Lec10A The Fast Fourier Transform

Section 4.5

February 10, 2021 11:49 PM

[Cooley and Tukey 1965]

$a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}$

Let $a(x) = \sum_{i=0}^{n-1} a_i x^i$ with $a_i \in F$ a field and $\alpha_1 \neq \alpha_2 \neq \cdots \neq \alpha_n \in F$

How fast can we evaluate $a(x)$ at α_i and interpolate $a(x)$ given $a(\alpha_i)$?

$$\underset{\text{coefficients.}}{a = \underbrace{[a_0, a_1, \ldots, a_{n-1}]}_{\in F^n}} \underset{\text{interpolation.}}{\overset{\text{evaluation}}{\rightleftarrows}} \underset{\text{points.}}{\underbrace{[a(\alpha_1), a(\alpha_2), \ldots, a(\alpha_n)]}_{\in F^n}}$$

Newton interpolation costs $O(n^2)$ operations in F.
Evaluation using Horner costs $n \cdot O(n) = O(n^2)$ ops. in F.

Idea of the FFT? Suppose $2|n$.

$$a(x) = (a_0 + a_2 x^2 + \cdots + a_{n-2} x^{n-2}) + x(a_1 + a_3 x^2 + \cdots + a_{n-1} x^{n-2})$$
$$a(x) = b(x^2) + x \cdot c(x^2)$$

where $b(x) = a_0 + a_2 x^1 + \cdots + a_{n-2} x^{(n-2)/2}$
and $c(x) = a_1 + a_3 x^1 + \cdots + a_{n-1} x^{(n-2)/2}$

Evaluate $a(x)$ at $x = \pm 1, \pm 2, \ldots, \pm n/2$.

$a(2) = a(-2) \qquad b(2) = b(-2)$.

This saves almost $\frac{1}{2}$ the work.

Suppose $4|n$. Then

$$a(x) = b(x^2) + x\,c(x^2) = d(x^4) + x^2 e(x^4) + x(f(x^4) + x^2 g(x^4))$$

where

$d(x) = a_0 + a_4 x + a_8 x^2 + \cdots \qquad f(x) = a_1 + a_5 x + a_9 x^2 + \cdots$
$e(x) = a_2 + a_6 x^1 + a_{10} x^2 + \cdots \qquad g(x) = a_3 + a_7 x + a_{11} x^2 + \cdots$

Evaluate at $\underset{x^4 = 1}{\pm 1, \pm i}$, $\underset{x^4 = 16}{\pm 2, \pm 2i}$, $\underset{x^4 = 81}{\pm 3 \pm 3i}$

Saving almost 3/4 of the work.

Definition. An element ω in a field F is an n'th root of unity if $\omega^n = 1$. ω is a primitive n'th root of unity (pnru) if $\omega^n = 1$ and $\omega^k \neq 1$ for $1 \leq k \leq n-1$.

Example. In $\mathbb{C}$ $\pm 1, \pm i$ are 4th roots unity. $i^2 = -1 \quad i^4 = (-1)^2 = +1$

of unity ...

Example. In $\mathbb{C}$ $\pm 1, \pm i$ are 4th roots unity. $i^2 = -1$ $i^4 = (-1)^2 = +1$

And i is a p4ru: $\underline{i^0 = 1}$, $i^1 = i$, $i^2 = -1$, $i^3 = -i$, $i^4 = 1$
$n=4$ $\qquad +1 \quad +i \quad -1 \quad -i$

Example. In $\mathbb{Z}_{13}$ $w=5$ is a p4ru.

$w^1, w^2, w^3, w^4 = 1$ $\quad 5^0 = 1$ $\quad 5^1 = 5$ $\quad 5^2 = 12 = -1$ $\quad 5^3 = -5$ $\quad 5^4 = -25 = 1$

$\ne 1 \quad \ne 1 \quad \ne 1 \qquad +1 \quad +5 \quad -1 \quad -5 \qquad 3+8$

Lemma 1. Let ω be a pnru. If $2 | n$ then

(me) (i) $\omega^j = -\omega^{j+\underline{n/2}}$ or $\omega^{j+n/2} = -\omega^j$

(you) (ii) ω^2 is a primitive $\frac{n}{2}$th root of unity.

from (i) (iii) $\omega^0 + \omega^1 + \cdots \omega^{n-1} = 0$.

Proof of (i). $(\omega^{j+n/2} - \omega^j)(\omega^{j+n/2} + \omega^j) = \omega^{2j+n} - \omega^{2j}$
$= \omega^n \omega^{2j} - \omega^{2j} = 0$.

This ↗ or ↗ is 0.

Suppose $\omega^{j+n/2} - \omega^j = 0 \Rightarrow \underset{\ne 0}{\omega^j} (\omega^{n/2} - 1) = 0 \Rightarrow \omega^{n/2} = 1$ $\underset{\ne 1}{}$ $\boxtimes$

Therefore $\omega^{j+n/2} + \omega^j = 0 \Rightarrow \omega^j = -\omega^{j+n/2}$.

Definition. Let ω be a pnru in F a field.
Let $a(x) \in F[x]$ of degree $\le n-1$. Then

$$B = [a(1), a(\omega), a(\omega^2), \ldots, a(\omega^{n-1})] \in F^n$$

is the (discrete) Fourier transform of $a(x)$.

How fast can we compute B?
Using Horner we do $n \cdot O(n) = O(n^2)$ adds and mults in F.

Algorithm DFFT (Discrete Fast Fourier Transform).
Inputs. $n = 2^k$, $A = [a_0, a_1, \ldots, a_{n-1}] \in F^n$, $\omega \in F$ a pnru.
Output. $[\underline{a(1)}, a(\omega), \ldots, a(\omega^{n-1})] \in F^n$ where $a(x) = \sum_{i=0}^{n-1} a_i x^i$

if $n = 1$ $\left(\begin{array}{l} A = [a_0] \\ a(x) = a_0 \end{array} \Rightarrow a(1) = a_0 \right)$ then return A.

$b \leftarrow [a_0, a_2, \ldots, a_{n-2}]$ $\quad | \quad$ $b(x) = a_0 + a_2 x^1 + \cdots + a_{n-2} x^{\frac{n-2}{2}}$

... x^{n-2}

$b \leftarrow [a_0, a_2, \ldots, a_{n-2}]$

$c \leftarrow [a_1, a_3, \ldots, a_{n-1}]$

$b(x) = a_0 + a_2 x^1 + \cdots + a_{n-2} x^{n-2}$

$c(x) = a_1 + a_3 x^1 + a_5 x^2 + \cdots + a_{n-1} x^{n-2}$

$B \leftarrow \mathrm{DFFT}(\frac{n}{2}, b, \omega^2)$ $= [b(1), b(\omega^2), b(\omega^4), \ldots, b(\omega^{n-2})]$

$C \leftarrow \mathrm{DFFT}(\frac{n}{2}, c, \omega^2)$ $= [c(1), c(\omega^2), c(\omega^4), \ldots, c(\omega^{n-2})]$

$\boxed{a(x) = b(x^2) + x\, c(x^2)}$

$y \leftarrow 1$

for $i = 0, 1, \ldots, n/2 - 1$ do

$\quad T \leftarrow y \cdot C_i$ // $y = \omega^i$

$\quad A_i \leftarrow B_i + \omega^i \cdot C_i\ T$ $= b(\omega^{2i}) + \omega^i c(\omega^{2i}) = a(\omega^i)$.

$\quad A_{i+\frac{n}{2}} \leftarrow B_i - \omega^i \cdot C_i\ T$ $= b(\omega^{2i}) - \omega^i c(\omega^{2i})$

$\quad y \leftarrow \omega \cdot y$

end.

$b((\omega^{i+n/2})^2) + \omega^{i+n/2} \cdot c((\omega^{i+n/2})^2)$

$= a(\omega^{i+n/2})$

return $[A_0, A_1, \ldots, A_{n-1}]$.

$A_0 = a(1)$, $A_1 = a(\omega)$, $A_{n-1} = a(\omega^{n-1})$

Cost? Let $T(n)$ be the # multiplications in F that algorithm DFFT does.

$n=1 \quad T(1) = 0$

$n>1 \quad T(n) = 2T(\frac{n}{2}) + 1 + n$

1: ω^2

two recursive calls to DFFT of size $n/2$

n: the loop.

$T(n) = 1 \cdot n \log_2 n + n - 1 \in O(n \log n)$.

Optimization. Precompute $W = [1, \omega, \omega^2, \ldots, \omega^{n/2-1}]$ in $\frac{n}{2} - 2$ mults.

$n>1 \quad T(n) = 2T(n/2) + n/2, \quad T(1) = 0.$

Show that $T(n) = \frac{1}{2} n \log_2 n$.

Total $\frac{1}{2} n \log_2 n + n - 2$

$n - 2$: W array.