**Algorithm constructing a spanning tree to get the best achievable bottleneck rate for the pair u, v in Graph G**

**Abstract:**

For a given graph G, we have to find a spanning tree which will give the best achievable bottleneck rate for the pair u, v in G. In the brute force approach, we have to look for every possible path from u to v and then consider the path which will give the best achievable bottleneck rate. With the Greedy technique, this problem can be solved easily by constructing the Maximum spanning tree.

1.Problem Statement:

A group of network designers at the communications company CluNet find themselves facing the following problem. They have a connected graph G = (V, E), in which the nodes represent sites that want to communicate. Each edge e is a communication link, with a given available bandwidth be. For each pair of nodes u, v ϵ V, they want to select a single u-v path P on which this pair will communicate.

The bottleneck rate b(P) of this path P is the minimum bandwidth of any edge it contains; that is, b(P) = min e ϵ P be. The best achievable bottleneck rate for the pair u, v in G is simply the maximum, over all u-v paths P in G, of the value b(P).

It’s getting to be very complicated to keep track of a path for each pair of nodes, and so one of the network designers makes a bold suggestion: Maybe one can find a spanning tree T of G so that for every pair of nodes u, v, the unique u-v path in the tree actually attains the best achievable bottleneck rate for u, v in G. (In other words, even if you could choose any u-v path in the whole graph, you couldn’t do better than the u-v path In T.) This idea is roundly heckled in the offices of CluNet for a few days, and there’s a natural reason for the skepticism: each pair of nodes might want a very different-looking path to maximize its bottleneck rate; why should there be a single tree that simultaneously makes everybody happy? But after some failed attempts to rule out the idea, people begin to suspect it could be possible.

Show that such a tree exists, and give an efficient algorithm to find one. That is, give an algorithm constructing a spanning tree T in which, for each u, v in V, the bottleneck rate of the u-v path in T is equal to the best achievable bottleneck rate for the pair u, v in G.

**2.The Brute Force approach:**

* Find all possible paths between two nodes (u,v)
* Find out the bottleneck rate for each of these paths
* Maximum among them will be the best achievable bottleneck rate
* Consider that path for communication

**2.1 Analysis:**

The path is a sequence of vertices that a path visit. Consecutive vertices in a path will have an edge between them. So, it is sufficient to generate every permutation of ‘V’ vertex. Every permutation is a valid path and calculating bottleneck for every path. Overall there will be V! permutations to check for a valid path.

Time complexity = O(V !)

**3. Greedy Approach:**

* Input G =(V, E )
* Construct Maximum Spanning Tree M
* P is a path between u,v in M
* Minimum bandwidth edge in path P is a bottleneck, Which will be the best achievable bottleneck rate.

**3.1Algorithm:**

G = (E,V)

M = ∅; //Set of Edges in Max Spanning tree

U = { V1 }; //Set of vertices with starting node

while (U ≠ V)

let (u, v) be the highest bandwidth edge such that u ∈ U and v ∈ V - U;

M = M ∪ {(u, v)}

U = U ∪ {v}

**3.2Proof:**

* M ={e1, e2, e3 .. en}

M is Maximum spanning tree by algorithm

* T={t1, t2, t3 .. tn }

T is spanning tree such that e1 ∉ T

If e1 is not in spanning tree T, it will be divided into two subtrees Sa and Sb.

Consider t1 is edge joining them and completing spanning tree.

1. e1 = t1, there will not be any effect on Best Achievable Bottleneck rate.
2. e1 < t1, it is not possible, as e1 is in Maximum spanning tree M.
3. e1 > t1, if we choose t1 then the Bottleneck rate is reduced, which is not desirable.

Algorithm constructing a spanning tree M in which, for each u, v in V,

the bottleneck rate of the u-v path in M is equal to the best achievable bottleneck rate for the pair u, v in G

**3.3Analysis:**

To construct a maximum spanning Tree:

O (E logV)

To find a path between u and v:

O (E)

Overall time complexity will be,

O (E logV) + O (E)

Where,

E: no. of edges

V: no. of Vertices

**4.Conclusion:**

By constructing a maximum spanning tree M in which, for each u, v in V, the bottleneck rate of the u-v path in M is equal to the best achievable bottleneck rate for the pair u, v in G.