Fast Fourier Transform

DFT

The discrete Fourier transformation of the sampled $f(x)$ is:

$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)e^{-j2\pi xu/N}$$

for $u = 0, 1, 2, 3, \ldots, N-1$

Inverse:

$$f(x) = \sum_{u=0}^{N-1} F(u)e^{j2\pi xu/N}$$
DFT $\Rightarrow$ FFT

$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) W_N^{ux}$, where $W_N^{ux} = e^{-j2\pi ux/N}$

Let $N = 2K$

$F(u) = \frac{1}{2} \left[ \frac{1}{K} \sum_{x=0}^{K-1} f(2x) W_{2K}^{ux(2x)} + \frac{1}{K} \sum_{x=0}^{K-1} f(2x+1) W_{2K}^{ux(2x+1)} \right]$

since $W_{2K}^{ux(2x)} = e^{-j2\pi ux(2x)/2K} = e^{-j2\pi u(x)/K} = W_K^{ux}$

$F(u) = \frac{1}{2} \left[ \frac{1}{K} \sum_{x=0}^{K-1} f(2x) W_K^{ux} + \frac{1}{K} \sum_{x=0}^{K-1} f(2x+1) W_K^{ux} W_{2K}^{u} \right]$

DFT $\Rightarrow$ FFT

$F(u) = \frac{1}{2} \left[ \frac{1}{K} \sum_{x=0}^{K-1} f(2x) W_K^{ux} + \frac{1}{K} \sum_{x=0}^{K-1} f(2x+1) W_K^{ux} W_{2K}^{u} \right]$

Define $F_{even}(u) = \frac{1}{K} \sum_{x=0}^{K-1} f(2x) W_K^{ux}$, $F_{odd}(u) = \frac{1}{K} \sum_{x=0}^{K-1} f(2x+1) W_K^{ux}$

For $u = 0\ldots K-1$

$F(u) = \frac{1}{2} \left[ F_{even}(u) + F_{odd}(u) W_{2K}^{u} \right]$  For $u = 0\ldots K-1$

$F(u) = \frac{1}{2} \left[ F_{even}(u) - F_{odd}(u) W_{2K}^{u} \right]$  For $u = K\ldots 2K-1$

since $W_K^{u+K} = e^{-j2\pi (u+k)/K} = W_K^{u} e^{-j2\pi} = W_K^{u} W_{2K}^{u} = W_{2K}^{u} e^{-j\pi}$
Divide and Conquer

\[ F(u) = \frac{1}{2} [F_{\text{even}}(u) + F_{\text{odd}}(u)W_{2K}^u] \quad \text{For } u = 0 \ldots K-1 \]

\[ F(u) = \frac{1}{2} [F_{\text{even}}(u) - F_{\text{odd}}(u)W_{2K}^u] \quad \text{For } u = K \ldots 2K-1 \]

Subdivision recursively until we get down to 2-point transforms.

Implementation

• Bottom-up, i.e., starting with two points

\[ F(0) = W_2^0 f(0) + W_2^0 f(1) = f(0) + W_2^0 f(1) \]
\[ F(1) = W_2^0 f(0) + W_2^1 f(1) = f(1) + W_2^1 f(1) \]
Four-point FFT

\[ f(0) \rightarrow F(0) \]
\[ f(2) \rightarrow F(1) \]
\[ f(1) \rightarrow F(2) \]
\[ f(3) \rightarrow F(3) \]

Generic butterfly graph

\[ W_N^s = W_N^{s+\frac{N}{2}} = -W_N^s \]
Eight-point FFT

In-place Implementation

- Pair-wise computation

Input: 000, 001, 010, 011, 100, 101, 110, 111

Level 1: (000, 010, 100, 110), (001, 011, 101, 111)

Level 2: ((000, 100), (010, 110)), (001, 101), (011, 111)

Final Order 000, 100, 010, 110, 001, 101, 011, 111

Sorting the numbers in bit-reversed order
1D FFT Steps

Given N point (N that is a power of two), calculate its N-point FFT.
1. Gather your samples into a buffer of size N
2. Sort the samples in bit-reversed order and put them in a complex N-point buffer (set the imaginary parts to zero)
3. Apply the first stage butterfly using adjacent pairs of numbers in the buffer
4. Apply the second stage butterfly using pairs that are separated by 2
5. Apply the third stage butterfly using pairs that are separated by 4
6. Continue butterflying the numbers in your buffer until you get to separation of N/2
7. The buffer will contain the Fourier transform

Code Samples

```c
void Fft::Transform () // this code assumes that the input has been scrambled
{
// step = 2 ^ (level-1)
// increm = 2 ^ level;
int step = 1;
for (int level = 1; level <= _logPoints; level++)
{
    int increm = step * 2;
    for (int j = 0; j < step; j++)
    {
        Complex U = exp ( - 2 PI j / 2 ^ level )
        for (int i = j; i < _Points; i += increm)
        {
            // butterfly
            Complex T = U*X[i+step];
            X[i+step] = X[i] - T
            X[i] = X[i] + T;
        }
    }
}
step *= 2;
}
```
Separability of 2D FFT

More Implementation Details

- Center the frequency image
  - $F(0, 0) @ (M/2, N/2)$
  - Multiplying $f(x,y)$ with $(-1)^{x+y}$
- Inverse Transform with forward FFT routine
  - $f^*(x,y)/MN$ - a scaled conjugate of $f(x,y)$

Read 4.6 in the textbook for more