

CECS 528, Homework Assignment 1, Fall 2025, Dr. Ebert

Directions: Please review the Homework section of the syllabus including a list of all rules and guidelines for writing and submitting solutions.

Due Date: Wednesday, September 17th as a PDF-file upload to the HW1 Canvas dropbox.

Problems

1. Apply both the FFT and IFFT algorithms for determining the coefficients of the product $C(x)$ of the linear polynomials $A(x) = 2 + x$ and $B(x) = 5 - 3x$. Do so as follows.
 - (a) Compute $\text{DFT}_4(A)$ using FFT. Show the entire recursion tree as was done in the lecture notes. (5 points)
 - (b) Compute $\text{DFT}_4(B)$ using FFT. Show the entire recursion tree as was done in the lecture notes. (5 points)
 - (c) Compute the values $C(\omega_4^i)$, for each $i = 0, 1, 2, 3$. (5 points)
 - (d) Use the solution to the previous problem to compute $\text{DFT}_4^{-1}(p)$ for an appropriate polynomial p . How is p defined? Show the entire recursion tree as was done in the lecture notes. (5 points)
 - (e) Verify that $\text{DFT}_4^{-1}(p)$ gives the correct coefficients of $A(x) \cdot B(x)$. Show all work. (5 points)
 - (f) What roots of unity would we use if A and B were both cubic polynomials? Explain. (5 pts)
2. This problem is inspired by my supervisor at Arcadia Design Systems who told me “linear and log-linear [algorithms]: good. Anything else, be very careful how you use it”. Suppose that a computer’s cpu is capable of executing a single instruction in 5×10^{-10} seconds. The plan is to run a time-intensive program on the computer for one full week. For each of the following scenarios determine the largest problem size n that can be solved during this time. Show all steps for full credit.
 - a. The program solves instances of **3SAT** and implements the best known **3SAT** algorithm which requires about $50 \left(\frac{4}{3}\right)^n$ instructions to solve a **3SAT** instance that depends on n variables. (5 pts)
 - b. The program solves instances of **Matrix Multiplication** and requires about $75n^3$ instructions to multiply two $n \times n$ matrices. **Side note:** although n is a natural size parameter for **Matrix Multiplication**, an instance of the problem is two $n \times n$ matrices, and hence n^2 better representst the actual instance size. Thus, **Matrix Multiplication** officially requires $O(m^{\frac{3}{2}})$ steps, where $m = n^2$. In any case, the traditional algorithm for multiplying matrices still grows cubically with respect to the dimension of the input matrices. (5 pts)

- c. The program solves instances of **Matrix Addition** and requires about $50n^2$ instructions to multiply two $n \times n$ matrices. (5 pts)