$T(n) = 7\,T(\frac{n}{4}) + O(n)$

$\Rightarrow O(n^{\log_4 7}) \leq O(n^{1.41})$

$(2r-1)\,T(\frac{n}{r}) + O(n)$; $\log_r(2r-1)$

$\Rightarrow O(n^{1+\varepsilon})$ for any const $\varepsilon > 0$.

Cooley & Tukey's Alg'm (1965)

Problem A (Multi-Point Evaluation)

Given polynomial P of deg $N-1$
& N distinct values $\alpha_0, \alpha_1, \ldots \alpha_{N-1}$
compute $P(\alpha_0), \ldots, P(\alpha_{N-1})$.

[trivial algm: $O(N)$ per α_k
$\Rightarrow O(N^2)$ time].

Problem B (Interpolation) (inverse problem)

Given $P(\alpha_0), \ldots, P(\alpha_{N-1})$,
reconstruct polynomial P.

[Lagrange's interpolation formula:
at least N^2 time].

To solve polynomial mult. problem: set $N = 2n-2$

1. compute $P(\alpha_0), \ldots P(\alpha_{N-1})$ by A
   $Q(\alpha_0) \ldots Q(\alpha_{N-1})$
2. compute $P(\alpha_0)\cdot Q(\alpha_0), \ldots, P(\alpha_{N-1})\cdot Q(\alpha_{N-1})$
   in $O(N)$ time
3. reconstruct $P \cdot Q$ by B.

$\Rightarrow$ $O(n \log n)$ time

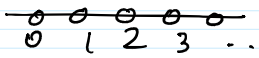
this works for any $\alpha_0, \ldots, \alpha_{N-1}$

find good choice of $\alpha_0, \dots, \alpha_{N-1}$
s.t. I can solve Probs A & B.

idea – choose

$$\alpha_k = e^{\frac{-2\pi i k}{N}}$$

for $k = 0, \dots, N-1$.

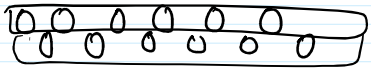


called roots of unity

(because they satisfy $z^N = 1$)

$$\left(e^{\frac{-2\pi i k}{N}}\right)^N = \left(e^{-2\pi i}\right)^k = 1$$

Alg'm for Problem A:



1. divide by odd-even:

$$P(x) = P_1(x^2)\,x + P_2(x^2)$$

2. recursively compute

$$u_k = P_1\left(e^{\frac{-2\pi i k}{N/2}}\right) \quad \text{for } k = 0, \dots, \frac{N}{2}-1$$

$$v_k = P_2\left(e^{\frac{-2\pi i k}{N/2}}\right) \quad \text{for } k = 0, \dots, \frac{N}{2}-1$$

3. for $k = 0, \dots, N-1$

$$z_k = P\left(e^{\frac{-2\pi i k}{N}}\right) = \underbrace{P_1\left(e^{\frac{-4\pi i k}{N}}\right)}_{\text{known from step 2}} e^{\frac{-2\pi i k}{N}} + \underbrace{P_2\left(e^{\frac{-4\pi i k}{N}}\right)}_{\text{known from step 2}}$$

(note: $e^{\frac{-4\pi i k}{N}} = e^{\frac{-2\pi i k}{N/2}}$ for $k < \frac{N}{2}$

$= e^{\frac{-2\pi i}{N/2}(k-\frac{N}{2})}$ for $k > \frac{N}{2}$

$\left(e^{\frac{-2\pi i}{N/2}\cdot\frac{N}{2}} = 1\right)$

i.e. $z_k = \begin{cases} u_k\, e^{\frac{-2\pi i k}{N}} + v_k & \text{for } k = 0, \dots, \frac{N}{2}-1 \\ u_{k-\frac{N}{2}}\, e^{\frac{-2\pi i k}{N}} + v_{k-N/2} & \text{for } k = \frac{N}{2}, \dots, N-1 \end{cases}$

$$\Rightarrow T(N) = 2T\left(\frac{N}{2}\right) + O(N)$$

$$\Rightarrow \boxed{O(N \log N)}$$

$\Rightarrow$ $\boxed{O(N \log N)}$

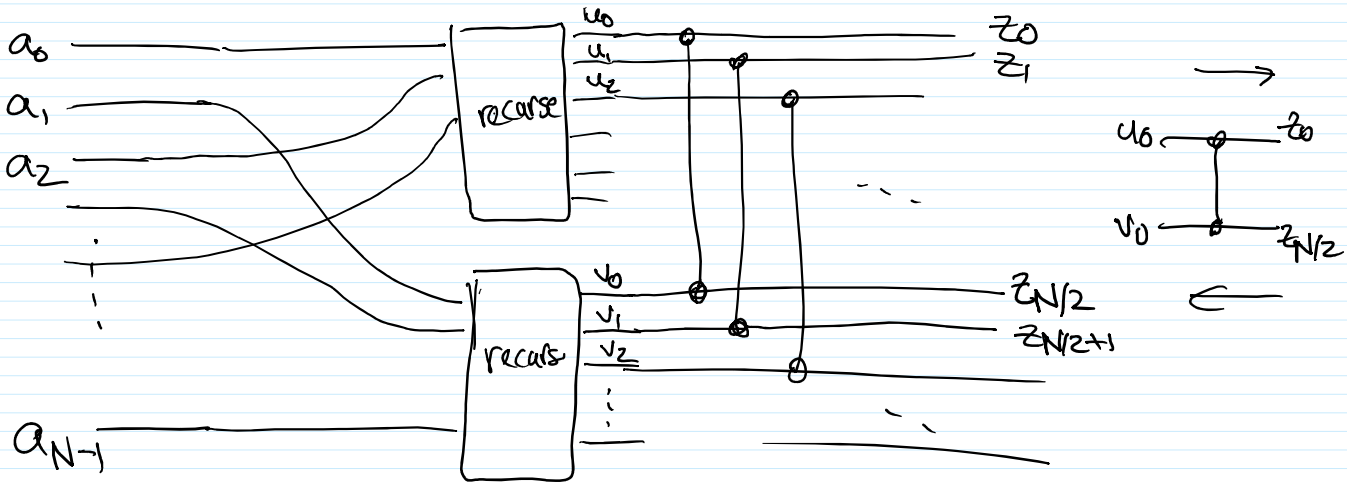
Rmk - above alg'm called Fast Fourier Transform (FFT)

Why? it computes $P(e^{\frac{-2\pi i k}{N}}) = \sum_{j=0}^{N-1} a_j e^{\frac{-2\pi i k j}{N}}$ $\forall k$

$P(x) = \sum_{j=0}^{N-1} a_j x^j$

(Similar to Fourier transform:

$\hat{f}(x) = \int_{-\infty}^{\infty} f(t)\, e^{-2\pi i x t}\, dt$)



"butterfly network"

Alg'm for Problem B:

Similar, $O(N \log N)$ time by going backwards

In fact, inverse-FFT equiv. to FFT!!
(ignoring scaling factor)

(inverse Fourier transform

$f(t) = \int_{-\infty}^{\infty} \hat{f}(x)\, e^{2\pi i t x}\, dx$)

Rmk - N power of 2
precision issues ...

$\widehat{f \circ g} = \hat{f} \cdot \hat{g}$)

precision issues ...

Appl'n 1. multiplying 2 large n-bit numbers $\left(\sum_{i=0}^{n-1} a_i 2^i\right) \cdot \left(\sum_{i=0}^{n-1} b_i 2^i\right)$

elementary-school method $O(n^2)$

⋮

FFT $O(n \log n)$ ops on $(\log n)$-bit numbers

Schönhage-Strassen '71 $O(n \log n \log\log n)$ bit ops

Fürer '07 $O(n \log n \log\log\log\log\cdots\log n)$ "

Harvey & van der Hoeven '19 $O(n \log n)$.

Appl'n 2. pattern matching with "don't cares"

given 2 strings $a_1 a_2 \cdots a_m \in (\Sigma \cup \{?\})^*$ "pattern"

$b_1 b_2 \cdots \cdots b_n \in \Sigma^*$ "text"

decide whether $\exists k$ s.t. $a_1 \cdots a_m = b_{k+1} \cdots b_{k+m}$

e.g. "i??u"

"algorithmsfun"