

Homework #3

cpe 512

Kyle Ray

October 10, 2017

Contents

[Part 1 Serial Test 1](#_Toc495349465)

[Code 1](#_Toc495349466)

[Output 1](#_Toc495349467)

[Part 2 Parallel Test 2](#_Toc495349468)

[Code 2](#_Toc495349469)

[Output NP = 2 2](#_Toc495349470)

[Output NP = 4 2](#_Toc495349471)

[Output NP = 8 3](#_Toc495349472)

[Part 3 Timing Analysis 3](#_Toc495349473)

[Serial 3](#_Toc495349474)

[Parallel 3](#_Toc495349475)

[Appendix 3](#_Toc495349476)

# Part 1 Serial Test

## Code

* See Appendix A for the source code

## Output

uahcls01@dmcvlogin1:Hw3> ./mm\_mult\_serial 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=1e-06 seconds

# Part 2 Parallel Test

## Code

* See Appendix B for the source code

## Output NP = 1

uahcls01@dmcvlogin2:Hw3> mpirun -np 1 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=1.7e-05 seconds

## Output NP = 2

uahcls01@dmcvlogin2:Hw3> mpirun -np 2 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=9.9e-05 seconds

## Output NP = 3

uahcls01@dmcvlogin2:Hw3> mpirun -np 3 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=0.000154 seconds

## Output NP = 4

uahcls01@dmcvlogin2:Hw3> mpirun -np 4 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=0.00015 seconds

## Output NP = 5

uahcls01@dmcvlogin2:Hw3> mpirun -np 5 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=0.004219 seconds

## Output NP = 6

uahcls01@dmcvlogin2:Hw3> mpirun -np 6 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=0.000299 seconds

## Output NP = 7

uahcls01@dmcvlogin2:Hw3> mpirun -np 7 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=0.004195 seconds

## Output NP = 8

uahcls01@dmcvlogin2:Hw3> mpirun -np 8 ./mm\_mult\_mpi 4 6 3

A matrix =

48.3962 65.3245 15.0385 72.383 25.8898 46.0265

15.4881 50.6507 6.74602 71.0055 12.2209 77.5441

61.5452 31.5127 46.8515 89.4849 70.0342 57.3195

75.4144 83.5553 91.7832 7.74197 40.0845 11.1709

B matrix =

26.5416 83.9488 86.5328

51.0444 65.3442 85.2683

76.9977 49.0015 46.6826

12.2581 99.9706 40.1026

58.6347 47.2069 4.06732

37.0919 22.9082 82.6622

C matrix =

9889.42 18581 17272.7

7979.16 14392.3 15281.2

14179 23086.6 18811.4

16193.3 19210.5 19332

time=0.000227 seconds

**(TODO: Possibly take this out before turn in)** I ran the original program with np = 2, 4, and 8 first and went back to run the rest of the cases after reading the homework description more closely. My connection to the dmc seemed slow on the night I ran np = 1, 3, 5, 6, and 7 which I think was the cause of the substantial increase in completion time.

# Part 3 Timing Analysis

## Serial

Matrix Size = 256, time = 0.107185 seconds

Matrix Size = 512, time = 0.862933 seconds

Matrix Size = 768, time = 4.647209 seconds

Matrix Size = 1024, time = 26.243087 seconds

Matrix Size = 1280, time = 46.751129 seconds

Matrix Size = 1536, time = 106.47022 seconds

Matrix Size = 1792, time = 180.9898 seconds

Matrix Size = 2048, time = 371.17542 seconds

Matrix Size = 2304, time = 490.22311 seconds

Matrix Size = 2560, time = 665.90334 seconds

Matrix Size = 2816, time = 950.09028 seconds

Matrix Size = 3072, time = 1154.8234 seconds

Matrix Size = 3328, time = 1447.418 seconds

Matrix Size = 3584, time = 2131.6487 seconds

**Figure 1. Serial Matrix Multiplication Run Time Characteristics**

The base algorithm behaves as expected. We know that the algorithm used has an order notation of O(n3) and looking at **Figure 1** it can be seen that the run time is exponentially increasing as the matrix size increases.

## Parallel

## NP = 2

Matrix Size = 256, time = 0.069588 seconds

Matrix Size = 512, time = 0.510835 seconds

Matrix Size = 768, time = 2.1731 seconds

Matrix Size = 1024, time = 13.790599 seconds

Matrix Size = 1280, time = 23.50243 seconds

Matrix Size = 1536, time = 55.698489 seconds

Matrix Size = 1792, time = 97.612683 seconds

Matrix Size = 2048, time = 188.87857 seconds

Matrix Size = 2304, time = 244.05105 seconds

Matrix Size = 2560, time = 347.84285 seconds

Matrix Size = 2816, time = 474.08665 seconds

Matrix Size = 3072, time = 576.89958 seconds

Matrix Size = 3328, time = 736.07543 seconds

Matrix Size = 3584, time = 1175.1372 seconds

**Figure 2. Parallel Matrix Multiplication Run Time Characteristics, NP(2)**

Comparing **Figure 2** with **Figure 1** it seems that utilizing one more process and splitting the work evenly between the two has cut the completion time almost in half for every case, which is in line with what we would expect to see.

## NP = 4

Matrix Size = 256, time = 0.054693 seconds

Matrix Size = 512, time = 0.289523 seconds

Matrix Size = 768, time = 1.221861 seconds

Matrix Size = 1024, time = 6.843977 seconds

Matrix Size = 1280, time = 11.904123 seconds

Matrix Size = 1536, time = 27.318558 seconds

Matrix Size = 1792, time = 48.401298 seconds

Matrix Size = 2048, time = 97.835424 seconds

Matrix Size = 2304, time = 121.94584 seconds

Matrix Size = 2560, time = 172.05514 seconds

Matrix Size = 2816, time = 236.1049 seconds

Matrix Size = 3072, time = 303.81367 seconds

Matrix Size = 3328, time = 370.86322 seconds

Matrix Size = 3584, time = 584.54626 seconds

**Figure 3. Parallel Matrix Multiplication Run Time Characteristics, NP(4)**

## NP = 8

Matrix Size = 256, time = 0.042516 seconds

Matrix Size = 512, time = 0.196104 seconds

Matrix Size = 768, time = 0.726653 seconds

Matrix Size = 1024, time = 3.569868 seconds

Matrix Size = 1280, time = 6.044348 seconds

Matrix Size = 1536, time = 13.962634 seconds

Matrix Size = 1792, time = 24.515898 seconds

Matrix Size = 2048, time = 49.485545 seconds

Matrix Size = 2304, time = 61.588715 seconds

Matrix Size = 2560, time = 88.130733 seconds

Matrix Size = 2816, time = 120.32 seconds

Matrix Size = 3072, time = 156.76207 seconds

Matrix Size = 3328, time = 187.06935 seconds

Matrix Size = 3584, time = 287.17158 seconds

**Figure 4. Parallel Matrix Multiplication Run Time Characteristics, NP(8)**

Examining the above data, we can see a trend forming. Every time the number of processes is doubled we are seeing the completion time cut almost in half for each run. The above implementations are behaving just like the serial portion but because the work is being distributed and has parallelized the program, it is able to finish in a timelier manner. We can see from each graph, of each parallel run, that the run time is increasing at an exponential rate as matrix size increases which is consistent with the behavior of the serial program.

**(What are the run time characteristics of these implementations?)**

# Appendix

Add anything else that might be pertinent to the assignment.