6.856 — Randomized Algorithms

David Karger

Handout #9, Mar. 9, 2011 — Homework 5, Due 3/16

1. **This problem should be done without collaboration.** Bloom filters can be used to estimate the difference between two sets. Suppose that you have sets X and Y, each with m elements, and with r elements in common. Create an n-bit Bloom filter for each, using the same k hash functions. Determine the expected number of bits where the two Bloom filters differ, as a function of m, n, k, and r. Explain how this could be used as a technique for estimating r.

2. MR7.2. Two rooted trees T_1 and T_2 are said to be isomorphic if there exists a one to one mapping f from the nodes of T_1 to those of T_2 satisfying the following condition: v is a child of w in T_1 if and only if $f(v)$ is a child of $f(w)$ in T_2. Observe that no ordering is assumed on the children of any vertex. Devise an efficient randomized algorithm for testing the isomorphism of rooted trees and analyze its performance. **Hint:** Associate a polynomial P_v with each vertex v in a tree T. The polynomials are defined recursively, the base case being that the leaf vertices all have $P = x_0$. An internal vertex v of height h with children $v_1, \ldots, v_k$ has its polynomial defined to be

$$(x_h - P_{v_1})(x_h - P_{v_2}) \cdots (x_h - P_{v_k}).$$

   Note that there is exactly one indeterminate for each level in the tree.

3. Consider the problem of finding a *minimum weight* (total weight of included edges) perfect matching in a bipartite graph whose edges are given integer weights of magnitude bounded by a polynomial in the number of vertices n. Note that it is not possible to apply the Isolating Lemma directly to this case since the random weights chosen there would conflict with the input weights.

   (a) Explain how you would devise an **RNC** algorithm for this problem. **Hint:** start by scaling up the input edge weights by a large polynomial factor. Apply random perturbations to the scaled weights and prove a variant of the Isolating Lemma for this situation.

   (b) The parallel complexity of the version where the edge weights can have a polynomial number of *bits* has not yet been resolved. Note that arithmetic operations on such weights are still tractable. Explain why the **RNC** algorithm you developed above does not work in this case.

(c) Devise an **RNC** algorithm for finding a *maximum matching* (i.e., most possible edges) in a graph (without weights) that may not have a perfect matching. **Hint:** use the min-weight perfect matching algorithm above as a "black box" by making nonexistent edges very expensive.

4. Suppose you are given a graph whose edge lengths are all integers in the range from 0 to B. Suppose also that you are given the all-pairs distance matrix for this graph (it can be constructed by a variant of Seidel's deterministic distance algorithm). Prove that you can identify the (successor matrix representation of the) shortest paths in $O(B^2 MM(n) \log^2 n)$ time, where $MM(n)$ is the time to multiply $n \times n$ matrices.

5. **Optional.** In the *exact matching* problem, a bipartite graph is given with a subset of the edges colored red, along with an integer k. The goal is to find a perfect matching with exactly k red edges. Devise an **RNC** algorithm for this problem using a (non-trivial) application of the Isolating Lemma. Note that this problem is not known to be solvable in **P**.