CS 317, Assignment 7

Due on 28 November in class

1. An AVL tree is a binary search tree where the following property holds for every node v in the tree: the difference between the height of the left child of v and height of the right child of v is at most 1.

   As in class, you are given a sorted array $A_{1\ldots n}$ of search keys (or strings) along with an array $p_{1\ldots n}$ with the respective search probabilities.

   (a) Design a backtracking algorithm that returns all the possible AVL trees for the given array of keys. Establish the correctness and analyze the running times of your algorithm. Also comment on the optimality of your algorithm.

   (b) Modify the algorithm you developed in Question 1a so that it finds *one* AVL tree for the given array of keys as efficiently as possible. Establish the correctness and analyze the running times of your algorithm.

   (c) Design now a backtracking algorithm that returns the optimal AVL tree for the given array of keys as quickly as possible. Establish the correctness and analyze the running times of your algorithm.

   (d) Design a nondeterministic algorithm that receives the two arrays $A_{1\ldots n}$ and $p_{1\ldots n}$ as above and also a positive real number T. The algorithm returns TRUE if and only if there exists an AVL tree for $A_{1\ldots n}$ whose average search time is less than or equal to T. Establish the correctness of your algorithm. Give an argument that the running time of your algorithm is polynomial.

   Note in passing that there is usually no need to go into a more precise run time analysis for nondeterministic algorithms, since their purpose is to make a point (such as establish membership in $\mathcal{NP}$), not to be efficient.

2. We talked in class about the complexity classes $\mathcal{P}$ and $\mathcal{NP}$ together with the concept of polynomial reductions being used to define the hardest problems in $\mathcal{NP}$ (that is, the $\mathcal{NP}$-complete problems). The concept of complete problems is not restricted to these classes, but can be defined for any pair of classes X and Y such that $X \subseteq Y$ as follows:

   - A problem π is Y-hard iff for all $\pi' \in Y$ it holds that $\pi' \trianglelefteq \pi$.
   - A problem π is Y-complete iff π is Y-hard and $\pi \in Y$.

   You will notice that the only missing piece is the definition of the $\trianglelefteq$ reduction. Define a suitable reduction to be used in the definition above. Explain carefully how your reduction is suitable for the purpose of supporting an eventual proof that either $X \subsetneq Y$ or $X = Y$ (as the case might turn out to be).

Make sure you review the submission guidelines posted on the course's Web site before submitting.