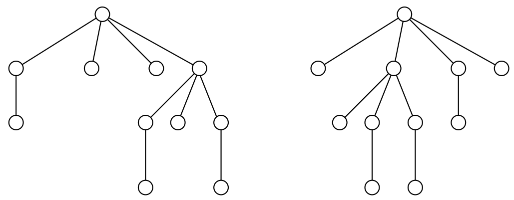
CS 466/666, Spring 2007
Timothy Chan

Assignment 3 (due June 27 Wednesday noon)

Please read `http://www.student.cs.uwaterloo.ca/~cs466/policies.html` first for general instructions.

1. [*15 marks*] Given two rooted trees T_1 and T_2, we want to decide whether T_1 and T_2 are *isomorphic* (i.e., equivalent up to reordering the children of every node). For example, the answer is yes for the two trees below:



   (a) [*5 marks*] First consider the subproblem of determining whether two lists are identical up to reordering (e.g., $\langle 3,5,9,3,4\rangle$ and $\langle 5,4,3,3,9\rangle$ are identical). Specifically, consider the following randomized approach: Fix a prime $p > n$. For a list $L = \langle a_1, \ldots, a_n\rangle$ of integers from $\{0, \ldots, p-1\}$ (where a value may appear multiple times), define its fingerprint to be $F_x(L) = (x - a_1)(x - a_2)\cdots(x - a_n) \bmod p$.

   Given two lists L_1 and L_2 each of n integers from $\{0, \ldots, p-1\}$, pick a new random integer $x \in \{0, \ldots, p-1\}$. Prove that if $F_x(L_1) = F_x(L_2)$, then the probability that L_1 and L_2 are not identical up to reordering is at most $(n-1)/p$.

   [Note: you may cite known results from class.]

   (b) [*10 marks*] Now design and analyze a linear-time Monte Carlo algorithm for deciding whether two rooted trees with n nodes are isomorphic. The probability of error should be at most $1/n$, and your algorithm is allowed to do arithmetic operations in constant time for $O(\log n)$-bit numbers only.

   [Hint: use part (a) repeatedly in a bottom-up manner...]

2. [*10 marks*] Consider the following *approximate median* problem: given a set S of n numbers, find an element of rank between $0.4n$ and $0.6n$ (the rank of x is the number of elements in S strictly less than x). Prove that any comparison-based deterministic algorithm must make at least $1.2n - o(n)$ comparisons in the worst case.

   [Hint: modify the $1.5n$ adversary-based lower bound proof for exact median from class.]

3. [*20 marks*] We are given n arrays, where each array contains n elements in increasing order. We want to report the first k smallest elements among all n^2 elements for a given k. These k elements do *not* need to be outputted in sorted order. You may assume the elements are distinct. (For example, if $n = 3$, the 3 arrays are $\langle 2,10,22\rangle$, $\langle 5,11,20\rangle$, and $\langle 1,30,32\rangle$ and $k = 5$, then we may output $2, 1, 10, 11, 5$.)

(a) [*3 marks*] Describe a deterministic algorithm for the above problem that runs in $O(n + k \log n)$ time. [Hint: use a standard heap.]

(b) [*6 marks*] Let S be a set of N elements. Choose random elements $x_1, \ldots, x_r$ independently from S, and let x^* be the minimum of $x_1, \ldots, x_r$. Let k^* denote the rank of x^* in S. Prove that

- the probability that $k^* \leq N/(2r)$ is at most 0.5.
- the probability that $k^* \geq 2N/r$ is at most 0.15.

[Hint: the inequality $1 - x \leq e^{-x}$ might come in handy.]

(c) [*4 marks*] Describe a (deterministic) algorithm for the above problem that runs in $O(n + k^*)$ time, under the special case where we are also given an element x^* of rank k^* with $k^* \geq k$. [Hint: use a linear-time selection algorithm.]

(d) [*7 marks*] Now design and analyze a faster Las Vegas algorithm for the above problem (in general case) that runs in $O(n + k)$ expected time. [Hint: if $k > n$, then use (b) with $N = n^2$ and $r = N/(2k)\ldots$]