15-451 Algorithms, Spring 2007

Homework # 3 **due: Thursday, March 8, 2007**

Please hand in each problem on a **separate** sheet and put your **name**, **andrew id** and **recitation** (time or letter) at the top of each page. You will be handing each problem into a separate box, and we will then give homeworks back in lecture. If a problem takes up more than one sheet of paper, you must **staple** all sheets of paper for that problem together.

If you DO NOT follow these instructions exactly, you will lose up to 30pts.

Remember: Group work is allowed, however each student must hand in a separate write-up. Moreover, you must explicitly state where you got your ideas from. The answers should always be in your own words.

Note: For every algorithm that you give briefly explain why it is correct.

Problems:

(20 pts) 1. **Augmented Binary Search Tree.** Call a binary search tree augmented if every node v in the tree also stores the size of the subtree rooted at v.

(a) Show that a rotation in an augmented binary search tree can be performed in constant time.

(b) Suppose you are given an augmented treap. Describe an algorithm $SELECT(k)$ which given an integer k, returns the kth smallest item in the treap in $O(\log n)$ expected time.

(30 pts) 2. **Dynamic Programming.** Let us define a multiplication operation on a finite size alphabet $\Sigma = \{s_1, \ldots, s_k\}$ by a multiplication table T, such that the result of multiplying s_i by s_j is stored in entry $T[i][j]$ (this result is some element of Σ). For example, the multiplication table for an alphabet $\{a, b, c\}$ might look like:

	a	b	c
a	b	b	a
b	c	b	a
c	a	c	c

In the above, the result of multiplying b by c is a. Notice that the multiplication operation defined by the table is not necessarily associative or commutative.

Find an efficient (polytime) algorithm that given a string $x = x_1 x_2 \ldots x_n$ of symbols from Σ, a multiplication table T, and a symbol $r \in \Sigma$, decides whether or not it is possible to parenthesize the string such that the value of the resulting expression (when using the multiplication table T) is r.

For example, given $x = bbbbac$, the above table, and $r = a$, your algorithm should return yes since $((b(bb))(ba))c = a$.

What is the running time of your algorithm in terms of n and k?

(30 pts) 3. **Strings and Sequences.**

(a) Given two strings $x = x_1x_2\dots x_n$ and $y = y_1y_2\dots y_m$ over some alphabet, a *common supersequence* of x and y is a string z such that both x and y appear in z as subsequences. That is, for each index $i = 1, \dots, n$ and $j = 1, \dots, m$, there is an indices p_i and q_j such that

- for all $i_1, i_2 \in \{1, \dots, n\}, i_1 < i_2 \Rightarrow p_{i_1} < p_{i_2}$, and
- for all $j_1, j_2 \in \{1, \dots, n\}, j_1 < j_2 \Rightarrow q_{j_1} < q_{j_2}$, and
- for all $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, m\} \Rightarrow z_{p_i} = x_i$ and $z_{q_j} = y_j$.

For example, for $x = abacb$ and $y = bichhb$, possible supersequences are *abacbichhb* and *abaichhb*.

Give an efficient algorithm to compute the length of the *shortest* common supersequence of two given strings of length m and n. Can you easily obtain an algorithm for the shortest supersequence of x and y if you use the algorithm given in class for computing the longest subsequence as a black box?

(b) Given two strings $x = x_1x_2\dots x_n$ and $y = y_1y_2\dots y_m$ over some alphabet, a *common substring* of x and y is a string $z = z_1z_2\dots z_s$ for which there are indices k and ℓ such that

- for all $i = 1, \dots, s$, $x_{k+i-1} = z_i$, and $y_{\ell+i-1} = z_i$.

For example, for $x = abbad$ and $y = adbbatt$ some possible substrings are *ad* and *bba*, but not *abb*.

Give a dynamic programming algorithm to compute the length of the *longest* common substring of two given strings of length m and n.

(c) Given two strings $x = x_1x_2\dots x_n$ and $y = y_1y_2\dots y_m$ over some alphabet, a *common superstring* of x and y is a string z such that both x and y appear in z as substrings. That is, there are indices k and ℓ such that

- for all $i = 1, \dots, n$, $z_{k+i-1} = x_i$, and
- for all $j = 1, \dots m$, $z_{\ell+j-1} = y_j$.

Give an $O(mn)$ time algorithm to compute the length of the *shortest* common superstring of two given strings of length m and n. Can you use the algorithm from part (b) as a black box to compute the shortest superstring of x and y? Why or why not?

(20 pts) 4. **Two Graph Problems.**

(a) A Hamiltonian path in a directed graph $G = (V, E)$ is a path going through each vertex of G exactly once. That is, it is a sequence of vertices $P = v_1 v_2 \dots v_n$ such that

- for every $i = 1, \dots, n-1$, $(v_i, v_{i+1}) \in E$,
- for all $i, j \in \{1, \dots, n\}, i \neq j$, $v_i \neq v_j$, and
- $|V| = n$.

Also, recall that a directed acyclic graph (DAG) is a directed graph containing no (directed) cycles. Give a linear time algorithm which given a DAG $G = (V, E)$, determines whether G contains a Hamiltonian path.

(b) A *vertex cover* of a graph $G = (V, E)$ is a subset of the vertices $S \subseteq V$ that includes at least one end point of every edge in E. Recall that a tree $T = (V, E)$ is a connected undirected graph with no cycles. Give a linear time algorithm which takes a tree T as an input and returns a vertex cover of T of smallest size. For instance, consider the following tree:

```
  A   D
 / \ /
B   E   G
|    \ /
C     F
```

The possible vertex covers include $\{A, B, C, D, E, F, G\}$ and $\{A, C, D, E, F\}$ but not $\{C, E, F\}$. A smallest vertex cover is $\{B, E, G\}$.

HINT: Start by considering the leaves.