CS 5633 Analysis of Algorithms – Spring 06

1/26/06

2. Homework

Due **2/2/06** before class

Always justify your answers.

1. ***d*-Heaps (8 points)**
   A *d-ary max-heap*, d-heap for short, is the generalization of a binary heap to a d-ary tree. The tree still has to be almost complete, and for every child of a parent the child's value is less or equal than the parent's value.

   (a) **(4 points)** Suppose a d-heap is stored in an array (that begins with index 1, not with index 0). For an entry located at index i in which location is its parent and in which locations are its children? (You do not have to formally prove your answer, but please give at least an example)

   (b) **(2 points)** What is the height of a d-heap that contains n elements? The height should be a function of n and d.

   (c) **(1 point)**What is the Θ-runtime of inserting an element into a d-heap of n elements? The runtime should be a function of n and d (so, do not consider d as a constant).

   (d) **(1 point)**What is the Θ-runtime of extracting the maximum from a d-heap of n elements? The runtime should be a function of n and d (so, do not consider d as a constant).

2. **Divide and Conquer (5 points)**

   (a) **(3 points)** Design a divide-and-conquer algorithm to find the position of the smallest element in an array of n distinct numbers. (Describe your algorithm in pseudo-code with verbal explanation.)

   (b) **(2 points)** Set up and solve a recurrence relation for the runtime of your algorithm. (You do not need to prove it by induction. A justification for your guess is enough.)

3. **3-Way Mergesort (3 points)**
   Consider a variant of Mergesort which divides the array into three parts, recurses on each of the three parts, and finally merges all three sorted parts. Merging still takes linear time. What is the runtime recurrence of this algorithm? What does this recurrence solve to? (You don't have to prove the solution of the recurrence, but please give a short persuasive argument)

FLIP OVER TO BACK PAGE $\Longrightarrow$

4. **Guessing and Induction (9 points)**

For each of the following recurrences find a good guess of what it could solve to (make your guess as tight as possible). Use either the recursion tree method or the expansion method to find your guess. Then prove that $T(n)$ is in big-Oh of your guess by induction (inductive step and base case). Hint: Appendix A in the book has a list of solved summations that might be helpful. For simplicity you may want to use $\log_4 n$ instead of $\log_2 n$ for the first question.

Every recursion below is stated for $n \geq 2$, and the base case is $T(1) = 1$.

(a) **(3 points)**
$T(n) = 4T(\frac{n}{4}) + 3n$

(b) **(3 points)**
$T(n) = 2T(\frac{n}{2}) + 5n^2$

(c) **(3 points)**
$T(n) = 4T(\frac{n}{2}) + n$