The FFT in three-and-a-half facts

Alex Townsend

April 28, 2015

The fast Fourier transform (FFT) is the world's best algorithm to compute the following sums called the discrete Fourier transform (DFT):

$$f_k = \sum_{n=0}^{N-1} c_n e^{-2\pi i n k/N}, \qquad 0 \leq n \leq N-1.$$

Instead of $\mathcal{O}(N^2)$ operations, the FFT does it in $\mathcal{O}(N \log N)$ operations. Just think about that saving: when $N = 10^6$ then we have reduced one trillion operations to about six million. But why do we need FFTs? Well, they are the discrete version of the Fourier transform. Fourier transforms are great, but most of the time one can not get neat analytic formulas and algebraic manipulations fail. We have to go numerical. The FFT always works, it's numerical, it's practical, it's fast, it's a top-ten algorithm.

The FFT algorithm is based on three-and-a-half facts. (For the rest of the document we take $N = 2^l$, a power of 2.)

Fact 1 is an observation. It makes the DFT easy to remember and understandable.

Fact 1: The DFT is polynomial evaluation at the roots of unity (in reverse order).

$$f_k = p(z_k) = \sum_{n=0}^{N-1} c_n z_k^n, \qquad z_k = e^{-2\pi i k/N}, \quad 0 \leq k \leq N-1$$

Fact 2 hints at the underlying divide-and-conquer approach that the FFT employs. It is a fact about polynomials.

Fact 2: An even degree polynomial can be decomposed in two parts.

$$p(z) = p_{even}(z^2) + z p_{odd}(z^2)$$

In the above fact we have

$$p(z) = \sum_{n=0}^{N-1} c_n z^n = \underbrace{\sum_{n=0}^{N/2-1} c_{2n} z^{2k}}_{p_{even}(z^2)} + z \underbrace{\sum_{n=0}^{N/2-1} c_{2n+1} z^{2k}}_{p_{odd}(z^2)}.$$

The FFT must also exploit properties of the roots-of-unity. Fact 3 is about roots-of-unity.

Fact 3: Roots-of-unity squared are roots-of-unity.

$$z_k^2 = \left(e^{2\pi ik/N}\right)^2 = e^{2\pi ik/(N/2)}$$

We now have every in place (up to half a fact) to write:

$$\begin{pmatrix} p(z_0) \\ \vdots \\ p(z_{N/2-1}) \\ p(z_{N/2}) \\ \vdots \\ p(z^{N-1}) \end{pmatrix} = \begin{pmatrix} I & I \\ & \\ I & -I \end{pmatrix} \begin{pmatrix} p_{even}(z_0^2) \\ \vdots \\ p_{even}(z_{N/2-1}^2) \\ z_0 p_{odd}(z_0^2) \\ \vdots \\ z_{N/2-1} p_{odd}(z_{N/2-1}^2) \end{pmatrix} \tag{1}$$

Please convince yourself that evaluating $p_{even}(z_k^2)$ and $p_{odd}(z_k^2)$ for $0 \le k \le N/2-1$ is just the DFT again with N replaced by $N/2$. How do you compute $p_{even}(z_k^2)$ and $p_{odd}(z_k^2)$? Use the three facts again, and then again, and then again. You will need to use it $\log_2(N)$ times.

Oh. You may need just half a fact more to be comfortable with (1).

Fact 3.5: Roots-of-unity contain their reflections (when N is even).

$$z_k = -z_{N/2+k}, \qquad 0 \le k \le N/2-1$$

Fact 3.5 tells us that $p_{even}(z_{N/2+k}^2) = p_{even}(z_k^2)$ and $z_{N/2+k}p_{odd}(z_{N/2+k}^2) = -z_k p_{odd}(z_k^2)$.