Homework 3

Due Sep 18 by 8am **Points** 20 **Submitting** a file upload **File Types** pdf
Available Sep 11 at 8am - Sep 18 at 8am

This assignment was locked Sep 18 at 8am.

Suggested reading

Chapter 2 [DPV].

Practice Problems (do not turn in)

[DPV] Problem 2.7 (Sum and product of roots of unity**)**

[DPV] Problems 2.8, 2.9(a) (FFT practice)

(FFT design)

Let $A(x) = 1 - 2x - 2x^2 + x^3$. You wish to run FFT to evaluate this polynomial.

(a) What is $A_{odd}(y)$ and $A_{even}(y)$?

(b) What is the appropriate root of unity to use?

(FFT as a black box)

Design an algorithm that takes as input a set $S = \{s_1, s_2, \ldots, s_n\}$ of distinct natural numbers such that $0 \leq s_i \leq 100n$, and a natural number N, and outputs True if the equation $s_i + s_j + s_k = N$ has at least one solution, and return False otherwise. There is a simple solution that runs in $O(n^3)$ time (can you find an $O(n^2)$ solution?), but you can improve on these times using FFT!

Example: For N = 6 and S = { 1; 2; 3; 5; 10 } your design should output True since 1+2+3 = 6. For N = 20 and the same set S the answer should be True again since 5+5+10 = 20 (yes, you can have $s_i = s_j = s_k$) but for N = 19 the answer is False since no three numbers add up to 19.

Graded Problem

Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of distinct real numbers. The k-th quantiles of S is a subset of exactly k-1 numbers $s'_1 < s'_2 < \ldots < s'_{k-1}$ such that the cardinality of the sets

$$S_j = \{s \in S \mid s'_{j-1} < s \leq s'_j\}$$

are the same (i.e. these numbers split the set into k subsets of equal size). Design a divide and conquer algorithm to find the k-th quantiles of a given set S of n numbers. You may assume that k is a power of 2

and that you can split the set S into k subsets of the same size. Your input is the set S, and the value of k. Note that S is not sorted.

Example: S={-1, 2, 4, 1, 3, 0, 18, -3} and k=2, your algorithm should output 1 (i.e.: the 2-th quantile is the median).

Example: S={-1, 2, 4, 1, 3, 0, 18, -3} and k=4 your algorithm should output {-1,1,3}.

Design a Divide & Conquer algorithm to solve this problem. Describe your algorithm in words (no pseudocode!) and justify its correctness. State and justify its runtime. Faster (and correct) in asymptotic Big O notation is worth more credit.

Brute force solutions will receive little to no credit.

You will upload a pdf of your **typed** solution. **Handwritten solutions will be penalized.**