Midterm Exam 1

Due by 12 noon, Friday, 10 October 2014

- **Honor code:** This exam is open-notes and open-book. However, *searching for solutions or ideas online, and collaboration of any kind are strictly prohibited.* ("Collaboration" includes, for example, discussion or exchange of material related to the problems on the exam with anyone other than the instructor.) If you have any questions or clarifications, please ask me.
- **Policy on Cheating:** Students who violate the above rules on scholastic honesty are subject to disciplinary penalties. Any student caught cheating will receive an **F** (failing grade) for the course, and the case will be forwarded to the Office of Student Conduct and Conflict Resolution.
- **Presentation of solutions:** While describing an algorithm, you may use any of the algorithms covered in class or in the text as a subroutine, without elaboration.
- **A note on grading:** Your grade on any problem that asks you to design an algorithm will be determined on the basis of its correctness, running time, and the clarity of the description. In case you are not able to present an algorithm that has the desired properties, give the best algorithm you have designed. Show your work, as partial credit may be given.
- **Problems and Points:** There are three problems, worth a total of 20 points. This exam counts for 20% of the total grade.
- **Good Luck!**

Problem 1. (3 + 3 = 6 points) Proofs and counterexamples

In each of the following parts, indicate True or False. If your answer is True, provide a proof. Otherwise, provide a counterexample.

(a) Let G be a connected graph with a positive weight on each edge. Let T be a minimum spanning tree of G. Let C be any cut of G, and let e be any edge of T crossing C. Then e is the minimum-weight edge crossing C.

(b) In any undirected connected graph G, there exists a vertex v, such that if we remove v and all its incident edges, the remainder of the graph is connected.

Problem 2. (4 + 4 = 8 points) Computing pairs in a set of integers

You are given a set S of n distinct positive integers, in arbitrary order. Each of these parts asks you to give an algorithm to find a desired pair of integers from S. Give the most efficient algorithm you can for each of the problems. The more efficient your algorithm is in terms of its worst-case running time, the more credit you will get.

(a) Find two distinct elements a and b of S, with $a > b$ such that a/b is at most x/y for any distinct integers $x, y \in S$ such that $x > y$.

(b) Let M and m be the largest and the smallest elements, respectively, in S. Find two distinct elements a and b such that $|a - b|$ is at most $(M - m)/(n - 1)$.

Problem 3. (6 points) Modeling a distance function by a hierarchy

Suppose you have a set S of n points. We say that $d : S \times S \to \mathbf{Z}$ is a *distance* function if $d(i,j) = d(j,i) > 0$ for all $i \neq j$, and $d(i,i) = 0$ for all i.

Such distance functions are very general. It is often easier to work with distances that are based on some hierarchical organization. We say that a distance function δ over S is *hierarchical* if it can be constructed as follows. We build a rooted tree T over n leaves, with one leaf for each of the n points. Each node v in the tree has a label $\ell(v)$. The label of every leaf node is 0, and all the labels satisfy the following condition: the label of a node is less than the label of its parent. The distance $\delta(i,j)$ between two distinct points i and j is given by the label of the least common ancestor of i and j in T.

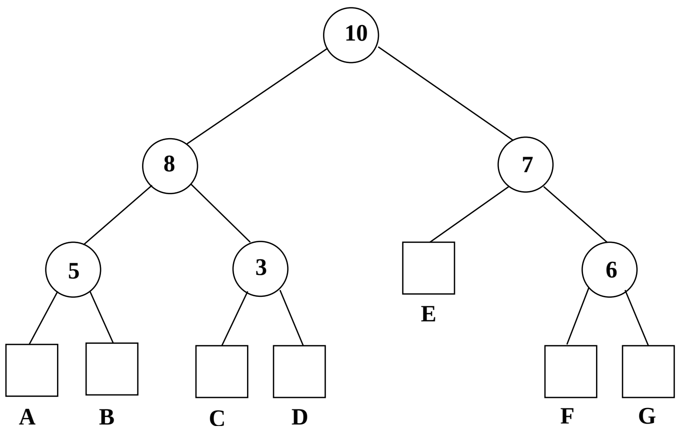

Figure 1: Illustration of the tree hierarchy. The machines are the (rectangular) leaves. The label of each non-leaf node is shown; these labels which are used in calculating the distances. For instance, $\delta(C,E)$ is 10 while $\delta(E,G)$ is 7.

Given two distance functions δ and d over a set S of points, we say that δ is *dominated by* d if $\delta(i,j) \leq d(i,j)$ for all $i, j \in S$.

The goal of this problem is to design, for a given distance function d over S, a hierarchical distance function δ that is dominated by d.

It is trivial to come up with one such δ; can you think of it? But we want to come up with a "good" one. Define the *goodness* of a distance function δ to be $\sum_{i,j \in S} \delta(i,j)$.

Give a polynomial-time algorithm that takes as input a set S of n points, a distance function d over S, and returns a maximum-goodness hierarchical distance function δ over S that is dominated by d.