CSC 226 Assignment 3

1 - Consider a union-find implementation that uses the same basic strategy as weighted quickunion but keeps track of tree height and always links the shorter tree to the taller one. Prove a logarithmic upper bound on the height of trees for N sites for this scheme.

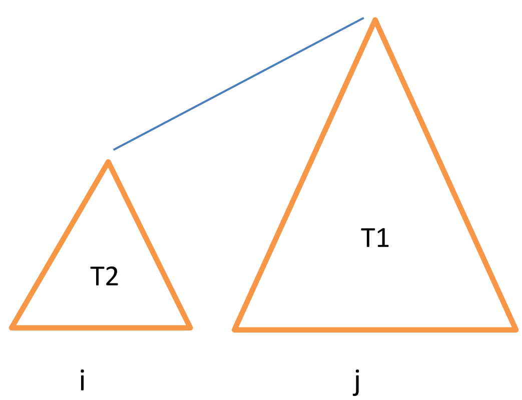
We are going to use the strong induction to prove a logarithmic upper bound on the height of trees for N sites for this scheme.
We claim the height of every tree of size k is at most $\log_2 K$ such that 1<= k<=N

a-Base case
When k=1, $\log_2(1) = 0$ so the height is 0. The claim is true for the base case.
b – Induction Hypothesis

Assume the claim is true for all trees of size < k

c- Induction Step

Let prove that the claim is true for tree of size k



i<=j and i + j = k

Since i<k height of T2 is $\log_2 i$. And since j< k, height of T1 is $\log_2 j$. Thus height of T = max { height(T1), 1+ height(T2)} which leads to two cases.

Case 1: height(T)= height(T1) = $\log_2 j$

$\quad\quad$ <= $\log_2 k$ (because j<k)

Case 2 : : height(T) = 1+ height(T2) = 1 +$\log_2 i$

$= \log_2(2i) <= \log_2(k)$ (because $i <= k/2$)

d- conclusion

By the strong induction, we conclude that the height of trees for N sites for this scheme has a logarithmic upper bound.

2 - Give a proof of correctness for Algorithm 4.10, for computing shortest paths in edgeweighted Directed Acyclie Graphs (DAGS). Use proof by contradiction technique

Suppose that v_i is the first vertex in the topological ordering such that $D[v_i]$ is not the distance from s to v_i. First, note that $D[v_i] <$ +infinity, for the initial D value for each vertex other than s is +infinity and the value of a D label is only ever lowered if a path from s is discovered. Thus, if $D[v_i]$ = +infinity, then v_j is unreachable from s. Therefore, v_i is reachable from s, so there is a shortest path from s to v_i. Let v_k be the penultimate vertex on a shortest path from s to v_i. Since the vertices are numbered according to a topological ordering, we have that $k<i$. Thus, $D[v_K]$ is correct (we may possibly have $v_k = s$). But when v_k is processed, we relax each outgoing edge from v_k, including the edge on the shortest path from v_k to v_i. Thus, D $[v_i]$ is assigned the distance from s to v_i. But this contradicts the definition of v_i, hence, no such v_i can exist.

3 - If the PQ is implemented as an unsorted sequence, show that Dijkstra's algorithm runs in O(n^2) time. For what type of graphs is this implementation preferred?

Insertion and decreaseKey will be O(1). However, in order to Extract the minimum key, we may have to search through the entire array for a running time O(n) where n is the number of vertices in the min PQ. We have *n vertices and* each extractMin operation can take $O(n)$ time. Thus the running time for all *n* extractMin operations altogether is O(n^2).

This implementation is preferred if the if the graph is dense. In other words, if the graph is a complete graph on n vertices (there is an edge between every pair of vertices).Using a heap in that situation, the running of Dijkstra's algorithm runs in time $O(n^2 \lg n)$, which is *worse* than using an unsorted array's O(n^2).

4. If at the end of the execution of Bellman-Ford algorithm, there is an edge (u, z) that can be potentially relaxed (that is, D[u] + w(u, z) < D[z], then show that the input digraph G contains a negative-weight cycle.

We know If there are no negative cycles from s, then for any v there is a shortest path from s to v using at most n-1 edges. Now we want to show that If at the end of the execution of Bellman-Ford algorithm, there is an edge (u, z) that can be potentially relaxed then the input digraph G contains a negative-weight cycle. We know that from the definition of the Bellman-Ford algorithm If the distance (weight) of at least one node changes in round n (number of vertices), then there is a negative cycle that is visible from s. So, by using the contrapositive argument, if there are no negative cycles visible from s, then the distances (weight) don’t change in round n. Since (u,z) can be potentially relaxed, the input diagraph G contains a negative-weight cycle.