BU CLA CS 530: Analysis of Algorithms
Spring 1995

Problem Set 3 (due Thursday 95.02.09)

---

1. Exercise 13.3-3, page 254 in [CLR].
2. Problem 13.2, page 260 in [CLR].
3. Problem 13.3, page 260 in [CLR].

The two next exercises are about recurrences. They are intended to be relatively easy and to review material studied in preceding courses.

4. In general it is sufficient to evaluate recurrences only for powers of 2 (or 3 or some other constant), but this is not always the case. Consider the following recurrence:

$$T(n) = \begin{cases} T(n/2) + 1, & \text{if } n \text{ even}, \\ 2 \cdot T((n-1)/2), & \text{if } n \text{ odd}, \end{cases}$$

   with $T(1) = 1$.

   a. Show that the solution of this recurrence for powers of 2 is $T(2^k) = 1 + k$, i.e. for all n that are powers of 2, $T(n) = 1 + \log\ n = \mathcal{O}(\log\ n)$.

   b. Show that, for an infinite number of values of n, it is the case that $T(n) = (n+1)/2 = \Omega(n)$.

   c. State a condition on a recurrence so that the usual restriction to powers of 2 (or some other constant) is sufficient, in order to correctly determine the asymptotic behavior of the solution for all n.

5. Fibonacci numbers are often encountered in the design and analysis of algorithms. (The definition is given page 36 in [CLR].) They can be viewed as the *range* of the solution of a certain recurrence over the non-negative integers. In many cases it is convenient to define F_n for *negative* n by assuming that the recurrence $F_{n+2} = F_{n+1} + F_n$ is over all integers n (and not only over the non-negative). Explore this possibility. What is F_{-1}? What is F_{-2}? Can F_{-n} be expressed in a simple way in terms of F_n?

Created: 95.02.02