

Fall 08. CS513. HW 4, Due Oct 9. You can assume any of the algorithms we discussed in the class: you should precisely state any such result you use without going into details. Your goal is as always to design the fastest algorithm possible and to use as little space as possible.

1. (Simple Ones)
 - (a) Given 2 n -bit integers X and Y , design an algorithm to compute XY by dividing X and Y into 3 pieces. Does this improve the algorithm I described in the class?
 - (b) Given two matrices X and Y , each $n \times n$, design an algorithm to compute XY .
 - (c) Given sorted array $A[1..n]$ of distinct integers, design an efficient algorithm to find an index i if any such that $A[i] = i$.
2. The Hadamard matrices H_0, H_1, \dots are defined as follows. H_0 is the 1×1 matrix $[1]$. For $k > 0$, H_k is the $2^k \times 2^k$ matrix:

$$\begin{bmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{bmatrix}$$

Given v , a column vector of length $n = 2^k$ for some k , design an efficient algorithm to compute $H_k v$. Assume as usual that all the basic arithmetic operations on matrix/vector elements take $O(1)$ time.

3. Input is a set S of points (x_i, y_i) on the plane. Output should be the pair of points from set S that are the closest. That is, output is i, j such that $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, is the minimum. Assume n is a power of 2 and all coordinates are distinct.
4. **Extra Credit.** Show that if an $n \times n$ matrix can be squared in $T(n) = \Omega(n^2)$ time, then any two $n \times n$ matrices can be multiplied in $O(T(n))$ time. **Hint.** Consider $n \times n$ matrices A and B . Compute $AB + BA$ using only squaring in $O(T(n))$ time. Then, choose A and B to be suitable $2n \times 2n$ matrices constructed out of $n \times n$ matrices X and Y so that $AB + BA$ will yield XY . Conclude that multiplying X and Y can be done in $O(T(n))$ time.