Intro to Algorithms, COMP-160, Homework #12

Benjamin Tanen, 04/29/2016

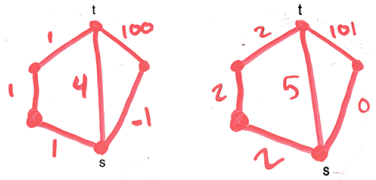
1. (a) **Let G be a graph where all edge weights are integers in the range $(1, ..., k)$, for some $k = O(1)$. Show how fast can you compute the MST of G.**

   We know that we will have E edges that have weights in the range $(1, \ldots, k)(k = O(1))$. In order to compute the MST of G, we can simply use Kruskal's algorithm, which we know takes $O(E \log V)$.

   However, since we know our weights are within the range $(1, \ldots, k)$, we can actually improve on Kruskal's algorithm by changing how we sort our edges. In our original consideration, we assume that it takes $O(E \log E)$ to sort our E edges. However, because of the arrangement of our edges, we can actually use a more efficient sorting algorithm like counting sort. By using counting sort, since we have E "values" ranging from $1 \to k$, we can sort all of our edges in $O(E + k)$ time.

   This makes it so that our total run time for Kruskal's will be the sum of $O(V)$ (to make a forest of vertices), $O(E + k)$ (to sort our edges), and $O(E + V \log V)$ (to check components and merge). If we add this all up, $O(E + V \log V)$ will dominate over our new sorting of $O(E + k)$, so our optimized algorithm has a running time of $O(E + V \log V)$.

   (b) **Prove or disprove that Dijkstra's algorithm can be modified to work on a graph with negative edge weights, if we first scan all edges, find the minimum weight w, and add $|w|$ to the weight of every edge.**



   Consider the two graphs above, where the graph on the left is G and the graph on the right is G' (the graph after we account for the negative edges by adding $|w|$). In G, we can see the minimized path from s to t goes along the left path with a sum weight of 3. We can see that the path including the negative edge has a sum weight of 99 so this is not the optimal path.

   If we update our graph G to account for the negative weights, we get the graph G' on the right above. In this graph, we can see that the optimal path is through the middle edge with the sum weight of 5, as opposed to the path that we should get. We can see this occurs because each path is incremented $|w|$ for every edge in the path. Thus, if the actual optimal path has a large number of edges, incrementing each edge by $|w|$ would unfairly advantage paths with fewer edges.

   Thus, from this example, we can see that Dijkstra's algorithm would not return the optimal path for the modified graph G' that we should get, thus proving this

method does not work.