**CS542- 2013 Fall**

**COMPUTER NETWORKS PROJECT REPORT**

**Implementation of Link State Routing Protocol**

Submitted by:

Nangong, Long

CWID: A20299162

Section: 01 (Live)

lnangong@hawk.iit.edu

**OBJECTIVE**

In this term project, I need to develop a program to implement Link-State Routing Protocol. My program should have two functions:

1. Simulate the process of generating routing tables for each router in a given network.

2. Compute optimal path with least cost between any two specific routers.

**PROBLEM DESCRIPTION**

Suppose we have a network with arbitrary number of routers. The network topology is given by a matrix, called the original routing table, which only indicates the costs of links between all directly connected routers. We assume each router only knows its own information and has no knowledge about others at the beginning.

In this project, to implement Link-State Routing Protocol, first our program is required to create the state of the links by each router after the input file containing the network information been loaded. This is called the link state packet or LSP. Then, the program need to flood LSPs of each router to all other routers and build the routing table for each router. A Dijkstra’s algorithm could be applied to find shortest path tree. Finally, our program should be able to output the routing table of any router, and output the optimal path between any two selected routers.

**LINK-STATE ROUTING PROTOCOL DESCRIPTION**

A link-state routing protocol is one of the two main classes of routing protocols used in packet switching networks for computer communications (the other is the distance-vector routing protocol).

The link-state protocol is performed by every *switching node* in the network (i.e., nodes that are prepared to forward packets; in the [Internet](http://en.wikipedia.org/wiki/Internet), these are called [routers](http://en.wikipedia.org/wiki/Router_(computing))). The basic concept of link-state routing is that every node constructs a *map* of the connectivity to the network, in the form of a [graph](http://en.wikipedia.org/wiki/Graph_theory), showing which nodes are connected to which other nodes. Each node then independently calculates the next best logical *path* from it to every possible destination in the network. The collection of best paths will then form the node's [routing table](http://en.wikipedia.org/wiki/Routing_table).

This contrasts with [distance-vector routing protocols](http://en.wikipedia.org/wiki/Distance-vector_routing_protocol), which work by having each node share its routing table with its neighbors. In a link-state protocol the only information passed between nodes is connectivity related.

**DESCRIPTION AND PSEUDO CODE OF DIJKSTRA'S ALGORITHM**

Dijkstra’s algorithm maintains a set S of vertices whose final shortest-path weights from the source s have already been determined. The algorithm repeatedly selects the vertex with the minimum shortest-path estimate, adds vertex to S, and update all edges leaving this vertex. In the following implementation, I use a min-priority queue Q of vertices, keyed by their distance values.

This algorithm has two stages - the first used when the network source node starts (initialization stage) and the second used to update its distance vectors in response to changes in costs received from other routers(update stage).

**1. INITIALIZATION STAGE**

Insert all nodes which performing the Link State Routing algorithms to the Priority-Q, with its distance value of infinity.

Initialization stage of Dijsktra’s Algorithm

1 Distance(v) = infinity;

2 Insert node(v);

**2. UPDATE STAGE**

Let u be the source node performing the Dijkstra’s algorithms, update Distance(u) = 0.

Update Stage of Dijkstra’s Algorithm

LOOP

1 find w not in N' such that D(w) is a minimum

2 add w to N' //N' is set of node whose minimum distance is known

3 update D(v) for all v adjacent to w and not in N' :

4 D(v) = min( D(v), D(w) + c(w,v) ) /\* new cost to v is either old cost to v or known shortest path cost to w plus cost from w to v \*/

5 until all nodes in N'

**DESIGN AND IMPLEMENTATION**

In my program of implementation of link-state routing protocol, there are four essential component. Namely,

**1. Priority-Q**

Priority-Q is the the core part of my program to calculate the shortest path between source and destination and generate optimal routing table for each router. It is implemented by min-heap, and it will support my program to implement Dijkstra's Algorithm to get the optimal routing path for each router. Each node in the priority-Q consists of information of router's id, distance calculated from source router, and previous router's id. Dijkstra's Algorithm is working by insert new node to Q, update existing node in the Q, and extract node with minimum calculated distance from the Q. The priority-Q will keeping maintain its property by implement heapify-up function while node is being manipulated by Dijkstra's Algorithm.

**2. LOAD FILE**

To design this model I have considered an original routing table which consists of the costs between two adjacent routers. Here in this text file uses 'space' to separate the two different cost values, also uses 'enter' to go to the next line, '-1' is used to represent two non-neighboring routers. This module's functionality is load the original routing table and create the state of the links for each router. This module gets the data from the original routing table and store into a 2D- array called link\_state\_packet. In that process, here it will read a character at a time, when it get read ' - ' character, record -1 instead of ' - ' to the ink\_state\_packet, and discard next '1' character read from text file, and when it encounters an enter then the row count is increased by one and column initialize to zero to get the next router's link state record into the corresponding space, and only record character 0~9 read from text file to the link\_state\_packet. At last the original routing table is printed on the console from the link\_state\_packet.

**3. COMPUTE OPTIMAL ROUTING TABLE FOR EACH ROUTER**

This module execute Dijkstra’s algorithm to calculate optimal routing table, by considering each router. For every router I update its optimal routing table values with the minimum cost calculated from the data available from its link-state packet. I use the data that is available at that point of time. First, this module will insert all the available routers to the priority-q, with data of router's id, distance with infinity, and previous router with NUL. Then apply Dijkstra's Algorithm to look into the adjacent routers of specific router, if the distance of the adjacent router is bigger than distance of present router plus cost to reach that adjacent router, then update that router's distance with less distance and previous router with present router in he priority-q. Keep extract min-node from priority-q and repeatedly do the same process to all the router until all routers get its minimum cost from the source router. During processing all the routers, program will record path and minimum cost to reach each router from source to the routing table. Then the computed routing table will be printed on the console from the optimal routing table in which it is stored and return to the menu.

**4. SHORTEST PATH AND MINIMUM COST**

This part is essentially same as compute optimal routing table module, and is also embedded in that part of program. Here user enters the source and destination number and this part is able to calculate the shortest path and minimum cost between source and destination. We input source and destination to the valuables called source and destination. Using Dijkstra's Algorithm to calculate optimal routing path for the source router just like what the compute optimal routing table module do. While compute optimal routing path, program will record previous router for each router in the shortest-path. Finally by recursively checking previous router from the destination until get reach the source, the shortest-path will be generated. Then I also print the minimum cost between source and destination from the data stored in the optimal routing table and return to the menu.

**CREDIBILITY OF THE CODE**

After these modules are done and are perfectly executing, I have tested my program on the given test file. I manually calculated the test file for the shortest path and then compared my results with the results produced by the program. They were actually the same and by this I concluded that the program is running perfectly fine. I have actually checked this for five more instances of routers.

1. 10 routers

2. 13 routers

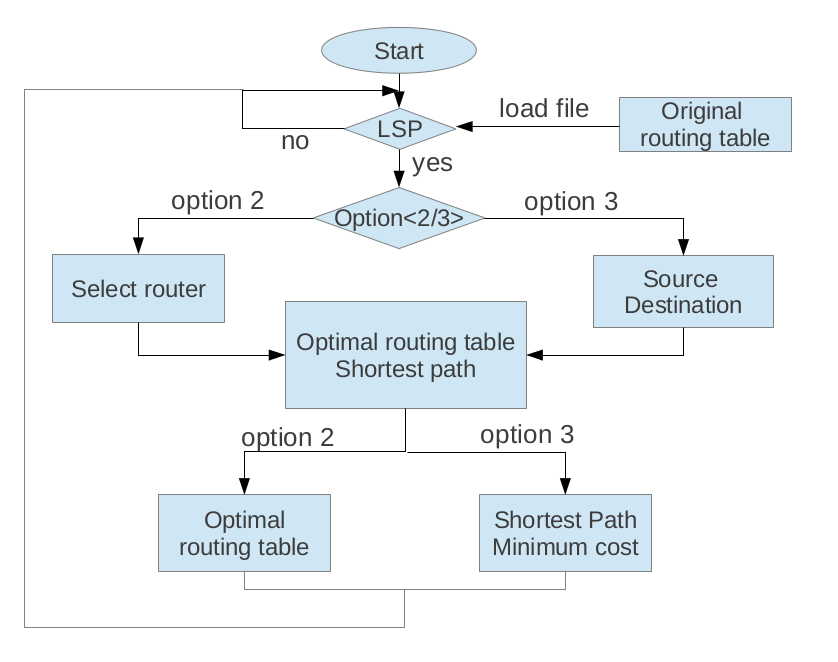
3. 15 routers

4. 17 routers

5. 20 routers

**These five instances are being documented in the following parts of the report(see Test Cases).**

**FLOW CHART OF PROGRAM**



**INSTRUCTIONS FOR COMPILING AND RUNNING THE PROGRAM**

Operating System: LINUX

Program Language: C

Compiler: GCC

**How to compile source file:**

Open the terminal, type command “gcc shortest-path -o routing”

Run this program by type command “./routing” into the terminal

Note:

1. It takes input as a text file which contains numerical values and '-1' is represented by the non-neighboring node.

2. The following menu is displayed when the program starts running.

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

3. The user selects an option from the above menu.

4. When option 1 is selected the program prompts the user to enter the text file and then it loads the input file, i.e. original routing table.

5. When option 2 is selected it prompts the user to enter a router number and then it prints the routing table of that specific router on the console.

6. When option 3 is selected, the programs prompt the user to enter source and destination routers. Given the inputs it will print the shortest path and the minimum cost to that destination from the source.

7. When we select the option 0 program will be terminated.

All these options are available all the time during running of the program.

After every option is executed user has a chance to select another option from the menu for further proceeding.

**TEST REPORT**

**Note:**

**I have considered that different direction may have different weight, then I have tested both undirected graph and directed graph.**

Test Case 1 :(Undirected graph) Sample original routing table

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):1

Please load original routing table data file:

network.txt

Original routing table is as follows:

0 2 5 1 -1

2 0 8 7 9

5 8 0 -1 4

1 7 -1 0 2

-1 9 4 2 0

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):2

Please select a router:

5

The routing table for router 5 is:

Destination:1 2 3 4 5

Cost:3 5 4 2 0

Next Router:4 4 3 4 0

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):3

Please input the source and the destination router number:

5 2

The shortest path from 5 to 2 is

5--4--1--2,the total cost is 5

Test Case 2 :(Undirected graph) 10 routers

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):1

Please load original routing table data file:

test1.txt

Original routing table is as follows:

0 6 2 7 -1 -1 -1 -1 -1 -1

6 0 4 -1 -1 -1 -1 9 6 -1

2 4 0 2 7 -1 -1 -1 -1 5

7 -1 2 0 2 -1 5 -1 -1 -1

-1 -1 7 2 0 2 -1 -1 -1 -1

-1 -1 -1 -1 2 0 8 -1 -1 -1

-1 -1 -1 5 -1 8 0 6 -1 5

-1 9 -1 -1 -1 -1 6 0 8 -1

-1 6 -1 -1 -1 -1 -1 8 0 6

-1 -1 5 -1 -1 -1 5 -1 6 0

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):2

Please select a router:

6

The routing table for router 6 is:

Destination: 1 2 3 4 5 6 7 8 9 10

Cost: 8 10 6 4 2 0 8 14 16 11

Next Router: 5 5 5 5 5 0 7 7 5 5

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):3

Please input the source and the destination router number:

6 10

The shortest path from 6 to 10 is

6--5--4--3--10,the total cost is 11

Test Case 3 :(Undirected graph) 13 routers

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):1

Please load original routing table data file:

test2.txt

Original routing table is as follows:

0 6 3 7 -1 -1 -1 -1 -1 -1 6 -1 -1

6 0 4 -1 -1 -1 -1 9 6 -1 -1 7 -1

3 4 0 3 7 -1 -1 -1 -1 5 -1 -1 -1

7 -1 3 0 3 -1 5 -1 -1 -1 -1 -1 -1

-1 -1 7 3 0 3 -1 -1 -1 -1 -1 -1 -1

-1 -1 -1 -1 3 0 8 -1 -1 -1 3 2 -1

-1 -1 -1 5 -1 8 0 6 -1 5 -1 -1 2

-1 9 -1 -1 -1 -1 6 0 8 -1 3 -1 6

-1 6 -1 -1 -1 -1 -1 8 0 6 -1 -1 -1

-1 -1 5 -1 -1 -1 5 -1 6 0 9 5 -1

6 -1 -1 -1 -1 3 -1 3 -1 9 0 -1 1

-1 7 -1 -1 -1 2 -1 -1 -1 5 -1 0 -1

-1 -1 -1 -1 -1 -1 2 6 -1 -1 1 -1 0

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):2

Please select a router:

3

The routing table for router 3 is:

Destination: 1 2 3 4 5 6 7 8 9 10 11 12 13

Cost: 3 4 0 3 6 9 8 12 10 5 9 10 10

Next Router: 1 2 0 4 4 4 4 1 2 10 1 10 4

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):3

Please input the source and the destination router number:

3 13

The shortest path from 3 to 13 is

3--4--7--13,the total cost is 10

Test Case 4 :(Directed graph) 15 routers

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):1

Please load original routing table data file:

test2.txt

Original routing table is as follows:

0 6 3 7 -1 -1 8 -1 -1 -1 6 -1 -1 -1 -1

4 0 4 -1 -1 -1 -1 9 6 -1 -1 -1 -1 -1 -1

8 5 0 3 7 -1 -1 -1 -1 5 -1 -1 -1 -1 -1

7 -1 7 0 3 -1 5 -1 -1 -1 -1 -1 -1 -1 -1

-1 -1 7 8 0 3 -1 -1 -1 -1 -1 9 -1 -1 -1

-1 -1 -1 -1 9 0 8 -1 -1 -1 3 -1 -1 -1 -1

4 -1 -1 8 -1 7 0 6 -1 5 -1 -1 -1 -1 -1

-1 8 -1 -1 -1 -1 8 0 8 -1 -1 -1 -1 -1 -1

-1 3 -1 -1 -1 -1 -1 7 0 6 -1 -1 -1 -1 3

-1 -1 1 -1 -1 -1 6 -1 3 0 9 -1 -1 -1 8

2 -1 -1 -1 -1 8 -1 -1 -1 8 0 7 -1 3 -1

-1 -1 -1 -1 4 -1 -1 -1 -1 -1 8 0 3 -1 -1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 5 0 5 -1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 6 -1 3 0 1

-1 -1 -1 -1 -1 -1 -1 -1 9 6 -1 -1 -1 4 0

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):2

Please select a router:

10

The routing table for router 10 is:

Destination: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Cost: 9 6 1 4 7 10 6 10 3 0 9 16 13 10 6

Next Router: 3 3 3 3 3 3 7 9 9 0 11 3 9 9 9

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):3

Please input the source and the destination router number:

10 2

The shortest path from 10 to 2 is

10--3--2,the total cost is 6

Test Case 5 :(Undirected graph) 17 routers

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):1

Please load original routing table data file:

test4.txt

Original routing table is as follows:

0 6 3 7 -1 -1 8 -1 -1 -1 6 -1 -1 -1 -1 -1 6

4 0 4 -1 -1 -1 -1 9 6 -1 -1 -1 -1 -1 -1 2 2

8 5 0 3 7 -1 -1 -1 -1 5 -1 -1 -1 -1 -1 3 -1

7 -1 7 0 3 -1 5 -1 -1 -1 -1 -1 -1 -1 -1 5 -1

-1 -1 7 8 0 3 -1 -1 -1 -1 -1 9 -1 -1 -1 -1 -1

-1 -1 -1 -1 9 0 8 -1 -1 -1 3 -1 -1 -1 -1 -1 -1

4 -1 -1 8 -1 7 0 6 -1 5 -1 -1 -1 -1 -1 -1 2

-1 8 -1 -1 -1 -1 8 0 8 -1 -1 -1 -1 -1 -1 -1 6

-1 3 -1 -1 -1 -1 -1 7 0 6 -1 -1 -1 -1 3 -1 3

-1 -1 1 -1 -1 -1 6 -1 3 0 9 -1 -1 -1 8 -1 3

2 -1 -1 -1 -1 8 -1 -1 -1 8 0 7 -1 3 -1 6 -1

-1 -1 -1 -1 4 -1 -1 -1 -1 -1 8 0 3 -1 -1 -1 -1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 5 0 5 -1 -1 -1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 6 -1 3 0 1 -1 -1

-1 -1 -1 -1 -1 -1 -1 -1 9 6 -1 -1 -1 4 0 4 -1

-1 2 3 5 -1 -1 -1 -1 -1 -1 6 -1 -1 -1 4 0 -1

6 2 -1 -1 -1 -1 2 6 3 3 -1 -1 -1 -1 -1 -1 0

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):2

Please select a router:

15

The routing table for router 15 is:

Destination: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17

Cost:10 6 7 9 12 15 10 14 9 6 10 12 7 4 0 4 8

Next Router:16 16 16 16 16 16 16 16 9 10 14 14 14 14 0 16 16

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):3

Please input the source and the destination router number:

15 17

The shortest path from 15 to 17 is

15--16--2--17,the total cost is 8

Test Case 6 :(Undirected graph) 20 routers

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):1

Please load original routing table data file:

test5.txt

Original routing table is as follows:

0 6 3 7 -1 -1 8 -1 -1 -1 6 -1 -1 -1 -1 -1 6 9 -1 6

4 0 4 -1 -1 -1 -1 9 6 -1 -1 -1 -1 -1 -1 2 2 -1 1 2

8 5 0 3 7 -1 -1 -1 -1 5 -1 -1 -1 -1 -1 3 -1 5 -1 9

7 -1 7 0 3 -1 5 -1 -1 -1 -1 -1 -1 -1 -1 5 -1 -1 -1 -1

-1 -1 7 8 0 3 -1 -1 -1 -1 -1 9 -1 -1 -1 -1 -1 6 -1 6

-1 -1 -1 -1 9 0 8 -1 -1 -1 3 -1 -1 -1 -1 -1 -1 7 -1 -1

4 -1 -1 8 -1 7 0 6 -1 5 -1 -1 -1 -1 -1 -1 2 1 -1 7

-1 8 -1 -1 -1 -1 8 0 8 -1 -1 -1 -1 -1 -1 -1 6 3 -1 6

-1 3 -1 -1 -1 -1 -1 7 0 6 -1 -1 -1 -1 3 -1 3 -1 -1 2

-1 -1 1 -1 -1 -1 6 -1 3 0 9 -1 -1 -1 8 -1 3 2 -1 9

2 -1 -1 -1 -1 8 -1 -1 -1 8 0 7 -1 3 -1 6 -1 4 -1 -1

-1 -1 -1 -1 4 -1 -1 -1 -1 -1 8 0 3 -1 -1 -1 -1 -1 -1 6

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 5 0 5 -1 -1 -1 -1 -1 -1

-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 6 -1 3 0 1 -1 -1 -1 1 7

-1 -1 -1 -1 -1 -1 -1 -1 9 6 -1 -1 -1 4 0 4 -1 -1 3 -1

-1 2 3 5 -1 -1 -1 -1 -1 -1 6 -1 -1 -1 4 0 -1 -1 -1 -1

6 2 -1 -1 -1 -1 2 6 3 3 -1 -1 -1 -1 -1 -1 0 -1 2 -1

9 -1 5 -1 6 7 1 3 -1 2 4 -1 -1 -1 -1 -1 -1 0 4 -1

-1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 3 -1 2 4 0 -1

6 2 9 -1 6 -1 7 6 2 9 -1 6 -1 7 -1 -1 -1 -1 -1 0

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):2

Please select a router:

7

The routing table for router 7 is:

Destination: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Cost: 4 4 4 7 7 7 0 4 5 3 5 12 8 5 6 6 2 1 4 6

Next Router: 1 17 18 18 18 6 0 18 17 18 18 18 17 17 17 17 17 18 17 17

\*Implementation of Link-State Routing Protocol\*

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

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

0-Exit Program

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

Choose operation(Enter option number):3

Please input the source and the destination router number:

7 20

The shortest path from 7 to 20 is

7--17--2--20,the total cost is 6