Homework 2

Released on September 23, 2016

1. Let T be a binary search tree whose keys are distinct. Let x be a leaf node and y be its parent. Show that y is a successor of x or x is a successor of y.

2. We may implement inorder tree walk of a binary search tree as follows: Find the minimum key first, then make $n-1$ calls to TREE-SUCCESSOR. Write down the pseudocode and prove that the running time is $\Theta(n)$.

3. We can sort a given set of n numbers by first building a binary search tree containing these numbers (using TREE-INSERT repeatedly to insert the numbers one by one) and then printing the numbers by an inorder tree walk.

1. What are the worst-case and best-case running times for this soring algorithm?
2. What are the worst-case and best-case running times when using red-black tree (instead of binary search tree) in the above algorithm?

4. Show that any arbitrary n-node binary search tree can be transformed into any other arbitrary n-node binary search tree using $O(n)$ rotations.
Hint: First show that at most $n-1$ right rotations suffice to transform the tree into a right-going chain.

5. An AVL tree is a binary search tree that is *height balanced*: for each node x, the heights of the left and right subtrees of x differ by at most one. To implement an AVL tree, we maintain an extra attribute in each node: x.h the height of node x.

1. Prove that an AVL tree of height h has at least F_h nodes, where F_h is the hth Fibonacci number. Further show that an AVL tree with n nodes has height $O(\log n)$. (You can use the fact that F_h is the closest integer to $\left(\frac{1+\sqrt{5}}{2}\right)^h/\sqrt{5}$.)
2. To insert into an AVL tree, we first place a node into the appropriate place in binary search tree order. Afterwards, the tree might be no longer balanced. Specifically, the heights of the left and right children of some node might differ by 2. Describe a procedure BALANCE(x), which takes a subtree rooted at x with $|x.\text{right.h} - x.\text{left.h}| \leq 2$, and alters the subtree rooted at x to be height balanced. Note: You do not need to write pseudocode. Just make case distinctions and describe your operations for each case.
   Hint: You may assume $x.right.h > x.left.h$ first, as the other case can be analyzed similarly. Under this assumption, let $y = x.right$. Then consider two cases $y.left.h > y.right.h$ and $y.left.h < y.right.h$ respectively. Use left or right rotations.

6. Following Q5.

1. Using part (2) of Q5, describe a recursive procedure AVL-INSERT(x, z) that takes a node x within an AVL tree and a newly created node z (whose key has already been filled in), and adds z to the subtree rooted at x, maintaining the property that x is the root of an AVL tree. (Assume z.left = z.right = nil and z.h = 0 and you may use sentinel T.nil as nil if necessary). Thus, to insert z into the AVL tree T, we call AVL-INSERT(T.root, z).
2. Show that AVL-INSERT takes $O(\log n)$ time and performs $O(1)$ rotations.

7. Use a recursion tree to determine a good asymptotic upper bound on the recurrence $T(n) = T(n-1) + T(n/2) + n$. Use the substitution method to verify your answer.