Assignment 1

CS 544: Topics in Networks

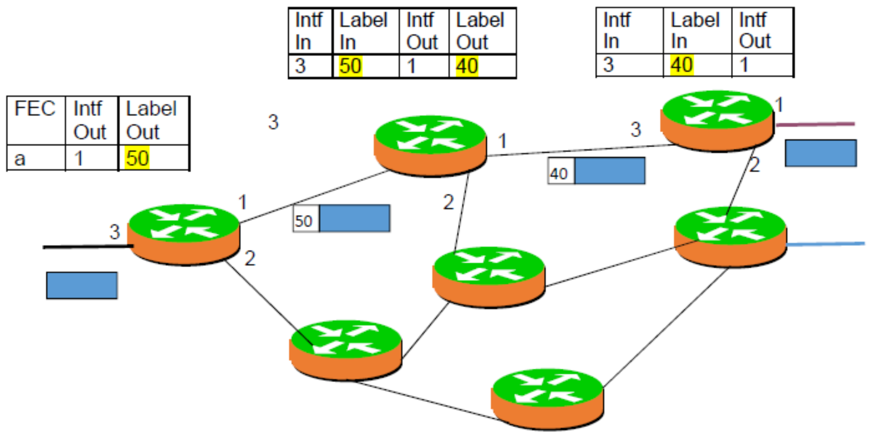
Things to note before you start:

- No extensions in submission are allowed. Delay in submission will lead to penalty in marks.
- The programs can be written in C/C++/Java.
- Assignments submitted before the deadline will only be considered for evaluation.
- Submissions should be done via Moodle only. Please do not email your assignments separately to the TAs, it will not be considered for evaluation.
- Your code will be checked for plagiarism. Any kind of academic dishonesty, plagiarism, etc. will lead to penalties.
- NO sharing of code between students, submission of downloaded code (from the Internet, Campus LAN, or anywhere else) is allowed.
- The first instance of code copying will result in ZERO marks for the assignment.
- The second instance of code copying will result in a `F' grade. Students may also be reported to the Students Disciplinary Committee, which can impose additional penalties.
- Please protect your Moodle account password. Do not share it with ANYONE, including your teammate. Do not share your academic disk drive space on Campus LAN.

Please read the questions carefully.

1. The purpose of this part of the assignment is to understand link-disjoint routing algorithms and MPLS.

MPLS OPERATION



FEC	Intf Out	Label Out
a	1	50

Intf In	Label In	Intf Out	Label Out
3	50	1	40

Intf In	Label In	Intf Out
3	40	1

Overview

- In this assignment you will be given topology file. By using shortest path and link disjoint concept create the routing table. In the routing table for each node to every other node, there will be 1st shortest distance and 2nd shortest distance. (Hint: all pair shortest path or apply Dijkstra for every node).
- After that create the label forwarding information base for each node using the routing table.
- Then using the optimistic or pessimistic approach create connection between nodes. A connection request file (Arpanet for 24 node and Nsfnet for 14 nodes) is given in which the connection request between 2 nodes and the required bandwidth is mentioned.
- All the output files format is given below.

Inputs

The command line will specify the following:

- topology of the network
- connections between nodes
- routing tables
- forwarding table
- path taken from one node to another

For e.g., the input can be given as below:

```
% ./-routing -top topology file -conn connectionsfile -rt routingtablefile -ft forwardingtablefile -path pathsfile - flag hopjdist - p 0|1.
```

- The topology file contains on the first line: NodeCount (N), Edge- count (E) of the network. Nodes are numbered from 0 through N-1.

On each subsequent line, there are four numbers - the first two integers represent the two endpoints of a bi-directional link, the third integer denote the link cost, and the fourth integer denotes the link's capacity (in Mbps).

- The connections file contains on the first line: Number of Connection

Requests (R). Connections are numbered from 0 through R - 1.

On each subsequent line, there are 5 numbers - the first two integers represent the source and destination node of a unidirectional connection, the remaining 3 integers denote the connection's stated capacity (in Mbps). The stated capacity of a connection request i specifies the requested bandwidth using three integers: (b_i^{min} ; b_i^{ave} ; b_i^{max}), which specify the minimum, average, and maximum bandwidth needed, respectively. Please note that there may be several connections between the same source-destination pair.

Routing

The program will first determine two shortest cost (not necessarily link-disjoint) paths for all node-pairs. You may use either hop or distance metric. The command line parameter will specify the choice. For hop, the link cost will be 1 and for distance third integer in the input file can be taken.

Connections

The program will then process the specified set of connection requests.

Optimistic Approach: This is used if -p command-line argument has value 0. Let C_l denote the total capacity of a link. Let $b_i^{equiv} = \min[b_i^{max}, b_i^{ave} + 0.25*(b_i^{max} - b_i^{min})]$. A connection is admitted if the following condition is met, along each link of the path selected for connection i:

$$b_i^{equiv} \leq \left(C_l - \sum_{j=1}^{n} b_i^{equiv} \right)$$

where n denotes the number of existing connections sharing a given link (along the path selected for the given connection) and j denotes the index of a connection assigned on this link.

Next, for each source-destination pair, select the two link-disjoint paths. If adequate bandwidth is not available along the first shortest-cost path, then the network attempts to set up the connection on the second shortest-cost path. If this also fails, then the connection is NOT admitted, i.e., it fails to meet the admission control test. Once an available path is identified, the connection will be set up along the path. This primarily involves setting up link-unique label id for a given connection request along all links of the path, and updating the corresponding forwarding tables at each intermediate node along the path.

Pessimistic Approach: If —p value in command-line argument is 1 then use this approach. Let C_l denote the total capacity of link l. A connection is admitted if the following condition is met, along each link of the path selected for connection i:

$$b_i^{max} \leq \left(C_l - \sum_{j=1}^{n} b_j^{max} \right)$$

where j denotes the index of a connection assigned on a given link along the path selected for the given connection. This gets repeated for the pessimistic approach.

Outputs

The **routing table** file will contain the routing table information for all the nodes. For each network node, the corresponding routing table (i.e. two link-disjoint paths from the given node to all other nodes) displayed with the following fields:

Destination Node	Path	Path Cost

Table 1: routingtablefile format

The **forwardingtable** file will contain the forwarding table information for all the nodes. For each network node, the corresponding forwarding table (for all established connections) will have following format:

Interface in	Label in	Interface out	Label out

Table 2: Label Information forwarding base format

paths will contain only one line consisting of two integers which represent the total number of requested connections and the total number of admitted connections, respectively.

For each connection that is admitted into the network, the output format is as follows:

Connection Id	Source	Destination	Label List	Path Cost

Table 3: Output format for each connection. Sort it based on connection ID.

Also calculate number of successful connections.

Grading

- *Routing setup: 15 points*
- *Connections setup: 25 points*
- *Output and viva voce: 10 points*

NOTE: Two example topology files are given. You may additionally use your own topologies. For the 14-node NSFNET and 24-node ARPANET topologies, compare the blocking probability for your own set of link capacities/traffic requests. Example files for 100 requests are given. Evaluation will be done on a different set of input files.

2.

In India migrant workers without proper identity increases day by day. So, the government has decided to have strict checking on border areas. To do so, a team of officers is assigned to do such a task. The arrival of each worker is considered as an event. Once the checking is done, the workers are free to roam anywhere inside India.

Case A:

To do the checking 2 special border officers are appointed in one check-post. Workers have to wait in separate queues before being checked by the two officers. Workers can join any of the two queues and get checked by the officers with equal probability. They are checked on FCFS basis by each officer.
The arrival pattern of the workers at each queue follows an exponential distribution with a mean service rate of λ_1. The service time taken by officers follows an exponential distribution with a mean service rate of μ_1. Simulate the given scenario with your own inputs and generate the following outputs for each queue:

a) Average number of workers getting checked.
b) Average response time for workers in getting checked.
c) Average time for which a worker has to wait until getting checked.
d) Average number of workers waiting in the queue before each officer.

You have to show the above in the console as well as in a file. Ensure that the program works for any input and/or output, verifies different test cases, and also throws exceptions, wherever needed.

Case B:

In case less number of workers arrive, only one queue is made. However, the number of serving officers remains the same. Workers can go to any of the two officers with equal probability. Any number of workers can stand in the queue but they are served on an FCFS basis. The arrival of workers follows exponential distribution with a mean arrival rate of λ_2. The service provided by officers follows exponential distribution in service time with a mean service rate of μ_2.
Simulate the above-given scenario with your own inputs and generate the following outputs:

a) Average number of workers getting checked.
b) Average response time for workers in getting checked.
c) Average time for which a worker has to wait until getting checked.
d) Average number of workers waiting in the queue before each officer.

You have to show the above in the console as well as in file. Ensure that the program works for any input and output, verifies different test cases, and also throws exceptions, wherever needed.

Case C:

For aged/unwell/pregnant workers, special counters are made with chairs. Here too, two officers are there and each officer's counter has 5 chairs. The workers join one of the queues (of chairs) with equal probability and are served on an FCFS basis. The probability of such an event happening is 0.02. The arrival of workers at each counter follows exponential distribution with a mean arrival rate of λ_3. The service provided by each of the officers follows an exponential distribution for the service time with a mean service rate of μ_3. Simulate the given scenario with your own inputs and generate the following outputs:

a) The average number of workers getting checked
b) Average response time for workers in getting checked.
c) Average time for which a worker has to wait until getting checked.
d) The average number of workers waiting in the queue before each officer

Grading

- *10x3 points*
- *Output and viva voce: 20 points*