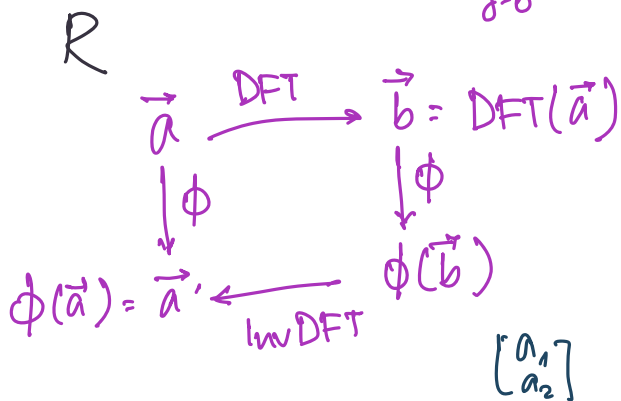
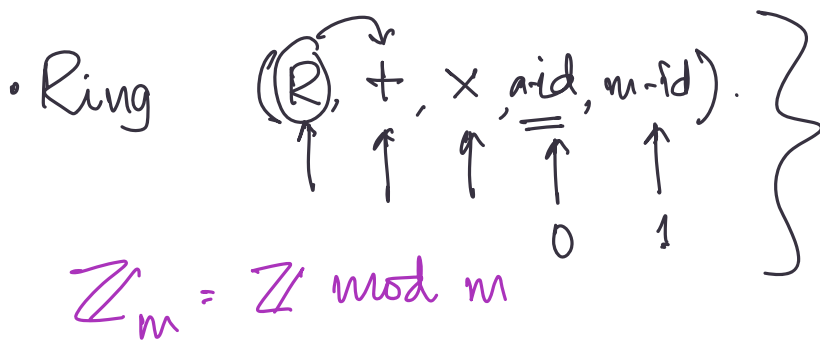


Discrete Fourier Transform.

$$b_i = \sum_{j=0}^{n-1} \omega^{ij} \cdot a_j.$$



Input: $\vec{a} = (a_0, a_1, \dots, a_{n-1})$

Discrete Fourier Transform $(\text{DFT}(\vec{a})) = A \cdot (\vec{a})^T$.

$\forall i, j \in [0, n-1], A_{i,j} = \omega^{ij}$.

Assume that $\frac{1}{n}$ is available in our ring R . Assume that n is a power of 2.

$(A^{-1})_{i,j} = \frac{\omega^{-i \cdot j}}{n}$.

$\text{DFT}: \mathbb{R}^n \rightarrow \mathbb{C}^n$
invertible linear map.

Primitive n^{th} root of unity

$\omega^k \neq 1 \quad \forall k \in [1, n-1]$.

Want: To compute the linear transform $(\text{DFT}(\vec{a}))$.

Remark: Naively, \vec{b} needs $O(n^2)$ operations.

$\text{DFT}(\text{InvDFT}(\vec{a})) = \vec{a}$
 $\text{InvDFT}(\text{DFT}(\vec{a})) = \vec{a}$

Rather any linear transformation takes at most $O(n^2)$ operations

Cooley-Tukey: If we consider a linear transformation by the DFT matrix, it can be done in $O(n \log n)$ operations.

$(b_i) = \sum_{j=0}^{n-1} \omega^{ij} \cdot a_j$

$$P_a(x) = \sum_{j=0}^{n-1} a_j x^j.$$

$$A \cdot (\vec{a})^T$$

$$b_0 = \sum_{i=0}^{n-1} (w^0)^i \cdot a_i = P_a(w^0)$$

$$\begin{bmatrix} b_0 \\ \vdots \\ b_{n-1} \end{bmatrix} = \begin{bmatrix} w^0 & w^1 & \dots & w^{n-1} \\ w^0 & w^2 & \dots & w^{2(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ w^0 & w^2 & \dots & w^{(n-1)^2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{bmatrix}$$

$$\begin{bmatrix} P_a(w^0) \\ \vdots \\ P_a(w^{n-1}) \end{bmatrix} = \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_{n-1} \end{bmatrix}$$

Evaluation of $P_a(x)$ at points w^0, w^1, \dots, w^{n-1} .

$$n' = 2^k$$

$$P_{n'}(x) = \sum_{j=0}^{n'-1} a_j x^j \quad \text{s.t. } a_j = 0 \text{ for } j > n-1.$$

$$(a_0, \dots, a_{n-1})$$

$$(a_0, \dots, a_{n-1}, 0, \dots)$$

\hookrightarrow A which is of size $n' \times n'$.

Given two polynomials $P(x)$ and $Q(x)$: Compute their product
 degree at most $n-1$.

$$\begin{array}{l} P(x) = A + x^{n/2} B \\ Q(x) = A' + x^{n/2} B' \end{array} \quad \left| \quad \begin{array}{l} PQ(x) = AA' + BB'x^n + (AB' + BA')x^{n/2} \\ \hookrightarrow n \log_2^3 \sim n^{1.58} \end{array} \right.$$

$P(x) \xrightarrow{\text{DFT}}$ evaluations of P at $\{w^i\}_{i \in [0, 2n-2]}$
 $Q(x) \xrightarrow{\text{DFT}}$ evals of Q at $\{w^i\}_{i \in [0, 2n-2]}$

Evaluations of $P(x) \cdot Q(x)$ at $\{w^i\}$
 \downarrow InvDFT
 Obtain coef. of the product.

$$w^i \rightarrow P(w^i) \cdot Q(w^i)$$

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

$$\downarrow \quad \downarrow \quad \downarrow$$

$$\sum_{i=0}^{n-1} p_i \cdot x^i \cdot \sum_{j=0}^{n-1} a_j x^j$$

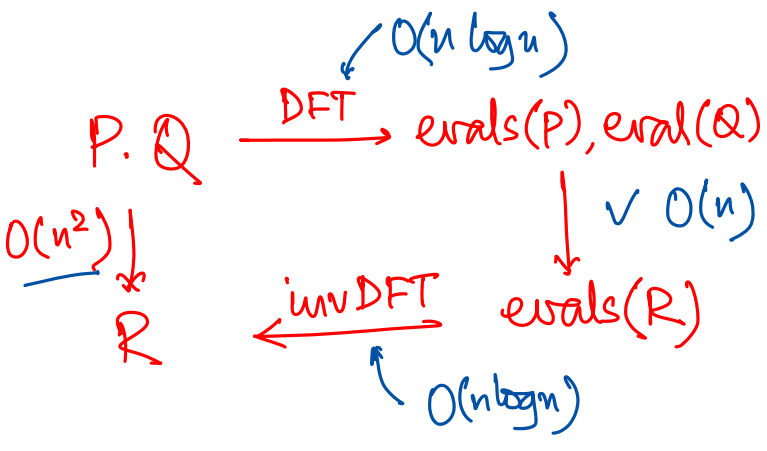
Suff: Consider $2n$ -th root of unity.

$$\left. \begin{aligned} P &= (p_0, \dots, p_{n-1}, 0, \dots, 0) \\ Q &= (a_0, \dots, a_{n-1}, 0, \dots, 0) \end{aligned} \right\} \xrightarrow{\text{DFT}} \left. \begin{aligned} P(w^0), \dots, P(w^{2n-1}) \\ Q(w^0), \dots, Q(w^{2n-1}) \end{aligned} \right\}$$

$$R(w^0), \dots, R(w^{2n-1})$$

↓ Recover Polynomial R.

$O(n \log n)$ algorithm



$$b_i = \sum_{j=0}^{n-1} w^{ij} \cdot a_j$$

$$= \sum_{j=0}^{n/2-1} w^{ij} a_j + \sum_{j=n/2}^{n-1} w^{ij} a_j$$

$$j = \frac{n}{2} + j'$$

$$n-1 = \frac{n}{2} + j'$$

$$= \sum_{j=0}^{n/2-1} w^{ij} a_j + w^{i \cdot n/2} \cdot \sum_{j'=0}^{n/2-1} w^{ij'} a_{j'+n/2}$$

$w^k \neq 1 \quad \forall k \in [0, n-1], w^n = 1.$

$$\sqrt[n]{w^n}$$

Obs: w^2 is $\frac{n}{2}$ th prim root of unity if w is n th prim root of unity.

Obs: $w^{n/2} = (-1)$
 $w^n = 1.$

$$= \sum_{j=0}^{\lfloor \frac{n}{2} \rfloor} \omega^{ij} \cdot a_j + (-1)^i \cdot \sum_{j'=0}^{\lfloor \frac{n}{2} \rfloor} \omega^{ij'} \cdot a_{j'+\frac{n}{2}}$$

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

$$b_i = \sum_{j=0}^{\lfloor \frac{n}{2} \rfloor} (\omega^{2p})^j a_j + \sum_{j'=0}^{\lfloor \frac{n}{2} \rfloor} (\omega^{2p})^{j'} \cdot a_{j'+\frac{n}{2}}$$

$$= \sum_{j=0}^{\lfloor \frac{n}{2} \rfloor} (\omega^{2p})^j (a_j + a_{j+\frac{n}{2}})$$

$$= \sum_{j=0}^{\lfloor \frac{n}{2} \rfloor} (\omega^{2p})^j \cdot C_j$$

$$C_j = a_j + a_{j+\frac{n}{2}}$$

$$\vec{C} = (C_0, \dots, C_{\frac{n}{2}-1})$$

If i is odd

$$b_i = \sum_{j=0}^{\frac{n}{2}-1} \omega^{(2p+1)j} \cdot a_j - \sum_{j'=0}^{\frac{n}{2}-1} \omega^{(2p+1)j'} \cdot a_{j'+\frac{n}{2}}$$

$$= \sum_{j=0}^{\frac{n}{2}-1} (\omega^{2p+1})^j (a_j - a_{j+\frac{n}{2}})$$

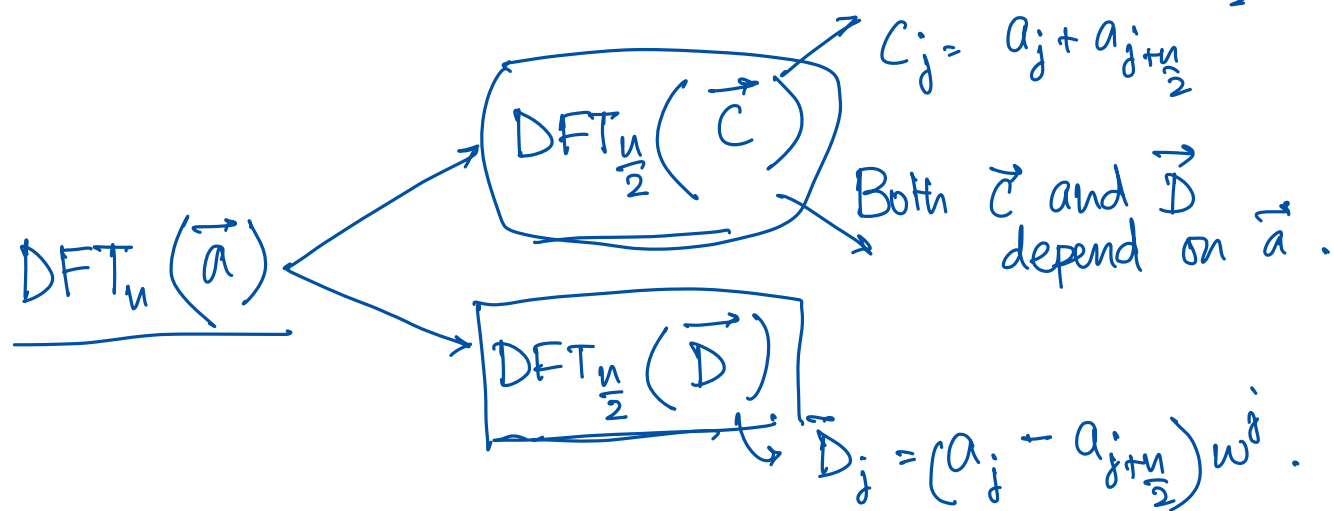
$$A_{ij}^{-1} = \frac{\omega^{-ij}}{n}$$

$$= \sum_{j=0}^{\frac{n}{2}-1} (\omega^{2p})^j \underbrace{(\omega^j)}_{\text{wavy line}} \cdot (a_j - a_{j+\frac{n}{2}})$$

$$= \sum_{j=0}^{\frac{n}{2}-1} (\omega^{2p})^j D_j \quad \text{where } D_j = \omega^j \cdot (a_j - a_{j+\frac{n}{2}})$$

$$\vec{D} = (D_0, \dots, D_{\frac{n}{2}-1})$$

Even locations of \vec{b} can be obtained by $DFT_{\frac{n}{2}}(\vec{C})$
 and Odd locations from $DFT_{\frac{n}{2}}(\vec{D})$.



$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + O(n)$$

$$= O(n \log n)$$

Rewriting Polynomial mult:

Convolution of two vectors \vec{a} and \vec{b} (denoted by $\vec{a} * \vec{b}$)

$$i \in [0, 2n-2]; \quad (a * b)_i = \sum_{\substack{j+k=i \\ 0 \leq j, k \leq n-1}} a_j \cdot b_k$$

Convolution gives coeffs of the product of the polys whose coeffs are \vec{a} and \vec{b} .

$$\left(\sum_{j=0}^{n-1} a_j x^j \right) \left(\sum_{k=0}^{n-1} b_k x^k \right) = \sum_i \left(\underbrace{\sum_{j+k=i} a_j b_k}_{\text{convolution}} \right) \cdot x^i$$

Convolution

$$a * b = \text{InvDFT}_{2n} \left(\text{DFT}_{2n}(\vec{a}') \odot \text{DFT}_{2n}(\vec{b}') \right)$$

$$\vec{a}' = (a_0, \dots, a_{n-1}, \underbrace{0, \dots, 0}_{2n})$$

$$\vec{b}' = (b_0, \dots, b_{n-1}, \underbrace{0, \dots, 0}_{2n})$$

point wise mult.

$$(a_1, \dots, a_n) \odot (b_1, \dots, b_n) \\ = (a_1 b_1, a_2 b_2, \dots, a_n b_n)$$

Using DFT: Integer mult is in $O(n \log n)$ }
if \mathbb{R} is "nice". }

↓ Else

$$O(n \log n \log \log n)$$

↑ Schonhage-Strassen
integer mult.

$$O(n \log n) \cdot 2^{O(\log^* n)}$$

→ [Furer 2008]

{ [De-Kurur-Saha.
- Saptarishi]

STOC 2008