FFT 1 algorithm

September 1, 2021 11:34 AM

Let F be a field with $\omega \in F$ a prim. n'th root of unity, i.e., $\omega^n = 1$ and $\omega^i \neq 1$ for $0 < i < n$.

Lemma 1 (n even): (i) $\omega^{n/2} = -1 \Rightarrow$ (ii) $\omega^{j+n/2} = -\omega^j$ and (iii) ω^2 is a prim. $n/2$th root of unity.

Ex. For $F = \mathbb{Z}_p$ where $\boxed{p = nq+1}$ is prime.
Let α a prim. elem in $\mathbb{Z}_p$.
$\Rightarrow \alpha^{p-1} = 1 \Rightarrow \alpha^{nq} = 1 \Rightarrow (\alpha^q)^n = 1$ so $\omega = \alpha^q$ satisfies $\omega^n = 1$ and $\omega^i \neq 1$ for $0 < i < n$.

Let $f(x) = a_0 + a_1 x + \dots + a_{n-1}x^{n-1} \in F[x]$. The Fourier transform of $u = [a_0, a_1, \dots, a_{n-1}] \in F^n$ is
$v = [f(1), f(\omega), f(\omega^2), \dots, f(\omega^{n-1})] \in F^n$.

For $n = 2^k$ the FFT(n, ω, u) computes v as follows.

1. if $n=1$ then $\boxed{\text{return } u.}$ $\quad [u = [a_0] \Rightarrow f(x) = a_0 \Rightarrow f(1) = a_0 \Rightarrow v = [a_0]]$

2. Write $f(x) = (a_0 + a_2x^2 + \dots + a_{n-2}x^{n-2}) + x(a_1 + a_3x^2 + \dots + a_{n-1}x^{n-2})$.
$\boxed{f(x) = a(x^2) + x\,b(x^2)}$ where
$a(x) = a_0 + a_2 x + \dots + a_{n-2}x^{n/2-1}$ and $b(x) = a_1 + a_3 x + \dots + a_{n-1}x^{n/2-1}$.
$A \leftarrow \text{FFT}([a_0, a_2, \dots, a_{n-2}], \omega^2, n/2) = [a(1), a(\omega^2), \dots, a(\omega^{n-2})]$
$B \leftarrow \text{FFT}([a_1, a_3, \dots, a_{n-1}], \omega^2, n/2) = [b(1), b(\omega^2), \dots, b(\omega^{n-2})]$

Need
$f(1) = a(1) + 1 \cdot b(1)$
$f(\omega) = a(\omega^2) + \omega \cdot b(\omega^2)$
$\vdots$
$f(\omega^{n/2-1}) = a(\omega^{n-2}) + \omega^{n/2-1} \cdot b(\omega^{n-2})$

$f(\omega^{n/2}) = a(\omega^n) + \omega^{n/2} \cdot b(\omega^n)$
$= a(1) + -b(1).$
$f(\omega^{n/2+1}) = a(\omega^{n+2}) + \omega^{n/2+1} \cdot b(\omega^{n+2})$
$= a(\omega^2) + -\omega\, b(\omega^2)$
$f(\omega^{n-1}) = \underline{\qquad\qquad\qquad}$

3. for $i = 0, 1, \dots, n/2-1$ do
$T := \omega^i \cdot B_i$
$C_i := A_i + T \qquad C_{i+n/2} := A_i - T$

4. Output C.

Let $T(n)$ be the # of multiplications in F that FFT does.

(4) Output c.

Cost. Let $T(n)$ be the # of multiplications in F that FFT does.

Assuming we pre-compute

$$F^n \ni W = [\underbrace{1, \omega, \omega^2, \dots, \omega^{n/2-1}}_{n/2}, \underbrace{1, \omega^2, \omega^4, \dots, \omega^{n/2-2}}_{n/4}, \underbrace{1, \omega^4, \omega^8, \dots, \omega^{n/2-4}}_{n/8}, \dots, 1, 0]$$

$$n/2 + n/4 + n/8 + \dots = n-1$$

$$T(n) = 2T(n/2) + n/2, \quad T(1) = 0 \Rightarrow T(n) = \frac{n}{2}\log_2 n \in O(n \log n).$$

The inverse transform obtains $u = [a_0, \dots, a_{n-1}]$ from $v = [f(1), \dots, f(\omega^{n-1})]$ using $\frac{1}{n}\text{FFT}(v, n, \omega^{-1})$.