Homework 2

Fundamental Algorithms, Spring 2003, Professor Yap

Due: Feb 24, in class

INSTRUCTIONS:

- Please read questions carefully. When in doubt, please ask.

---

1. (20 Points)
   Solve using the Master Theorem:
   (a) $T(n) = 3T(n/8) + 1$.
   (b) $T(n) = 3T(n/8) + \sqrt{n}\log n$.
   Be careful to justify all conditions required by the Master Theorem.

2. (10 Points)
   Give tight upper and lower bounds for the following:
   (a) $T(n) = 4T(n/2) + n^2/\log n$.
   (b) $T(n) = 4T(n/2) + n^2 \log n$.
   NOTE: we DO NOT want you to transform these recurrences to solve them, nor to use real induction. Just use the Master theorem, or from remarks like $T(n) = \Omega(d(n))$ where $d(n)$ is the driving function.

3. (15 Points)
   Let $T(n) = 2T(\frac{n}{2} + c) + n$ for some $c > 0$. You are told that this recurrence holds for $n \geq n_0$, and that for all $n < n_0$, we have $T(n) \leq C$ for some constant C. Prove that $T(n) \leq D(n-2c)\lg(n-2c)$ for D large enough. As usual, lg means $\log_2$. Describe carefully how you would choose D.

4. (20 Points)
   (a) Prove that $\sum_{i=1}^{n} \frac{\log i}{i} = \Theta(\log^2 n)$. Use the rules for polynomial-type or exponential-type sums.
   (b) Let $T(n) = aT(n/b) + w(n)\frac{\log\log n}{\log n}$ where $w(n) = n^{\log_b a}$ is the watershed function. Prove that $T(n) = O((\log\log n)^2)$. You must use range and domain transformation methods to solve this. You can combine the two transformations into one, by defining, for all $k \geq 0$,
   $$s(k) = \frac{T(b^k)}{a^k}.$$

5. (15 Points)
   (a) Implement the rotation operation in pseudo-code. Make the following assumption: each node u has three pointers, $u.Parent$, $u, Left$ and $u.Right$, with the obvious meaning. $u.Parent = null$ iff u is the root, and $u.Left = u.Right = null$ iff u is a leaf.
   (b) How may assignments did you perform in (a)? Can you argue that you have used the minimum number of assignments possible?

6. (15 Points)
   Exercise 3.3 (a) and (b) in Lecture III. Rotations to transform any binary search tree into any other equivalent one. Recall that two BST are equivalent if they store exactly the same set of keys.

7. (30 Points)
   (a) Insert the keys $1, 2, 3, \ldots, 14, 15$ into an AVL Tree, in this order. Please draw your tree at the end of each insertion.
   (b) Suppose you continue inserting in this manner indefinitely. Prove that when you insert key $2^n - 1$, you will have a complete binary tree (i.e., every leaf is at the same level).
   (c) Consider the AVL tree T_{12} after you have inserted keys $1, 2, \ldots, 11, 12$ as in (a). What is the minimum number of keys you need to delete from T_{12} before you cause a double rotation to happen? Please specify the order of keys you need to delete, and show the tree after each deletion. The last deletion must cause a double-rotation.

8. (30 Points) Exercise 4.9 (Lect.III), parts (a) and (b) only. For the algorithm in (b), describe it in the style that we use to explain the insertion algorithm. Draw pictures to help explain your ideas.