CS 170 HW 2

Due on 2017-02-05, at 11:59 pm

1 (★) Study Group

List the names and SIDs of the members in your study group.

2 (★) Fourier Transform Basics

Answer the following parts:

(a) What is the fourier transform of $(3, i, 2, 4)$?

(b) Find x and y such that $\mathrm{FT}(x) = (5, 2, 1, -i)$ and $\mathrm{FT}(y) = (4, 4, i, i)$?

(c) *Using the above*, find z such that $\mathrm{FT}(z) = (1, -2, 1-i, -2i)$?

(d) Compute $(2x^2 + 1)(x + 4)$ using fourier transform.

3 (★★) Modular Fourier Transform

You might be wondering how does computer take a fourier transform (FT) given that it has to deal with irrational numbers etc. which cannot even be represented on a computer! This problem illustrates how to do the fourier transform in modular arthmetic, for example, modulo 5.

(a) There is a number $\omega \in \{0, 1, 2, 3, 4\}$ such that all the powers $\omega, \omega^2, \ldots, \omega^4$ are distinct (modulo 5). When doing the FT in modulo 5, this ω will serve a similar role to the primitive root of unity in our standard FT. Find this ω, and show that $\omega + \omega^2 + \cdots + \omega^4 = 0$. (Interestingly, for any prime modulus there is such a number.)

(b) Using the matrix form of the FT, produce the transform of the sequence $(0, 1, 0, 2)$ modulo 5; that is, multiply this vector by the matrix $M_4(\omega)$, for the value ω you found earlier. Be sure to explicitly write out the FT matrix you will be using (with specific values, not just powers of ω). In the matrix multiplication, all calculations should be performed modulo 5.

(c) Write down the matrix necessary to perform the inverse FT. Show that multiplying by this matrix returns the original sequence. (Again all arithmetic should be performed modulo 5.)

(d) Now show how to multiply the polynomials $2x^2 + 3$ and $-x + 3$ using the FT modulo 5.

4 (★★★) Polynomial from roots

Given a polynomial with exactly n distinct roots at $r_1, \ldots, r_n$, compute the coefficient representation of this polynomial using (much) fewer than $O(n^2)$ computations. There may be multiple possible answers, but your algorithm should return the polynomial where the coefficient of the highest degree term is 1.

Note: A root of a polynomial p is a number r such that $p(r) = 0$. The polynomial with roots $r_1, ..., r_k$ can be expressed as $\prod_i (x - r_i)$.

5 (★★★★) Triple sum

Design an efficient algorithm for the following problem. We are given an array $A[0..n-1]$ with n elements, where each element of A is an integer in the range $0 \leq A[i] \leq n$. The algorithm must answer the following yes-or-no question: does there exist indices i, j, k such that $A[i] + A[j] + A[k] = n$?

Design an $O(n \lg n)$ time algorithm for this problem. Note that you do not need to actually return the indices; just yes or no is enough.

Hint: define a polynomial of degree $O(n)$ based upon A, then use FFT for fast polynomial multiplication.

Reminder: don't forget to include explanation, pseudocode, running time analysis, and proof of correctness.

6 (★★★) Finding Clusters

We are given a directed graph $G = (V, E)$, where $V = \{1, \ldots, n\}$, i.e. the vertices are integers in the range 1 to n. For every vertex i we would like to compute the value $m(i)$ defined as follow: $m(i)$ is the smallest j such that vertex j is reachable from vertex i. (As a convention, we assume that i is reachable from i.) Show that the values $m(1), \ldots, m(n)$ can be computed in $O(|V| + |E|)$ time.

7 (★★★) Updating Labels

You are given a tree $T = (V, E)$ with a designated root node r, and a non-negative integer label $l(v)$. If $l(v) = k$, we wish to relabel v, such that $l_{\text{new}}(v)$ is equal to $l(w)$, where w is the kth ancestor of v in the tree. We follow the convention that the root node, r, is its own parent. Give a linear time algorithm to compute the new label, $l_{\text{new}}(v)$ for each v in V

Slightly more formally, the *parent* of any $v \neq r$, is defined to be the node adjacent to v in the path from r to v. By convention, $p(r) = r$. For $k > 1$, define $p^k(v) = p^{k-1}(p(v))$ and $p^1(v) = p(v)$ (so p^k is the kth ancestor of v). Each vertex v of the tree has an associated non-negative integer label $l(v)$. We want to find a linear-time algorithm to update the labels of all vertices in T according to the following rule: $l_{\text{new}}(v) = l(p^{l(v)}(v))$.