CSCI-GA.1170-003/004 Fundamental Algorithms April 30, 2020

Solutions to Problem 3 of Homework 11 (12 points)

Name: Rahul Zalkikar (rz1567) *Due: Thursday, 8 pm on April 30*

Collaborators: ua388, jni215, nmd353

(a) (4 points) Let e be the maximum weight edge on some cycle of a connected graph $G = (V, E)$. Prove that there exists an MST T of $G' = (V, E \backslash \{e\})$ which is also an MST of G. Namely, some MST of G does not include e.

Solution:

If $e \notin T$, we are done.

Now suppose $e \in T$. If we remove e from T (but do not remove e's vertices), T will be split into two trees, which we call T_1 and T_2. Now when there is a cycle in the graph there are more than two paths between some pair of nodes, and if we remove the maximum weight edge e from the cycle we are eliminating some path that is larger than the shortest path. So if a cycle C connects T_1 and T_2 by e, C must also connect T_1 and T_2 at another edge, which we call e'. Therefore T_1 together with T_2 and e' is an MST of G that does not contain e. $\square$

(b) (4 points) Given a graph $G = (V, E)$ with edge weight function $w : E \mapsto \mathbb{N}$ such that no two edges have equal weight, give an algorithm to find a second smallest spanning tree. Assume that there exists a second smallest spanning tree with exactly one edge different from a minimum spanning tree. State the running time of your algorithm.

Solution:

FIND-T2(G, s)

$A = $ MST-KRUSKAL(G, w)

$A_{sorted} = HeapSort(A)$

$T = (V, A_{sorted})$

$\Delta T.w = \infty$

$E_{new} = -1$

$E_{old} = -1$

for each edge $e \in A_{sorted}$:

  $A_{temp} = A_{sorted} - \{e\}$

  $T_{temp} = (V, A_{temp})$

  for each edge $k \in G.E$ and $k \notin A_{sorted}$:

    $A_{tempk} = A_{temp} + \{k\}$

    $T_{tempk} = (V, A_{tempk})$

    $\Delta w = e.w - k.w$:

    if $\Delta w < \Delta T.w$:

      $\Delta T.w = \Delta w$

$E_{new} = k$
$E_{old} = e$
$T_2 = (V, A_{sorted} - \{E_{old}\} + \{E_{new}\})$
return T_2

In this algorithm we find the set of edges in the MST with Kruskal in $O(E)$ and then sort the edges in $O(E \log E)$ before checking each possibility of removing one edge of the MST and replacing it with another and returning the spanning tree with smallest weight difference to MST, or the second smallest spanning tree. So we have a runtime of $O(E + E \log E + VE) = O(VE)$. □

(c) (4 points) Prove the assumption you used in part (b).

Solution:

We want to prove there always exists a second smallest spanning tree with exactly one edge different from a minimum spanning tree for a graph G that is not a tree.

Let T represent the minimum spanning tree and let T_2 represent the second smallest spanning tree. Now consider an edge $(u, v) \in T - T_2$. Then, $T_2 \cup (u, v)$ must have a cycle where exactly one of the edges, $(x, y) \notin T$. So we must have $w(u, v) < w(x, y)$ because otherwise $(x, y) \in T_2$ could be replaced by (u, v) to attain an MST better than T_2. Now we can see that $T_2 - (u, v) \cup (x, y)$ is also a spanning tree since (u, v) and (x, y) are in the same cycle, and that $w(T_2 - (u, v) \cup (x, y)) < w(T_2)$. Finally, $T_2 - (u, v) \cup (x, y)$ is the smallest spanning tree (the MST), and by uniqueness of MST we know that $T = T_2 - (u, v) \cup (x, y)$ and therefore there always exists a T_2 with exactly one edge that differs from T. □