CS 575 Parallel Programming

Project #7

Fourier Analysis using MPI

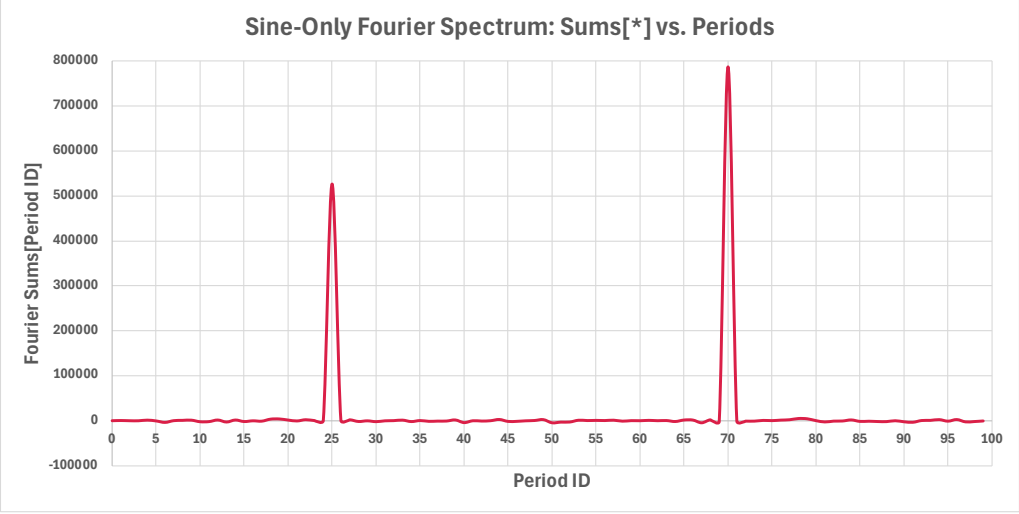
Name: Rachel Xing (#934229189)

Email Address: xingru@oregonstate.edu

Description: This project uses MPI to implement the Fourier analysis to detect if there are regular patterns given a thought-to-be-random signal data with a size of 1,048,576. In this program, the boss processor loads all the signal data into memory and scatters it evenly to all other processors. Each processor, including the boss, performs a Fourier transform by projecting its data portion onto the sine waves of 100 periods. Then, the non-boss processors send their transform results back to the boss, and the boss processor combines all results to produce a sine-only Fourier spectrum across 100 periods based on the given signal data. The program is written in C++ and is run in five trials with 1, 2, 4, 6, and 8 CPUs on the OSU HPC cluster with the node list of cn-b02, cn-b03, cn-b04, cn-b05, cn-b06, cn-b10, cn-b11, cn-b12 from 25 Dell Intel Sandy-bridge/Ivy-bridge compute nodes.

Commentary:

1. **Show the Sums[*] vs. period scatterplot.**

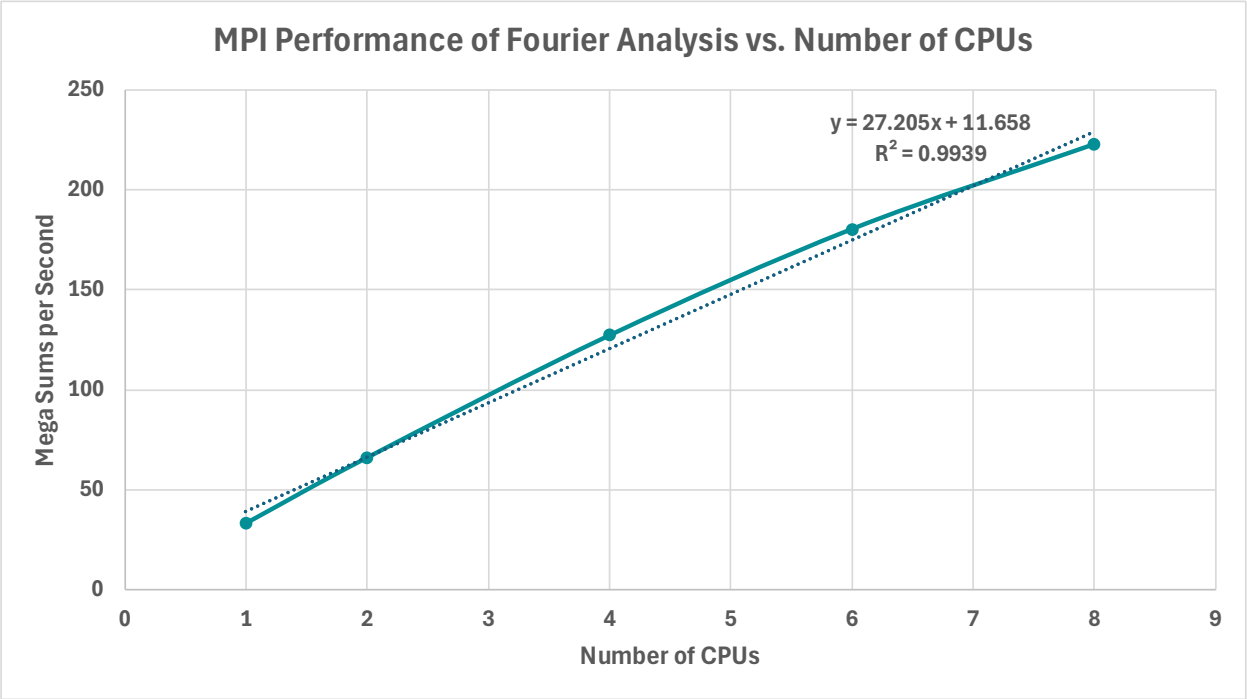


2. **State what the two secret sine-wave periods are.**

   The two sine-wave periods are at period **index 25** with a Fourier sum of 526793.56, and at period **index 70** with a Fourier sum of 787514.81 (index starts from 0 to 99 for 100 periods).

3. **Show your graph of Performance vs. Number of Processors used.**

Signal Data Size	1048576	1048576	1048576	1048576	1048576
CPU #	1	2	4	6	8
Performance (MegaSumsPerSec)	33.41	66.17	127.2	180.23	222.59



4. **What patterns are you seeing in the performance graph?**

   We can observe that as the number of CPUs increases from 1 to 8, there is a **highly linear** growth from 33.41 to 222.59 megasums per second in performance. This growth trend has a linear correlation coefficient of 0.99, and the approximated linear equation is $MPI\ Performance = 27.205 * (Number\ of\ CPUs) + 11.658$.

5. **Why do you think the performances work this way?**

   Such linear growth in performance is because MPI follows a Single-Program Multiple-Data (SPMD) model. In this program, the signal data is evenly distributed among multiple CPUs. Each CPU performs the same Fourier computation independently in parallel. Since overhead only occurs during scattering and gathering the data (only twice for the communication between the boss CPU and each other CPU), it yields significantly less overhead compared to OpenMP, which usually involves the scheduling and management of thousands of threads.

   Thus, MPI can achieve a near-linear speedup (a factor of ~N with N CPUs) compared to the performance running with one CPU. However, such linear performance growth can degrade if the task requires more frequent inter-CPU communication or if the dataset is too large to exceed the individual CPU's capacity.