Homework #6 Parallel Matrix-Matrix Multiply Part 1

Overview

You are going to write and evaluate Cannon's algorithm for parallel matrix-matrix multiplication. For this assignment you must use a Cartesian topology.

Software Development

Write a program called `multiply` that multiplies two matrices. Your program should accept five inputs:

- First matrix input filename: specified as `-m1 <filename>`
- Second matrix input filename: specified as `-m2 <filename>`
- Output filename (optional): if specified, use `-o <filename>`
- Verbose (optional, see discussion below): if used, specify `--verbose`

Examples:

```
mpirun -np 4 ./multiply -m1 m1000.hdf5 -m2 m1000.hdf5
mpirun -np 8 ./multiply -m1 m100.hdf5 -m2 m100.
mpirun -np 4 ./multiply -m1 ma10.hdf5 -m2 mb10.hdf5 -o out
./multiply -m1 m1000.hdf5 -m2 m1000.hdf5 --verbose
```

You may assume your input matrices will be square and that the size will always be evenly divisible by the square-root of the number of processors. Your code must also run in serial.

You should use the example HDF5 code provided on the d2L site to assist you with reading and writing HDF5 files. Use the partial read from the previous homework where one rank reads the matrix information from the specified files and distribute the partial matrices to the other ranks. Once a rank has received its information, the process continues until all ranks have their portion of the matrix.

You can use the code provided from the last homework to generate matrix files in HDF5. *Modify this code to make your performance better.* You should chunk the files for optimal performance based on a range of processors.

Only output your data if the -o <filename> flag is specified. Output is an HDF5 2D matrix (see helper code).

The verbose mode should collect timing information. Please output the number of processors used, the problem size, the total time, and the parallel processing time. Remember, you may want to warm-up the `MPI_Wtime`.

Analysis

You will be running a series of experiments. Consider writing bash scripts (or your favorite scripting language) for managing the execution of the different trials.

I. Chunking

You should experiment with different chunking sizes to determine which one is best for reading both files. Use this for your analysis.

- What size and shape of chunking works best for a given problem size?
- Why?

II. Serial analysis and Amdahl's Law (again)

Profile your serial implementation and estimate the percentage of time your program spends doing serial work (everything but the multiplication). Do this for problems of size 1024, 2048, and 4096. Based on this information and **Amdahl's Law**, how do you anticipate your algorithm will scale as you increase the number of processors for a give problem size? Remember, Amdahl's law doesn't require you to account for communication or additional MPI overhead. Show your data and explain your calculations.

III. Parallel analysis: variable number of processors

Run your experiments using 1, 2, 4, 8, 16, and 64 processors using problem sizes 1024, 2048, 4096, and 8192. Do this experiment for only a single IO implementation (your choice). Repeat these experiments 10 times and take the average.

1. Plot the **speedup** for each problem size as a function of the number of processors and algorithm type.
2. How does the speedup change with problem size?
3. Do you notice **Amdahl's effect** for this problem? Why or why not?

Assignment Submission

Your lab write up, answers to questions, plots and all other bits should be presented using formal scientific write-up guidelines including, as a minimum, the following sections; Introduction, Methods, Results, and Conclusions. If you use outside

information make sure you properly reference them. Make sure all graphs are properly scaled, labeled, titled and referenced.

Create a file named username-hw6.tar.gz that contains a softcopy of your report, your source code and Makefiles. Please do not include object files, executables, or any *matrix files*. Submit the tar file to the d2L site by the due date. Bring a printed copy of your report to lab. The assignment is due *before lab* on Friday, March 9th.