**Yueling Qin**

CS 4800

**Problem1 MATCHING DANCERS**

Goal: There are n men and n women; we would like to match the men and women into n pairs according their height.

Way:

* Firstly, we can assume n men in an array list; guarantee the array order by using merger sort. We have running time O (n log (n)) for input n men.
* Secondly, same thing with women, we sort the women array list by using merger sort. We have running time also O (n log (n)) for input n women.
* We pair men and women, first men match with first women, second men match with second women. At the end of matching, every men and women matched, it proof in CLAIM.

Using this way, we can get matching in which the sum, over all pairs, of the absolute differences of the heights of the man and the women in the pair, is minimized. Having a running time O (n log (n).

To proof that is best way (greedy) , we from some simple example:

We have 2 men: 180, 176

2 women: 178, 163

* 180-178=2
* 176-163= 13
* abs:|2+13=15|
* 180-163=17
* 176-178=-2
* abs: |17|+|-2|=19
* So, we get 19>15, 1 is better than 2 in this matching problem.

We have 3 men: 198, 177, 164

3 women: 188, 169, 164

* 198-188=10
* 177-169=8
* 164-164=0
* abs: 10+8+0=18
* 198-188=10
* 177-164=13
* 164-169=-5
* abs: 10+13+|-5|=28
* 198-169=29
* 177-188=-11
* 164-164=0
* abs:29+11+0 = 40
* 198-169=29
* 177-164=1
* 164-188=-24
* abs:29+1+|-24|=54
* 198-164=34
* 177-169=8
* 164-188=-24
* abs: 34+8+|-24|=56
* 198-164=34
* 177-188=-11
* 164-169=-5
* abs:34+|-11|+|-5|=50

so if we have n men, n women, we get the array list of men from larger to small, (m1, m2, m3…..mn)

and women array list (w1, w2, w3…….wn)

m1-w1< m1-w2<m1-w3<…..< m1-wn

and m2-w1> m2-w2>m2-m3……

continue to do this thing, we will get more larger difference.

According to the strategy of greedy algorithm, we always try my best to get we want, therefore, the way I do must get minimum sum of absolute differences value.

**Problem2 LIMITING BANDWIDHT**

Goal: We define the limiting bandwidth of a path p as the bandwidth of the minimum-bandwidth link in p. Design and analyze a polynomial-time algorithm to determine a path with the largest limiting bandwidth from a given node s to a given node t.

Way:

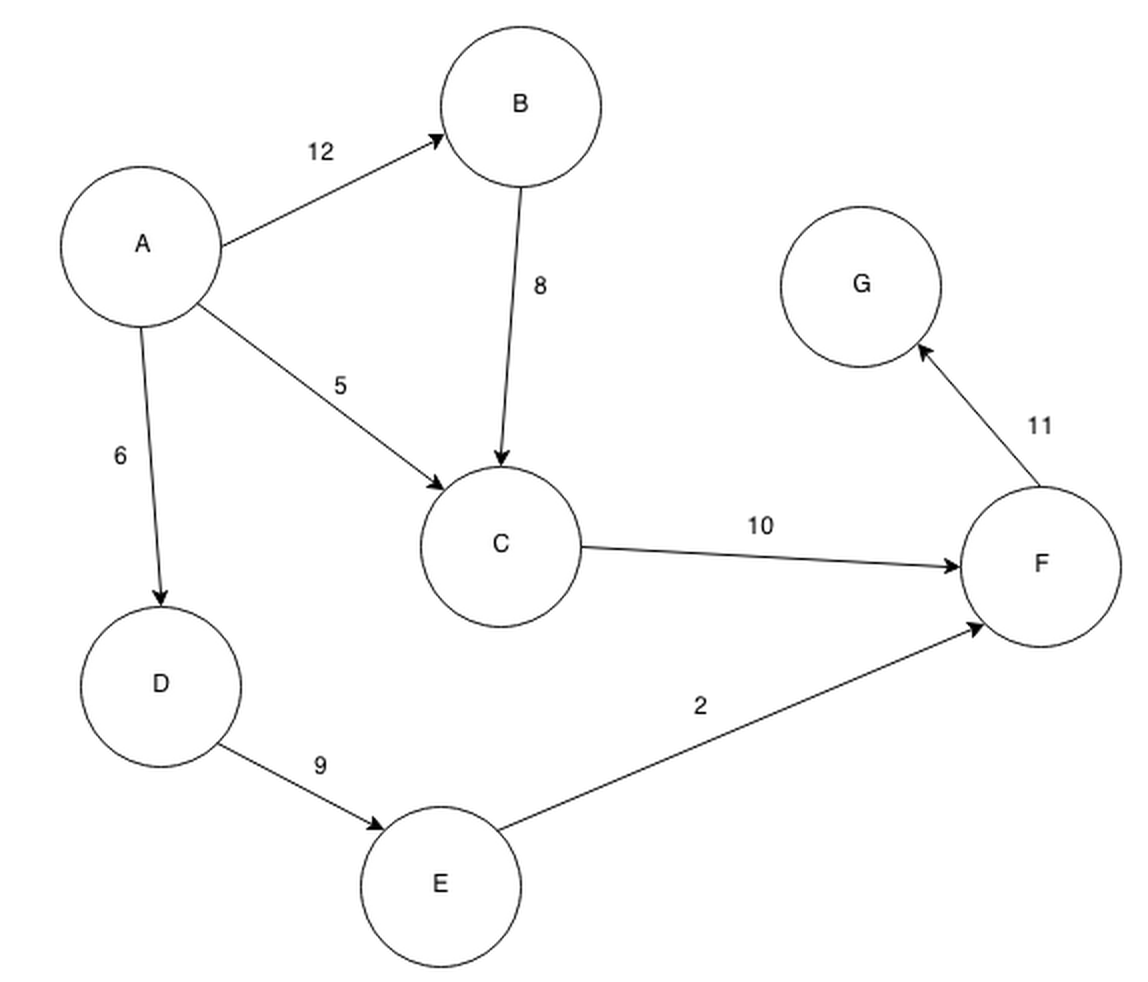
Run DFS check is there a path from node s to node t O(n+m)

If yes, we using the algorithm by modifying Dijkstra’s algorithms O (n^2)

* Maintain a set of explored nodes S for which we have determined the path distance d (u) from s to u.
* Initialized S ={s}, d(s)=0.
* We choose a maximum weight of node, we keep travel the next point connected with this node, if the weight is less than other weight form starts point to second one, and we try other path. Otherwise, we don’t need to pass other point; we keep track this path.

Min (d (u))= limiting bandwidth.

Otherwise, return “there is no a path from s to t”.



Firstly, we need to check is there a path from A to G,

If yes, we go to next step, A to B, and then we keep going to next point, and also we follow the algorithm, we go to C, and F, G

If we have this diagram, the largest limiting bandwidth from A to G in this graph would be A, B, C, F, G, and the limiting bandwidth would be 8.

Other path A, D, E, F, G limiting bandwidth is 2,

Path A, C, F, G limiting bandwidth is 5.

**Problem3 UNIQUENESS OF MST WHEN WEIGHTS ARE DISTINCT**

True, we assume there is a e be the edge with minimum weight.

By using contradiction, if G’ is a MST which does not include the minimum weight e,

Now, we add e to G’, it create a cycle in G’, then we remove any edge (because other edges always bigger then edge e). So, we have a new MST G’’, W (G’’)<W”(G).

Moreover, if the minimum weight edge e exist in G’’, it mustn’t in the other one. Because the G’’ and G1 are two spanning trees, according to the definition of tree, we can know e in G’’ must not in G1, otherwise, there is only have one spanning tree.