Today:

- Fast Fourier Transform
- Polynomial multiplication

Poly Mult

Input: $A(x) = a_0 + a_1 x + \cdots a_{d-1} \cdot x^{d-1}$

$B(x) = b_0 + b_1 x + \cdots b_{d-1} x^{d-1}$

Goal: compute vector of coeffs of $C = A \cdot B$

$C(x) = c_0 + \cdots c_{2d-2} x^{2d-2}$

$\boxed{\text{def: } n := 2d-1}$

$c_0 = a_0 b_0$

$c_1 = a_0 b_1 + a_1 b_0$

$\cdots \; c_k = \sum_{j=0}^{k} a_j b_{k-j}$

Ex: Integer mult

$\begin{array}{r} 1074 \swarrow \alpha \\ \times 2351 \leftarrow \beta \end{array}$ | $A(x) = 4 + 7x + 0x^2 + 1 \cdot x^3$ $B(x) = 1 + 5x + 3x^2 + 2x^3$ $\begin{pmatrix} \alpha = A(10) \\ \beta = B(10) \end{pmatrix}$

$\alpha \cdot \beta = (A \times B)(10)$

Alg 1 (poly mult)

- Two nested for loops $= O(n^2)$

(also $\Omega(n^2)$ since computing each of $c_{n/2}, \ldots c_{n-1}$

requires $\geq \frac{N}{2}$ flops each $\geq \frac{N^2}{2^3}$ flops) $\Rightarrow \Theta(N^2)$

Alg 2. Karatsuba

$$A(x) = \underbrace{a_0 + a_1 x + \cdots}_{A_L(x)} \; \underbrace{+ a_{N-1} x^{N-1}}_{A_h(x) \cdot x^{\frac{N}{2}}}$$

$$B(x) = \underbrace{b_0 + b_1 x + \cdots}_{B_L(x)} \; \underbrace{- b_{N-1} x^{N-1}}_{B_h(x) \cdot x^{\frac{N}{2}}}$$

$$(A \cdot B)(x) = A_L \times B_L + x^{\frac{N}{2}} (A_L B_h + A_h B_L) + x^N (A_h B_h)$$

Interpolation: any degree $<N$ poly is determined by its evaluation on N distinct points

why? $x_0, x_1, \ldots x_{N-1}$ are distinct pts

Vandermonde matrix

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{N-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{N-1} & & \cdots & x_{N-1}^{N-1} \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ \vdots \\ y_{N-1} \end{bmatrix}$$

$$V \cdot c = y \overset{?}{\Rightarrow} c = V^{-1} y$$

$\det(V) = \prod_{i > j} (x_i - x_j)$

$$C(x) = (A \times B)(x) = A(x) \cdot B(x)$$

```
def eval(p, x):
    ans ← 0
    cpow ← 1
    for i = 0 to n-1:
        ans ← ans + p_i · cpow
        cpow ← cpow · x
    return ans.
```

linear time

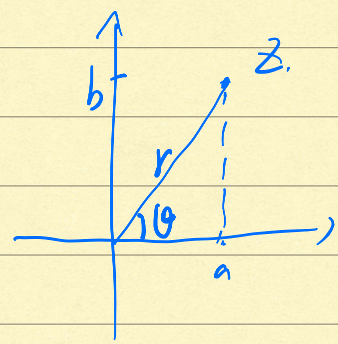
Types: • Fast Fourier Transform (FFT) is an algorithm
• Discrete Fourier Transform (DFT) is a matrix

DFT: $F_{ij} = (w^i)^j$, $w := e^{2\pi\sqrt{-1}/N}$

Complex numbers: $z \in \mathbb{C}$

• $z = a + \sqrt{-1} \cdot b$

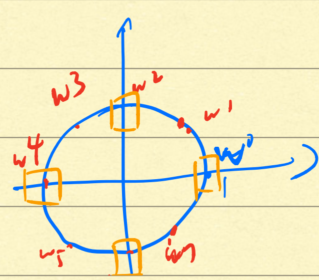
• $z = r \cdot e^{\sqrt{-1}\theta}$



$\left(r = \sqrt{a^2+b^2},\ \theta = \tan^{-1}\left(\frac{b}{a}\right) \right)$

$e^{\sqrt{-1}\theta} = \cos\theta + \sqrt{-1}\sin\theta$

$N = 8$, $w^0, w^1, \ldots w^7$



• p : degree $< N$, want $p(w^i)$ for all i

$p(z) = p_0 + p_1 z + p_2 z^2 + \cdots p_{N-1} z^{N-1}$

$= (p_0 + p_2 z^2 + p_4 (z^2)^2 + \cdots) \leftarrow P_{even}$

$+ (p_1 + p_3 (z^2) + p_5 (z^2)^2 + \cdots) \cdot z. \leftarrow P_{odd}$

$p(z) = P_{even}(z^2) + z \cdot P_{odd}(z^2)$

$\Rightarrow$ to obtain $p(w^0), p(w^1), \ldots, p(w^{N-1})$, actually N

we just need $P_{even}(w^0), P_{even}(w^2) \ldots P_{even}(w^{2N-2})$

$P_{odd}(\cdot) \ldots \ldots$

• $p(x) = P_{odd}(x^2) \cdot x + P_{even}(x^2)$

$$T(N) = \begin{cases} 2T\left(\frac{N}{2}\right) + \Theta(N) & N > 1 \\ 1 & N = 1 \end{cases}$$

$\Rightarrow T(N) = \Theta(N \log N)$

$$\left[\; F \; \right] \begin{bmatrix} a_0 \\ \vdots \\ a_{d-1} \end{bmatrix}$$

$F \cdot \vec{c} = \hat{c}$

Game plan: (given A, B)

① compute $F \cdot \vec{a}$ to get $\hat{a}$ (FFT)

② compute $F \cdot \vec{b}$ to get $\hat{b}$ (FFT)

③ compute $\hat{c}_j = \hat{a}_j \cdot \hat{b}_j$ for $j = 0, \ldots, N-1$

④ return $F^{-1} \cdot \hat{c} = \frac{1}{N} \vec{\bar{F}} \cdot \hat{c} = \frac{1}{N} \cdot \overline{F \cdot \bar{\hat{c}}}$

claim : $F^{-1} = \frac{1}{n}\bar{F}$

remind : $\bar{z} = a - b\sqrt{-1}$
$= r \cdot e^{\sqrt{-1}\cdot(-\theta)}$

claim : $\bar{A} \cdot v = \overline{A\bar{v}}$

---

cross-correlation (successive dot products)

$x = \boxed{x_0, - - - x_{m-1}}$

$y = \boxed{y_0\ y_1\ -\ -\ \cdots\ y_{n-1}}$

- want to know

$\sum_{i=0}^{m-1} x_i y_i$

$\sum_{i=0}^{m-1} x_i y_{i+1}$

$\sum_{i=0}^{m-1} x_i y_{i+2}$

list of $n-m+1$ numbers.

Naive :
$\Theta((n-m)m)$.

Fast algo via fast poly multi (FFT)

$\omega^{n-1}\ \boxed{F_n}\ y = \hat{y}$
$\omega^{n-1}\ \boxed{F_n}\ x = \hat{x}$

$Y(z) = y_0 + y_1 z + \cdots + y_{n-1} z^{n-1}$
$X(z) = x_{m-1} + x_{m-2} z + \cdots + x_0 \cdot z^{m-1}$
$\Rightarrow Q := Y \cdot X$.

$Q(z) = (Y \cdot X)(z) = q_0 + q_1 z + \cdots + q_{n+m-2} \cdot z^{n+m-2}$

$q_0 = y_0 x_{m-1}\ \cdots\ \boxed{q_{m-1}} = x_{m-1} y_{m-1} + x_{m-2} \cdot y_{m-2} + \cdots x_0 y_0$

$q = \hat{q}$

return $q_{m-1}, q_m, \ldots, q_{n-1}$, $\left(\underline{\qquad} \right)_{w^{n+m-q}}$