CS 573: Algorithms, Fall 2009

Homework 2, due Monday, October 5, 23:59:59, 2009

Version 1.0

Name:	
Net ID:	Alias:

#	Score	Grader
1.		
2.		
3.		
4.		
5.		
Total		

Neatly print your name(s), NetID(s), and the alias(es) you used for Homework 0 in the boxes above. Staple this sheet to the top of your homework. If you are on campus, submit the homework by submitting it in SC 3306 (or sliding it under the door).

This homework should be easier than hw0. You are encouraged to discuss problems in this homework with people, but should submit your homework on your own.

To acknowledge the corn – This purely American expression means to admit the losing of an argument, especially in regard to a detail; to retract; to admit defeat. It is over a hundred years old. Andrew Stewart, a member of Congress, is said to have mentioned it in a speech in 1828. He said that haystacks and cornfields were sent by Indiana, Ohio and Kentucky to Philadelphia and New York. Charles A. Wickliffe, a member from Kentucky questioned the statement by commenting that haystacks and cornfields could not walk. Stewart then pointed out that he did not mean literal haystacks and cornfields, but the horses, mules, and hogs for which the hay and corn were raised. Wickliffe then rose to his feet, and said, "Mr. Speaker, I acknowledge the corn".
– Funk, Earle. "A Hog on Ice and Other Curious Expressions

Required Problems

1. Greedy algorithm does not work for independent set.
   [20 Points]

   A natural algorithm, **GreedyIndep**, for computing maximum independent set in a graph, is to repeatedly remove the vertex of lowest degree in the graph, and add it to the independent set, and remove all its neighbors.

   (a) [**5 Points**] Show an example, where this algorithm fails to output the optimal solution.

(b) **[5 Points]** Let G be a $(k, k+1)$-uniform graph (this is a graph where every vertex has degree either k or $k+1$). Show that the above algorithm outputs an independent set of size $\Omega(n/k)$, where n is the number of vertices in G.

(c) **[5 Points]** Let G be a graph with average degree δ (i.e., $\delta = 2\left|E(G)\right| / \left|V(G)\right|$). Prove that the above algorithm outputs an independent set of size $\Omega(n/\delta)$.

(d) **[5 Points]** For any integer k, present an example of a graph G_k, such that **GreedyIndep** outputs an independent set of size $\leq \left|OPT(G_k)\right| / k$, where $OPT(G_k)$ is the largest independent set in G_k. How many vertices and edges does G_k has? What it the average degree of G_k?

2. Greedy algorithm does not work for coloring. Really.
[20 Points]

Let G be a graph defined over n vertices, and let the vertices be ordered: $v_1, \ldots, v_n$. Let G_i be the induced subgraph of G on $v_1, \ldots, v_i$. Formally, $G_i = (V_i, E_i)$, where $V_i = \{v_1, \ldots, v_i\}$ and

$$E_i = \left\{ uv \in E \;\middle|\; u, v \in V_i \text{ and } uv \in E(G) \right\}.$$

The greedy coloring algorithm, colors the vertices, one by one, according to their ordering. Let k_i denote the number of colors the algorithm uses to color the first i vertices.

In the i-th iteration, the algorithm considers v_i in the graph G_i. If all the neighbors of v_i in G_i are using all the k_{i-1} colors used to color G_{i-1}, the algorithm introduces a new color (i.e., $k_i = k_{i-1} + 1$) and assigns it to v_i. Otherwise, it assign v_i one of the colors $1, \ldots, k_{i-1}$ (i.e., $k_i = k_{i-1}$).

Give an example of a graph G with n vertices, and an ordering of its vertices, such that even if G can be colored using $O(1)$ (in fact, it is possible to do this with two) colors, the greedy algorithm would color it with $\Omega(n)$ colors. (Hint: consider an ordering where the first two vertices are not connected.)

3. Graph coloring revisited

(a) **[5 Points]** Prove that a graph G with a chromatic number k (i.e., k is the minimal number of colors needed to color G), must have $\Omega(k^2)$ edges.

(b) **[5 Points]** Prove that a graph G with m edges can be colored using $4\sqrt{m}$ colors.

(c) **[10 Points]** Describe a polynomial time algorithm that given a graph G, which is 3-colorable, it computes a coloring of G using, say, at most $O(\sqrt{n})$ colors.

4. Find kth smallest number.
[20 Points]

This question asks you to design and analyze a *randomized incremental* algorithm to select the kth smallest element from a given set of n elements (from a universe with a linear order).

In an *incremental* algorithm, the input consists of a sequence of elements $x_1, x_2, \ldots, x_n$. After any prefix $x_1, \ldots, x_{i-1}$ has been considered, the algorithm has computed the kth smallest element in $x_1, \ldots, x_{i-1}$ (which is undefined if $i \leq k$), or if appropriate, some other invariant from which the kth smallest element could be determined. This invariant is updated as the next element x_i is considered.

Any incremental algorithm can be *randomized* by first randomly permuting the input sequence, with each permutation equally likely.

(a) [**5 Points**] Describe an incremental algorithm for computing the kth smallest element.

(b) [**5 Points**] How many comparisons does your algorithm perform in the worst case?

(c) [**10 Points**] What is the expected number (over all permutations) of comparisons performed by the randomized version of your algorithm? (Hint: When considering x_i, what is the probability that x_i is smaller than the kth smallest so far?) You should aim for a bound of at most $n + O(k \log(n/k))$. Revise (a) if necessary in order to achieve this.

5. MAJORITY TREE
[**20 Points**]

Consider a uniform rooted tree of height h (every leaf is at distance h from the root). The root, as well as any internal node, has 3 children. Each leaf has a boolean value associated with it. Each internal node returns the value returned by the majority of its children. The evaluation problem consists of determining the value of the root; at each step, an algorithm can choose one leaf whose value it wishes to read.

(a) Show that for any deterministic algorithm, there is an instance (a set of boolean values for the leaves) that forces it to read all $n = 3^h$ leaves. (hint: Consider an adversary argument, where you provide the algorithm with the minimal amount of information as it request bits from you. In particular, one can devise such an adversary algorithm.).

(b) Consider the recursive randomized algorithm that evaluates two subtrees of the root chosen at random. If the values returned disagree, it proceeds to evaluate the third subtree. If they agree, it returns the value they agree on. Show the expected number of leaves read by the algorithm on any instance is at most $n^{0.9}$.