CS201c: Programming Evaluation 5
Shortest paths and Dijkstra's algorithm
Due date: Monday, Nov 18 by 11:59 pm

Problem Description:

You are given a computer network, which is modeled as an undirected graph G(V,E). G has |V|=n vertices and |E|=m edges, Each edge e of G has two attributes:

(i) color (either red or blue), and
(ii) p, 0<=p<1, the probability of link (edge) failure.

Further, there is a source node (computer) A and a destination node (computer) B in graph G.

Suppose a path P in G consists of the sequence of edges e_1, e_2, …, e_l. Then, *success probability* s(P) of the path equals:

s(P) = (1-p_1)*(1-p_2)*...*(1-p_l)

, where p_1, p_2, …, p_l are link failure probabilities for edges e_1, e_2, …, e_l respectively.

[A practical interpretation of success probability is as follows. In the real world, it is possible that some network links (or edges) in the communication path fail. If p_i is the probability that link e_i will fail, then s(P) is the probability that a packet sent from the start vertex of P will reach the end vertex of P safely. A large p_i means the link is less reliable, and vice versa.]

The *cost* of path P is equal to the number of red edges in P.

[A practical interpretation of cost is as follows. Blue edges belong to A's network provider, whereas the red edges belong to a different network provider. When A communicates with B along a path P, it has to pay extra (say 1 Rupee per red link used) for using the services of the different network provider.]

Your objective is to answer the following:

Given: An undirected graph G(V,E), real number 0<g<=1, and an integer k.

Output. Print `Yes' if there exists a path P from A to B in graph G satisfying the following two conditions: (i) P has at most k red edges, and (ii) success probability of P is at least g. Otherwise, print `No'.

Requirements.

1. *Running time.* The running time of your algorithm should be O(k * (n+m) * log (n)).

2. You cannot use any in-built libraries (including standard template library). All data structures should be implemented in C++ from scratch.

3. Collaboration is not permitted on this assignment. Your submitted code should be completely your own.

See section titled ``Honor Code" in course outline already shared with you.

Input and Output Format.

Input Format.

The first line contains the real number g.
The second line contains a single integer k.

The third line contains two integers n and m, where n is the number of vertices in G and m is the number of edges in G.

After this there are m more lines, with one line for each edge of G. A line with the format

"n1 n2 p c"

specifies that there is an edge between vertices n1 and n2 with failure probability p and color c.

[In the above, n1 and n2, with n1 < n2, are two distinct integers between 1 and n. p is a real number between 0 and 1. c is either 0 (blue edge) or 1 (red edge).]

You can assume that all edges are distinct.

All real numbers in the input are specified to two decimal places. Our convention is that node 1 is A and node n is B.

Output Format.

The output is either `Yes' or `No' on a line by itself..

Example Input and Output.

Input: (the graph is shown in figure on page 5 below)

2.43
4
16
20
1 2 0.2 1
1 3 0.1 0
2 4 0.2 0
3 4 0.1 1
4 5 0.1 1
4 6 0.2 0
5 7 0.1 0
6 7 0.2 1
7 8 0.6 0
7 9 0.05 1
8 10 0.6 0
9 10 0.05 1
10 11 0.1 1
10 12 0.2 0
11 13 0.1 1
12 13 0.2 0
13 14 0.1 0
13 15 0.2 1
14 16 0.1 0
15 16 0.2 1

Output.
Yes

Note 1. Instead of g, the first line of input contains the value of "-log_{10}(g)", where 10 is the base of the logarithm. We will follow this modification in input format.

Note 2. The same input with k=3 will give answer 'No'.

Hints.
(i) You can assume that 0<=k<=n. Further, for every edge e, the failure probability p(e) satisfies 0 < p(e) < 1.
(ii) For an edge e of graph G, define its ``length" l(e) as

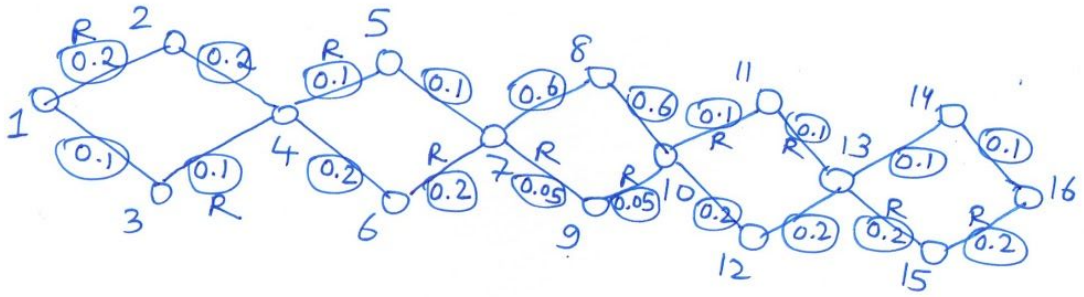
$$l(e) = - \log (1-p(e))$$

Then, with respect to these lengths, finding the path with maximum success probability (without any constraint on the number of red edges) is equivalent to finding the path with minimum total length.

You can use “math.h” to compute logarithm of a real number.

(iii) Consider the wave propagation analogy:

What extra information would you like to attach to individual wave packets, if you have a constraint on how many red edges you can use?



- numbers in circles are failure probabilities
- R means red edge.