02-713 Homework #2

Due: Jan. 31 by 9:30am

You may discuss these problems with your current classmates, but you must write up your solutions independently, without using common notes or worksheets. You must indicate at the top of your homework who you worked with. Your write up should be clear, and concise. You are trying to convince a skeptical reader that your answers are correct. Your homework should be submitted via Autolab (`https://autolab.cs.cmu.edu/02713-s14/`) as a typeset PDF. A LaTeX tutorial and template are available on the class website if you choose to use that system to typeset. For problems asking for an algorithm: describe the algorithm, an argument why it is correct, and an estimation of how fast it will run. For this assignment, your argument about run times need not be sophisticated: just argue that it will be practical.

1. Let G be a graph with distinct, positive edge weights $c(e)$ on every edge e. Show that if T is a minimum spanning tree of G, then T is a minimum spanning tree of G' where G' is the same as G except it has costs $c'(e) = c(e)^2$ on every edge (the cost of every edge is squared).

2. Let G be a graph that represents a computer network, and let $b(e)$ be the *bottleneck* of edge e — assume all edges have distinct, positive bottlenecks. When two nodes u and v want to communicate over a path P, the rate at which they can communicate is the *minimum* bottleneck along that path: $b(P) = \min_{e \in P} b(e)$. The best path $B(u, v)$ between two nodes u and v is the path P that maximizes $b(P)$ (i.e. has the largest, smallest bottleneck).

   It turns out there is a single spanning tree T such that for every pair of nodes u and v, a best path $B(u, v)$ for them is contained in the tree T. That is, a single tree can encode best paths between every pair of nodes.

   Give an efficient algorithm to find such a tree T and argue why your algorithm is correct.

3. Rank the following functions in order of their asymptotic growth. That is if $f_i(n) = O(f_j(n))$ then $f_i(n)$ should come before $f_j(n)$ in your list. If $f_i(n) = \Theta(f_j(n))$ then the two functions should be given the same rank.

   - $f_1(n) = n^3$
   - $f_2(n) = n!$
   - $f_3(n) = n \log_2 n$
   - $f_4(n) = 1$
   - $f_5(n) = 2^{\log_2 n}$
   - $f_6(n) = 10n \log_{10} n$
   - $f_7(n) = (n + 1)!$
   - $f_8(n) = 2^{\log_{50} n}$
   - $f_9(n) = 4^{\log_2 n}$
   - $f_{10}(n) = n^{\log_2 \log_2 n}$

4. Let $f(n)$ and $g(n)$ be functions that take on positive values for all $n > 1$ and such that $f(n) = O(g(n))$. Prove or give a counterexample to the following statements:

   (a) $2^{n+1} = O(2^n)$
   (b) $2^{2n} = O(2^n)$
   (c) $3^n = O(2^n)$
   (d) $f(n) = \Theta(f(n/2))$
   (e) $f(n) = O(g(n))$ implies $g(n) = \Omega(f(n))$
   (f) $\log_2 f(n) = O(\log_2 g(n))$
   (g) $2^{f(n)} = O(2^{g(n)})$
   (h) $f(n)^2 = O(g(n)^2)$