 0 6 5 3 1 16

2 6 0 1 3 5 16

3 5 1 0 2 4 16

4 3 3 2 0 2 16

5 1 5 4 2 0 16

6 16 16 16 16 16 0

Our distance vector and routes:

destination 1 2 3 4 5 6

------------------------------------------------------------------------------------

Link 1 5 5 5 5 1

cost 0 6 5 3 1 16

Current state for router 1 at Cycle 3

Distance table:

1 2 3 4 5 6

------------------------------------------------------------------------------------

1 0 6 5 3 1 16

2 6 0 1 3 5 16

3 5 1 0 2 4 16

4 3 3 2 0 2 16

5 1 5 4 2 0 16

6 16 16 16 16 16 0

Our distance vector and routes:

destination 1 2 3 4 5 6

------------------------------------------------------------------------------------

Link 1 5 5 5 5 1

cost 0 6 5 3 1 16

At cycle 3, there are no distance vector values update. ex: Destination router 2, by neighbor router 5, link cost is 6. you can verify this value in above figure.

And There must not be negative values, because initial link cost values are positive values and calculate the link cost value’s sum.

**The same pattern follows (the distance, destination, link and cost varies) for all other routers from 2 to 6 as well.**

***Reference: github.com, youtube.com***