PARALLELIZATION OF FAST FOURIER TRANSFORM ALGORITHMS

MUSAU MOSES KIMUYU
DFT is very useful in the aid of analysis of many problems in engineering such as signal processing, speech and image processing.

The number of computations done in the computation of DFT using a direct approach is given by $O(n^2)$ whereas the FFT basically computes the DFT with a reduced number of computations that is $O(n \log n)$. 
PROBLEM DEFINITION

Historically, most computers have had one processor, with a single processing unit or core.

Unfortunately, a single processor only allows for genuine sequential programming.

Sequential programming is constrained in terms of speed.
OBJECTIVES

• Implementation of FFT algorithms in C++
• Use of multithreading to achieve concurrency for the algorithms
• Analysis of the sequential and multithreaded implementations of the algorithms
• Comparison of the sequential and parallelized algorithms
PROJECT SCOPE

• Implementation of the naïve approach to the DFT
• The radix-2 Decimation in Time (DIT) and Decimation in Frequency (DIF) as well as the Goertzel FFT algorithms have been considered
The frequency domain is basically a more suitable way of viewing the signal information that may be difficult to see in the time domain.

A periodic signal can be synthesized using a linear combination of sine and cosine waves using the Fourier series concept.

\[ x(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos \left( \frac{2\pi nt}{T} \right) + \sum_{n=1}^{\infty} b_n \sin \left( \frac{2\pi nt}{T} \right) \]
When the period of a signal tends to infinity, the Fourier series (which is essentially a line spectrum) tends to a continuous spectrum that is referred to as the Fourier transform (FT).

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} \, dt$$
Digital Signal Processing (DSP) works with discrete signals hence the FT input and output must be discrete signals.

When the input to the FT is sampled, the Discrete-Time Fourier Transform (DTFT) is obtained.

\[ x(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x(n)e^{-jn\omega} \]

However, the DTFT is continuous and periodic.
Since we are interested in discrete signals in DSP, the DTFT is sampled over one period to obtain the DFT.

\[ X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi \frac{kn}{N}} \]

where \( \Delta W = \frac{2\pi}{N} \)
The equations for both the DFT and its inverse are usually simplified by the definition of twiddle factor

\[ W_N = e^{j\frac{2\pi}{N}} \]

\[ x(n) = \sum_{k=0}^{N-1} x(n)W^{kn} \]
The evaluation of the DFT equation directly is hectic especially for large number of input samples. However, The DFT equation has an arithmetic form that renders it possible to improve on its efficiency in terms of time taken. FFT algorithms take advantage of the periodicity, symmetries, and orthogonality in order to reduce redundancy and unnecessary operations in the computation of DFT directly.
DECIMATION IN TIME FFT

The input sequence can be broken down into odd and even-indexed parts. Taking these two sequences into consideration, the initial sum can be split to calculate the N/2 point DFTs of the even and odd sequences i.e the butterfly equations

\[ X(k) = X_e(k) + W_N^k X_o(k) \]

\[ X(k + N/2) = X_e(k) - W_N^k X_o(k) \]
DECIMATION IN FREQUENCY FFT

In this FFT, the DFT sum is broken into the initial \( \frac{N}{2} \)-points and last \( \frac{N}{2} \)-points. This corresponds to the odd and even parts of \( X(k) \) that is in the frequency domain.

\[
X(2k) = \sum_{n=0}^{N/2-1} [x_0(n) + x_1(n)] W_N^{nk/2}
\]

\[
X(2k + 1) = \sum_{n=0}^{N/2-1} [x_0(n) - x_1(n)] W_N^n W_N^{nk/2}
\]
The Goertzel FFT is efficient when it is required to compute a small number of selected frequency components.

Its main calculation takes the form of a digital filter since it expresses the DFT sum as a convolution sum using the identity

$$W_N^{-kn} = 1$$
The direct DFT, radix-2 DIT FFT, radix-2 DIF FFT and the Goertzel algorithm were implemented as sequentially using C++.

The programs were parallelized using the multithreading techniques.
## RESULTS

### SEQUENTIAL IMPLEMENTATION

<table>
<thead>
<tr>
<th>Input/Output sequence</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFT</td>
<td>178ms</td>
<td>231ms</td>
</tr>
<tr>
<td>DIT</td>
<td>159ms</td>
<td>185ms</td>
</tr>
<tr>
<td>DIF</td>
<td>161ms</td>
<td>202ms</td>
</tr>
</tbody>
</table>
## RESULTS

### PARALLEL IMPLEMENTATION

<table>
<thead>
<tr>
<th>Input/Output sequence</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFT</td>
<td>136ms</td>
<td>164ms</td>
</tr>
<tr>
<td>DIT</td>
<td>155ms</td>
<td>170ms</td>
</tr>
<tr>
<td>DIF</td>
<td>150ms</td>
<td>166ms</td>
</tr>
</tbody>
</table>
## RESULTS

<table>
<thead>
<tr>
<th>GOERTZEL ALGORITHM</th>
<th>Sequential Implementation</th>
<th>Parallel Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>238ms</td>
<td>202ms</td>
</tr>
</tbody>
</table>
CONCLUSION

- FFT algorithms speed up the computation of the DFT.
- Parallelization speeds up the execution as compared to sequential programs.