Abstract—This paper studies the algorithmic complexity of FFTs using the method of Asymptotic analysis. There are many ways in which the FFTs are computed. This paper looks at one of the most commonly used algorithms called the Cooley-Tukey FFT Algorithm [1]. This algorithm employs a divide and conquer approach. Then the paper studies an algorithm for the computation of FFT of a 2D matrix, a technique commonly used in Image Signal Processing applications.

1. INTRODUCTION

The Discrete Fourier Transform is widely used across a large number of fields. It’s used for spectral analysis of time domain signals, in the field of data compression, partial differential equations etc. And hence its implementation on computers has been extensively researched and as a result several algorithms have been developed for its implementation. FFT is used to implement DFT. There are several algorithms implementing FFT. This paper studies one of the most commonly used algorithms called the Cooley-Tukey FFT Algorithm [1]. It employs a divide and conquer approach along with recursion. We study the performance improvement due to this approach and the direct approach. Then we look at a technique to compute 2D FFT. This algorithm also employs a divide and conquer approach in an unconventional sense, as we will study in the corresponding section.

2. 1D FFT

In this section, we assume that \( x(n), n \in \{0,1,\ldots,N-1\} \) is a complex input sequence and \( X(k), k \in \{0,1,\ldots,N-1\} \) is the FFT of the input sequence.

2.1. Direct Computation of DFT

The DFT of an input sequence is given by:

\[
X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi nk}{N}}
\]

To compute one DFT point, \( N \) complex multiplications and \( N-1 \) complex additions are required. And hence to compute the whole of \( N \) point FFT, \( N^2 \) complex multiplications and \( N(N-1) \) complex additions are required. Now each complex multiplication requires 4 real multiplications and 2 real additions and each complex addition requires 2 more real additions. Hence, in all, the \( N \) point FFT requires \( 4N^2 \) real multiplications and \( 4N^2 - 2N \) real additions. Considering that multiplication is a much more complex operation than that of addition, here we make an assumption and say that the complexity of the algorithm is the number of multiplications, i.e. \( 4N^2 \). In asymptotic notations, the algorithm is \( O(N^2) \).

2.2. FFT

Let us assume that the size of input sequence \( N \) is an integer exponent of 2, i.e. \( N = 2^k \), where \( k \) is an integer. Now as the first step of the divide and conquer algorithm, we divide the \( N \) point FFT into two \( \frac{N}{2} \) point FFTs with odd indices in one and the even indices in the other FFT. Let the odd indices be \( 2k+1 \) and the even indices be \( 2k, k \in \{0,1,\ldots,N/2 - 1\} \). Hence,

\[
X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi nk}{N}}
\]

Dividing,

\[
X(k) = \sum_{r=0}^{N/2-1} x(2r)e^{-j\frac{2\pi 2rk}{N}} + \sum_{r=0}^{N/2-1} x(2r+1)e^{-j\frac{2\pi (2r+1)k}{N}}
\]

\[
X(k) = \sum_{r=0}^{N/2-1} x(2r)e^{-j\frac{2\pi rk}{N/2}} + e^{-j\frac{2\pi k}{N}} \sum_{r=0}^{N/2-1} x(2r+1)e^{-j\frac{2\pi rk}{N/2}}
\]

\[
X(k) = X_{even}(k) + e^{-j\frac{2\pi k}{N}}X_{odd}(k)
\]

Where \( X_{even}(k) \) and \( X_{odd}(k) \) are the FFTs of the odd and the even subsequence of the original input sequence respectively. Similarly each of these even and odd sequences can be divided further \( \log(N) \) times, the end step having \( N \) different sequences with one sample in each. This hence has a complexity of \( O(N \log(N)) \).

In case when the size of input sequence is not an integer exponent of 2, the divide and conquer step can be still used, however it will not simplify to computing 1 multiplication in the last step.

Following is a plot comparing the complexity orders of both the algorithms:
3. 2D FFT

Just like 1D FFTs, 2D FFTs are also used in various applications especially in Image Signal Processing, for doing spectral domain analysis. In this section, we will assume \( X[N, N] \) be an input square matrix of size \( N \times N \). Its 2D FFT is given by the matrix \( X[k, l] \).

3.1. Direct Computation of 2D FFT

The FFT of a 2D matrix is given by:

\[
X[k, l] = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x[m, n] e^{-j \frac{2\pi (mk + nl)}{N}}
\]

Where \( X[k, l] \) refers to the \( k, l \) entry in the matrix.

Using the formula directly, the complexity of the algorithm is of the order \( O(N^3) \).

3.2. Computation of 2D FFT using Row Column Decomposition

The formula of the FFT can be written as:

\[
X[k, l] = \frac{1}{N} \sum_{m=0}^{N-1} e^{-j \frac{2\pi km}{N}} \sum_{n=0}^{N-1} x[m, n] e^{j \frac{2\pi nl}{N}}
\]

This is similar to taking 1D FFTs of the rows first and forming an intermediate matrix and then taking the 1D FFTs of its columns to get the ultimate 2D FFT. This algorithm hence does \( 2N \) 1D FFT computations and hence its complexity is of the order \( O(N^2 \log(N)) \).

In case the matrix is not a square matrix, and of the size \( M \times N \) instead, the row column decomposition can still be used by taking the FFTs of the \( M \) rows first and then computing the FFTs for the \( N \) columns.

Following is a plot comparing both the algorithms:

4. CONCLUSION

To conclude, we learnt that for an algorithm as important as FFT, there is immense research to improve its complexity and ease of implementation. One of the most commonly used algorithms is the Cooley Tukey algorithm, which improves the complexity to \( O(N \log(N)) \) for a \( N \) point DFT. Then we looked at the computation of 2D matrices, which uses Row Column Decomposition and the Cooley Tukey algorithm to reduce its complexity to \( O(N^2 \log(N)) \).

ACKNOWLEDGMENT

I acknowledge Dr. Nitin Chandrachoodan to have given us this paper as an assignment as it helped me learn in details about an algorithm which I have been using a lot as a student of Communication Engineering. I also acknowledge Dr. A. N. Rajagopalan for having taught me the techniques of FFT in Image Processing and inspiring me to take up this topic for the paper.

REFERENCES

[2] Dr. A. N. Rajagopalan’s notes for the course Image Signal Processing at IIT Madras