Part 2

The Fast Fourier Transform

Let \( w_n \) be a primitive \( N \)th root of unity, i.e.

i) \( w_N = 1 \) (\( N \)th root of unity)

ii) \( w_N \neq 1 \) \( \forall j \neq N \)

Note all powers of \( w_N \) are \( N \)th roots of unity.

\[(w_N^k)^N = (w_N^k)^1 = 1 \]

In fact there are the only \( N \)th roots of unity, since they all satisfy all roots of the polynomial \( N \)th degree polynomial \( X^N - 1 \), and a polynomial of degree \( N \) can have at most \( N \) roots. Note not all \( N \)th roots of unity are primitive (eq. 1).

In the complex number

\[ e^{2\pi i / N} = \cos (2\pi / N) + i \sin (2\pi / N) \]

is a primitive \( N \)th root of unity.
Euler's Identity
\[ e^{i \theta} = \cos(\theta) + i \sin(\theta) \]
\[ e^{i \pi/N} = \left( e^{i \pi/N} \right)^N = e^{i \pi} = \cos \left( \frac{i \pi}{N} \right) + i \sin \left( \frac{i \pi}{N} \right) \]
\[ \Rightarrow \text{powering } \equiv \text{rotating } \frac{1}{N} \text{ around } \text{ unit circle.} \]

E.g., \( N = k \)

Rmk: \( \omega_N = \left( \omega_N^R \right)^S \implies \omega_N^R = \omega_N^S = 1 \)

If \( k < S \), \( \omega_S^k = \left( \omega_N^R \right)^k = \omega_N^S \neq 1 \)
Discrete Fourier Transform

Definition:

\[ y_{k} = \sum_{j=0}^{N-1} w_{N}^j x_{j}, \quad 0 \leq k < N \]

\[ y = \text{DFT}_{N} X \quad (\text{linear transform}) \]

FFT (rewrites sum as a nested sum)

\[ \begin{align*}
\hat{y}_{j} &= \hat{y}_{j}, \quad 0 \leq j < S \\
K &= K_{2}R + K_{1}, \quad 0 \leq K < S \\
\end{align*} \]

\[ \begin{align*}
\hat{y}_{K_{2}R + K_{1}} &= \sum_{i=0}^{S-1} \sum_{j=0}^{R-1} w_{N}^{(K_{2}R + K_{1})(jS + i)} x_{iS + j} \\
&= \sum_{i=0}^{S-1} \sum_{j=0}^{R-1} w_{N}^{K_{2}jRS + K_{1}i} w_{S}^{Ki} w_{N}^{K_{2}i} w_{R}^{jR} x_{iS + j} \\
&= \text{DFT}_{S}^{K_{2}R + K_{1}} \rightarrow \text{DFT}_{S}^{K_{2}R} \times \text{DFT}_{R}^{K_{1}}
\end{align*} \]
Observe that for fixed \( j \), the inner sum computes \( \text{DFT}_R \)
and for fixed \( k \), the outer sum computes \( \text{DFT}_S \).

Therefore the outer sum computes \( S \text{ DFT}_R \)'s followed by multiplication by \( \omega_{\text{SN}}^k \), called twiddle factors, and the \( R \text{ DFT}_S \)'s.

Here is the algorithm:

1. \( X_{\hat{s}} = X\hat{R}_{\hat{r}}, \hat{s} = X \hat{R}_{\hat{r}}; \hat{s} : \mathbb{N} \)
2. \( U_{\hat{s}} = \text{DFT}_R X_{\hat{s}} \), \( 0 \leq \hat{s} < \hat{s} \)
3. \( t_k(\hat{s}) = \omega_{\text{SN}}^k U_{\hat{s}}(k) \)
4. \( V_{\hat{k}} = \text{DFT}_S t_k, \quad 0 \leq \hat{k} < \hat{R} \)
5. \( U_{\hat{j}k} = V_{\hat{k}}(k) \),
6. \( U_{\hat{j}k} \hat{R} + \hat{k} = V_{\hat{k}}(k) \)

\( U_{\hat{j}k} \hat{R} + \hat{k} = V_{\hat{k}} \)
\[ R = \frac{N}{2}, \quad S = 2 \]

\[ u_0 = \text{DFT}_{N/2} x_{0,2} \]

\[ u_1 = \text{DFT}_{N/2} x_{1,2} \]

\[ t_{k_1}(0) = u_0(k_1) \quad 0 \leq k_1 < N/2 \]

\[ t_{k_1}(1) = \omega_{N/2}^k u_1(k_1) \]

\[ v_{k_1} = \text{DFT}_{N/2} t_{k_1} \quad 0 \leq k_1 < N/2 \]

\[ y_{k_1,N/2} = v_{k_1} \]

\[ y = \text{DFT}_{N/2} x \]

\[ y(0) = x(0) + x(1) \]

\[ y(1) = x(0) - x(1) \]

\[ u_0 = \text{DFT}_{N/2} x_{0,2} \]

\[ u_1 = \text{DFT}_{N/2} x_{1,2} \]

\[ y(k_1) = u_0(k_1) + \omega_{N/2}^k u_1(k_1) \]

\[ y(k_1 + N/2) = u_0(k_1) - \omega_{N/2}^k u_1(k_1) \]
\[ N = RS \]

\[ T(N) = RT(S) + ST(R) + \Theta(N) \]

Using a \( \Theta(N^2) \) split for recursive calls,

\[ \Theta(rs^2 + sr^2) = \Theta(N^2) \]

Comparing to \( \Theta(N^2) \).

Can when \( N = 2^k \)

and \( S = 2^r \), \( R = N/2 \)

and the recursive calls use the same split recursively.

\[ T(N) = 2T\left(\frac{N}{2}\right) + \Theta(N) \]

\[ T(1) = 1 \]

\[ \Rightarrow T(N) = \Theta(N \log N) \]

Exercise: Solve recurrence by repeated substitution.

\[ T(N) = 2T\left(\frac{N}{2}\right) + CN \]

\[ = 2\left[2T\left(\frac{N}{2^2}\right) + CN/2\right] + CN \]

\[ = 2^2T\left(\frac{N}{2^2}\right) + 2CN \]

\[ = 2^kT\left(\frac{N}{2^k}\right) + kCN = \Theta(N \log N). \]