CSE521: Design and Analysis of Algorithms October 2, 2017

Homework 1

Anup Rao Due: October 11, 2017

Read the fine print[1]. Each problem is worth 10 points:

1. Assume have functions f, g such that $f = O(g)$, and that $f(x), g(x) > 1$ for every x. For each of the following statements, decide whether you think it is true or false and give a proof or counterexample:
   (a) $\log f(n) = O(\log g(n))$.
   (b) $2^{f(n)} = O(2^{g(n)})$.
   (c) $f(n) = O(g(n)^2)$.

2. Arrange the following in increasing order of asymptotic growth rate:
   (a) $f_1(n) = 100n^2$.
   (b) $f_2(n) = n^3$.
   (c) $f_3(n) = 2^{\sqrt{n}}$.
   (d) $f_4(n) = n(\log n)^{1000}$
   (e) $f_5(n) = 2^{n \log n}$
   (f) $f_6(n) = 2^{(\log n)^{0.9}}$

3. Prove that in any tree with n vertices, the number of nodes with 3 or more neighbors is at most $2(n-1)/3$. Use the fact that every tree on n vertices has exactly $n-1$ edges, and apply the identity $\sum_v deg(v) = 2m$.

4. A walk of length k in a graph is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that v_i is a neighbor of v_{i+1} for $i = 0, 1, 2, \ldots, k-1$. Suppose the product of two $n \times n$ matrices can be computed in time $O(n^\omega)$ for a constant $\omega \geq 2$. Give an algorithm that counts the number of walks of length k in a graph with n vertices in time $O(n^\omega \log k)$. HINT: If A is the adjacency matrix, prove that the $(i, j)'th$ entry of A^k is exactly the number of walks of length k that start at i and end at j. Repeatedly square the adjacency matrix to compute A^k.

5. In class we discussed an algorithm to color the vertices of an n vertex graph with 2 colors so that every edge gets exactly 2 colors (assuming such a coloring exists). We know of no such algorithm for finding 3-colorings in polynomial time. Here we'll figure out how to color a 3-colorable graph with $O(\sqrt{n})$ colors.

[1] In solving the problem sets, you are allowed to collaborate with fellow students taking the class, but **each submission can have at most one author**. If you do collaborate in any way, you must acknowledge, for each problem, the people you worked with on that problem. The problems have been carefully chosen for their pedagogical value, and hence might be similar to those given in past offerings of this course at UW, or similar to other courses at other schools. Using any pre-existing solutions from these sources, for from the web, constitutes a violation of the academic integrity you are expected to exemplify, and is strictly prohibited. Most of the problems only require one or two key ideas for their solution. It will help you a lot to spell out these main ideas so that you can get most of the credit for a problem even if you err on the finer details. Please justify all answers. Some other guidelines for writing good solutions are here: `http://www.cs.washington.edu/education/courses/cse421/08wi/guidelines.pdf`.

(a) Give a polynomial time algorithm to color the vertices with $\Delta + 1$ colors, where Δ is the maximum degree of vertices in the graph.

(b) Give a polynomial time algorithm that colors the graph with $O(\sqrt{n})$ colors, if the input graph is promised to have a 3-coloring. HINT: If a vertex v has more than $\sqrt{n}$ neighbors, then use the algorithm from class to color v and its neighbors with 3 colors. Continue until every vertex has less than $\sqrt{n}$ neighbors and then use the algorithm you developed to answer the first part.