CS 3500 Section A, Spring 2000

Homework 7 — due Tuesday, March 28

In problems that ask you to design an algorithm, give either the pseudocode, or an un-ambiguous description in plain English of your algorithm. For all problems, give the recurrence relation for the running time of the algorithm, clearly state its solution, and prove this solution using one of the methods studied in class.

Problem 1 Using a divide-and-conquer approach, design an algorithm for simultaneously finding the minimum and the maximum in an array of n elements with *exactly* $3n/2 - 2$ comparisons, where n is a power of 2.

Problem 2 Strassen's algorithm that we discussed in class (also presented in Section 31.2 of CLR) computes the product of two $n \times n$ matrices in time $O(n^{2.81})$, using the fact that the product of two 2×2 matrices can be found with only 7 multiplications instead of 8 with the normal matrix-multiplication algorithm.

Suppose we were to come up with a variant of Strassen's algorithm based on the fact that the product of two 3×3 matrices can be found with only m multiplications instead of the normal 27. Give the recurrence relation for the running time of this variant. How small would m have to be for this algorithm to be asymptotically faster than Strassen's algorithm?

Problem 3 This problem is on polynomial multiplication.

(a) Show how to multiply two linear polynomials $ax + b$ and $cx + d$ using only three multiplications. Hint: One of the multiplications is $(a + b) \cdot (c + d)$.

(b) Give a divide-and-conquer algorithm for multiplying two polynomials of degree n in time $\Theta(n^{\log_2 3})$. Hint: Split the input polynomials into two parts, based on the parity of the powers of x. Then use part (a), with polynomials of degree $n/2$ (in x^2) instead of constants.

Problem 4 Exercise 10.3–1, page 191 of CLR.

Problem 5 Exercise 10.3–8, page 192 of CLR.