UNIVERSITY OF PORTSMOUTH

The Fast Fourier Transform

HAMIDREZA KHALEGHZADEH

Overview

The Fast Fourier Transform (FFT) is a very important algorithm with practical applications in many areas including:

- Audio processing
- Image processing and image reconstruction
- Solving mathematical equations (e.g. in N-body problems)
- Large integer multiplication

According to IEEE Computing in Science & Engineering, it is one of the "Top 10 Algorithms of 20th Century" [†].

[†] http://cse8803hnc.gatech.edu/Lecture2/Dongarra_TOP10_Algorithms_CSE_2000.pdf

DFT Reprise

As we saw last week, the basic formulae of the Discrete Fourier Transform and its inverse are:

$$C_k = 1/N \sum_{n=0}^{N-1} X_n e^{-2\pi i\, kn/N}$$

$$X_n = \sum_{k=0}^{N-1} C_k e^{2\pi i\, kn/N}$$

Here the N numbers X_n are discrete samples of some function (like a density or pressure) of position or time, taken at regular intervals.

The N complex numbers C_k are Fourier components of this function, with k representing frequency or wave number of the component.

Complexity of "Naïve" DFT

A serious problem with using the formulae on the previous slide in practice is their time complexity, in computational terms.

For example each C_k is a sum over N terms, so each element takes $O(N)$ time. There are N such elements, so the overall complexity of implementing the first formula for the DFT directly is $O(N^2)$.

The inverse transform (second formula) has the same complexity.

Small DFTs

To understand these formulae better, let's consider some small examples.

We will consider the forward transformation (first formula on slide 4), but, to avoid distraction, ignore the overall 1/N normalization factor.

First, for $N = 2$, original formula gives us.

$$C_0 = X_0 + X_1$$

$$C_1 = X_0 + e^{-i\pi} X_1$$

Where I have only used the fact $e^0 = 1$.

I could also use $e^{-i\pi} = -1$ if I wished, but for now I'll leave that as it is.

DFT for $N = 4$

Original formula gives us:

$$C_0 = X_0 + X_1 + X_2 + X_3$$

$$C_1 = X_0 + e^{-i\pi/2}X_1 + e^{-i\pi}X_2 + e^{-3i\pi/2}X_3$$

$$C_2 = X_0 + e^{-i\pi}X_1 + e^{-2i\pi}X_2 + e^{-3i\pi}X_3$$

$$C_3 = X_0 + e^{-3i\pi/2}X_1 + e^{-3i\pi}X_2 + e^{-9i\pi/2}X_3$$

DFT for $N = 4$

Collecting even and odd n terms separately:

$$C_0 = (X_0 + X_2) + (X_1 + X_3)$$

$$C_1 = \left(X_0 + e^{-i\pi} X_2\right) + e^{-i\pi/2}\left(X_1 + e^{-i\pi} X_3\right)$$

$$C_2 = (X_0 + X_2) + e^{-i\pi}(X_1 + X_3)$$

$$C_3 = \left(X_0 + e^{-3i\pi} X_2\right) + e^{-3i\pi/2}\left(X_1 + e^{-3i\pi} X_3\right)$$

DFT for $N = 4$

Just use fact $e^{-2i\pi} = 1$ to simplify last line:

- $e^{-3i\pi} = 3^{-2i\pi} * e^{-i\pi} = 1 * e^{-i\pi} = e^{-i\pi}$

$$C_0 = (X_0 + X_2) + (X_1 + X_3)$$

$$C_1 = \left(X_0 + e^{-i\pi} X_2\right) + e^{-i\pi/2}\left(X_1 + e^{-i\pi} X_3\right)$$

$$C_2 = (X_0 + X_2) + e^{-i\pi}(X_1 + X_3)$$

~~$$C_3 = \left(X_0 + e^{-3i\pi} X_2\right) + e^{-3i\pi/2}\left(X_1 + e^{-3i\pi} X_3\right)$$~~

$$C_3 = \left(X_0 + e^{-i\pi} X_2\right) + e^{-3i\pi/2}\left(X_1 + e^{-i\pi} X_3\right)$$

DFT for $N = 4$

Extracting some "common subexpressions":

$$C_0 = C_0^{even} + C_0^{odd}$$

$$C_1 = C_1^{even} + e^{-i\pi/2} C_1^{odd}$$

$$C_2 = C_0^{even} + e^{-i\pi} C_0^{odd}$$

$$C_3 = C_1^{even} + e^{-3i\pi/2} C_1^{odd}$$

where we defined:

$$C_0^{even} = X_0 + X_2, \qquad C_1^{even} = X_0 + e^{-i\pi} X_2$$

$$C_0^{odd} = X_1 + X_3, \qquad C_1^{odd} = X_1 + e^{-i\pi} X_3$$

A Recursion?

Comparing with slide 5, we see that the C^{even} coefficients correspond to an $N = 2$ DFT on the even elements $[X_0, X_2]$ of the original X vector, and the C^{odd} coefficients correspond to an $N = 2$ DFT on the odd elements $[X_1, X_3]$.

DFT($[X_0, X_1]$) for N = 2 (slide 5):

$$C_0 = X_0 + X_1 \qquad C_1 = X_0 + e^{-i\pi} X_1$$

Therefore, for a signal $[X_0, X_1, X_2, X_3]$:

DFT($[X_0, X_2]$) inclduing even indexed points:

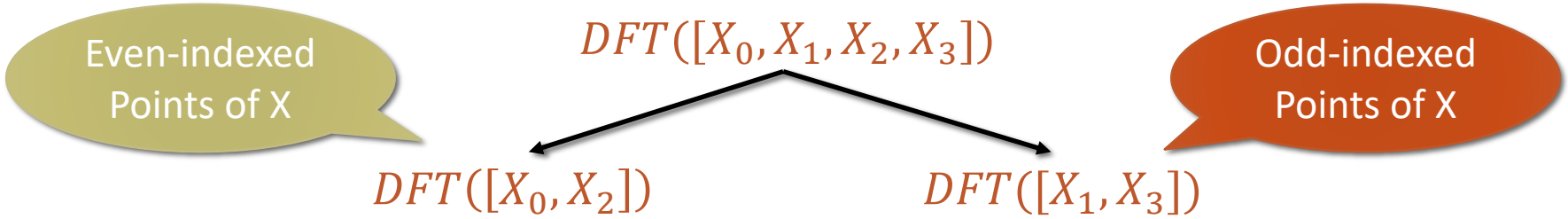
$$C_0^{even} = X_0 + X_2 \qquad C_1^{even} = X_0 + e^{-i\pi} X_2$$

DFT($[X_1, X_3]$) including odd indexed points:

$$C_0^{odd} = X_1 + X_3 \qquad C_1^{odd} = X_1 + e^{-i\pi} X_3$$

A Recursion?

So, we can reduce the $N = 4$ DFT to two $N = 2$ DFTs, with a few extra operations to combine them.



It doesn't take too much algebra to show that this pattern generalizes, in the way shown on the next slide.

Divide and Conquer

Can fairly easily generalize the $N = 4$ case to show that if N is any even number, a size N DFT for a signal $[X_0, \ldots, X_{N-1}]$ can be written as follows:

$$C_k = \begin{cases} C_k^{even} + e^{-2i\pi k/N} C_k^{odd}, & k < N/2 \\ C_{k-N/2}^{even} + e^{-2i\pi k/N} C_{k-N/2}^{odd}, & k \geq N/2 \end{cases}$$

where the C^{even} coefficients correspond to a size N/2 DFT on the even elements:

$$[X_0, X_2, \cdots, X_{N-2}]$$

of the original X vector, and the C^{odd} coefficients correspond to a size $N/2$ DFT on the odd elements:

$$[X_1, X_3, \cdots, X_{N-1}]$$

Cooley-Tukey FFT

The recursive form on the previous slide for the coefficients C_k on the previous slide is the foundation for the Cooley-Tukey Fast Fourier Transform.

Suppose the time to compute the transform using this formula is a function $T(N)$. The additions, multiplications, and complex exponentials in the formula take time $O(N)$, and calculating the two size $N/2$ DFTs takes time $2T(N/2)$. So:

$$T(N) = O(N) + 2T(\frac{N}{2})$$

For simplicity assume N a power of 2. The depth of recursion will be $\log_2 N$. So the overall complexity is:

$$T(N) = O(N \log N)$$

$O(N^2)$ vs $O(N \log N)$

N	1000	10^6	10^9
$O(N^2)$	10^6	10^{12}	10^{18}
$O(N\ log_2 N)$	10^4	$20 * 10^6$	$30 * 10^9$

Suppose it took 1 nanosecond to perform one operation. So, a DFT analysis with the size of $N = 10^9$

- Naïve DFT: 10^{18} ns -> 31.2 years
- FFT: $30 * 10^9$ ns -> 30 seconds

Twiddle Factors

The imaginary exponentials in the recursive formula for the coefficients on side 11 are technically known as *twiddle factors*.

It is common to use this formula:

$$e^{-2i\pi k/N} = e^{-i\pi} . e^{-2i\pi(k-\frac{N}{2})/N} = -e^{-2i\pi(k-\frac{N}{2})/N}$$

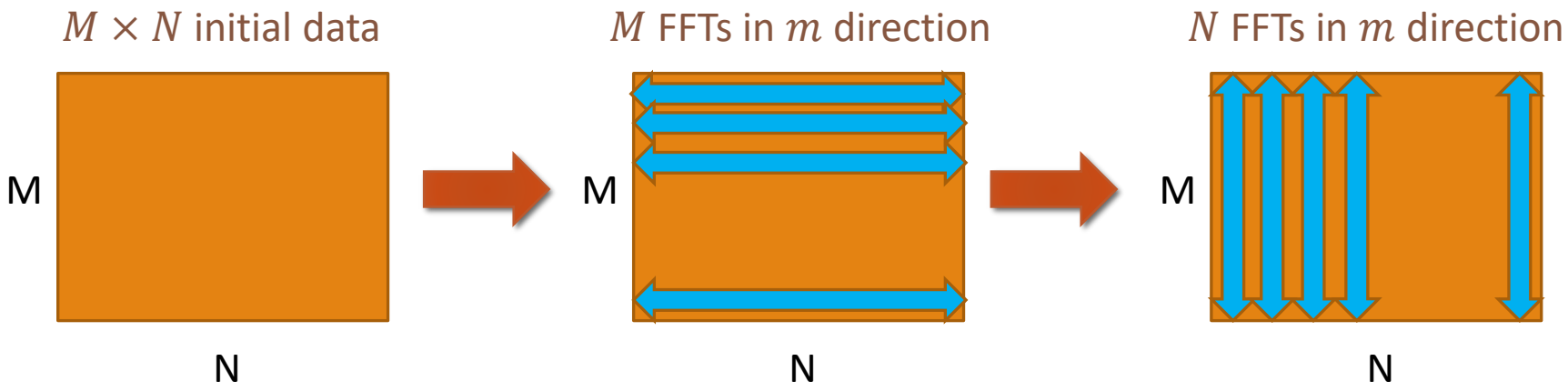
to rewrite the basic formula as:

$$C_k = \begin{cases} C_k^{even} + e^{-2i\pi k/N} C_k^{odd}, & k < N/2 \\ C_{k-N/2}^{even} - e^{-2i\pi(k-\frac{N}{2})/N} C_{k-N/2}^{odd}, & k \geq N/2 \end{cases}$$

which means we only need to compute the $N/2$ twiddle factors for $k < N/2$ (and reuse them in the second case, $k \geq N/2$).

Multidimensional FFT

In problems with more than one (e.g. spatial) dimension, can perform the multidimensional DFT fast by simply applying the FFT to each dimension in turn.



Complexity:

$$M \cdot N \log N + N \cdot M \log M = MN(\log N + \log M) = MN \log MN$$

Examples

Over the next couple of weeks we will cover a couple of interesting scenarios for use of Fourier Transforms and FFTs.

In the mean time, simple examples include pitch analysis and correction in sound engineering, or emphasizing high frequency or low frequency components in image filtering.

Further interesting applications depend on other mathematical properties of the Fourier Transform.