COP5520

PROJECT PROGRESS REPORT

FAROOQ MAHMUD AND DAE HYUN SUNG

1. PROGRESS SUMMARY

We chose to do the project on Fast Fourier Transform (FFT). We were able to successfully create a program that can take samplings of multiple sine waves for the period of 2π, and sum them to create a complex waveform upon which to perform FFT. We successfully recreated the Cooley-Tukey algorithm with big $O(nlogn)$ using complex math operations to perform calculations at each sampling. We verified the calculation results of the Cooley-Tukey algorithm to be correct which sets us up for the next portion of project which will be to parallelize the algorithm.

2. COMPLEX NUMBER MATH BACKGROUND

The main operations performed in the program are trigonometric sin(Θ), cos(Θ), complex summation, and complex multiplication. Sine wave defined using complex numbers equals

$$r * cos(\Theta) + ir * sin(\Theta)$$

where Θ is phase and r is amplitude. The summation operation of complex umbers is simple in that one only must add the real parts together and the imaginary parts together. The multiplication of complex numbers is defined as below.

$$(a + bi) * (c + di) = (ac - bd) + (ad + bc)i$$

3. ALGORITHM & OPTIMIZATION

There exist multiple variations of the Cooley-Tukey algorithm implementation for FFT, all fundamentally having big $O(nlogn)$. We therefore stuck with the most prevalent algorithm that uses recursion to separate the samples to even and odd, dividing the data set in half at each step until data size equals one. This accounts for the $logn$ portion of $O(nlogn)$. Once division is completed, the children are calculated and assigned to the parent in a n size for loop which accounts for the prefixing n in $O(nlogn)$. The algorithm is straight-forward and well-defined, optimizations mainly consisted of choosing the appropriate data types, allocating the right amount of memory, performing the correct complex math operations, and freeing all allocated memory before closing.