CS-542 LINK STATE ROUTING SIMULATOR

***lINK sTATE routing simulation using “dijkshtra’s algorithm”***

Name: Vivek Sagar Sahu

CWID - A20375371

Course Title: Computer Networks-1

Fall 16

Date: 11/20/2016

***Contents***

1. Introduction to Link State Algorithm

- Phases

- Flooding

- Shortest Path

2. Types of Link State Algorithm

- OSPF

- IS IS

3. Dijkstra’s Algorithm

- Implementation

4. Link State Routing Simulator System Design

- Create Topology

- Connection Table

- Shortest Path

- Modification in Topology

- Best Router

5. Modification in topology

- Delete a Router

6. Test Cases

7. User Manual

8. Conclusion

9. Reference

Introduction

The basic concept of link-state routing is that every node creates a map of the connectivity to the network in the form of a graph, showing which nodes are connected to which other nodes. Each node then independently calculates the best logical path from it to every possible destination in the network. The collection of best paths will create a shortest path tree, which then forms the node’s routing table. Link state routing assumes that each node has partial knowledge: it knows the state (cost) of its links. In other words, the whole topology can be compiled from the partial knowledge of each node. In this way, it allows the topology to be dynamic.

Phases

In link state routing, four different tasks are required to ensure that each node has the routing table showing the least-cost node to every other node.

1. Creation of the states of the links by each node, called the link state packet or LSP.

2. Dissemination of LSP data to every other router, called flooding.

3. Formation of a shortest path tree for each node.

4. Calculation of a routing table based on the shortest path tree.

Flooding

**Reliable Flooding:** Tell all routers what you know about your local topology.

a. Each router transmits a Link State Packet (LSP) on all links

b. A neighboring router forwards out all links except incoming

c. Keep a copy locally; don’t forward previously-seen LSPs

Shortest Path

After receiving all LSPs, each node will have a copy of the whole topology. Dijkstra’s Algorithm finds the shortest path to every node using this topology data. The algorithm is explained in detail in next section.

Types of Link State Algorithm

Examples of link-state routing protocols include the following:

1. Open Shortest Path First (OSPF) and
2. intermediate system to intermediate system (IS-IS).

**OSPF**: The OSPF (Open Shortest Path First) protocol is one of a family of IP Routing protocols, and is an Interior Gateway Protocol (IGP) for the Internet, used to distribute IP routing information throughout a single Autonomous System (AS) in an IP network.

The OSPF protocol is a link-state routing protocol, which means that the routers exchange topology information with their nearest neighbors. The topology information is flooded throughout the AS, so that every router within the AS has a complete picture of the topology of the AS. This picture is then used to calculate end-to-end paths through the AS, normally using a variant of the Dijkstra algorithm. Therefore, in a link-state routing protocol, the next hop address to which data is forwarded is determined by choosing the best end-to-end path to the eventual destination.

**IS - IS**: The IS-IS (Intermediate System - Intermediate System) protocol is one of a family of IP Routing protocols, and is an Interior Gateway Protocol (IGP) for the Internet, used to distribute IP routing information throughout a single Autonomous System (AS) in an IP network.

IS-IS is a link-state routing protocol, which means that the routers exchange topology information with their nearest neighbors. The topology information is flooded throughout the AS, so that every router within the AS has a complete picture of the topology of the AS. This picture is then used to calculate end-to-end paths through the AS, normally using a variant of the Dijkstra algorithm. Therefore, in a link-state routing protocol, the next hop address to which data is forwarded is determined by choosing the best end-to-end path to the eventual destination.

Dijkstra’s Algorithm

**Steps implemented in the algorithm**

1. The program first asks for a network topology file. It validates the data and  
 store in matrix format.

2. The next step is to create the connection table. The program takes the source  
 router as input, and performs the Dijkstra Algorithm on it.

3. For Dijkstra Algorithm, the program follows below steps:

A. It selects the source node as the root of the tree and add it to the path.  
 Then it sets the shortest distances for all the neighbors of the root to the cost  
 between the root and those neighbors. Finally it sets the shortest distance of the root to zero.  
 B. Then it repeats the following two steps in loop until all nodes are added to the path:  
 a. It searches the for nodes which are not in the path. It then selects the one with minimum shortest distance and add it to the path.  
 b. It updates the shortest distance for all remaining nodes using the shortest distance of the node just moved to the path in previous step.

4. At every step, it keeps track of two type of nodes:

A. The interface used to go to next router. (For connection table.)  
 B. The parent node of last added node. (To create the final path.)

5. Once both connection table and parent table are ready, the shortest path is  
 found from given source to destination by following way :

A. Starting from the destination node, it follows the parent node from the parent  
 table to reach to the source, and provide the reverse path.  
 B. The total cost is found by adding the cost of all the nodes in previous step.

6. If there is no path from given source and destination, the program returns with  
 such message.

Dijkstra’s pseudocode

1 **function** Dijkstra(*Graph*, *source*):

2 create vertex set Q

3 **for each** vertex *v* in *Graph*: *// Initialization*

4 dist[*v*] ← INFINITY *// Unknown distance from source to v*

5 prev[*v*] ← UNDEFINED *// Previous node in optimal path from source*

6 add *v* to *Q* *// All nodes initially in Q (unvisited nodes)*

7 dist[*source*] ← 0 *// Distance from source to source*

8 **while** *Q* is not empty:

9 *u* ← vertex in *Q* with min dist[u] *// Source node will be selected first*

10 remove *u* from *Q*

11 **for each** neighbor *v* of *u*: *// where v is still in Q.*

12 *alt* ← dist[*u*] + length(*u*, *v*)

13 **if** *alt* < dist[*v*]: *// A shorter path to v has been found*

14 dist[*v*] ← *alt*

15 prev[*v*] ← *u*

16 **return** dist[], prev[]

17 *S* ← empty sequence

18 *u* ← *target*

19 **while** prev[*u*] is defined: *// Construct the shortest path with a stack S*

20 insert *u* at the beginning of *S* *// Push the vertex onto the stack*

21 *u* ← prev[*u*] *// Traverse from target to source*

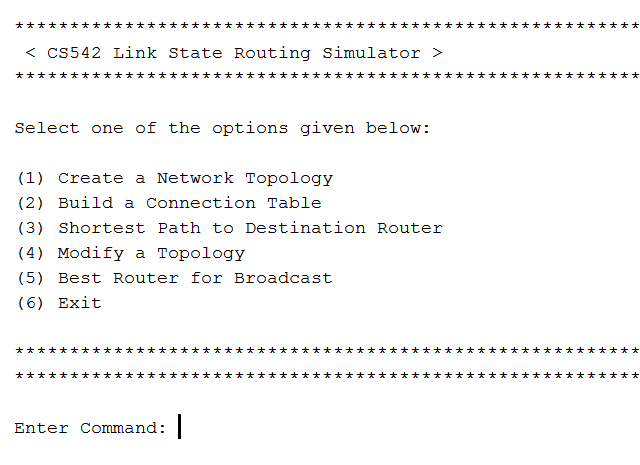
22 insert *u* at the beginning of *S* *// Push the source onto the stack*

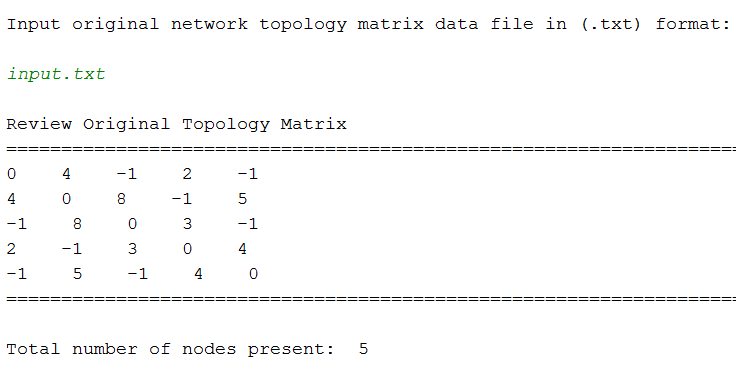
System Design

The document provides the basic system features in detail of how it works. As far as the software features are concerned it is very important that they are simple to use and display high performance in possibly all aspects from both users and the developer. Here are some main points which describe the system features effectively.

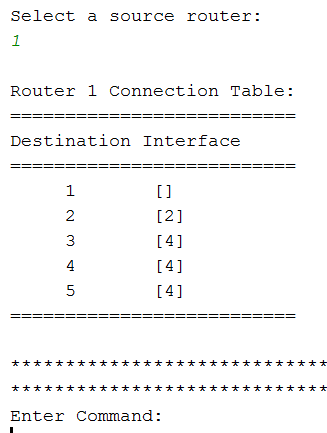
1. **User Defined Topology File:** The system provides its users an option to choose their owntopology as an input to the system. With help of user defined topology file, the system carries out all the required operations which provides user an efficient way to evaluate its own topology and get a result regarding the connection tables at each phase of the system as well as the shortest path from source to destination routers.
2. **Default Connection Table :** The system has been developed in such a way that it facilitatesthe user to the Default Connection Tables possessed by source router, which is involved in the topology. This option proves to be very useful if any user is in need of having a detailed view of the connection table of the source router.
3. **Modification of Default Topology:** The system gives its user an opportunity to modify theDefault Input Topology by asking to enter the source and destination router where it wants to delete a router in the topology. This option modifies the default topology and displays the modified topology connection table and then performs all the required operations to get the results of shortest distance between source and destination.
4. **Connection Table:** This feature has been implemented to check the Connection Table atthe phase of the source router. This option asks for the router number, for which the user wants to view individual connection table
5. **Finding Shortest Path:** The system has been implemented to find the shortest path betweenthe mentioned Source router to Destination router. This feature allows user to quickly view the shortest path present between the given input routers. System also demonstrates the route of reaching from source to destination using shortest path
6. **Efficient Navigation:** The system has been implemented in a such way that, it gives nooverheads to its user for navigating within the system flow and provides efficient access to all its features and results

Create Topology

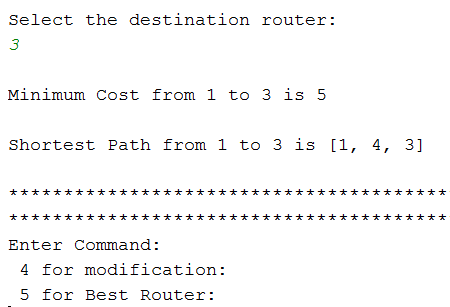




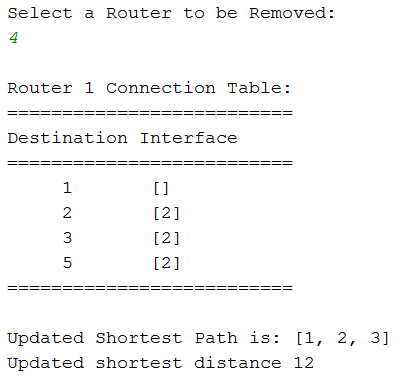
**Connection Table:**



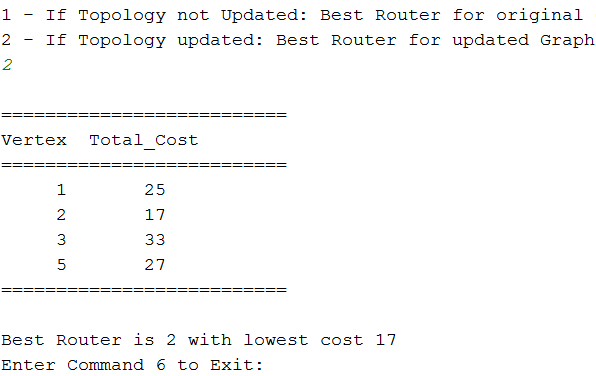
**Shortest Path:**



**Modification of Network Topology:**



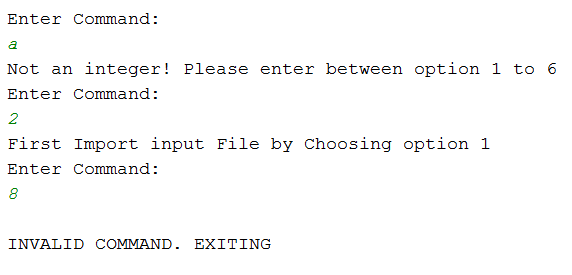
**Best Router for given network topology:**



Test Cases

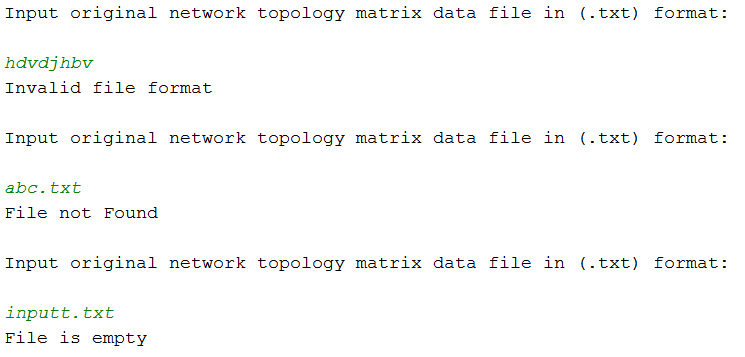
**Invalid Command:**

* If entered invalid command, it will display the error message, and prompt for it again.
* If entered command other then 1 to 6, it will display the error message, and exit the command prompt.

****

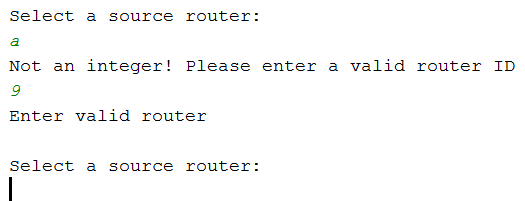
**Invalid File:**

* If invalid file format is entered, it will display the error message, and prompt for new file again.
* If invalid topology is entered, it will display the error message, and prompt for new file again.
* If empty topology file is entered, it will display the error message, and prompt for new file again.

****

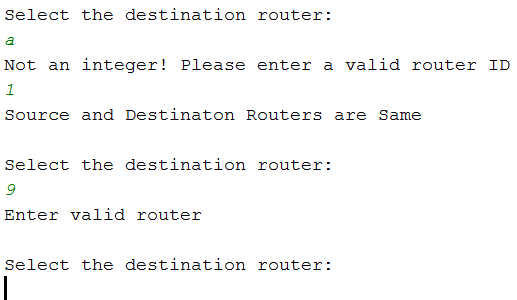
**Invalid Source Router:**

* If entered invalid source router, it will display the error message, and prompt for it again



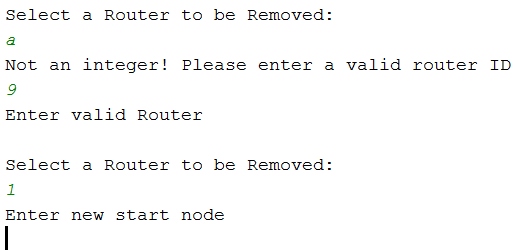
**Invalid Destination Router:**

* If entered invalid source router, it will display the error message, and prompt for it again.
* If both source router and destination router are same, it will display the error message, and prompt for it again



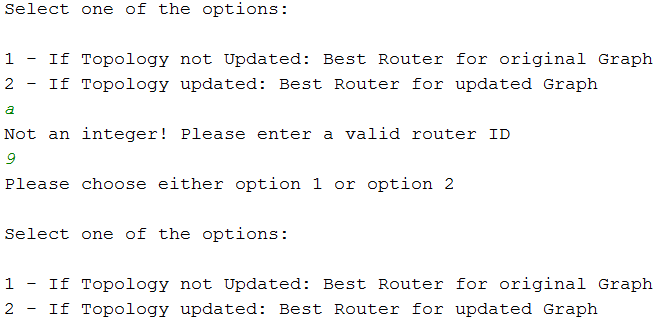
**Invalid Router for modification:**

* If entered invalid source router, it will display the error message, and prompt for it again.
* If entered router same as source or destination router already being used, then it will prompt for new start or destination router.



**Invalid Selection for Best Path:**

* If entered invalid option, it will display the error message, and prompt for it again.
* If entered valid option but unrelative to updated topology, it will display the error message, and prompt for it again.



User Manual

**Steps To Execute the Simulator:**

* As we cant run python 3.5 file as executable file .exe format we have to open it in python shell or pyCharm
* Open the source code in python shell or pyCharm.
* Add the input matrix file in the same directory as the source code.
* In the input file, all the elements should be separated by single space.
* Either right click and run as python shell or from the toolbar select Run the code.
* Enter the desired choice to get the required output.

Conclusion

* The implemented program for Link State Algorithm works for any network topology regardless of the size of network.
* With every node having partial information about the network topology, it can create shortest path tree for the network.
* Given valid topology data, it will provide you with the shortest path between source router and destination router.
* You can also delete any node including source and destination router already being chosen in beginning by user.
* It will give you the best router which has the lowest cost with every other routers in the given topology.

Reference

1. <https://en.wikipedia.org/wiki/Dijkstra's_algorithm>
2. <https://en.wikipedia.org/wiki/Link-state_routing_protocol>
3. <https://www.youtube.com/watch?v=gdmfOwyQlcI>
4. https://www.youtube.com/watch?v=2\_0AwfQWKUk