Justin Bahr
NUID: 001592707

High Performance Computing
Final Project Proposal

Rationale:
I played trumpet and wrote a lot of music in high school, so I have always been interested in audio processing. I am also currently enrolled in Digital Signal Processing, where we have covered discrete time convolutions and digital FIR filters in depth using MATLAB. I would like to connect these by completing a project evaluating an optimized audio processing workload in C++. My goal is to discover which optimization techniques (likely a combination of AVX and multi-threading) can produce the fastest execution time for a high-order low-pass FIR filter.

In addition to audio processing, I would also like to accelerate Sobel Filter edge detection using one or more GPUs. I have not worked with GPUs or image processing before, but I think this would be a valuable learning experience to become familiar with CUDA programing, GPU architecture, and edge detection algorithms.

Workloads:
I will evaluate audio signal processing through convolution using a high-order low-pass FIR filter as well as Sobel Filter edge detection.

Input Datasets:
For the audio signal processing workload, I plan to use audio samples from the Star Wars movies as datasets since most of the soundtrack is fairly well known, the songs have a wide range of frequencies, and they are long enough to obtain noticeable speedups in parallel. I found MP3 files available here: https://archive.org/details/13BinarySunsetAlternate
which I can convert into stereo (2-channel) WAV files for processing. For the Sobel Filter edge detection workload, I plan to use PPM files since they are easy to parse in C++. From https://filesamples.com/formats/ppm I found an image with noticeable edges available in different sizes ranging from 640x426 to 5184x3456 pixels, which will be good to evaluate weak scaling. This image is shown below:

Platforms:
I will evaluate the audio processing workload on CPU nodes available on the Explorer cluster, and possibly the Vector COE server. I will evaluate the Sobel Filter edge detection workload on one or more GPUs available on the Explorer cluster. I will write my programs using C++.

Experiments:
For the audio processing workload, I will verify my solutions by outputting and listening to a filtered WAV file to ensure high frequencies have been removed. To design the initial filter and generate FIR coefficients, I will use MATLAB. I can also check my results by testing the same input data in an equivalent MATLAB process. Additionally, I will time my results for different numbers of threads, and with/without AVX support. For the Sobel Filter edge detection algorithm, I will visually compare my results to the original image to ensure edge detection was successful. Finally, I will time the GPU acceleration to evaluate the speedup obtained. I will use the high precision clock available with chrono to time results.

Results and Grade Expectations:
A =

- Audio processing workload evaluated with at least 3 WAV input/outputs on the Explorer cluster, including a comparison a serial implementation as well with OpenMP, AVX support, and a combination of the two
- Sobel Filter edge detection workload evaluated with at least different 3 image sizes on at least one GPU node on the Explorer cluster
- All results reported and analyzed thoroughly in the project writeup, including a strong scalability report for the audio processing workload and a weak scaling scalability report for the Sobel Filter edge detection workload

A- =

- Audio processing workload evaluated with at least 3 array input/outputs (WAV files may be parsed and written outside my C++ program using Python or MATLAB) on the Explorer cluster, including a comparison a serial implementation as well with OpenMP, AVX support, and a combination of the two
- Sobel Filter edge detection workload evaluated with at least different 3 image sizes on one GPU node on the Explorer cluster
- All results reported and analyzed thoroughly in the project writeup, including a strong scalability report for the audio processing workload and a weak scaling scalability report for the Sobel Filter edge detection workload

B+ =

- Audio processing workload evaluated with at least 3 WAV input/outputs on both the Explorer cluster, including a comparison a serial implementation as well with OpenMP, AVX support, and a combination of the two OR Sobel Filter edge detection workload evaluated with at least different 3 image sizes on two GPU nodes on the Explorer cluster
- All results reported and analyzed thoroughly in the project writeup, including either a strong scalability report for the audio processing workload or a weak scaling scalability report for the Sobel Filter edge detection workload

B =

- Audio processing workload evaluated with at least 3 array input/outputs (WAV files may be parsed and written outside my C++ program using Python or MATLAB) on the Explorer cluster, including a comparison a serial implementation as well with OpenMP, AVX support, and a combination of the two OR Sobel Filter edge detection workload evaluated with at least different 3 image sizes on one GPU node on the Explorer cluster
- All results reported and analyzed thoroughly in the project writeup, including either a strong scalability report for the audio processing workload or a weak scaling scalability report for the Sobel Filter edge detection workload

B- =

- Audio processing workload evaluated with at least 3 array input/outputs (WAV files may be parsed and written outside my C++ program using Python or MATLAB) on the Explorer cluster, including a comparison a serial implementation vs a multithread implementation OR Sobel Filter edge detection workload evaluated with at least different 3 image sizes on one CPU node on the Explorer cluster, and another CPU node on the COE Vector system.
- All results reported and analyzed thoroughly in the project writeup, including either a strong scalability report for the audio processing workload or a weak scaling scalability report for the Sobel Filter edge detection workload

C+ =

- Audio processing workload evaluated with at least 3 array input/outputs (WAV files may be parsed and written outside my C++ program using Python or MATLAB) on the Explorer cluster, including a comparison a serial implementation vs a multithread implementation OR Sobel Filter edge detection workload evaluated with at least different 3 image sizes on one CPU node on the Explorer cluster and one CPU node on the COE Vector system, including a comparison a serial implementation vs a multithread
- All results reported in the project writeup and some analysis included

C =

- Audio processing workload evaluated with at least 3 array input/outputs (WAV files may be parsed and written outside my C++ program using Python or MATLAB) on the Explorer cluster using a multithreaded implementation OR Sobel Filter edge detection workload evaluated with at least different 3 image sizes on one CPU node on the Explorer cluster and one CPU node on the COE Vector system using a multithreaded implementation.
- All results reported in the project writeup and some analysis included

C- =

- Audio processing workload evaluated with one array input/outputs (WAV files may be parsed and written outside my C++ program using Python or MATLAB) on the Explorer cluster using a multithreaded implementation OR Sobel Filter edge detection workload evaluated with at one image size on one CPU node on the Explorer cluster and one CPU node on the COE Vector system using a multithreaded implementation.
- All results reported in the project writeup and some analysis included

Below C- =

- Exploration of either workload, without a fully successful implementation
- Writeup includes details about progress made, challenges faced, and strategies to achieve better results for future work

Extra Credit Opportunities:

- Audio processing workload evaluated on the COE Vector system
- Audio processing workload evaluated using Pthreads
- Multiple FIR filters tested for the audio processing workload
- Sobel Filter edge detection evaluated on one or more CPU nodes to compare to GPU performance
- An additional image processing workload evaluated