



CS 542 Project Report

LINK-STATE ROUTING ALGORITHM

Professor: Dr. Michael Y. Choi

TA : Viswatej Kasapu

By: Song Wang

CWID: A20435988

SEAT# : 29



Contents

Abstract.....	3
Objectives.....	3
Link-State Routing Protocol.....	3
Dijkstra's Algorithm.....	5
Steps / Pseudocode.....	6
Explanation of the project.....	6
Execution of the Program.....	7
Important Screen shots.....	10
Reference.....	15

Abstract

The project will calculate the least-cost path in network topology and use the Dijkstra's Algorithm in JAVA.

Objectives

The main operations of the project as shown in following:

1. Input a Network Topology
2. Create a Forward Table
3. Paths from Source to Destination
4. Update Network Topology
5. Best Router for Broadcast
6. Exit

Link-State Routing Protocol

Each node independently runs an algorithm over the map to determine the shortest path from itself to every other node in the network; generally, some variant of Dijkstra's algorithm is used. This is based around a link cost across each path which includes available

bandwidth among other things. A node maintains two data structures: a tree containing nodes which are "done", and a list of candidates. The algorithm starts with both structures empty; it then adds to the first one the node itself. The variant of a Greedy Algorithm then repetitively does the following: All neighboring nodes which are directly connected to the node are just added to the tree (excepting any nodes which are already in either the tree or the candidate list). The rest are added to the second (candidate) list. Each node in the candidate list is compared to each of the nodes already in the tree. The candidate node which is closest to any of the nodes already in the tree is itself moved into the tree and attached to the appropriate neighbor node. When a node is moved from the candidate list into the tree, it is removed from the candidate list and is not considered in subsequent iterations of the algorithm. The above two steps are repeated as long as there are any nodes left in the candidate list. (When there are none, all the nodes in the network will have been added to the tree.) This procedure ends with the tree containing all the nodes in the network, with the node on which the algorithm is running as the root of the tree. The shortest path from that node to any other node is indicated by the list of nodes one

traverses to get from the root of the tree, to the desired node in the tree![1]

Dijkstra's Algorithm

```

1  function Dijkstra(Graph, source):
2
3      create vertex set Q
4
5      for each vertex v in Graph:
6          dist[v] ← INFINITY
7          prev[v] ← UNDEFINED
8          add v to Q
9
10     dist[source] ← 0
11
12     while Q is not empty:
13         u ← vertex in Q with min dist[u]
14
15         remove u from Q
16
17         for each neighbor v of u:           // only v that are
18         still in Q
19             alt ← dist[u] + length(u, v)
20             if alt < dist[v]:
21                 dist[v] ← alt
22                 prev[v] ← u
23
24     return dist[], prev[]

```

[2]

Steps / Pseudocode

1. Initialization
2. Mark the source u = visited
3. For all nodes v
4. If : v is neighbor of u
5. then $D(v) = c(u, v)$
6. Else : $D(v) = \text{Infinity}$
7. Loop :
8. Find node w not visited and who's $D(w) = \min$
9. Mark node w as visited
10. Update $D(v)$ for all the neighbor v of w , not visited
11. Set $D(v) = \min (D(v) + D(w) + c(w, v))$
12. /* new cost to v is either the old cost to v or least cost of the path to w + cost from w to v */
13. Repeat unless all the nodes are visited

Explanation

The following are the notations used:

u : Source node , w : Not visited node

$D(v)$: least cost from source u to destination v

$C(u, v)$: It is the cost on the link which is incurred to send a packet from node u to v and is infinity when there is no direct connection or link between the node and I have used -1 for the same in the project.

Execution of the Program

Steps:

1. Open the Eclipse Application and move the main.java and getfile.java into a new package, which is called CS542_2019SP_Project_29_Wang_Song.
2. Starting with Run button and show the Main operations.
3. Thereafter the Main menu is displayed then choose the options you want the main to perform.

Note: User need to install the Eclipse App for Windows System firstly.

 **The project has 2 java files:**

1. Main.java

- 1) This is the main class that will display the main menu.
- 2) The user will be asked to select the option one by one for which the user wants this project to perform.



- 3) The main file could call the different functions store in `getfile()` class for every steps.
- 4) The following are the functions present in this file:
 - a) `Add_New_Router()`: function to add new router in topology
 - b) `Update_Router(int New_R,int End_R, int Value)`: update edge weight
 - c) `Show_New_Matrix()`:print the new matrix
 - d) `Del_Router()`: Delete Router from end
 - e) `main(String[] args)`: Showing the operation menu and six opinions and select different operations

2. getfile.java

- 1) This file contains all the functions which are being called by the main file
- 2) The following are the functions present in this file:
 - a) `Pair()`: to use store the values of `shortestPath` and `shortestDest`
 - b) `take_file()`: show the network topology and calculate how many routers in the topology
 - c) `GetForwardTable()`: the function to get forward table from network topology matrix



- d) `LeastPath(int Source_Router_Number, int[][] Network_Topology_Matrix)`: find the neighboring node that has the least-cost path
- e) `TakeCostLeastPath(int Network_Topology_Matrix[][], int Source_A_Number, int Destination_A_Number)`: showing the shortest distance from Source to destination
- f) `findMinimum(int[] cost, boolean[] visited)`: Finds the minimum cost by comparing `currMin` and `cost[i]`
- g) `getBroadcasting(int[][] matrix)`: broadcasting method and get the total value from source to destination
- h) `getTotalCost(int Source_Router_Number, int[][] Network_Topology_Matrix)`: get the total least cost from source to every other routers

The user interface will show in the following:

```

Problems @ Javadoc Declaration Console x
Main (1) [Java Application] D:\JAVA\bin\javaw.exe (2019年4月28日 下午3:17:44)
### Welcome Using Link-State Routing Algorithm and it was created by Song ###
Please choose one number of operation according to the following menu!
1.Input a Network Topology
2.Create a Forward Table
3.Paths from Source to Destination
4.Update Network Topology
5.Best Router for Broadcast
6.Exit
<

```

Network Topology & Adjacency Matrix

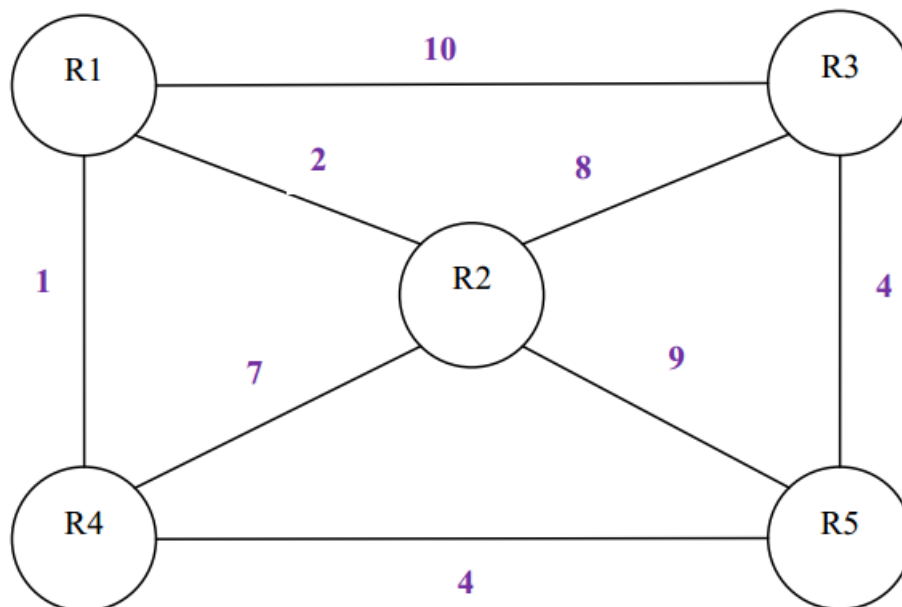
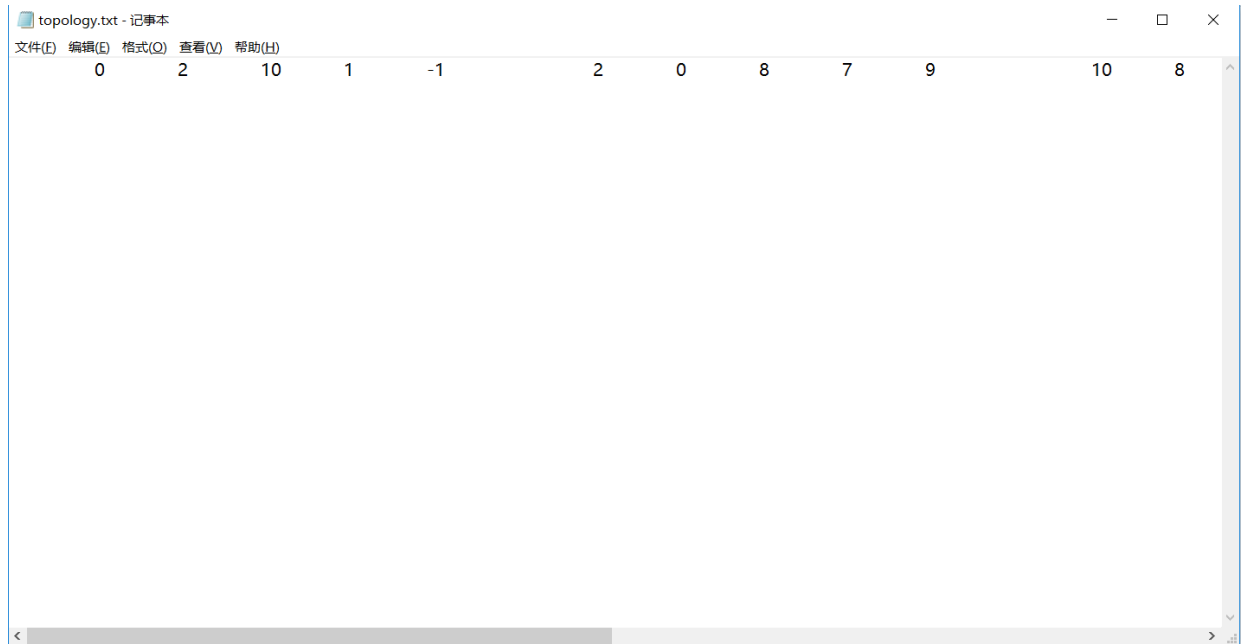


Figure: A sample network topology with costs on links



Output –Command 1

```

Problems @ Javadoc Declaration Console X
Main (1) [Java Application] D:\JAVA\bin\javaw.exe (2019年4月28日 下午3:17:44)
### Welcome Using Link-State Routing Algorithm and it was created by Song ###
Please choose one number of operation according to the following menu!
1.Input a Network Topology
2.Create a Forward Table
3.Paths from Source to Destination
4.Update Network Topology
5.Best Router for Broadcast
6.Exit
1
Please enter the file path in this line
D:\JAVA\Songcs542\topology.txt
The # of Routers = 5
***** The network topology is shown below: *****
      R1      R2      R3      R4      R5
R1      0       2      10       1      -1
R2      2       0       8       7       9
R3     10       8       0      -1       4
R4      1       7      -1       0       4
R5     -1      9       4       4       0

```

Output –Command 2

```

Problems @ Javadoc Declaration Console x
Main (1) [Java Application] D:\JAVA\bin\javaw.exe (2019年4月28日 下午3:17:44)
4.Update Netwok Topology
5.Best Router for Broadcast
6.Exit
1
Please enter the file path in this line
D:\JAVA\Songcs542\topology.txt
The # of Routers = 5
***** The network topology is shown below: *****
      R1      R2      R3      R4      R5
R1      0       2      10       1      -1
R2      2       0       8       7       9
R3     10       8       0      -1       4
R4      1       7      -1       0       4
R5     -1       9       4       4       0

### Welcome Using Link-State Routing Algorithm and it was created by Song ###
Please choose one number of operation according to the following menu!
1.Input a Network Topology
2.Create a Forward Table
3.Paths from Source to Destination
4.Update Netwok Topology
5.Best Router for Broadcast
6.Exit
2
Select a number of source router!
1
Destination      Interface Link
1                -
2                 2
3                10
4                 1
5               10000

```

Output –Command 3

```

Problems @ Javadoc Declaration Console x
Main (1) [Java Application] D:\JAVA\bin\javaw.exe (2019年4月28日 下午3:17:44)
3
Enter a number of Source Router!
1
Enter a number of Destination Router!
5
The Cost of Shortest distance from 1 to 5 is: 5
Shortest path from 1 to 5 is: (Source Router is in rightmost)
5<--4<--1

```

Output –Command 4

4

Please make a choice according to the following funcations:

- (1)Add a new router in the topology matrix
- (2)update a new edge in the topology matrix
- (3)delete a route in the topology matrix

1

```
0,2,10,1,-1,-1,
2,0,8,7,9,-1,
10,8,0,-1,4,-1,
1,7,-1,0,4,-1,
-1,9,4,4,0,-1,
-1,-1,-1,-1,-1,-1,
```

Output –Command 4

***** The network topology is shown below: *****

	R1	R2	R3	R4	R5
R1	0	2	10	1	-1
R2	2	0	8	7	9
R3	10	8	0	-1	4
R4	1	7	-1	0	4
R5	-1	9	4	4	0



4

Please make a choice according to the following funcations:

- (1)Add a new router in the topology matrix
- (2)update a new edge in the topology matrix
- (3)delete a route in the topology matrix

2

Please input a new router number and weight: r1,r2,weight

1,2,3

```
0,3,10,1,-1,
2,0,8,7,9,
10,8,0,-1,4,
1,7,-1,0,4,
-1,9,4,4,0,
```

Output –Command 4

***** The network topology is shown below: *****

	R1	R2	R3	R4	R5
R1	0	2	10	1	-1
R2	2	0	8	7	9
R3	10	8	0	-1	4
R4	1	7	-1	0	4
R5	-1	9	4	4	0



4

Please make a choice according to the following funcations:

- (1)Add a new router in the topology matrix
- (2)update a new edge in the topology matrix
- (3)delete a route in the topology matrix

3

|
0,2,10,1,
2,0,8,7,
10,8,0,-1,
1,7,-1,0,

Output –Command 5

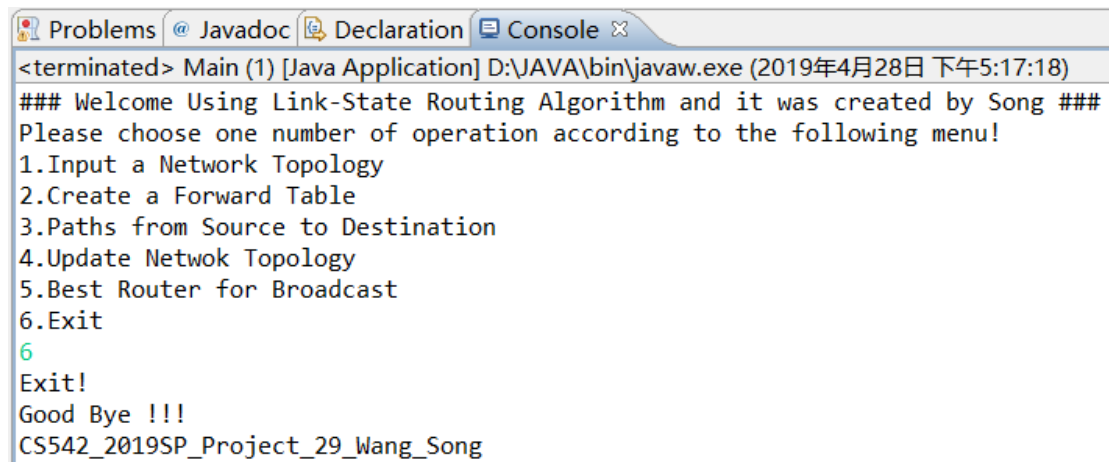
Welcome Using Link-State Routing Algorithm and it was created by Song ###
Please choose one number of operation according to the following menu!

- 1.Input a Network Topology
- 2.Create a Forward Table
- 3.Paths from Source to Destination
- 4.Update Netwok Topology
- 5.Best Router for Broadcast
- 6.Exit

5

The route:1total weight is 17
The route:2total weight is 20
The route:3total weight is 29
The route:4total weight is 16
The route:5total weight is 20
The best Router for Broadcasting is : 4
The weight is:16

Output –Command 6



```
<terminated> Main (1) [Java Application] D:\JAVA\bin\javaw.exe (2019年4月28日 下午5:17:18)
### Welcome Using Link-State Routing Algorithm and it was created by Song ###
Please choose one number of operation according to the following menu!
1.Input a Network Topology
2.Create a Forward Table
3.Paths from Source to Destination
4.Update Network Topology
5.Best Router for Broadcast
6.Exit
6
Exit!
Good Bye !!!
CS542_2019SP_Project_29_Wang_Song
```

[1] https://en.wikipedia.org/wiki/Link-state_routing_protocol

[2] https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm