DEPARTAMENTO DE ELECTRÓNICA, TELECOMUNICAÇÕES E INFORMÁTICA

MESTRADO INTEGRADO EM ENG. DE COMPUTADORES E TELEMÁTICA

ANO 2017/2018

DESEMPENHO E DIMENSIONAMENTO DE REDES

ASSIGNMENT GUIDE NO. 4

TRAFFIC ENGINEERING OF PACKET SWITCHED NETWORKS

Assignment Guide No. 4

1. Preamble

The aim of this assignment is to address the traffic engineering of an ISP (Internet Service Provider) core network based on MPLS (Multi-Protocol Label Switching). For a given network and a given set of estimated traffic flows to be supported, the traffic engineering task addressed in this assignment is to select a routing path for the LSP (Label Switched Path) of each traffic flow such that the performance of the network is optimized. The network performance is assessed by the Kleinrock approximation.

KLEINROCK APPROXIMATION:

Each network link is a $M/M/1$ queuing system. Consider a network composed by a set of unidirectional links (i,j), each one with a capacity μ_{ij} (in packets/second) and a propagation delay d_{ij} (in seconds). The network supports S packet flows $s = 1, \ldots, S$, each one with a packet arrival rate λ_s (in packets/second) and a routing path composed by the links (i,j) defined in set R_s. The total arrival rate on connection (i,j) is:

$$\lambda_{ij} = \sum_{s:(i,j)\in R_s} \lambda_s$$

and the total traffic supported by the network is:

$$\gamma = \sum_{s=1..S} \lambda_s$$

Then, by the Kleinrock approximation, the network average delay is:

$$W = \frac{1}{\gamma} \sum_{(i,j)} \left(\frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}} + \lambda_{ij} d_{ij} \right)$$

and the average delay of each flow s is:

$$W_s = \sum_{(i,j)\in R_s} \left(\frac{1}{\mu_{ij} - \lambda_{ij}} + d_{ij} \right)$$

2. Assignment

Consider an MPLS (Multi-Protocol Label Switching) network of an ISP (Internet Service Provider) with the topology presented in Figure 1 where the gray nodes are transit nodes (they just provide connectivity between the other nodes). The thick connections in Figure 1 have a capacity of 10 Gbps and the other connections have a capacity of 1 Gbps. In Annex I, the R matrix defines, in MATLAB format, the network topology shown in Figure 1 where element $R(i,j)$ gives the capacity (in Gbps) of the connection from node i to node j. Consider that, in all connections, the propagation delay is given by the fiber light speed (2×10^8 meters/second). In Annex I, matrix L defines, in MATLAB format, the connection lengths (in kilometers) where element $L(i,j)$ indicates the length of the connection from node i to node j.

Consider that the network must support an estimated set of traffic flows. In Annex I, matrix T defines the estimated traffic flows in the following way: row i defines traffic flow i with origin $T(i,1)$, destination $T(i,2)$, bandwidths (in Mbps) from origin to destination $T(i,3)$ and from destination to node $T(i,4)$. Assume that in all traffic flows, the packet arrivals are Poisson processes and the packet sizes are exponentially distributed with an average size of 1000 Bytes.

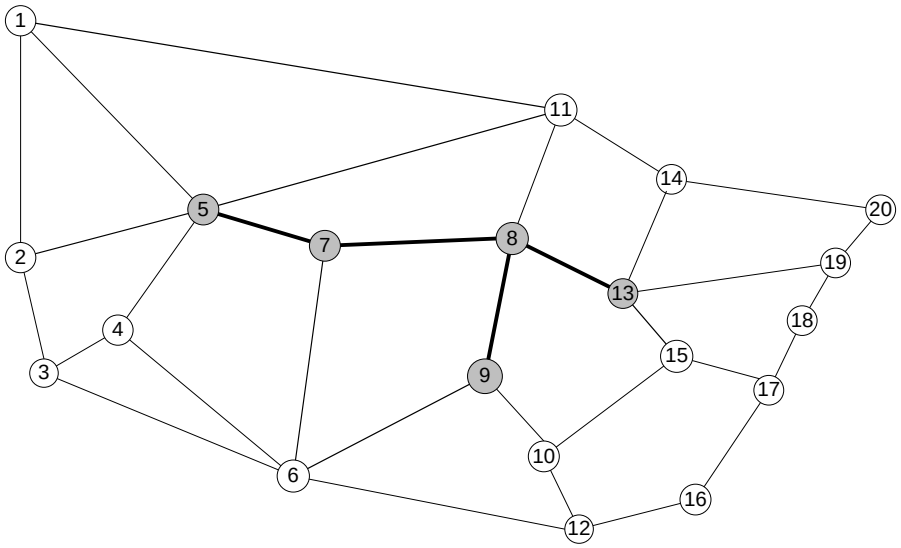

Figure 1: Topology of an ISP Network

Consider that the ISP requires each traffic flow to be routed through a single LSP (Label Switched Path). The LSPs between the same nodes are required to be symmetrical (*i.e.*, the network connections of an LSP from node *i* to node *j* must be the same network connections of the LSP from node *j* to node *i*).

For the implementation of this assignment, a MATLAB function is provided (file `ShortestPathSym.m`) that, for a given pair of nodes, determines the routing path that minimizes the sum of the connection costs in both directions between them. The function sintax is:

```
function [route]= ShortestPathSym(costs,node1,node2)
```

Input parameters:

`costs` – a square matrix of 20×20 elements where element `costs(i,j)` defines the cost of the connection from node *i* to node *j*;
`node1` – the first end node;
`node2` – the second end node.

NOTES:
- the cost values must be non-negative;
- the function ignores the values `costs(i,j)` for pairs of nodes such that there is no connection in the ISP network (as defined in Figure 1).

Output parameter:

`route` – a row array of 20 elements with the sequence of nodes in the routing path from `node1` up to `node2` (the remaining elements are zero).

Error codes:

When the output parameter `route` is a single element, its content indicates the following error in the input parameters:

`-1` : the matrix `costs` is not a square matrix of 20×20 elements
`-2` : the `node1` and/or `node2` are either equal or invalid
`-3` : at least one cost value is negative

a) Assume that the routing path of each flow is selected by the shortest path given by the sum of the lengths of the connections. Consider this solution as solution A. In Annex II, a MATLAB script is provided that computes solution A. The provided script also computes the average connection load in variable `AverageLoad`) and the maximum connection load in variable `MaximumLoad` (note that the connection load is the bandwidth supported by the connection divided by its capacity and each connection has two load values, one on each of its directions). Analyze the code and determine the average and maximum connection load of solution A.

b) Add to the previous MATLAB script an appropriate code to determine, based on the Kleinrock approximation: (i) the network average round-trip delay and (iii) the maximum average round-trip time among all flows. Determine these values of solution A.

c) Add to the previous MATLAB script an appropriate code to plot in decreasing order: (i) the average packet round-trip delay of each flow and (ii) the load of each direction of each connection. Register the obtained plots.

d) Change your previous MATLAB script to compute a different solution in the following way. Start by considering the network empty. Then, for each flow, compute as its routing path the one that minimizes the sum of the connection loads that resulted from the previous selected routing paths. Consider this solution as solution B. Repeat **a)**, **b)** and **c)** for solution B.

e) Change your previous MATLAB script to compute another solution in the following way. Start by considering the network empty. Then, for each flow, select as its routing path the one that has the minimum average round-trip delay resulted from the previous selected routing paths (use the M/M/1 assumption for the delay of each connection). Consider this solution as solution C. Repeat **a)**, **b)** and **c)** for solution C.

f) Compare solutions A, B and C (both the performance values and the obtained plots) and draw conclusions on which solution is better and in what cases.

g) Develop a multi-start local search algorithm that ends when the best solution does not improve in n consecutive cycles (n is an input parameter). The algorithm's objective is to find a solution with the lowest average round-trip delay and, among all such solutions, the one with the lowest maximum average round-trip time among all flows. Run the algorithm with $n = 20$ and consider this solution as solution D. Repeat **a)**, **b)** and **c)** for solution D.

h) Adapt the previous algorithm with the objective to find a solution with the lowest maximum connection load and, among all such solutions, the one with the lowest average round-trip delay. Run the algorithm with $n = 20$ and consider this solution as solution E. Repeat **a)**, **b)** and **c)** for solution E.

i) Compare solutions D and E (both the performance values and the obtained plots) and draw conclusions on which solution is, in your opinion, the one that the ISP should adopt on its network. Explain the arguments that support your opinion.

ANNEX I – PROBLEM DATA MATRICES

```
R= [0  1  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0
    1  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
    0  1  0  1  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
    0  0  1  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
    1  1  0  1  0  0  10 0  0  0  1  0  0  0  0  0  0  0  0  0
    0  0  1  1  0  0  1  0  1  0  0  1  0  0  0  0  0  0  0  0
    0  0  0  0  10 1  0  10 0  0  0  0  0  0  0  0  0  0  0  0
    0  0  0  0  0  0  10 0  10 0  1  0  10 0  0  0  0  0  0  0
    0  0  0  0  0  1  0  10 0  1  0  0  0  0  0  0  0  0  0  0
    0  0  0  0  0  0  0  0  1  0  0  1  0  0  1  0  0  0  0  0
    1  0  0  0  1  0  0  1  0  0  0  0  0  1  0  0  0  0  0  0
    0  0  0  0  0  1  0  0  0  1  0  0  0  0  1  0  0  0  0  0
    0  0  0  0  0  0  0  0  10 0  0  0  0  0  1  1  0  0  0  1  0
    0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0  0  0  0  0  1
    0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  1  0  0  0
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  0  1  0  0
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  1  0
    0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  1  0];

L=[0   119 0   0   147 0   0   0   0   0   292 0   0   0   0   0   0   0   0   0
   119 0   61  0   105 0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   61  0   40  0   136 0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   40  0   83  116 0   0   0   0   0   0   0   0   0   0   0   0   0   0
   147 105 0   83  0   0   65  0   0   0   181 0   0   0   0   0   0   0   0   0
   0   0   136 116 0   0   120 0   136 0   0   157 0   0   0   0   0   0   0   0
   0   0   0   0   65  120 0   95  0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   95  0   71  0   80  0   67  0   0   0   0   0   0   0
   0   0   0   0   0   136 0   71  0   51  0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   51  0   0   32  0   0   87  0   0   0   0   0
   292 0   0   0   181 0   0   80  0   0   0   0   0   73  0   0   0   0   0   0
   0   0   0   0   0   157 0   0   0   32  0   0   0   0   0   61  0   0   0   0
   0   0   0   0   0   0   0   67  0   0   0   0   0   68  40  0   0   0   101 0
   0   0   0   0   0   0   0   0   0   0   73  0   68  0   0   0   0   0   0   103
   0   0   0   0   0   0   0   0   0   87  0   0   40  0   0   0   45  0   0   0
   0   0   0   0   0   0   0   0   0   0   0   61  0   0   0   0   81  0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   45  81  0   52  0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   52  0   42  0
   0   0   0   0   0   0   0   0   0   0   0   0   101 0   0   0   0   42  0   46
   0   0   0   0   0   0   0   0   0   0   0   0   0   103 0   0   0   0   46  0];

%T(i,1) - node1
%T(i,2) - node2
%T(i,3) - Mbps from node1 to node2
%T(i,4) - Mbps from node2 to node1

T=  [1     2    32    60
     1     3    78    82
     1     4    23    18
     1     6    60    72
     1    10    13    17
     1    11    14    12
     1    12    15    14
     1    14    55    54
     1    15    17    16
     1    16    14    14
     1    17    12    21
     1    18    15    14
     1    19    10    13
     1    20    12    11
     2     3    80    84
     .     .     .     .
     .     .     .     .
     .     .     .     .
    18    19    14    32
    18    20    15    25
    19    20    24    14];
```

ANNEX II – MATLAB SCRIPT FOR A)

```
Matrizes;
miu= R*1e9/(8*1000);
NumberLinks= sum(sum(R>0));
T(:,3:4)= T(:,3:4)*1e6/(8*1000);
gama= sum(sum(T(:,3:4)));
d= L*1e3/2e8;
nT= size(T,1);
lambda= zeros(20);
routes= zeros(nT,20);
for i=1:nT
    origin= T(i,1);
    destination= T(i,2);
    lambda_od= T(i,3);
    lambda_do= T(i,4);
    r= ShortestPathSym(d,origin,destination);
    routes(i,:)= r;
    j= 1;
    while r(j)~= destination
        lambda(r(j),r(j+1))= lambda(r(j),r(j+1)) + lambda_od;
        lambda(r(j+1),r(j))= lambda(r(j+1),r(j)) + lambda_do;
        j= j+1;
    end
end
Load= lambda./miu;
Load(isnan(Load))= 0;
MaximumLoad= max(max(Load))
AverageLoad= sum(sum(Load))/NumberLinks
```