

Problem Set 1

CS157 - Spring 2009

Due: Tuesday February 10, 2008 10:30am

Problem 1

1. Let $A = [a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7]$ and $B = [b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7]$ such that $a_i < a_j$ and $b_i < b_j$ for all $i < j$ and such that $a_i \neq b_j$ for all i, j . Given that $a_3 < b_3$ what are the possible positions of the 8th smallest element in $A \circ B = [a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7, b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7]$?
2. You are given two sorted lists A and B of size m and n , respectively. Give an $O(\log m + \log n)$ time algorithm for computing the k th smallest element in $A \circ B$. You may assume that every element of AB is unique. Feel free to assume the input is in any basic data structure (e.g. linked list, array, queue) and state what data structure you chose in the proof. Prove correctness and running time.

Problem 2

Consider problem 2.23 in the textbook on page 75. Using part b as inspiration to create a linear-time algorithm that solves the problem. Be sure to be precise! Prove correctness and running time.

Problem 3

In Section 1.2.3, we studied Euclid's algorithm for computing the *greatest common divisor* (gcd) of two positive integers: the largest integer which divides them both. Here we will look at an alternative algorithm based on divide-and-conquer.

1. Show that the following rule is true:
$$\text{gcd}(a, b) = \begin{cases} 2\text{gcd}(a/2, b/2) & \text{if } a, b \text{ are even} \\ \text{gcd}(a, b/2) & \text{if } a \text{ is odd and } b \text{ is even} \\ \text{gcd}((a-b)/2, b) & \text{if } a, b \text{ are odd and } b < a \\ \text{gcd}(b, a) & \text{otherwise} \end{cases}$$
2. Give an efficient divide-and-conquer algorithm for greatest common divisor.
3. How does the efficiency of your algorithm compare to Euclid's algorithm if a and b are n -bit integers? (In particular, since n might be large you cannot assume that basic arithmetic operations like addition take constant time).

Problem 4

1. Let matrix $M_1 = \begin{bmatrix} -3 & 3 \\ 2 & -2 \end{bmatrix}$ and vector \mathbf{v} be an 2-dimensional vector $[v_0, v_1]$ whose entries are randomly and independently chosen to be 0 or 1 (each with probability $1/2$). For what values of \mathbf{v} will $M_1\mathbf{v} = 0$?
2. Let matrix $M_2 = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \neq \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$. What are the upper and lower bounds of $\Pr[M_2\mathbf{v} \neq 0]$?
3. Now suppose you are given $n \times n$ matrices A, B, C and you wish to check whether $AB = C$. You can do this in $O(n^{\log_2 7})$ steps using Strassen's algorithm. Propose a faster algorithm that on input matrices A, B, C outputs either $AB \neq C$ or with probability 99.9% $AB = C$. Prove correctness and running time.