

## Homework 2 – Due Friday, February 2, 2007

**Reminders** Collaboration is permitted, but you must write the solutions by yourself without assistance, and be ready to explain them orally to a member of the course staff if asked. You must also identify your collaborators. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

When you describe an algorithm, remember that clarity is the main goal. Pseudocode is preferred, but an English-language description is also acceptable if it is unambiguous.

**Exercises** These should not be handed in, but the material they cover may appear on exams.

- Asymptotic notation: CLRS Ex. 3.1-1, 3.1-2, Problems 3-1, 3-2, 3-3.
- Recurrences: CLRS Ex. 4.1-1, 4.1-2, 4.2-1, 4.2-4.
- Divide and Conquer: CLRS Problem 2-4.

### Problems to be handed in

1. (a) Rank the following functions by increasing order of growth, that is, find an arrangement  $g_1, \dots, g_{22}$  of the functions satisfying  $g_1(n) = O(g_2(n)), g_2(n) = O(g_3(n)), \dots$ . Break the functions into classes so that  $f$  and  $g$  are in the same class if and only if  $f(n) = \Theta(g(n))$ . Note that  $\log(\cdot)$  is the base 2 logarithm,  $\log_b(\cdot)$  is the base  $b$  logarithm, and  $\ln(\cdot)$  is the natural logarithm.

$$\begin{array}{cccccccc}
 18 \log_7 n & \log^2(n) & \log(n^2) & \log(n!) & n \log n & \sum_{i=5}^n (2i + 3) & & \\
 \\
 \left(\frac{3}{2}\right)^n & n! & \ln(\ln n) & n & 2^{\log n} & 4^{\log n} & & \\
 \sqrt{n} & 2^n & n^2 & n^{\log 7} & 2^{\sqrt[3]{n}} & \sum_{i=1}^n 2^i & & \\
 \\
 \sum_{i=1}^n \left(\frac{1}{2}\right)^i & (\log n)! & 2^{\log_4 n} & 2^{\log^2(n)} & & & & 
 \end{array}$$

- (b) For each of the following statements, decide whether it is always true, never true, or sometimes true for asymptotically nonnegative functions  $f$  and  $g$ . If it is always true or never true, explain why. If it is sometimes true, give one example for which it is true, and one for which it is false.
  - i.  $f(n) + g(n) = \Theta(\max(f(n), g(n)))$
  - ii.  $f(n) = O(f(n)^2)$

- iii.  $f(n) = \Omega(g(n))$  and  $f(n) = o(g(n))$  (note the little- $o$ )
  - iv.  $f(n) = O(g(n))$  or  $g(n) = O(f(n))$  (or both)
2. Suppose you are given two sorted arrays, each with  $n$  elements. There are  $2n$  values in total, and you may assume that no two values are the same. You would like to determine the median of this set of  $2n$  values. We define the median as the  $n^{\text{th}}$  smallest value.
- Give a divide and conquer algorithm which finds the median using as few queries as possible to the input. How many queries does your algorithm use asymptotically?
3. Given two polynomials

$$A(x) = \sum_{i=0}^n a_i x^i \quad \text{and} \quad B(x) = \sum_{i=0}^n b_i x^i$$

of degree at most  $n$ , you want to compute the product  $C(x) = A(x)B(x)$ . Polynomials are given as arrays of coefficients. For example, the polynomial  $A(x)$  is represented as  $[a_0 \ a_1 \ a_2 \ \cdots \ a_n]$ .

- (a) Write a formula for the coefficients of  $C(x)$  in terms of the coefficients of  $A(x)$  and  $B(x)$  (that is, the  $a_i$ 's and  $b_i$ 's).
  - (b) How much time does the straightforward algorithm take to multiply two polynomials? Assume that adding or multiplying two individual coefficients takes constant time.
  - (c) Give a divide-and-conquer algorithm for multiplying two polynomials in time  $O(n^{\log_2 3})$  (again, assume that adding and multiplying individual coefficients takes constant time).
4. Recurrences:

For each of the following recurrences: (i) Draw a recursion tree. Indicate the height of the tree as well as the approximate sum of the nodes at each level (e.g. for the integer multiplication recurrence, for  $k \geq 1$ , the sum of the level  $k$  nodes is roughly  $(3/2)^k cn$ ). What is the growth rate of  $T(n)$  based on the recursion tree?

- (a)  $T(n) = 7T(n/2) + n$
- (b)  $T(n) = T(n - 1) + n \log n$
- (c)  $T(n) = T(\sqrt{n}) + 1$
- (d)  $T(n) = \sqrt{n}T(\sqrt{n}) + n$

(ii) Pick two of the four recurrences above and use the substitution method to prove asymptotic upper and lower bounds for  $T(n)$ .

5. **(Extra credit.)** In the integer multiplication from Lecture 3, the divide step consists of splitting the arguments  $a$  and  $b$  into their most and least significant bits.

Consider instead breaking the integers into their even bits and odd bits (e.g. the binary integer 11001110 would be written  $10001010 + 01000100$ , but in each of the two pieces half of the 0's are redundant). Give a divide and conquer algorithm based on this decomposition which runs in time  $\Theta(n^{\log_2 3})$ .