Performance Evaluation of Cooley Tukey FFT and Chirp z-transform Algorithm on Audio Signals

Nitika Khurana, Srishty Saha
Computer Science and Electrical Engineering
University of Maryland, Baltimore County
nkhur1, srishty1@umbc.edu
October 30, 2017

***Abstract*—In this paper, we are evaluating the performance of two Fast Fourier transform(FFT) algorithms: Cooley-Tukey FFT and Chirp Z transform FFT when applied to an audio signal. We first introduce the different FFT algorithms developed over time and then compute the computational complexities, such as running time and memory requirements, for Cooley Tukey's Radix 2 FFT and Bluestein Chirp Z transform when applied to an audio signal and then evaluate their performance in spectral analysis used in sound recognition apps such as Shazam.**

***Keywords*—Analysis of algorithms, time complexity, DFT, FFT, Cooley-Tukey FFT, Chirp Z transform, spectral analysis.**

I. Introduction

With the advancement of technology, it is impossible to believe that there was a time when there was no digital media. All this digital data is processed by computers by a variety of signal processing operations to be in a form that we can use. This digital data is available in the form of continuous time signals and hence, difficult to analyze. In 1800s, Jean-Baptiste Joseph Fourier [1] discovered that these can be converted into equivalent discrete frequency signals thereby simplifying the processing of these signals by researchers for spectrum analysis to detect specific tones, frequencies or other signatures. This frequency domain analysis is used in several voice recognition applications such as matching voice patterns, or displaying spectrum information on the face of an amateur radio [2], or detection of songs from a small chunk of music, etc. Shazam [3], is such an application that is used to identify a song from a small chunk of music. As of now, the application uses Cooley Tukey FFT algorithm to discretize the music sample and then create the fingerprint of the song. We, thus, are evaluating Cooley Tukey FFT with Chirp Z transform FFT to identify the best fit algorithm for this application and if a switch of algorithms would help in providing better recognition. We do this by implementing these two algorithms on an audio signal and then comparing the frequency distribution obtained with them. We, also, further compute the time complexity and space requirements for the algorithms under research.

A. Background

The Fourier Transform is a mathematical operation which expresses time domain function in frequency domain. The traditional methodology Discrete Fourier Transform (DFT), is essentially a Fourier Transform that takes a discrete-time input and transforms it to frequency. But at the same time, DFTs are computationally expensive in terms of both time and space. The Fast Fourier Transform is the computational optimization of DFT by providing an algorithm of running time $O\,(Nlog_2N)$ compared to O(N^2). The following is the mathematical equation of DFT of N sample signal:

$$X_k = \sum_{m=0}^{N/2-1} x_n e^{-j2\pi k \frac{n}{N}} \qquad k = 0, .., N-1. \tag{1}$$

Thus, FFT algorithms are (log_2N/N) times faster than DFT and hence, in the signal processing field, researchers most likely use FFT for various applications ranging from transient and frequency analysis of discrete audio signals to spectral analysis in chemistry. There are many variations of FFT which have been proposed. Each of the variations have been developed to improvise the performance of FFT for different applications[4]:

1) Cooley-Tukey FFT: : It is a recursive divide-and-conquer method proposed by Cooley and Tukey [5] back in 1965 of splitting up a DFT of size N into two smaller DFTs N1 and N2 such that, N=N1*N2. In radix-2 implementation of Cooley-Tukey FFT, the N-sized sample signal is divided into 2 equal halves i.e. N1 = N2 = N/2 summing over the even*(n=2m)* and odd*(n=2m+1)* indices.

$$X_k = \sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{2\pi i}{N}(2m)k} + \sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{2\pi i}{N}(2m+1)k} \tag{2}$$

Several algorithm were developed

2) Prime-Factor FFT: : The Prime-Factor FFT splits a size-N DFT into two separate DFTs of size N into two parts N1 and N2 as prime number. This algorithm is of order O(NlogN) and it is used only when there are dataset of signals are prime-sized.

3) Bruun's FFT: : It is a fast Fourier transform (FFT) algorithm based on an unusual recursive polynomial-factorization approach, proposed for powers of two size signal dataset.Its operations involve only real coefficients until the last computation stage. Though the algorithm is fast but it is not accurate in most of the applications.

4) Rader's FFT: :It tranforms DFT into a circular convolution. It only works for prime bases of the DFT, and typically it is more effective and simpler to use the Cooley-Tukey algorithm. The only real use for Raders FFT is for large prime bases. Computationally, this algorithm is order O(NlogN) (FFT opeartion) plus O(N) (circular convolution).

5) Bluestein Chirp-Z Transform: : Bluestein algorithm also transforms DFT as circular convolution but unlike Rader's Transform, it performs operations on arbitrary sized DFTs. The product in the DFt, nk, is substituted with:

$$n\,k = -\frac{(n-k)^2+n^2+k^2}{2} \tag{3}$$

and the transform changes to:

$$X_{\,k} = e^{-\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} \left(x_n e^{-\frac{\pi i}{N}n^2}\right) e^{\frac{\pi i}{N}(k-n)^2} \quad k = 0, ..., N-1. \tag{4}$$

The convolution sequences of length N are:

$$a_{\,n} = x_n e^{-\frac{\pi j}{N}n^2}, \qquad a_n = e^{-\frac{\pi j}{N}n^2}. \tag{5}$$

and the algorithm becomes:

$$X_{\,k} = b_k^* \sum_{n=0}^{N-1} a_n b_{k-n} \qquad\qquad k = 0, .., N-1. \tag{6}$$

The computational complexity is also of order $O\,(Nlog_2N)$ although in practical implementation, slower than Cooley Tukey. Bluestein algorithm can also be implemented to calculate Z-transform as follows:

$$X_{\,k} = \sum_{n=0}^{N-1} x_n z^{kn} \qquad\qquad k = 0, .., M-1. \tag{7}$$

for any number of N or M inputs. This algorithm is termed as Bluestein Chirp-Z transform or simply Chirp Z transform (CZT) because, for the Fourier-transform case ($|\,z\,| = 1$), the sequence $b_{\,n}$ becomes a sinusoid function called "chirping" in radar systems. This algorithm believes that the values of the z-transform on a circular or spiral contour can be expressed as a discrete convolution. It is used to provide a more evenly spaced interpolation of some portion of the spectrum to obtain better analysis of a range of frequencies, enhance arbitrary poles in transfer-function analyses, etc.

B. How Shazam works

Shazam Entertainment, Ltd. was started in 2000 to provide a service to bring people more closer to music by implementing an algorithm to help them recognize music in the environment using their mobile phones. The algorithm recognizes a song from a small chunk of music against a large music database and providing a high recognition rate[6]. The algorithm employs a process called fingerprinting for matching, It first converts the time-domain sample music into a spectrogram of equivalent frequency signals and then uses this to create a fingerprint of the song by identifying the highest magnitude of frequency in each range of frequencies in the signal. It then uses this fingerprint against a large database of songs and returns the identified song[7].

The remainder of this paper is organized as follows: Section 2 presents related work in the area of FFT algorithms, the Section 3 details the implementation of Cooley Tukey algorithm and the input signal collected and used for our analysis. We then describe the results obtained in Section 4. Section 5 presents the discussions explaining the significance of our analysis in the light of the application of these two FFT algorithms in digital signal processing. We have explained the set of open problems in FFT in Section 6 followed by our conclusion in Section 7.

II. Previous Work

The concept of FFT was introduced in early 1800s by Carl Gauss [8] and later developed by James Cooley and John Tukey in 1960s. They provided an algorithm that could perform signal discretization in NlogN time compared to DFT that consumed N^2 time thereby providing substantial saving. Bluestein introduced Chirp Z transform [9] that uses z-transform to calculate the frequency signal. Lawrence R. Rabiner et. al [10] presented various applications [11] of Chirp Z FFT. In K. Marcolini et. al [4], various Fast Fourier Transform algorithms have been described and running time of FFT algorithms on the audio signal data-set have been compared. But, it failed to measure and analyze the effect of applying FFT algorithms on audio-signal properties such as frequency resolutions. D.N. Rockmore in 2000 stated the role of FFT in the digital world [12].

III. Methodology

This section describes our framework for evaluating best-fit Fast Fourier Transform to be applied on audio-signal processing applications for better frequency resolution and low cost. For applications like song recognition [] or Guitar stroke recognition, it is necessary to understand the zooming effect of frequency. In order to analyze the zooming effect, there is need to apply appropriate Fast Fourier Transform on time-series audio signal. The following are the modules which describe our methodology to find best-fit FFT algorithm for audio processing applications:

A. Data Collection and Pre-Processing

We recorded an audio tape in wav format of length 5 secs. The data-set is of size 963 KB. The audio tape was discretized into floating point numbers through Numpy python library [] into a linear array. The size of the array was [1, 246578].

B. FFT Implementation

As a use-case, we implemented two most commonly used FFT algorithms, i.e, Cooley-Tukey FFT and Bluestein Chirp-Z Transform. We used Scipy[] and Numpy[] python libraries to implement FFT algorithms. Below are description of psedo code and underline mathematical aspect of Fast Fourier Transform algorithm:

1) Cooley-Tukey FFT: : This algorithm basically splits DFT of N size into two halves as N1 and N2 such that N=N1*N2. And in radix-2 implemenation, N1=N2= (N)/2 i.e FFT is applied separately on even and odd indices.
Mathematical Computation of Cooley-Tukey:

$$X_{\,k} = \sum_{m=0}^{N1-1} x_{2m} e^{-\frac{2\pi i}{N}(2m)k} + \sum_{m=0}^{N2-1} x_{2m+1} e^{-\frac{2\pi i}{N}(2m+1)k}$$

Algorithm 1 Cooley-Tukey

```
X0,...,N1 = fft2(x, N, s):
if N = 1 then then
    X0  x0
else
    X0,...,N/21 = fft2(x, N/2, 2s)
    X(N)/2,...,N1 = fft2(x+s, N/2, 2s)
    for k = 0 to (N)/21 do
        performfull DFT:
        t = Xk
        Xk = t+ exp(2πik/N) * Xk + N/2
return fft transformed signal
```

The running time complexity of radix-2 implementation of Cooley-Tukey Transform is O(NlogN).

2) Bluestein Chirp-Z Transform: Here, z-Transform is applied along with DFT in the circular convolution form on arbitrary sized signals. Z-Transform results in better frequency resolution in most of the applications of audio-signal processing. Mathematical Computation of Chirp-Z Transform:

$X_k = e^{-\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} \left(x_n e^{-\frac{\pi i}{N}n^2} \right) e^{\frac{\pi i}{N}(k-n)^2}$

where k = 0.....N-1 and N is size of signal.

Algorithm 2 Bluestein Chirp-Z Transform

```
n = len(vector)
m = 1 number of points to be considered
while m < n:
m=m*2
[x,y]= z-Transform of signal
b = conjugate of matrix [x,y]
for i to m do
    if (i < n or m - i < n) then  0
        c = convolve(a, b, False)
        for i = 1 to n do
            c[i] *= exptable[i]
            return chirp-z transformed signal
```

IV. Results

We have implemented Radix-2 implementation of Cooley-Tukey Fast Fourier Transform and analyzed the spectral analysis of signal after performing transform on the signal. Figure 1 shows our initial spectral analysis. As in Figure 1, it is clearly visible the range of frequencies which need to be focused for audio-signal processing. We also recovered the original signal by performing inverse Fast Fourier Transform and analyzed the quality of recovered audio through Signal to Noise Ratio (PSNR). The PSNR values obtained was 5.96 + 1.06j. Higher PSNR value signifies good quality of retrieved audio. As part of our ongoing work, we are implementing Bluestein Chirp-Z Transform and we will compare the quality of recovered signals for both of FFT variations through various evaluation metrics.

V. Discussion

In this paper, we aim to analyze the best-fit Fast Fourier Transform algorithm to be applied on various audio signal processing applications. We chose two major FFT algorithms i.e, Cooley-Tukey and Bluestein Chirp-Z Transform for our analysis on song recognition application -"Shazam". Till now, we have implemented and performed spectral analysis of Radix-2 implementation of Cooley-Tukey transform. As a part of our ongoing work, we are implementing Bluestein Chirp-Z Transform and we will analyze best fit FFT algorithm for "Shazam"- song recognition applications based on various evaluation metrics.

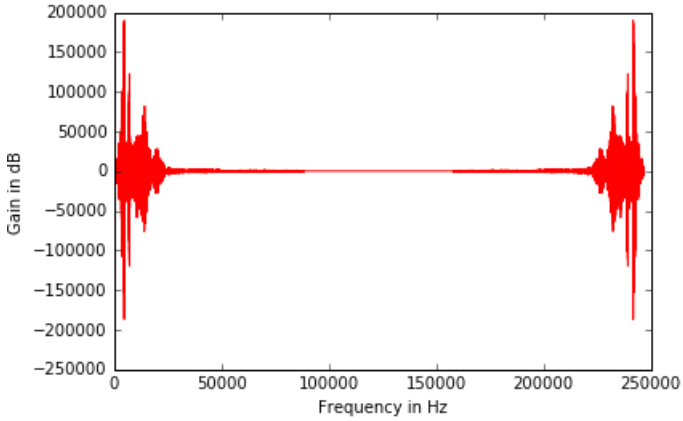

Fig. 1. The Spectral Analysis of Audio Signal after Cooley-Tukey Transform

VI. Open Problems

While implementing the Chirp-Z FFT algorithm, we faced challenges while executing the algorithm. The running time of Chirp-Z Transform was unexpectedly higher. So, to overcome thi challenge we plan to use processors (CPUs) of better configuration. Though upper bound of theoretical running time of both the algorithms i.e Cooley-Tukey and Chirp-Z Transform are same but still Chirp-Z Transform's running time was unexpectedly higher. In future, we aim to tackle this challenge. Secondly, Fast Fourier Transform has no definite lower bound complexity and thus it is of one of open problems in the area of digital signal processing. We aim to address this challenge in our research as well.

VII. Conclusion

Acknowledgment

The authors would like to thank Dr. Sherman for his guidance and sharing his knowledge on algorithms and their analysis and also for providing insightful comments in the proposal document. Also, we thank our reviewers Shylla and Rajat for their initial feedback.

References

[1] http://lpsa.swarthmore.edu/Fourier/Series/FourierBio.html

[2] http://www.arrl.org/voice-modes

[3] Shazam: https://www.shazam.com/

[4] K. Marcolini et. al , Various Fast Fourier Transform algorithms

[5] Cooley et. al "The Fast Fourier Transform and Its Applications," IEEE Transaction Education

[6] ALC Wang et. al "An Industrial-Strength Audio Search Algorithm"

[7] https://www.toptal.com/algorithms/shazam-it-music-processing-fingerprinting-and-recognition

[8] M. Heideman, D. Johnson, and C. S. Burrus, "Gauss and the history of the FFT," IEEE Signal Processing Magazine

[9] Bluestein et. al A Linear Filtering Approach to the Computation of Discrete Fourier Transform

[10] Rabiner et. al "The Chirp Z transform, Rabiner, Schafer, IEEE"

[11] Lawrence R. Rabiner et. al "The Chirp z-Transform Algorithm and Its Application", Rade, 6 May 1969, Bell Labs Technical Journal

[12] The FFT: an algorithm the whole family can use,D.N. Rockmore, IEEE Journal, Computing in Science Engineering (Volume: 2, Issue: 1, Jan/Feb 2000)

ADDITIONAL NOTES

We planned on following the below mentioned schedule:

TABLE I. PROPOSED SCHEDULE

Oct-10 Oct 20	Oct 21-Nov 15	Nov15- Nov 25	Nov 25 - Dec 5
Collect audio signals	Implement algos & analyze	Find best fit FFT	Report & slides

We have completed the work that we planned to complete by the current date; we have collected the audio signal that we would use for our analysis, we have implemented the Cooley-Tukey FFT algorithm and calculated its performance using a frequency plot displaying the spectral analysis and computed the SNR for the audio signal. We plan to implement Chirp-Z transform as per the schedule and continue with the evaluation. We are facing problems while implementing the Chirp-Z transform FFT algorithm. The conversion of the pseudocode to a running code is giving us some errors and consuming a lot of time and we therefore, we will execute the algorithm on another system with better configuration and check if the problem is arising because of the processor.
We have made some changes after our proposal. We have provided a focused research question as to how the performance evaluation of Cooley Tukey FFT and Chirp Z transform FFT on a sample audio signal help us to identify a better algorithm for spectrum analysis. We are trying to identify if changing the FFT algorithm in the process of song recognition as employed by Shazam would enhance the recognition process thereby decreasing the false positives. We are suggesting that if an algorithm produces better SNR ratio and frequency discretization, then it might provide increased efficiency while fingerprinting the music for song recognition.
Since, we are running as per our proposed schedule, there are not many changes to it:

TABLE II. REVISED SCHEDULE

Oct 30-Nov 15	Nov 15-Nov 25	Nov 25- Dec 5
Implement Chirp Z FFT	Find best fit FFT	Report and Slides