
You're expected to work on the discussion problems before coming to the lab. Discussion session is not meant to be a lecture. TA will guide the discussion and correct your solutions if needed. We will not release 'official' solutions. If you're better prepared for discussion, you will learn more. TAs will record names of the students who actively engage in discussion and report them to the instructor; they are also allowed to give some extra points to those students at their discretion. The instructor will factor in participation in final grade.

For MST, assume that all edges have distinct weights unless otherwise stated.

1. (Basic) What is the definition of tree?

   Sol. A connected (undirected) graph with no cycle.

2. (Basic) Consider two trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ over disjoint sets of vertices, i.e., $V_1 \cap V_2 = \emptyset$. For any two vertices $x \in V_1$ and $y \in V_2$, if we add (x, y) to the two trees, we obtain another tree $(V_1 \cup V_2, E_1 \cup E_2 \cup \{(x, y)\})$. True or False?

   Sol. True. Why: the graph is connected, and has $|V| - 1$ edges.

3. (Intermediate) Suppose we have a mysterious data structure that supports set operations, Make-Set, Union, and Find-Set, each with $O(1)$ run time. What would be the running time of the Kruskal's algorithm if we use this data structure.

   Sol. Still, we get $O(E \log V)$ since we need sort edges in the order of their weights (assuming that sorting n elements takes $O(n \log n)$ time).

4. (Basic) What is the definition of safe *?*dges (assuming that edges have all distinct weights)? The lecture slides use a definition that is different from the textbook. Use the definition in the lecture slides, which is simpler.

   Sol. An edge is safe if it is the cheapest edge crossing a certain cut $(X, V \setminus X)$.

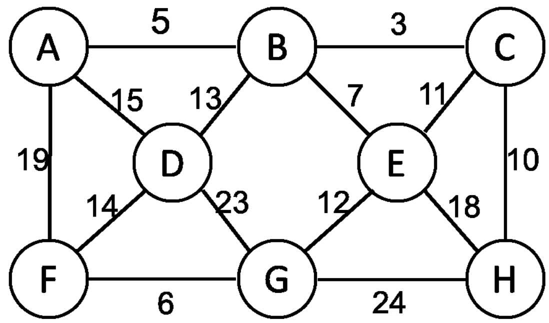
5. (Intermediate) Consider an undirected connected graph $G = (V, E)$ with $|E| \geq |V|$. It is always the case that the heaviest edge is not safe, therefore is not included in any MST of G. True or False?

   Sol. False. A counter example: Consider a graph which has edges $(a, b), (b, c), (c, a)$, (a, d). Say (a, d) is the heaviest edge. The edge (a, d) is safe.

6. (Basic) Minimum Spanning Tree (MST). Consider the above weighted undirected graph.

   For each of the following two algorithms, list the edges appearing in the MST *in the order they are added*. For example, the first edge added by Kruskal's algorithm is (E, H).

   (a) Kruskal's algorithm.
   Sol. $BC, AB, FG, BE, CH, EG, BD$.



(b) Prim's algorithm using A as initial vertex.

Sol. $AB, BC, BE, CH, EG, FG, BD$.

7. (Basic) Note that in the above graph, all edges have distinct weights. Hence the graph has a unique MST. To show that edge (B, C) is included in the MST, we only need to find a cut such that (B, C) is a light (cheapest) edge crossing the cut. For example, $(\{B\}, V - \{B\})$; here V denotes the set of vertices. .

   (a) (5 points) Show that (B, D) must be included in the MST.

   Sol. (B, D) is the unique light edge crossing cut $(\{D\}, V - \{D\})$.

   (b) (5 points) Show that (F, G) must be included in the MST.

   Sol. (F, G) is the unique light edge crossing cut $(\{G\}, V - \{G\})$.

8. (Intermediate) When we run Kruskal's algorithm, we can stop as soon as we've chosen $|V| - 1$ edges as those of the MST. True or False?

   Sol. True. A tree has exactly $|V| - 1$ edges, and we know that Kruskal's algorithm only adds safe edges.

9. (Advanced) We're given as in input an undirected connected graph G with distinct weights on edges along with its MST T. Say the weight $w(x, y)$ of an edge (x, y) of the tree T has changed to $w'(x, y)$. Then, the new MST T' could be different from T. We would like to compute T' as fast as possible. As before, let's assume that edges have distinct weights both before and after the change. Consider the following two cases.

   (a) When $w'(x, y) < w(x, y)$.

   Sol. T' is the same as T. To see this consider any edge e of T. If we cut the edge, we will have two trees T_1 and T_2. Note that e is the cheapest edge crossing the cut $(V(T_1), V(T_2))$, hence safe. Decreasing (x, y)'s weight has no effect on this: if $e = (u, v)$, e surely remains safe, otherwise (x, y) is not a crossing edge of the cut. So every edge of T remains safe even after the change. Hence we conclude that $T = T'$. So we have nothing to do.

(b) When $w'(x, y) > w(x, y)$.

Sol. It is easy to see that all edges of T except (x, y) remain safe after increasing (x, y)'s weight. Let T_1 and T_2 be the two trees we get after removing (x, y) from T. We add the cheapest edge crossing the cut $(V(T_1), V(T_2))$. To filter out edges that do not cross the cut, we need $O(E)$ time. To find the edge with the lowest weight among the remaining edges, we need $O(E)$ time. So, our algorithm runs in $O(E)$ time.