Implementation of Link-State Routing Protocol

CS 542 Term Project (Fall 2013)

1. Overview

Link-state routing protocol is a main class of routing protocols. It is performed by every router (switching node) in the network. The basic concept of link-state routing is that every node constructs a map of the connectivity to the network, 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.

2. Objective

In this term project, you are asked to develop a program to implement Link-State Routing Protocol. Your 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.

3. 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 your 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, your program should be able to output the routing table of any router, and output the optimal path between any two selected routers.

4. Sample Input

1) Sample Network Topology Diagram

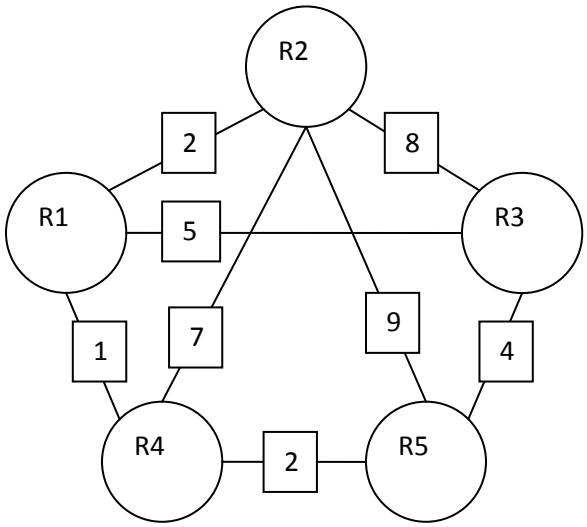

Figure: A sample network topology with costs on links

2) Sample Original Routing Table

Based on the above network topology diagram which is an undirected graph, we set the cost between a router and itself to 0, the cost between two indirectly connected routers to -1. Then we have the following original routing table.

	R1	R2	R3	R4	R5
R1	0	2	5	1	-1
R2	2	0	8	7	9
R3	5	8	0	-1	4
R4	1	7	-1	0	2
R5	-1	9	4	2	0

Table: A sample original routing table

3) Sample Input File (network.txt)

The above table will be given as an input in a text file form named ***network.txt***. Please notice that this is only a sample table and your program is required to be able to process a network with arbitrary number of routers.

5. Expected Program Menu

Program prompts in blue. User input in red.

After start to run your program, the following menu is expected to prompt:

1-Load File

2-Build Routing Table for Each Router

3-Out Optimal Path and Minimum Cost

Input: 1

Prompt: Please load original routing table data file:

Input: network.txt

Prompt:

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

(Automatically exit to menu)

Input: 2

Prompt: Please select a router:

Input: 1

Prompt: The routing table for router 1 is:

(The routing table for router 1 computed by your program)

(Automatically exit to menu)

Input: 3

Prompt: Please input the source and the destination router number:

Input: 1 5

Prompt: The shortest path from 1 to 5 is 1-?-…-?-5, the total cost is (value)?

(Automatically exit to menu)

6. Deliverables and Grading Policy

1. (5%) Inline documented source code (attach the file and include its printout in the project report).

2. (20%) Design and test report. The design of your program, what you did to convince yourself that the program works correctly, briefly explain link-state routing protocol. The test report for input files must be given in the project report. Also, you are required to specify at least five instances (10 routers at least) you developed additional and report the corresponding results. TA will test your program with another file (in the same format as network.txt) which won't be released to you until the demo.

3. (10%) Detailed description of your algorithm applied to find the shortest path between two selected routers.

4. (5%) Detailed instructions how to compile and run your program.

5. (30%) Results output by your program during demo.

6. (30%) TA will test your program with his own data file not released to the students. Your program should be able to process the input matrix regardless of its size when you show a demo.

7. Submission Guidelines

- **Due date: Monday, Nov. 25, noon**
- Late submissions will **not** be accepted.
- A five minutes demo is required on the due date (will be scheduled later on)
- A maximum of **three** students per team, you may work individually if you like.
- Only one submission per team.
- The instructor and TA are not responsible for debugging your code.
- Your submission should consist of two items only, a project report and the source code (Java or C/C++ only) of your program. Do not submit 100 documents/files/attachments. Such submissions will be disregarded.
- The live sections and all the Internet students registered on the Main Campus, please submit a hard copy of your project report, a printout of your source code to TA's mailbox located in the CS department office. TA is responsible for grading the project.
- Students not in the live section, put your project report and the source code into one zip file and upload it to "the digital drop box" on the Blackboard. Please do not email your files to the instructor or TA. The name of your zip file should follow this format: CS542_Lastname_Firstname_Project.zip (Again, one submission per team only).
- Please put down your name (Last, First, Middle) student ID and section ID you are enrolled in (section number or simply Live, Internet, Bangalore) on the front page of the project report.
- If you have any project related questions/doubts, please contact TA directly (cma3@iit.edu, xiayuhuahxy@gmail.com).