The Radix-2 Fast Fourier Transform

Under the constraint $N = r^v$, we have additional symmetry. Let us consider the case $r = 2$, and reexamine the divide and conquer algorithm with $M = N/2$ and $L = 2$.

Define:

$$f_1(n) = x(2n)$$

$$f_2(n) = x(2n+1)$$

This is referred to as a decimation-in-time approach. Why?
Let us derive a simplified expression for $X(k)$:

$$\begin{aligned} X(k) &= \sum_{n=0}^{N-1} x(n) W_N^{kn} \\ &= \sum_{n\ even} x(n) W_N^{kn} + \sum_{n\ odd} x(n) W_N^{kn} \\ &= \sum_{m=0}^{(N/2)-1} x(2m) W_N^{2mk} + \sum_{m=0}^{(N/2)-1} x(2m+1) W_N^{k(2m+1)} \end{aligned}$$

But, $W_N^2 = W_{N/2}$:

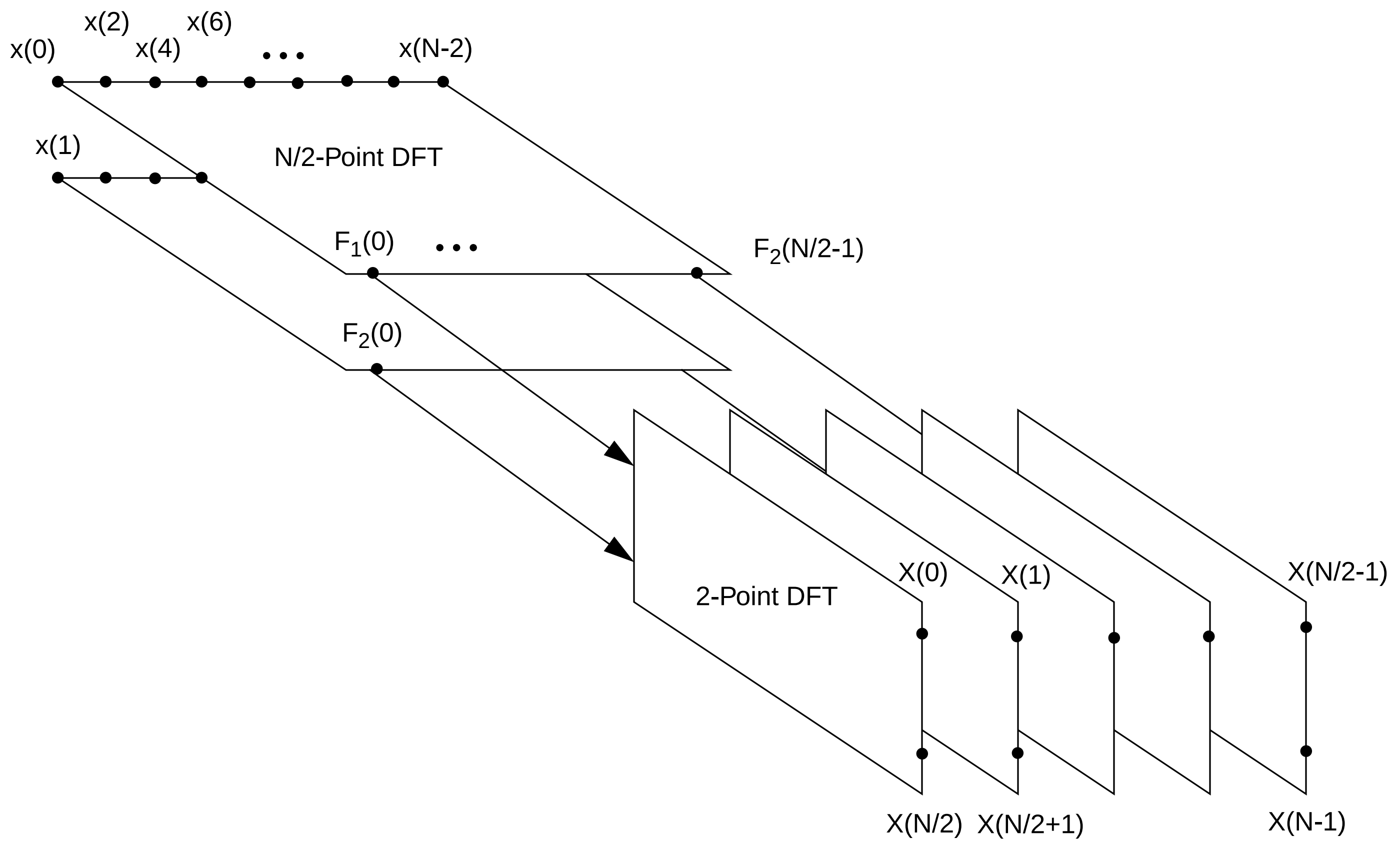
$$X(k) = \sum_{m=0}^{(N/2)-1} f_1(m) W_{N/2}^{km} + W_N^k \sum_{m=0}^{(N/2)-1} f_2(m) W_{N/2}^{km}$$

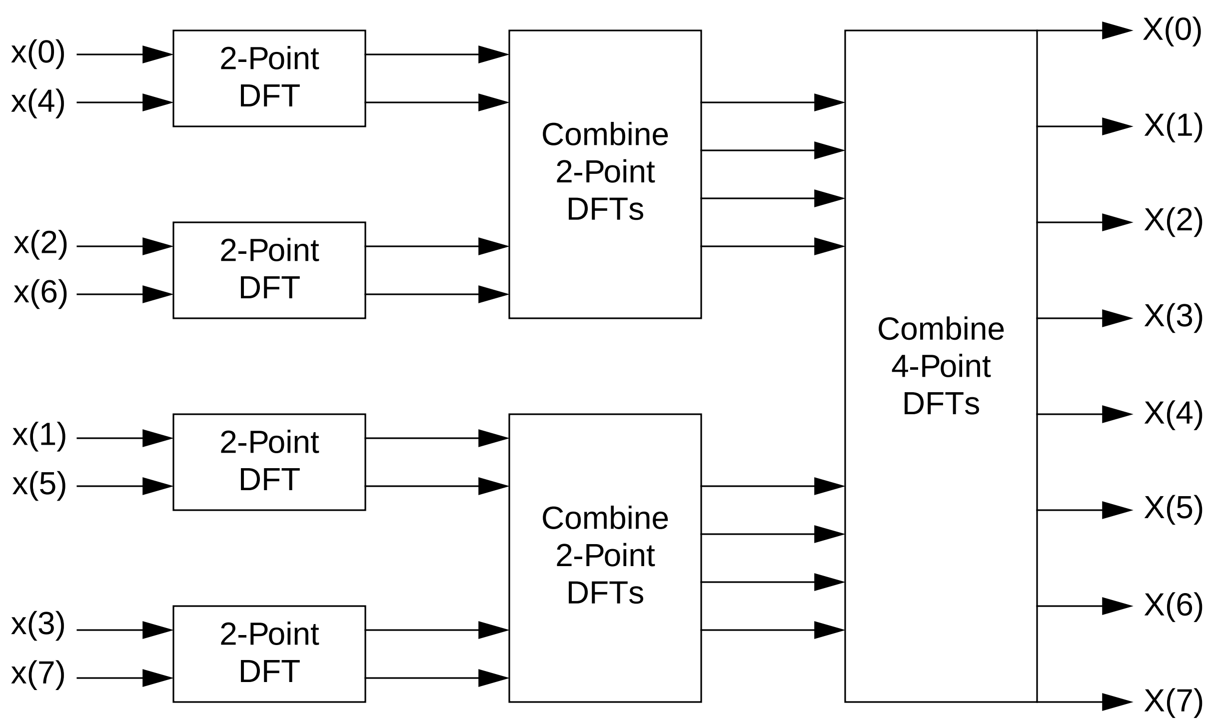
Note that $F_1(k)$ and $F_2(k)$ are $N/2$-point DFTs. This implies a reduction of a factor of 2 for large N. We can repeat the process by reducing $F_1(k)$ (and $F_2(k)$) from $N/2$ to $N/4$-point transforms, and so on. This gives a complexity of $O(N \log N)$ as opposed to $O(N^2)$.

This approach gives rise to a family of algorithms known as the Radix-2 Fast Fourier Transform. It has a simple graphical interpretation.

We can also do decimation-in-frequency.

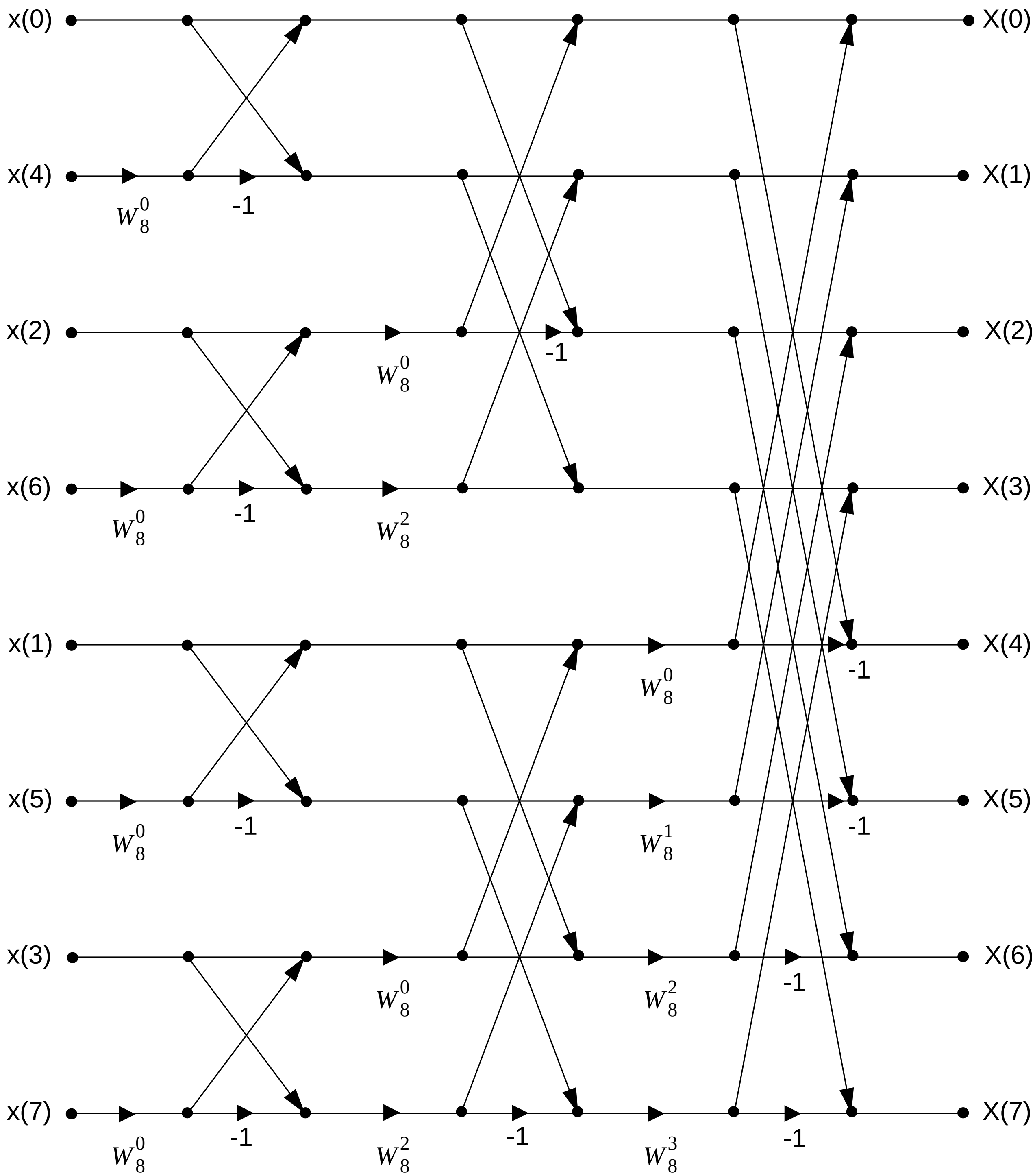
The Decimation-In-Time FFT





Signal Flow Graphs For Decimation-In-Time FFTs
(Butterfly Diagrams)

An implementation of an 8-point DFT:



DFT of Two Real Sequences

Consider:

$$x(n) = x_1(n) + jx_2(n)$$

Because the DFT is a linear operator:

$$X(k) = X_1(k) + jX_2(k)$$

But,

$$x_1(n) = [x(n) + x^*(n)]/2$$

$$x_2(n) = [x(n) - x^*(n)]/2j$$

Therefore,

$$X_1(k) = \frac{1}{2}\{DFT[x(n)] + DFT[x^*(n)]\}$$

$$X_2(k) = \frac{1}{2j}\{DFT[x(n)] - DFT[x^*(n)]\}$$

But, $DFT[x^*(n)] = X^*(N-k)$, which implies:

$$X_1(k) = \frac{1}{2}[X(k) + X^*(N-k)]$$

$$X_2(k) = \frac{1}{2j}[X(k) - X^*(N-k)]$$

Strategy:

(1) Compute the FFT of $x(n)$.

(2) Use simple math to split the result apart.

DFT of a 2N-Point Real Sequence

Consider $g(n)$ as a 2N-point real signal. Let us define two signals:

$$x_1(n) = g(2n)$$

$$x_2(n) = g(2n+1)$$

and, of course,

$$x(n) = x_1(n) + jx_2(n)$$

We can show:

$$G(k) = X_1(k) + W_{2N}^k X_2(k)$$

$$G(k+N) = X_1(k) - W_{2N}^k X_2(k)$$

Strategy:

(1) Form $x_1(n)$ and $x_2(n)$.

(2) Take N-point DFTs.

(3) Combine

Note the reduction from length 2N to length N in complexity.

Quantization Properties

One of the reasons the DFT/FFT is so popular is that is very robust to quantization noise (because it is a linear operator).

We can show that the numerical properties for a DFT are:

$$\frac{\sigma_x^2}{\sigma_q^2} = \frac{2^{2b}}{N^2} \qquad 0 \le |x(n)| \le \frac{1}{N}$$

Note that the SNR decreases with the length of the FFT.

If the input is scaled to have a maximum magnitude of 1, then:

$$\frac{\sigma_x^2}{\sigma_q^2} = 2^{2b} \qquad |x(n)| \le 1$$

For an FFT with scaling,

$$\frac{\sigma_x^2}{\sigma_q^2} = 2^{2b - v - 1}$$

where v is the radix base of the FFT.

Note that the SNR in dB is linearly related to the number of bits (similar to what we observed with the uniform quantizer).