CS-542 Project

The Distance Vector (DV) Routing Algorithm

- **Due date: Monday, Dec. 8, 1 pm**

1. Objective

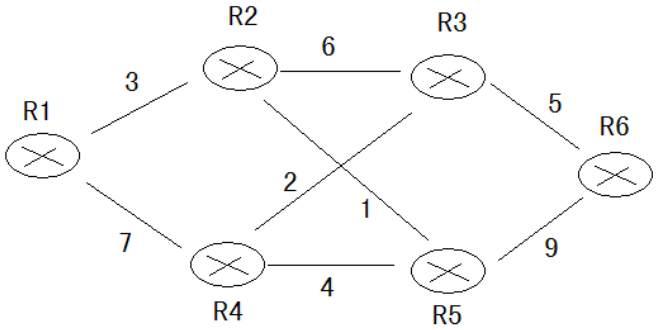
The objective of this project is to implement the DV routing algorithm. In order to make sure that the TA will be able to compile and execute your program, write your code in C/C++ or Java. Your program should take a text file (specified below) describing the network topology as an input. Your results should be given in the project report.

2. Algorithm overview

The detailed description of the DV algorithm is given in the posted accompanying document (Section 11.3 of the textbook by Forouzan). Please read it first. The network topology is given as a matrix $\boldsymbol{M}^{(0)}$ (0 denotes the initial conditions) representing an undirected graph. An element $\boldsymbol{M}^{(0)}[i][j]$ of this matrix is a cost of a link connecting the node i with its direct neighbor j. We assume that $\boldsymbol{M}^{(0)}[i][i]=0$ and $\boldsymbol{M}^{(0)}[i][j]$ is identical to $\boldsymbol{M}^{(0)}[j][i]$ (the graph is undirected). If the node b is not a direct neighbor of a, $\boldsymbol{M}^{(0)}[a][b]$ is equal to infinity, which we denote by 999, i.e. $\boldsymbol{M}^{(0)}[a][b]=999$. The matrix $\boldsymbol{M}$ is updated in the following way. Assume that the matrix $\boldsymbol{M}^{(k-1)}$ is given. Its i-th row is the distance vector for the router R_i, which can be sent to its neighboring routers. For every source-destination pair (i,j) a new current cost of the path from i to j is determined based on $\boldsymbol{M}^{(k-1)}$ and stored in the entry $\boldsymbol{M}^{(k)}[i][j]$, i.e. $\boldsymbol{M}^{(k)}$ is the updated cost matrix after the k-th iteration. When n iterations are completed we obtain the final matrix $\boldsymbol{M}^{(n)}$ whose element $\boldsymbol{M}^{(n)}[i][j]$ is the length of the shortest path from i to j.

3. Input data

3.1. Network topology diagram



3.2. The initial matrix $\boldsymbol{M}^{(0)}$ reflecting the network topology

	R1	R2	R3	R4	R5	R6
R1	0	3	999	7	999	999
R2	3	0	6	999	1	999
R3	999	6	0	2	999	5
R4	7	999	2	0	4	999
R5	999	1	999	4	0	9
R6	999	999	5	999	9	0

4. Requirements

Your program should produce the following results for the given data set *datafile1.txt* specifying the initial matrix $\boldsymbol{M}^{(0)}$:

- (1) The number of iterations *n* required to obtain the final matrix $\boldsymbol{M}^{(n)}$. How do you know that *n* is the final iteration?
- (2) Find the final matrix $\boldsymbol{M}^{(n)}$.
- (3) List the shortest paths for all the source-destination pairs *(i,j)*, i.e. *i*->2->5->6->*j*. Assume *i*<*j*. Note that if the shortest path for e.g. *(2,4)* is listed, there is no need to list it again for *(4,2)*. They are identical because the graph is undirected.
- (4) Simulate a failure of the router *R2*, i.e. in $\boldsymbol{M}^{(0)}$ set all the costs between *R2* and the remaining routers to 999. This specifies the new initial matrix $\boldsymbol{F}^{(0)}$.
- (5) Find the final matrix $\boldsymbol{F}^{(n)}$. What is the value of *n* ?
- (6) List the shortest paths for all the source-destination pairs *(i,j)* for $\boldsymbol{F}^{(n)}$ (cf. item (3) above).

Intermediate nodes of the shortest path [see items (3) and (6) above] can be determined as a by-product of the DV algorithm. How did you find these nodes? Describe your algorithm. (There are many algorithms for finding the shortest path in a graph - **do NOT implement any of them!** Again, your algorithm should be a by-product of the Distance Vector Routing Algorithm).

Warning. *There is an error in line 29 of the pseudocode of the DV algorithm given on page 288 in the textbook by Forouzan (see the scanned accompanying document). Finding this error is also a part of the project.*

5. Sample run of the program

Enter the input file name: (program prompt)

datafile1.txt (user input)

The initial matrix is as follows:

0	3	999	7	999	999
3	0	6	999	1	999
999	6	0	2	999	5
7	999	2	0	4	999
999	1	999	4	0	9
999	999	5	999	9	0

The required number of iterations n=?

(The symbol "?" should be replaced with a numerical value computed by your program.)

The final matrix computed by the DV algorithm is as follows:

?	?	?	?	?	?
?	?	?	?	?	?
?	?	?	?	?	?
?	?	?	?	?	?
?	?	?	?	?	?
?	?	?	?	?	?

Enter the source and destination nodes:

6 2

The shortest path from node 6 to node 2 is as follows:

6 ->? ->? ->?->2

The length of this path is ?

Another source-destination pair?

Yes

Enter the source and destination nodes:

1 3

The shortest path from node 1 to node 3 is as follows

1 ->? ->?->?->3

The length of this path is ?

Another source-destination pair?

No

Enter the number of the router whose failure is to be simulated:

X

The final matrix computed by the DV algorithm for the network with the Router X failure is as follows:

? ? ? ? ? ?

? ? ? ? ? ?

? ? ? ? ? ?

? ? ? ? ? ?

? ? ? ? ? ?

? ? ? ? ? ?

Enter the source and destination nodes:

6 2

The current shortest path from node 6 to node 2 is as follows:

6 ->? ->? ->?->2

The current length of this path is ?

Another source-destination pair?

Yes

Enter the source and destination nodes:

1 3

The current shortest path from node 1 to node 3 is as follows

1 ->? ->?->?->3

The current length of this path is ?

Another source-destination pair?

No

6. **Deliverables and grading policy**

1. (4%) Inline documented source code (attach the file and include its printout in the project report).

2. (5%) Design and test report (the design of your program, what you did to convince yourself that the program works correctly, briefly explain the DV algorithm).

3. (10%) Detailed description of your algorithm applied to find intermediate nodes of the shortest path (cf. problems (3) and (6) of Section 4).

4. (1%) Instructions how to compile and run your program. **If the TA can't compile/run your program, your score will be reduced by 80%.**

5. Problem report: if your code does not work properly you should report this.

6. (40%) Report your results (see problems (1)-(6) of Section 4).

7. (40%) The 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 if its size.

7. Submission guidelines

- Due date: Monday, Dec. 8, 1 pm
- Weight of this project is 20%.
- Max. 3 students/team. You may work individually if you like.
- Only one submission per team. Do NOT submit multiple copies of your project. It's very confusing.
- The instructor and TA are not responsible for debugging your code.
- Your submission should consist of two items ONLY: your project report (with a printout of your source code) and a file with the source code of your program. Do NOT submit 100 documents/files/attachments. Such submissions will be disregarded.
- Please submit a hard copy of your project report, a printout of your source code and a CD with your source code directly to my TA Xin Liu. His mailbox is in the CS Dept. office. Xin is responsible for grading this project.
- Please put down your name (LAST, FIRST, MIDDLE - THE ORDER DOES MATTER) and student ID on the front page of your project report.