CS3205 Networks Assignment 3 Report

Author Name: **Anumala Venu Madhava Reddy**

Roll no: **CS18B051**

Department of Computer Science and Engineering

April 20, 2021

1 Aim/objective

- The purpose of this lab is to implement a simplified version of the Open Shortest Path First (OSPF) routing protocol.

2 Introduction

- Given multiple alternative paths, how to route information to destinations should be viewed as a policy decision
- One such shortest path policy is Open Shortest Path First (OSPF)
- Open Shortest Path First (OSPF) is based on Dijkstra's algorithm to calculate shortest paths from one node to all other nodes.
- The topology information is flooded throughout the network, so that every router within the network has a complete picture of the topology of the network.

3 Experimental details

- We are given a set of N routers with M connections between these edges , the goal for each router is to ,
- Exchange Hello packets with neighbours every x seconds
- create Link State Advertisement (LSA) packets based on neighboring nodes info we got through Helloreply.
- broadcast the LSA packets to all other routers in the network , through flooding
- construct the network topology based on the LSA packets received from other routers
- Determe the routing table entries based on this topology, by using Dijkstra' algorithm (single source - all nodes shortest paths). If multiple equalcost paths exist, any one of them can be reported.

3.1 Experimental/Simulation setup

- The input format is as follows, The first entry on the first line specifies the number of routers (N) The node indices go from 0 to (N-1) ,The second entry on the first line specifies the number of links M
- Each subsequent row contains the tuple (i, j, MinC_{ij} , MaxC_{ij}). This implies a bidirectional link between nodes i and j . The use of minimum and maximum will be defined later.
- Each OSPF router (running as a Linux process) will perform the following actions:
  - Obtain necessary parameters including its node identifier (id) , Read the input file and find out its neighboring node identifiers.

- Establish a UDP socket on port number (10000 + id) for all OSPF communications.

- Each OSPF router then Sends a HELLO message to its neighbors, once every HELLO INTERVAL seconds. This value is specified in the command line as x; Default: 1 second.One thread implements this part.
- Packet Format: HELLO srcid
- When a router receives a HELLO message on an interface, it will reply with the HELLOREPLY message along with the cost. The cost reported for link ij by node j for a packet received from node i is a RANDOM number between MinC_{ij} and MaxC_{ij} . Node i, on receiving this message, will store this value as the cost for link ij
- The message format is: HELLOREPLY j i costforlinkij
- Send a Link State Advertisement (LSA) message to its neighbors, once every LSA INTERVAL seconds. This value is specified in the command line; Default: 5 seconds.One thread implements this part.
- packet Format : LSA srcid seqno noofentries neigh1 cost1 neigh2 cost2 ...
- Using the LSA packets recieved from all other nodes , determine the network topology and use Dijkstra's algorithm to compute the shortest path and write it to the outfile.

3.2 Entities Involved

- id , it denotes the node no of the current router
- infile , the file from which input is read
- outfile , the file to which routing table is written to , every z seconds
- x , HELLO INTERVAL Time , for Every x seconds HELLO packets are sent
- y , LSA INTERVAL time , for every y seconds LSA packets are sent
- z , SPF INTERVAL time , for every z seconds The routing table is computed

3.3 Additional Details

- We are using locks in between threads , so that graph won't change during it's copy for dijkstra's algorithm.
- Because we are using only 8 nodes and 20 to 28 edges in the network topology , the time taken for computation in the threads sendhello , sendlsa , dodix is usually small , so we just used sleep(x) , sleep(y) , sleep(z) in the threads as a means to call the threads every x , y , z seconds.

4 Results and Observations

- input-1
  8 22
  0 1 4 10
  1 2 3 9
  2 0 6 10
  3 1 4 10
  3 2 3 9
  0 3 6 10
  0 4 2 5
  4 1 7 20
  2 4 3 7
  4 3 9 17
  0 5 10 15
  5 1 13 20
  2 5 20 27
  5 3 25 26
  0 6 12 16
  6 1 13 17
  2 6 4 6
  6 3 1 5
  0 7 9 15
  7 1 15 20
  2 7 19 24
  7 3 30 35
  $x = 1, y = 5, z = 20$

- Table-1
  ================================
  Routing Table for Node number. 0 at Time 20

Destination	Path	Cost
1	01	7
2	02	8
3	03	8
4	04	5
5	05	11
6	026	12
7	07	10

  ================================
  Routing Table for Node number. 0 at Time 40

  | Destination | Path | Cost |

1	01	4
2	02	6
3	03	8
4	04	2
5	05	10
6	036	9
7	07	11

- We can clearly see the graph has changed from what the graph is at time = 20.
- The costs have changed and also the path for node 6 have changed.
- This is because of hello packets , each x seconds the costs changes , but all nodes realizes the changes and updates the changes every y seconds when they recieve lsa packets.
- input-2
  8 20
  1 0 5 20
  1 2 12 30
  2 0 13 17
  3 1 11 16
  3 2 7 19
  0 3 2 10
  4 5 16 25
  4 6 27 31
  4 7 21 24
  5 6 31 35
  5 7 10 15
  6 7 23 27
  0 4 1 30
  4 1 9 19
  1 5 12 14
  7 2 13 20
  3 6 11 17
  4 2 4 7
  3 7 10 13
  6 2 20 27
  $x = 1, y = 5, z = 20$
- Table-2
  ================================
  Routing Table for Node number. 0 at Time 20

Destination	Path	Cost
1	01	12

2	02	15
3	03	2
4	04	15
5	015	24
6	036	17
7	037	13

================================

Routing Table for Node number. 0 at Time 40

Destination	Path	Cost
1	01	10
2	02	16
3	03	10
4	04	9
5	015	24
6	036	21
7	037	20

- We can see , the output for node 0 at time = 20 for input1 and input2 are different , this is beacuse the network topology has changed and min and max C_{ij} also given different.
- We can clearly see the graph has changed from what the graph is at time = 20.
- The costs have changed and also the path for node 6 have changed.
- This is because of hello packets , each x seconds the costs changes , but all nodes realizes the changes and updates the changes every y seconds when they recieve lsa packets.

5 Learnings

- As we have a common memory which is accessed by all the threads , we need to use locks , especially while doing dijkstra's algorithm , other wise the graph changes during the copying of graph itself .
- For a small network topology , the number of lsa packets transferred will be small , so OSPF routing protocol can be efficiently used.

6 Additional Thoughts

- We can keep y and z constant and change only x and see how it effects the output.
- This OSPF is not scalable as the no of routers increases the time taken to get the network topology increses , because of time taken by lsa to reach all the nodes increases.

7 conclusion

- We can use OSPF routing protocol to send messages between two not directly connected routers,by calculating the shortest path , with the network topology at that time and send the message along the shortest path.

8 References

https://www.geeksforgeeks.org/mutex-lock-for-linux-thread-synchronization/
https://www.geeksforgeeks.org/thread-functions-in-c-c/
https://www.geeksforgeeks.org/udp-server-client-implementation-c/
https://www.cyberciti.biz/faq/run-execute-sh-shell-script/
https://www.geeksforgeeks.org/substring-in-cpp/