

CS 213

Homework Assignment 7

Given: March 05, 2009

Due: March 12, 2009

This assignment is due by the end of the class on the due date. Unless all problems carry equal weight, the point value of each problem is shown in []. To receive full credit all your answers should be carefully justified. Each solution must be the student's own work. Assistance should be sought or accepted only from the course staff. Any violation of this rule will be dealt with harshly.

1. There may be several different min-cut sets in a graph. Using the analysis of the randomized min-cut algorithm, argue that there can be at most $n(n-1)/2$ distinct min-cuts sets.
2. Give a smallest possible counterexample to the following claim.

Consider the following property of binary search trees. Suppose that the search for key k in a binary search tree ends up in a leaf. Consider three sets: A , the keys to the left of the search path; B , the keys on the search path; and C , the keys to the right of the search path. Then any three keys $a \in A$, $b \in B$, and $c \in C$ must satisfy $a \leq b \leq c$.

3. Given two strings $a = a_0a_1 \dots a_p$ and $b = b_0b_1 \dots b_q$, where each a_i and each b_j is in some ordered set of characters, we say that a string is *lexicographically less than* string b if either

1. there exists an integer j , where $0 \leq j \leq \min(p, q)$, such that $a_i = b_i$ for all $i = 0, 1, \dots, j-1$ and $a_j < b_j$, or
2. $p < q$ and $a_i = b_i$ for all $i = 0, 1, \dots, p$.

For example, if a and b are bit strings, then $10100 < 10110$ by rule 1 (letting $j = 3$) and $10100 < 101000$ by rule 2. This is similar to the ordering used in English-language dictionaries.

The *binary string tree* data structure shown in the figure below stores the bit strings 1011, 10, 011, 100, and 0. When searching for a key $a = a_0a_1 \dots a_p$, we go left at a node of depth i if $a_i = 0$ and right if $a_i = 1$. Let S be a set of distinct binary strings whose lengths sum to n . Show how to use a binary string tree to sort S lexicographically in $\Theta(n)$ time. For the example in the figure, the output of the sort should be the sequence 0, 011, 10, 100, 1011.

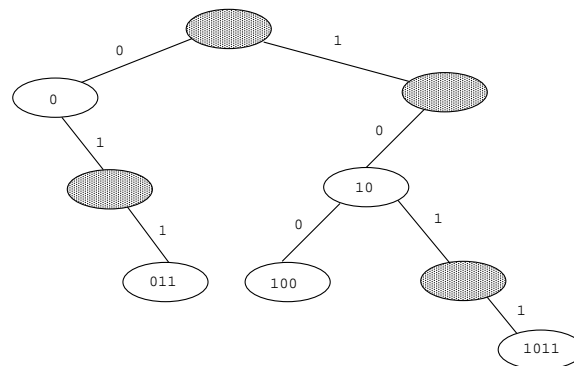


Figure 1: A binary string tree storing the bit strings 1011, 10, 011, 100, 0. Each node's key can be determined by traversing the path from the root to that node. There is no need, therefore, to store the keys in the nodes; the keys are shown here for the illustrative purposes only. Nodes are heavily shaded if the keys corresponding to them are not in the tree; such nodes are present only to establish a path to other nodes.