CS3205 Assignment 3 - OSPF Routing Algorithm Report

CS18B054 - Dipesh Tandel

April 20, 2021

Objective

The objective of this lab is to implement a simplified version of the Open Shortest Path First (OSPF) routing protocol.
Given a set of N routers, the goal is for EACH router to:

(a) exchange HELLO packets with neighbours,

(b) create Link State Advertisement (LSA) packets based on neighboring nodes' info,

(c) broadcast the LSA packets to all other routers in the network,

(d) construct the network topology based on the LSA packets received from other routers,

(e) determining the routing table entries based on this topology, by using Dijkstra' algorithm (single source – all nodes shortest paths). If multiple equal cost paths exist, any one of them can be reported.

Introduction

The Open Shortest Path First (OSPF) protocol is one of a family of IP Routing protocols.It is an intra-domain routing protocol used to distribute IP routing information throughout a single Autonomous System(AS) in an IP network.
In OSPF protocol the routers exchange network topology information with their neighbours periodically through flooding so that every router has the complete picture of the network topology so that it can calculate the shortest path to all the routers(using Dijkstra's algorithm).

Flooding is done in the following way:

- Each router periodically creates Link State Packet(LSP) which contains router id, list of all neighbours and costs, sequence number and time to live.
- Sequence number is used to identify the latest LSP.
- LSP is then forwarded to all the neighbours except the neighbour where the LSP came from.
- LSP is also generated when a link's state changes (failed or restored).Note that this flooding is reliable as we use acknowledgement and retransmission.

Experimental Details

In this lab we implement a simplified version of OSPF routing protocol on a single machine with one Linux process per OSPF router. To each process/router we provide the following parameters -

- Node identifier value (id) - unique identifier for each router (varies from 0 to N-1, where N is number of routers)
- Input file ($infile$) - Contains the network topology
- Output file ($outfile$) - Location where the output(paths) is stored
- HELLO interval (hi) - The HELLO packets will be exchanged every hi seconds ;
- LSA interval ($lsai$) - The LSA updates will be sent every lsai seconds
- SPF interval ($spfi$) - the routing table computation will be done every spfi seconds

Each process will :

- Establish a UDP socket on port number (10000 + id) for the OSPF communications.
- Create 3 threads for sending HELLO message to neighbours periodically, sending LSA message to neighbours periodically and calculating the paths periodically.
- When a router receives a HELLO message it replies with a HELLOREPLY message which also the contains the edge cost between the two routers involved.
- When a router receives a HELLOREPLY message it stores the edge cost.
- When a router receives a LSA message it will check the sequence number. If it is strictly greater than the last known sequence number from the sender then we store the LSA information and forward the LSA to all neighbours except to the neighbour who sent the message.

Functions involved:

- **sendHELLO()** - sends HELLO message to all its neighbours every hi seconds.
- **sendLSA()** - sends LSA message to all its neighbours every $lsai$ seconds.
- **calcSPF()** - calculates the shortest paths for the network topology after every $spfi$ seconds.
- **sendmsg(int destid, string msg)** - sends the message msg to the router number $destid$.
- **rand_int(int min, int max)** - returns a random number between min and max.
- **dijkstraAndwrite()** - finds the shortest path and writes to the output file.
- **writePath()** - utility function for **dijkstraAndwrite()** to find all the elements in the shortest path.
- **init()** - Does all the initialization and also reads the input file and computes th neighbour details.

Results and Observations

INPUT FILE 1 - 'input1.txt'

14		22	
0	1	2	8
0	2	5	10
1	2	6	20
2	3	11	13
2	5	8	13
2	4	6	13
3	11	4	23
1	6	7	16
4	7	8	10
5	6	4	18
5	7	7	19
6	7	17	28
6	9	13	29
7	9	4	16
7	8	5	10
8	9	3	9
4	10	2	8
8	10	5	25
12	13	6	23
10	13	8	21
10	11	8	18
11	13	3	19

OUTPUT FILE 1 for Router 0 - 'output1-0.txt'

Routing Table for Node No.0 at Time 80		
Destination	Path	Cost
1	0-1	3
2	0-2	6
3	0-2-3	17
4	0-2-4	17
5	0-2-5	15
6	0-1-6	10
7	0-2-5-7	25
8	0-1-6-9-8	30
9	0-1-6-9	25
10	0-2-4-10	25
11	0-2-3-11	38
12	0-2-4-10-13-12	45
13	0-2-4-10-13	38

OUTPUT FILE 1 for Router 5 - 'output1-5.txt'

Routing Table for Node No.5 at Time 80		
Destination	Path	Cost
0	5-2-0	15
1	5-2-0-1	17
2	5-2	9
3	5-2-3	21
4	5-2-4	20
6	5-6	5
7	5-7	14
8	5-7-8	20
9	5-6-9	20
10	5-2-4-10	28
11	5-2-3-11	34
12	5-2-3-11-13-12	46
13	5-2-3-11-13	39

OUTPUT FILE 1 for Router 10 - 'output1-10.txt'

Routing Table for Node No.10 at Time 80		
Destination	Path	Cost
0	10-4-2-0	24
1	10-4-2-0-1	26
2	10-4-2	18
3	10-11-3	22
4	10-4	7
5	10-4-2-5	27
6	10-4-2-5-6	32
7	10-4-7	16
8	10-8	21
9	10-8-9	25
11	10-11	9
12	10-13-12	18
13	10-13	11

INPUT FILE 2 - 'input2.txt'

18		22	
0	1	2	8
0	2	5	10
1	6	7	16
4	7	8	10
5	6	4	18
5	7	7	19
6	7	17	28
6	9	13	29
7	9	4	16
7	8	5	10
8	9	3	9
2	4	6	13
3	11	4	23
12	13	6	23
10	13	8	21
10	11	8	18
11	13	3	19
4	10	2	8
8	10	5	25
1	2	6	20
2	3	11	13
2	5	8	13
14	17	2	18
15	16	8	14
15	17	3	14
16	17	5	13
4	17	21	30
15	12	19	26
16	6	16	21
14	0	18	23

OUTPUT FILE 2 for Router 1 - 'output2-1.txt'

Routing Table for Node No.1 at Time 80		
Destination	Path	Cost
0	1-0	3
2	1-2	6
3	1-2-3	19
4	1-2-4	14
5	1-2-5	16
6	1-6	7
7	1-2-4-7	24
8	1-6-9-8	29
9	1-6-9	23
10	1-2-4-10	19
11	1-2-3-11	23
12	1-2-3-11-13-12	52
13	1-2-3-11-13	29
14	1-0-14	23
15	1-6-16-15	33
16	1-6-16	25
17	1-0-14-17	27

OUTPUT FILE 2 for Router 10 - 'output2-10.txt'

Routing Table for Node No.10 at Time 80		
Destination	Path	Cost
0	10-4-2-0	17
1	10-4-2-1	18
2	10-4-2	12
3	10-4-2-3	24
4	10-4	2
5	10-4-2-5	24
6	10-4-2-1-6	25
7	10-4-7	10
8	10-4-7-8	15
9	10-4-7-9	17
11	10-11	15
12	10-13-12	32
13	10-13	9
14	10-4-17-14	29
15	10-4-17-15	35
16	10-4-17-16	32
17	10-4-17	25

OUTPUT FILE 2 for Router 15 - 'output2-15.txt'

Routing Table for Node No.15 at Time 80		
Destination	Path	Cost
0	15-17-14-0	30
1	15-17-14-0-1	33
2	15-17-14-0-2	35
3	15-17-14-0-2-3	46
4	15-17-4	27
5	15-16-6-5	42
6	15-16-6	29
7	15-17-4-7	35
8	15-17-4-7-8	40
9	15-17-4-7-9	42
10	15-17-4-10	29
11	15-17-4-10-11	44
12	15-12	21
13	15-12-13	37
14	15-17-14	8
16	15-16	9
17	15-17	4

There are two input files(two topologies) with 14 and 18 routers but only 3 router tables are shown for each input file. The second latest timestamped table is chosen beacuse the latest timestamp may be running and we may have killed it in between.

- Number of computations (computing shortest paths , processing the messages received, etc.) and number of communication messages (sending and receiving HELLO HELLOREPLY, LSA messages ,etc.)involved is high.
- Requires much memory to store the topology details
- As the number of routers increase, the time taken for all LSA messages to reach all the routers increases.
- As the number of routers increases, the size of topology updates increases. Also the frequency of this updates increases.

Learnings

- OSPF Routing Protocol
- Link State Advertisement
- Dijkstra's Algorithm

Conclusion

From the above tables we can see that we have successfully sent the topology details to all the routers and computed shortest path table for all the routers. Hence we have successfully implemented the OSPF routing protocol.

References

GFG - Dijkstras algorithm Note : The code on this site is slightly incorrect
GFG - UDP Server-Client