CMOR 421/521 Assignment 3: Using MPI to implement Cannon's algorithm and SUMMA algorithm

Yuhao Liu

April 22, 2025

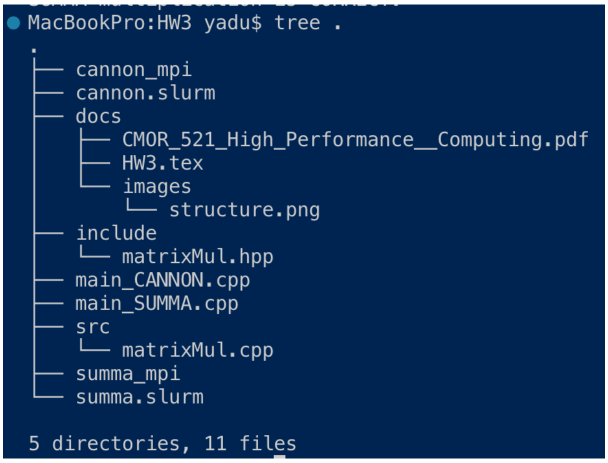
Contents

1 Directory Structure 2

2 How to Build and Run the Code (In NOTXs) 3

1 Directory Structure

Below is my file organization for this assignment. My final zip file follows this structure (`docs/` for LaTeX, `src/` for source files, and `include/` for header files):



```
MacBookPro:HW3 yadu$ tree .
.
├── cannon_mpi
├── cannon.slurm
├── docs
│   ├── CMOR_521_High_Performance__Computing.pdf
│   ├── HW3.tex
│   └── images
│       └── structure.png
├── include
│   └── matrixMul.hpp
├── main_CANNON.cpp
├── main_SUMMA.cpp
├── src
│   └── matrixMul.cpp
├── summa_mpi
└── summa.slurm

5 directories, 11 files
```

Figure 1: structure

- The `include` folder and `src` folder have `matrixMul.hpp` and `matrixMul.cpp`. In `matrixMul.cpp`, there are 2 help function
  - `void testMul(const int N, double* serialC, double* mpiC);`
  - `void serialMatMult(const int N, double* C, const double* A, const double* B);`
- The `main_CANNON.cpp` and `main_SUMMA.cpp` contain the Cannon's algorithm and SUMMA algorithm respectively.
- The `cannon_mpi` and `summa_mpi` are executable files. Both of them already complied by `-O3` optimization flag.
- The `cannon.slurm` and `summa.slurm` are sbatch scripts.

2 How to Build and Run the Code (In NOTXs)

- **Build and run**
  - For each algorithm you can using

    ```
    sbatch summa.slurm
    sbatch cannon.slurm
    ```

    to run these 2 different algorithm with $2 \times 2, 3 \times 3, 4 \times 4$ grids with $N = 512, 768, 1024$ and $k = 64$. All configuration parameters are defined in the `*.slurm` job submission script. The following example illustrates the contents of `summa.slurm` used for executing the SUMMA algorithm:

    ```
    #!/bin/bash
    #SBATCH --job-name=SUMMA
    #SBATCH --account=commons
    #SBATCH --partition=commons
    #SBATCH --ntasks=16             # Total number of MPI processes
    #SBATCH --cpus-per-task=1       # One CPU core per MPI process
    #SBATCH --mem-per-cpu=16000     # Memory per core (in MB)
    #SBATCH --time=00:30:00         # Walltime limit
    #SBATCH --mail-user=yl336@rice.edu
    #SBATCH --mail-type=END,FAIL

    echo "Job running on nodes: $SLURM_NODELIST"
    echo "=================================================="

    # Change to working directory
    cd /home/yl336/cmor521/CMOR-521-High-Performance-Computing/Assignments/H

    # Load compiler and MPI module
    module load GCC/13.2.0
    module load OpenMPI/4.1.6

    # Compile the SUMMA program
    mpic++ -O3 -o summa_mpi main_SUMMA.cpp ./src/matrixMul.cpp

    # Execute the program on different processor grids
    echo "=== Running SUMMA on 4 procs (2×2 grid) ==="
    mpirun -n 4  ./summa_mpi 512 64

    echo "=== Running SUMMA on 9 procs (3×3 grid) ==="
    mpirun -n 9  ./summa_mpi 768 64

    echo "=== Running SUMMA on 16 procs (4×4 grid) ==="
    mpirun -n 16 ./summa_mpi 1024 64

    echo "All runs complete."
    ```

- **Results:**
  - **SUMMA Algorithm**

    - **4 processes (2×2 grid)**
      - Matrix size: 512
      - SUMMA time: `0.0419424` seconds
      - Serial time: `0.337914` seconds
      - Total error: `1.55078e-08`
      - Status: `CORRECT`
    - **9 processes (3×3 grid)**
      - Matrix size: 768
      - SUMMA time: `0.084537` seconds
      - Serial time: `0.861577` seconds
      - Total error: `7.07854e-08`
      - Status: `CORRECT`
    - **16 processes (4×4 grid)**
      - Matrix size: 1024
      - SUMMA time: `0.0979639` seconds
      - Serial time: `7.28481` seconds
      - Total error: `1.72492e-07`
      - Status: `CORRECT`
  - **Cannon’s Algorithm**
    - **4 processes (2×2 grid)**
      - Matrix size: 512
      - Cannon time: `0.0235929` seconds
      - Serial time: `0.62207` seconds
      - Total error: `1.13176e-08`
      - Status: `CORRECT`
    - **9 processes (3×3 grid)**
      - Matrix size: 768
      - Cannon time: `0.0345131` seconds
      - Serial time: `1.03666` seconds
      - Total error: `6.90086e-08`
      - Status: `CORRECT`
    - **16 processes (4×4 grid)**
      - Matrix size: 1024
      - Cannon time: `0.039058` seconds
      - Serial time: `5.11798` seconds
      - Total error: `1.63165e-07`
      - Status: `CORRECT`