COP5520
PROJECT PROPOSAL

FAROOQ MAHMUD AND DAE HYUN SUNG

1. Introduction

We propose to parallelize the Fast Fourier Transform (FFT) algorithm used for solving the Discrete Fourier Transform. A preliminiary search of existing FFT non-paralllel implementations in C have not been fruitful although there are several C++ and Python implementations to be found. If we cannot find a suitable C implementation we will implement our own non-parallel version based on available pseudocode [2]. We will then implement parallel versions using UNIX pThreads, OpenMP, and MPI.

2. Performance Evaluation

The input to the FFT is a vector representing the samples of a function's value between a given time interval. Therefore we will assess the performance with varying vector lengths. Ideally the FFT algorithm has a runtime of $\Theta\ n\ log(n)$ [2]. We'd like to verify that runtime in the non-parallel and parallel implementations. We will also assess the runtime improvement of the parallel algorithm based on the availability of additional threads (in the case of pThreads and OpenMP) and processor cores (in the case of MPI).

3. Potential Challenges

The non-parallel implementation of the FFT is a typical divide-and-conquer algorithm [1]. Therefore the algorithm is well-suited to parallelization. We anticipate the challenge will be to understand the mathematics behind the algorithm so that we can verify correctness. We will write tests in order to verify the algorithm is correct as we write our code.

References

[1] Guy E. Blelloch and Bruce M. Maggs. *Parallel Algorithms*, page 25. Chapman and Hall/CRC, 2 edition, 2010.

[2] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Polynomials and the FFT*, page 906–911. Mit Press, 2009.