CSL 356: Analysis and Design of Algorithms
Instructor: Ragesh Jaiswal

We first prove the following lemma.

Lemma 1.0.1 *Let G be a weighted graph and let T be a MST of G. There exists an ordering of the edges of G such that the weights of these edges are in increasing order and the Kruskal's algorithm, when run on this ordering of edges, outputs T.*

Proof: If all the edge weights are distinct, then T is the unique MST of G and since the Kruskal's algorithm outputs an MST, it outputs T. Now if all the edge weights are not distinct, then we can argue in the following manner:

We will slightly modify the edge weight of each edge so that all edge weights become distinct. Let $E_T = \{t_1, t_2, ..., t_{n-1}\}$ denote the edges of T such that they are sorted in increasing order of their weights, i.e., $W(t_1) \leq W(t_2) \leq \ldots \leq W(t_{n-1})$. Let $E'_T = \{s_1, s_2, ..., s_k\}$ denote the remaining edges in the graph. We construct the following new graph that has the same vertices and edges as G but the edge weights are slightly modified. Let W denote the weight function of G and W' be the weight function of G' that is defined in the following manner. Let β be the minimum value of the difference of weights of two edges of G. Let $\alpha = \beta/n^3$. For edges in E_T the weights are given by $W'(t_1) = W(t_1) - (n-1) \cdot \alpha$, $W'(t_2) = W(t_2) - (n-2) \cdot \alpha$, $\ldots$, $W'(t_{n-1}) = W(t_{n-1}) - 1 \cdot \alpha$. For edges in E'_T, the weight function is defined as $W'(s_1) = W(s_1) + 1 \cdot \alpha, W'(s_2) = W(s_2) + 2 \cdot \alpha, \ldots W'(s_k) = W(s_k) + k \cdot \alpha$. Note that the way we have defined the weight function W', we have made sure that all the weights of edges are distinct. We now show the following two things:

Lemma 1.0.2 *The MST of G' is an MST of G.*

Lemma 1.0.3 *T is the MST of G'.*

We will defer the proof of the above lemmas and see how these lemmas imply Lemma 1.0.1. Let $e_1, e_2, ..., e_m$ be an ordering of edges such that $W'(e_1) < W'(e_2) < ... < W'(e_m)$. Note that due to the way we have defined the weight function W', we get that $W(e_1) \leq W(e_2) \leq ... \leq W(e_m)$. This is because for any two edges e_i, e_j such that $W(e_i) < W(e_j)$, we have

$$W'(e_i) \leq W(e_i) + (W(e_j) - W(e_i)) \cdot (n^2/n^3) < W(e_j) - (W(e_j) - W(e_i)) \cdot (n/n^3) \leq W'(e_j).$$

Note that the Kruskal's algorithm, when executed using the ordering $e_1, ..., e_m$, returns T. This proves the lemma 10.0.1.

We now prove Lemmas 1.0.2 and 1.0.3.

Proof: [Proof of Lemma 1.0.2] Let V be the total weight of any MST of G. Note that $V = W(t_1) + W(t_2) + ... + W(t_{n-1})$. Consider the weight V' of T in G'. $V' = V - n(n-1)/2 \cdot \alpha$. The proof follows from the fact that any other spanning tree will have weight more than V' in G'. ∎

Proof: [Proof of Lemma 1.0.3] Consider any edge e in T. When e is removed from T, it gets disconnected into two subsets of vertices, say V_1, V_2. Note that the weight of e is one of the smallest among the edges going accross the cut (V_1, V_2) in G. This implies that the weight of e in G' is the smallest weight edge that goes accross the cut (V_1, V_2) in G'. This implies that all $MSTs$ of G' should contain e. Similarly, we can argue for all edges in T. ■

This completes the proof of Lemma 1.0.1. ■

1.1 ~~Problem 2 of Homework 1~~ Problem 2 of Tutorial 3

If this edge is already in the MST, then the algorithm returns the same MST. Otherwise, the algorithm adds this edge to the MST and this creates a cycle. The algorithm then goes around the cycle and throws out the edge with maximum weight.

Here, we will discuss the proof of correctness of this algorithm. In Lemma 1.0.1, we have shown that given any weighted graph G and an MST T of G, there is an ordering of edges (in increasing order of weight) of G such that the Kruskal's algorithm when executed using this ordering returns T. Let $e_1, ..., e_m$ be this an ordering. Suppose that we have decreased the weight of edge e_i. Let us re-insert this edge e_i into this ordering such that the as per the new ordering, the edge weights are again sorted. This ordering can be denoted as $e_1, ..., e_j, e_i, e_{j+1}, ..., e_{i-1}, e_{i+1}, ..., e_m$. We now examine the MST returned by the Kruskal's algorithm on this ordering. If the algorithm does not pick e_i, then the MST returned is exactly T. If the algorithm picks e_i, then the only edge from T that it does not pick is the one that completes a cycle involving the edge e_i. This is precisely the maximum weight edge along the cycle that is created when e_i is added to T.