PSO #8 Solutions Sketch (Week 10)

Week of 2021-10-25

1 Minimum Spanning Trees

1. Let G be an undirected weighted graph, and T a minimum spanning tree of G. In each case, construct an algorithm to modify T so that it remains a MST.

   (a) The weight of some edge $e \in T$ is decreased

   (b) The weight of some edge $e \notin T$ is increased

   (c) The weight of some edge $e \in T$ is increased

   (d) The weight of some edge $e \notin T$ is decreased

   Solution:

   (a) No change needed. Proof by contradiction

   (b) No change needed. Think about running Kruskal's.

   (c) Let $e = (u, v)$ and let T_u, T_v be the subtrees resulting from removing e from T. Examine each edge and keep the minimum weight edge e' that connects T_u, T_v.
   Runtime: $O(|V| + |E|)$. $O(|V|)$ to partition into T_u, T_v and $O(|E|)$ to find the new edge.

   (d) Let $e = (u, v)$. Add e to T, creating a cycle. BFS on $T \cup \{e\}$ can be used to find the cycle, then remove the max weight edge on that cycle.
   Runtime: $O(|V|)$ for the BFS, then $O(|V|)$ to find the max weight edge on the cycle

2. Let G be a graph where k edges are marked as essential. How does one modify Prim's or Kruskal's such that a spanning tree can be formed that contains all essential edges (the k edges are given to not form a cycle)? Which one is better (no correct answer per se, so discuss)?

   Solution: Kruskal's is trivial modification, just sort all the non-essential edges and add starting from the minimum non-essential edge that does not create a cycle. Prim's is a bit weirder, 'cuz you have to either always pick an essential edge as the connection between an adjacent vertex as you're constructing the tree or you have to try to create a tree first that is composed of all the vertices connected by essential edges and then add non-essential edges as you go.

3. Let there be a tree T that is claimed to be the minimum spanning trees of a graph G, but you are not sure. Construct an algorithm to ensure the given tree is a MST.

   Solution: Two approaches the follow the same idea are as follows. First, sort the remaining edges in G that aren't in T and perform Kruskal's until the minimum weight edge not in T is larger than the largest weight edge in T and T is still a tree. However, this approach still has worst case $O(V * E)$, thus leading to the second approach. The second one is to simply create a new minimum spanning tree and compare the total weights of T and the new tree.

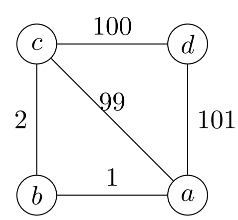
4. Let $G = (V, E)$ be an undirected weighted graph. Let the weight of every edge be from the set $\{1, 10, 25\}$. Construct a linear time algorithm to find an MST of G.

   Solution:

   We can modify Prim's Algorithm. Make three buckets for 1-weight, 10-weight, and 25-weight edges. It takes $O(|E|)$ to sort the edges into these buckets. Find and Decrease-Key both run in constant time using these buckets. Thus, the algorithm runs in $O(|V| + |E|)$

5. Let G be an undirected graph with distinct costs on every edge. Let $e*$ denote the least expensive edge on a cycle C in G. Is it true that any minimum spanning tree of G contains the edge $e*$?

   Solution: No.



   The smallest edge in the cycle $a \to c \to d \to a$ has weight 99, but this edge is not included in the MST