The Fast Fourier Transform

October 11, 2023 4:54 PM

Algorithm DFFT (optimized).

Input $A = [a_0, a_1, \ldots, a_{n-1}] \in F^n$ representing $a(x) = \sum_{i=0}^{n-1} a_i x^i$.

$n = 2^k$ ω is a pnru i.e $\omega^n = 1$.

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

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

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

two temporary arrays s.d size $\frac{n}{2}$.

$B \leftarrow [\underbrace{a_0, a_2, a_4, \ldots, a_{n-2}}]$ $C \leftarrow [a_1, a_3, \ldots, a_{n-1}]$

$b(x) = \sum_{i=0}^{n/2-1} a_{2i} x^i$ $c(x) = \sum_{i=0}^{n/2-1} a_{2i+1} x^i$

$$a(x) = b(x^2) + x \cdot c(x^2)$$

DFFT$(B, n/2, W + n/2)$ // $B = [b(1), b(\omega^2), b(\omega^4), \ldots, b(\omega^{n-2})]$

DFFT$(C, n/2, W + n/2)$ // $C = [c(1), c(\omega^2), c(\omega^4), \ldots, c(\omega^{n-2})]$.

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

$T \leftarrow W_i \cdot C_i$ // $= \omega^i \cdot c(\omega^{2i})$

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

$A_{i+\frac{n}{2}} \leftarrow B_i - T$ // $= a(\omega^{i+n/2})$.

return.

Let $M(n)$ be the # of multiplications in F done.

$M(n) = \underset{\substack{\uparrow \\ \text{two recursive calls}}}{2} M(n/2) + n/2$ $\leftarrow$ loop.

$M(1) = 0$.

$M(n) = \frac{1}{2} n \log_2 n \in O(n \log n)$.

Let $S(n)$ be the # units of storage for the temporary arrays.

$S(n) = \underset{\substack{\uparrow \\ B \text{ and } C}}{2 \frac{n}{2}} + \underset{\substack{\uparrow \\ 2 \text{ recursive calls}}}{2} S(n/2) = 2S(n/2) + n$.

$$S(n) = \underbrace{2\tfrac{n}{2}}_{\uparrow \text{ B and C}} + \underbrace{2S(n/2)}_{\uparrow \text{ 2 recursive calls}} = 2S(n/2) + n.$$

$$S(1) = 0.$$

$$\Rightarrow S(n) = n \log_2 n.$$

How can we use $O(n)$ space?

The Second FFT Algorithm

Modern Computer Algebra 8.2

Let $a(x) = (a_0 + a_1 x + \cdots + a_{\frac{n}{2}-1} x^{n/2-1}) + (a_{\frac{n}{2}} x^{n/2} + a_{\frac{n}{2}+1} x^{\frac{n}{2}+1} + \cdots + a_{n-1} x^{n-1})$

(1) $a \div x^{\frac{n}{2}} - 1$ $\quad a(x) = q_0(x)\cdot(x^{\frac{n}{2}} - 1) + r_0(x).$ $\quad r_0(x) = a(x^{\frac{n}{2}} = 1).$

$$r_0(x) = (a_0 + a_{\frac{n}{2}}) + (a_1 + a_{\frac{n}{2}+1})\cdot x + \cdots + (a_{\frac{n}{2}-1} + a_{n-1})\cdot x^{\frac{n}{2}-1}.$$

(2) $a \div \underline{x^{n/2}+1}$ $\quad a(x) = q_1(x)\cdot(x^{\frac{n}{2}} + 1) + r_1(x)$ $\quad r_1(x) = a(x^{\frac{n}{2}} = -1).$

$$r_1(x) = (a_0 - a_{\frac{n}{2}}) + (a_1 - a_{\frac{n}{2}+1})\cdot x + \cdots + (a_{\frac{n}{2}-1} - a_{n-1})\cdot x^{\frac{n}{2}-1}.$$

We can compute r_0 in $n/2$ additions and r_1 in $n/2$ subtractions.

Observe

$$a(\underline{\omega^{2i}}) \overset{(1)}{=} q_0(\omega^{2i})\cdot(\underbrace{\omega^{\frac{2in}{2}}}_{=1} - 1) + \underline{r_0(\omega^{2i})} = r_0(\omega^{2i}) \leftarrow$$

2 recursive calls to to DFFT

$$a(\underline{\omega^{2i+1}}) \overset{(2)}{=} q_1(\omega^{2i+1})(\underbrace{\omega^{\frac{2in}{2}+\frac{n}{2}}}_{=\omega^{n/2}=-1} + 1) + r_1(\omega^{2i+1}) = r_1(\omega^{2i+1}).$$

$$\omega^{2i+1} = \omega^{2i}\cdot\omega \qquad = r_1^*(\omega^{2i}).$$

$0 \le i < \frac{n}{2}$

where $r_1^*(x) = r_1(\omega\cdot x) = \sum_{i=0}^{n/2-1} (a_i - a_{\frac{n}{2}+i})(\omega x)^i = \sum_{i=0}^{n/2-1} \left[(a_i - a_{n/2+i})\cdot\omega^i\right]\cdot x^i$

We can obtain $r_1^*(x)$ from $a(x)$ by doing $\frac{n}{2}$ subs and $\frac{n}{2}$ mults.

Algorithm FFT_2

Input $A = [a_0, a_1, \ldots, a_{n-1}] \in F^n$, $n = 2^k$ and
$W = [1, \omega^1, \omega^2, \ldots \;] \in F^n$.

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

if $n = 1$ then return.

$B \leftarrow \text{Array}(0..\frac{n}{2}-1)$.

$C \leftarrow \text{Array}(0..\frac{n}{2}-1)$.

for $i = 0, 1, \ldots, \frac{n}{2}-1$ do

  $B_i \leftarrow A_i + A_{i+n/2}$ // $B = r_0(x)$.

  $C_i \leftarrow (A_i - A_{i+n/2}) \cdot W_i$ // $C = r_1^*(x)$.

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

FFT2$(B, \frac{n}{2}, W + \frac{n}{2})$ // $B = [\, r_0(\omega^{2i}) : 0 \le i \le \frac{n}{2}-1 \,]$

FFT2$(C, \frac{n}{2}, W + \frac{n}{2})$ // $C = [\, r_1^*(\omega^{2i}) : 0 \le i \le \frac{n}{2}-1 \,]$.

$= a(\omega^{2i+1})$.

for $i = 0, 1, \ldots, \frac{n}{2}-1$ do

  $A_{2i} \leftarrow B_i$

  $A_{2i+1} \leftarrow C_i$

return.

Let $M(n)$ be the # mults in F done.

$$M(n) = \frac{n}{2} + 2M(\frac{n}{2})$$

$$M(1) = 0$$

$\Rightarrow M(n) = \frac{1}{2}\log_2 n$ mults.

This is the same as the first algorithm.