Algorithms, Spring 1997

Assignment 2

Due in one week.

There are two assigned problems which you should turn in and two practice problems which you should do, but not turn in . I give a sample solution to another problem; your solutions to the first two should have about the same level of detail as the sample solution.

Do : Problem 1 : 4.10 of Manber.

Do : **Problem 2** Design a data structure to support each of the following operations in $O(\log n)$ time :

Insert (x) (with the usual meaning)

Delete (x) (with the usual meaning)

Find-rank (x) : should return the number of elements less than or equal to x in current data structure if x is present. Otherwise should return "not present".

Solve, but do not turn in : 4.8 and 4.20 of Manber.

Sample Answer for Problem 4.15 of Manber : (Manber gives an answer at the end of the book; I have modified it because it is not very clear.)

We use an AVL tree to store the data. So, insertions and deletions can be done in time $O(\log n)$ time each. To do $Find\_Next(x)$, we use the regular search algorithm for x. There are three cases :

Case 1 The search finds x as the key at an internal node v of the tree. Then we go the right child of v and then on keep going left until we reach a leaf. This leaf is the successor of v in the in-order traversal and so has the least key greater than x as required.

Case 2 The search reaches a leaf v and the key at v is greater than x. Then we return this key.

Case 3 The search reaches a leaf v and the key at v is less than or equal to x. Then the node we are looking for is the last node on the path from the root to v where we branched to the left (if we always branched to the right, no key in the tree is larger than x). So, as we do the search for x, we keep track of the last node where we branched left and this then gives us the answer.