Problem Set 1

Randomized Algorithms

Due Tuesday, September 19

1. [Karger] Suppose we have access to a source of unbiased random bits. This problem looks at constructing biased coins or dice from this source.

   (a) Show how to construct a biased coin, which is 1 with probability p and 0 otherwise, using $O(1)$ random bits in expectation. [Hint: First show how to construct a biased coin using an arbitrary number of random bits. Then show that the expected number of bits examined is small.]

   (b) Show how to sample from $[n]$, with probabilities $p_1, \ldots, p_n$, using $O(\log n)$ random bits in expectation.

   (c) Show that the "in expectation" caveat is necessary: for example, one cannot sample uniformly over $\{1, 2, 3\}$ using $O(1)$ bits in the worst case.

2. [MR 1.8]. Consider adapting the min-cut algorithm of the first class to the problem of finding an s–t min-cut in an undirected graph. In this problem, we are given an undirected graph G together with two distinguished vertices s and t. An s–t min-cut is a set of edges whose removal disconnects s from t; we seek an edge set of minimum cardinality. As the algorithm proceeds, the vertex s may get amalgamated into a new vertex as the result of an edge being contracted; we call this vertex the s-vertex (initially s itself). Similarly, we have a t-vertex. As we run the contraction algorithm, we ensure that we never contract an edge between the s-vertex and the t-vertex.

   (a) Show that there are graphs (not multi-graphs) in which the probability that this algorithm finds an s–t min-cut is exponentially small.

(b) How large can the number of s–t min-cuts in an instance be?

3. [MR 2.3]. 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.

   (b) Show that there is a nondeterministic algorithm can determine the value of the tree by reading at most $n^{\log_3 2}$ leaves. In other words, prove that one can present a set of this many leaves from which the tree value can be determined.

   (c) 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 sub-tree. Show the expected number of leaves read by the algorithm on any instance is at most $n^{0.9}$.