Fundamental Algorithms, Assignment 10

Due April 19, in Recitation

I cannot live without people. – Pope Francis

1. Suppose we are given the Minimal Spanning Tree T of a graph G. Now we take an edge $\{x, y\}$ of G which is not in T and reduce its weight $w(x, y)$ to a new value w. Suppose the path from x to y in the Minimal Spanning Tree contains an edge whose weight is bigger than w. Prove that the old Minimal Spanning Tree is no longer the Minimal Spanning Tree.

2. Suppose we ran Kruskal's algorithm on a graph G with n vertices and m edges, no two costs equal. *Assume* that the $n - 1$ edges of minimal cost form a tree T.

   (a) Argue that T will be the minimal cost tree.

   (b) How much time will Kruskal's Algorithm take. *Assume* that the edges are *given* to you an array in increasing order of weight. Further, *assume* the Algorithm stops when it finds the MST. Note that the total number m of edges is irrelevant as the algorithm will only look at the first $n - 1$ of them.

   (c) We define Dumb Kruskal. It is Kruskal without the SIZE function. For $UNION[u, v]$ we follow u, v down to their roots x, y as with regular Kruskal but now, if $x \neq y$, we simply reset $\pi[y] = x$. We have the same assumptions on G as above. How long could dumb Kruskal take. Describe an example where it takes that long. (You can imagine that when the edge u, v is given an adversary puts them in the worst possible order to slow down your algorithm.)

3. Consider Kruskal's Algorithm for MST on a graph with vertex set $\{1, \ldots, n\}$. Assume that the order of the weights of the edges begins $\{1, 2\}, \{2, 3\}, \{3, 4\}, \ldots, \{n-1, n\}$. (Note: When $SIZE[x] = SIZE[y]$ make the first value the parent of the second. In particular, set $\pi[2] = 1$, not the other way around.)

   (a) Show the pattern as the edges are processed. In particular, let $n = 100$ and stop the program when the edge $\{72, 73\}$ has been processed. Give the values of $SIZE[x]$ and $\pi[x]$ for all vertices x.

(b) Now let n be large and stop the program after $\{n-1, n\}$ has been processed. Assume the ordering of the weights of the edges was *given* to you, so it took zero time. How long, as an asymptotic function of n, would this program take. (Reasons, please!)

4. Consider Prim's Algorithm for MST on the complete graph with vertex set $\{1, \ldots, n\}$. Assume that edge $\{i, j\}$ has weight $(j - i)^2$. Let the root vertex $r = 1$. Show the pattern as Prim's Algorithm is applied. In particular, Let $n = 100$ and consider the situation when the tree created has 73 elements and π and key have been updated.

   (a) What are these 73 elements.

   (b) What are $\pi[84]$ and $key[84]$.

5. `Do NOT hand in -- but give it a try!` In Kruskal, a student asked about using DEPTH rather than SIZE. Here we show this works. When z is a root we want $DEPTH[z]$ to be the largest l so that there is a "path" $x_0, x_1, \ldots, x_l$ with $x_{j+1} = \pi(x_j)$ for $0 \leq j < l$ and $x_l = z$. (That is, the longest "slide down the bannister" to z.) Initially all $DEPTH[z] = 0$. The FOR loop starts as before

   $x \leftarrow x[i]$; $y \leftarrow y[i]$
   WHILE $\pi(x) \neq x$
           $x \leftarrow \pi(x)$ (*sliding down the bannister*)
   WHILE $\pi(y) \neq y$
           $y \leftarrow \pi(y)$ (*sliding down the bannister*)

   When $x = y$ we do nothing. Otherwise we now use DEPTH. Flip if necessary so that $DEPTH[x] \leq DEPTH[y]$. Then
   $\pi(x) \leftarrow y$ *redirect to bigger depth*
   IF $DEPTH(x) = DEPTH(y)$ THEN $DEPTH(y) + +$

   (a) Show that the new value of $DEPTH(y)$ is correct with the changed $\pi(x)$. This has two parts.

      i. If $DEPTH(x), DEPTH(y)$ were equal, the new longest slide down the bannister to y is one more than it was.

      ii. If $DEPTH(x), DEPTH(y)$ were unequal, the new longest slide down the bannister to y is the same as what it was.

   (b) Show (use induction on t) that if z is a root and $DEPTH[z] = t$ then the cluster containing z (that is, the set of all x, including z itself, that slide down the bannister to z) has at least 2^t vertices.

(c) Deduce that the WHILE loop will have at most $\lg(V)$ steps, V being the total number of vertices.

I had ridden in taxi cab number 1729 and remarked that the number seemed to me rather a dull one, and that I hoped it was not an unfavorable omen. "No, [Ramanujan] replied, "it is a very interesting number; it is the smallest number expressible as the sum of two cubes in two different ways.
G.H. Hardy