ESO207: Data Structures and Algorithms

Programming Assignment *1* | *Due: Sep 2 midnight*

Instructions: The precise input-output format will be specified by Programming TAs.

Problem 1. FFT

1. Write the following program. *Input:* Given a polynomial $A(x)$ that is specified by providing its degree $n-1$, and its n coefficients $a = [a_0, a_1, \ldots, a_{n-1}]$. *Output* $DFT(a, n)$ using the FFT routine. You may use the recursive FFT routine or an iterative routine. See note 1 below.

2. Write the following program. Given an n-dimensional vector of complex numbers $y = [y_0, y_1, \ldots, y_{n-1}]$ output $DFT_n^{-1}(y)$ using FFT or a close variant of FFT that has time complexity $O(n \log n)$. See note 2 below.

3. Given two n-dimensional vectors $a = [a_0, \ldots, a_{n-1}]$ and $b = [b_0, b_1, \ldots, b_{n-1}]$ over complex numbers, use FFT and its inverse to output the convolution $c = a \otimes b$, where, $c_k = \sum_{j=0}^{k} a_j b_{k-j}$, for $k = 0, 1, \ldots, 2n-2$. See note 3 below.

Notes.

1. Note that the input coefficients for $A(x)$ can be complex numbers. If n is not a power of 2, then, let N be the closest power of 2 that is larger than or equal to n, and extend a to make it an N dimensional vector by padding with additional coefficients $a_n, \ldots, a_{N-1}$ that are zeros. Recall that $DFT(a, n)$ is defined as

$$DFT(a,n) = \begin{bmatrix} A(w_n^0) \\ A(w_n^1) \\ \vdots \\ A(w_n^{n-1}) \end{bmatrix} .$$

2. Let F_n denote the $n \times n$ DFT matrix. That is,

$$F_n = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \ldots & \omega_n^{2(n-1)} \\ & \vdots & \vdots & \vdots & \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \ldots & \omega_n^{(n-1)^2} \end{bmatrix}$$

Recall by taking inner-product of any two columns that $F_n^* F_n = nI$. Hence, $F_n^{-1} = \frac{1}{n} F_n^*$ and therefore,

$$(DFT)_n^{-1}(y) = (1/n) F_n^*(y) \ .$$

We obtain two ways of computing DFT_n^{-1}. From definition of F_n^*, we have that F_n^* is the same as that of F_n^* with ω_n replaced by $\overline{\omega_n} = \omega_n^{-1} = e^{-2\pi i/n}$. So, in the computation of $F_n y$, if we replace the role of ω_n by ω_n^{-1} appropriately throughout, and divide by n, we should obtain $DFT^{-1}(y)$. The second method comes by observing the rows of F_n^* and relating them to rows of F_n. Note that $\overline{\omega_n}^k = \omega_n^{-k} = \omega_n^{n-k}$, for $0 \leq k \leq n-1$.

$$F_n^* = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \overline{\omega_n} & \overline{\omega_n}^2 & \dots & \overline{\omega_n}^{n-1} \\ 1 & \overline{\omega_n}^2 & \overline{\omega_n}^4 & \dots & \overline{\omega_n}^{2(n-1)} \\ & \vdots & \vdots & \vdots & \\ 1 & \overline{\omega_n}^{n-1} & \overline{\omega_n}^{2(n-1)} & \dots & \overline{\omega_n}^{(n-1)^2} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \dots & \omega_n^{(n-1)^2} \\ 1 & \omega_n^{n-2} & \omega_n^{2(n-2)} & \dots & \omega_n^{(n-2)(n-1)} \\ & \vdots & \vdots & \vdots & \\ 1 & \omega_n & \omega_n^2 & \dots & \omega_n^{n-1} \end{bmatrix}$$

Thus row indexed 0 of F_n^* is the same as row indexed 0 of F_n. Row 1 of F_n^* is same as row $n-1$ of F_n, row 2 of F_n^* is same as row $n-2$ of F_n, row $n-i$ of F_n^* is same as row i of F_n, for $i = 1, 2, \dots, n-1$. This same relation therefore holds between $F_n^* y$ and $F_n y$.

3. Part 3 of problem 1 can be solved by just combining the functions written for part 1 (FFT) and part 2 (inverse DFT using FFT). Since, a and b are both n dimensional, first pad a and b each with n zero new coefficients $a_n, \dots, a_{2n-1}$ and $b_n, \dots, b_{2n-1}$ that are all zeros to make them $2n$ dimensional vectors. Now compute $c = a \otimes b$ as follows. Let N be the closest power of 2 that is equal to or larger than $2n$.

$$c = DFT_N^{-1}(FFT_N(a) \bullet FFT_N(b))$$

where, for any k-dimensional vectors u and v, $(u \bullet v)_j = u_j \cdot v_j$, for $j = 0, \dots, k-1$.