Practice Problem Set: MST

Problem 1. Let $G = (V, E), w$ be a weighted undirected graph. Suppose we want to find a connected spanning subgraph of minimum total weight. Show that if all weights are positive, then the solution cannot include a cycle.

Problem 2. Let G be a graph with distinct weights, show that the MST of G is unique.

Problem 3. Let $e = (u, v)$ be the lightest edge crossing the cut $[A, \bar{A}]$. Show that e is part of some MST of G.

Problem 4. Let $G = (V, E), w$ be a graph. Show that G is connected if and only if there is no non-trivial empty cut in G.

Problem 5. Let $[A, \bar{A}]$ be a non-trivial cut in a graph $G = (V, E), w$. Prove that any cycle C in G crosses the cut $[A, \bar{A}]$ an even number of times.

Problem 6. Show that if $e = (u, v)$ is part of some MST of G, then e is the lightest edge across some cut in G.

Problem 7. Show that if $e = (u, v)$ be the lightest edge of the graph G, then e is part of some MST of G.

Problem 8. Let $G = (V, E), w$ be a graph such that all weights are distinct. Let $e = (u, v)$ be the heaviest edge on some cycle C in G. Prove that e cannot be part of the MST of G.

Problem 9. Show that a graph has a unique MST if for every cut in the graph the lightest edge across the cut is unique. Show that the converse is not true, i.e. make a small graph such that a cut has two lightest edges (of equal weights), still the MST is unique.

Problem 10. Let $G = (V, E), w$ be a graph. Suppose $V' \subset V$ and let G' be the subgraph of G induced by V'. i.e. $G' = (V', E')$, where $E' = \{(u, v) \in E(G) : u \in V' \wedge v \in V'\}$. Let T be a MST of G and T' be a subgraph of T induced by V'. Show that if T' is connected, then T' is a MST of G'.

Problem 11. We know that if weights on edges of a graph G are not distinct, then G may have more than one MST's. Make a small graph and show that with different sorted orders of edge weights (depending on how we break ties) Kruskal's algorithm produce different MST's.

Problem 12. Note that the implementation of Prim's algorithm we discussed in class assumed that the graph was input as an adjacency list. Give a simple implementation of Prim's algorithm if the graph is input as an adjacency matrix. Your algorithm should run in $O(n^2)$, where $n = |V|$.

Problem 13. For Kruskal's algorithm we used a linked list implementation of the disjoint set (union-find) data structure. It's complexity was $O(m \log n) + O(n \log n)$, where the first term was for sorting the edges by their weights and the second term was for the union-find operations. Suppose we are given that all weights are integers in the range $[0, \ldots, n-1]$. Give an implementation of Kruskal's algorithm that runs in $O(m) + O(n \log n)$.

Problem 14. With our simple implementation union-find data structure with linked list for each set and a pointer to the head of the list at each node (pointer to the representative of the set), we know that in the worst case a union of two sets $Union(Find(x), Find(y))$ could take time equal to the length of the longer list (for head pointers update). Suppose we do not save sizes of the sets (hence sizes of the lists), come up with a sequence of union operations that would take time $\Omega(n^2)$. Assume we have already done n make-set operations.

Problem 15. Suppose now we keep sizes of the lists at the head node for each list (set). Show that any sequence of m Find operations and n Union operations take time $O(m + n \log n)$, where n is the size of the universal set and assume that we have already performed n make set operations. Argue that there can't be more than $n-1$ union operations.

Problem 16. Suppose we want to find a MST of an undirected connected graph $G = (V, E, w)$ but vertices are given one by one. When vertex v arrives we are also given a list of its adjacent vertices along with weights of edges among the already known vertices. Suppose we grow T incrementally as follows: When a new vertex u_j is input along with its incident edges we select the minimum weight edge (u_j, u_i) where $i < j$ and add it to T.

a. Show that T is a tree.

b. Give a small counter example (3 or 4 vertices) where this algorithm does not produce a MST.

c. The algorithm seems to be applying the blue rule (the cut property), why is it wrong then?

d. Briefly give an idea of how to correct this algorithm.

Problem 17. Consider the following divide-and-conquer approach to computing MST of a graph $G = (V, E), w$. Suppose $|V| = n = 2^k$ for some integer k. We partition V into V_1 and V_2 such that $|V_1| = |V_2| = \frac{n}{2}$. Let G_1 and G_2 be the subgraphs induced by V_1 and V_2 respectively. We recursively compute a MST's in T_1 of G_1 and T_2 of G_2. Now we add the lightest edge $e = (u, v)$ crossing the cut $[V_1, V_2]$ in G and use it to unite T_1 and T_2. Either prove that $T = T_1 \cup T_2 \cup \{(u, v)\}$ is a MST of G or give a small counter example on $n = 4$ vertices on which this algorithm fails.