

# Divide and Conquer (Contd.)

→ Linear transforms:

$$\bar{b} = A \bar{a} \quad \left. \begin{matrix} m \times 1 & m \times n & n \times 1 \end{matrix} \right\}$$

↳ For each entry of  $\bar{b}$ , we have inner product of  $n$  length vectors.

$$\left[ \begin{matrix} & & \\ & & \\ & & \end{matrix} \right]_{m \times n} \left[ \begin{matrix} \\ \\ \end{matrix} \right]_{n \times 1}$$

Naive:  $O(mn)$

$L \leftarrow$

$$L_{ij} = \omega^{ij}$$

$\omega^k \neq 1 \quad \forall k \in [1, n-1]$   
 $\omega = e^{\frac{2\pi i}{n}}$   
 $\omega$  is a primitive  $n$ th root of unity

$$\{\alpha_1, \dots, \alpha_n\}$$

1-indexing for  $j$

$$\begin{bmatrix} \alpha_1^0 & \alpha_1^1 & \dots & \alpha_1^{n-1} \\ \vdots & \vdots & & \vdots \\ \alpha_n^0 & \alpha_n^1 & \dots & \alpha_n^{n-1} \end{bmatrix}$$

$$V(\alpha_1, \dots, \alpha_n) =$$

Vandermonde matrices.

$$\omega^4 = 1$$

$$\omega^1 \neq 1$$

$$\omega^2 \neq 1$$

$$\omega^3 \neq 1$$

Primitive fourth root  
 $\omega = -i, i$

If  $\omega$  is primitive fourth root, then  $\omega^2$  is prim 2nd root.

$$P(z) = \sum_{i=0}^{n-1} c_i z^i$$

$$P(\alpha_i) = \langle i\text{th row of } V(\bar{\alpha}), \begin{bmatrix} c_0 \\ \vdots \\ c_{n-1} \end{bmatrix} \rangle$$

$$\begin{bmatrix} P(\alpha_1) \\ \vdots \\ P(\alpha_n) \end{bmatrix} =$$

$$V \cdot$$

$$\begin{bmatrix} c_0 \\ \vdots \\ c_{n-1} \end{bmatrix}$$

$$= \sum_{j=0}^{n-1} c_j \cdot \alpha_i^j$$

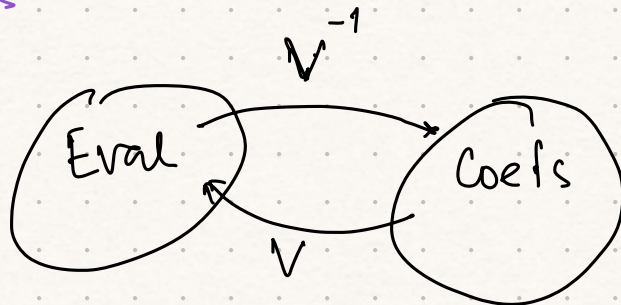
zero-indexing for  $j$ .

Given coeffs of a polynomial, we are computing their evaluations.

Coefs

↓ Evaluations

$$\begin{bmatrix} C_0 \\ \vdots \\ C_{n-1} \end{bmatrix} = V^{-1} \begin{bmatrix} P(\alpha_1) \\ \vdots \\ P(\alpha_n) \end{bmatrix}$$



We are interested in complexity of linear transformations when  $\alpha_1, \dots, \alpha_n$  are  $\omega^1, \omega^2, \dots, \omega^n$ .

$$\omega^{ij}$$

$$(L^{-1})_{ij} = \frac{\omega^{-ij}}{n} = \frac{\omega^{n-ij}}{n}$$

If transformation by  $L$  and  $L^{-1}$  were efficient (much better than  $O(n^2)$ ) then we can do polynomial mult. efficiently.

$$P(z) = \sum_{i=0}^n C_i z^{i-1}$$

↓  $T(n)$

$$P(\omega), \dots, P(\omega^n)$$

$$Q(z) = \sum_{i=0}^n C'_i z^{i-1}$$

↓  $T(n)$

$$Q(\omega), \dots, Q(\omega^n)$$

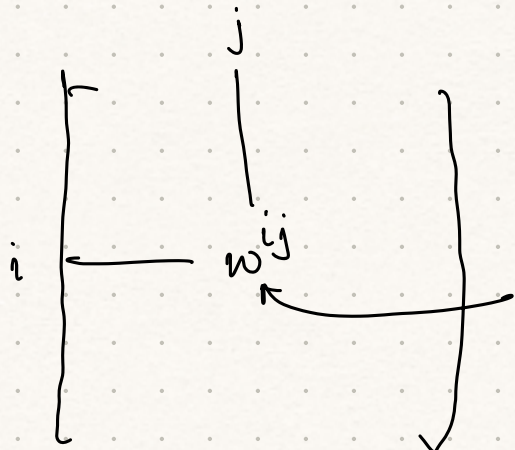
$$R(z) = P(z) \cdot Q(z)$$

↓  $T(n)$

$$R(\omega), \dots, R(\omega^n)$$

Caveat:  $R$  need not be a deg  $n$  poly.

DFT(n)



primitive  $n^{\text{th}}$  root of unity.



$$\begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} | \\ \hline w_{ij} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}$$

$$b_i = \sum_{j=1}^n w^{ij} \cdot a_j = \sum_{j=1}^{n/2} w^{ij} a_j + \underbrace{\sum_{j=\frac{n}{2}+1}^n w^{ij} a_j}$$

$$\sum_{j=\frac{n}{2}+1}^n w^{ij} a_j = \sum_{j'=1}^{n_2} w^{i(j'+\frac{n}{2})} \cdot a_{j'+\frac{n}{2}}$$

$$j' = j - \frac{n}{2} \Rightarrow j = j' + \frac{n}{2} \quad \Rightarrow \quad \omega^{\frac{n}{2}i} \sum_{j'=1}^{n/2} \omega^{i \cdot j'} \cdot a_{j' + \frac{n}{2}}$$

$$b_i = \sum_{j=1}^{n/2} w_{ij} \cdot a_j + w^{\frac{n}{2}i} \cdot \sum_{j'=1}^{n/2} w_{ij'} \cdot a_{j+\frac{n}{2}}$$

If  $\omega^n = 1$ , then what is  $\omega^{\frac{n}{2}}$ ?  $\omega^{n/2} = -1$ .

$$b_i = \sum_{j=1}^{n/2} \omega^{ij} \cdot a_j + (-1)^i \cdot \sum_{j=1}^{n/2} \omega^{ij'} \cdot a_{j+\frac{n}{2}}$$

Is this a DFT $_{\frac{n}{2}}$ ? Ans: No because  $w$  is  $n^{\text{th}}$  root and not  $\frac{n}{2}^{\text{th}}$  root.

Obs:  $\omega^2$  is  $\frac{n^{\text{th}}}{2}$  primitive root of unity.

If  $i$  is even; Say  $i = 2p$ .

$$b_i = \sum_{j=1}^{n/2} (\omega^2)^{p \cdot j} \cdot a_j + \sum_{j'=1}^{n/2} (\omega^2)^{p \cdot j'} \cdot a_{j'+\frac{n}{2}}$$

$j'$  is a notation and can be replaced with  $j$ .

$$= \sum_{j=1}^{n/2} (\omega^2)^{p \cdot j} (a_j + a_{j+\frac{n}{2}})$$

Else:  $i = 2p+1$ .

$$b_i = \left( \sum_{j=1}^{n/2} \omega^{2 \cdot p \cdot j} \cdot a_j \cdot \omega^j \right) - \left( \sum_{j'=1}^{n/2} \omega^{2 \cdot p \cdot j'} \cdot a_{j'+\frac{n}{2}} \cdot \omega^{j'} \right)$$

$$= \sum_{j=1}^{n/2} (\omega^{2 \cdot p \cdot j}) \cdot \omega^j \cdot (a_j - a_{j+\frac{n}{2}})$$

$i \leftarrow \text{odd}$

$$v_e = \begin{bmatrix} a_1 + a_{\frac{n}{2}+1} \\ a_2 + a_{\frac{n}{2}+2} \\ \vdots \\ a_{\frac{n}{2}} + a_n \end{bmatrix}$$

$$v_o = \begin{bmatrix} \omega(a_1 - a_{\frac{n}{2}+1}) \\ \omega^2(a_2 - a_{\frac{n}{2}+2}) \\ \vdots \\ \omega^{n/2}(a_{\frac{n}{2}} - a_n) \end{bmatrix}$$

DFT<sub>n</sub>( $\bar{a}$ ) from DFT <sub>$\frac{n}{2}$</sub> ( $v_e$ ) and DFT <sub>$\frac{n}{2}$</sub> ( $v_o$ ).

$$\begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} \left\{ \begin{array}{l} \text{even terms} \end{array} \right. \begin{bmatrix} b_2 \\ b_4 \\ \vdots \\ b_n \end{bmatrix} = \text{DFT}_{\frac{n}{2}}(v_e)$$

$$\begin{bmatrix} b_1 \\ b_3 \\ \vdots \\ b_n \end{bmatrix} = \text{DFT}_{\frac{n}{2}}(v_o)$$



# Fast (Discrete) Fourier Transform

FFT<sub>n</sub>( $\bar{a}$ ):

W.L.O.G.  $n$  is a power of 2.

- Find primitive  $n^{\text{th}}$  root of unity
- Construct vectors  $v_e$  and  $v_o$ .
- Recursively call FFT <sub>$\frac{n}{2}$</sub> ( $v_e$ ) and FFT <sub>$\frac{n}{2}$</sub> ( $v_o$ )
- "Put together" the results
- Return  $\bar{b}$  thus obtained.

If  $n$  is not a power of 2, append 0's at the bottom to make the vector  $2^k$

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n) \rightarrow O(n \log n).$$

$$P(z) = \sum_{i=0}^n c_i z^i$$

$$Q(z) = \sum_{i=0}^n c'_i z^i$$

$$R(z) = \sum_{i=0}^{2n} e_i z^i$$

$$\downarrow$$

$$P(z) = \sum_{i=0}^{2n} c_i z^i$$

$$\text{s.t. } c_i = 0 \forall i > n$$

$$\downarrow$$

$$Q(z) = \sum_{i=0}^{2n} c'_i z^i$$

$$\text{s.t. } c'_i = 0 \forall i > n$$

$\omega \leftarrow 2n^{\text{th}}$  primitive root

$$\left. \begin{array}{l} P(\omega), \dots, P(\omega^{2n}) \\ Q(\omega), \dots, Q(\omega^{2n}) \end{array} \right\} R(\omega), \dots, R(\omega^{2n})$$

Inv DFT  $\rightarrow$  coeffs of  $R$ .

[De, Kurur, Saha, Saptharishi  
STOC 2008]

[Fürer, ~]

Best integer mult. algos

$$2 \cdot T(2n) + T_{\text{inv}}(2n)$$