Daniel Murphy
CSCI 6454 – Fall 2020
Homework 3 – report

Question 1

Please see code documentation in file /HW3_DANIEL_MURPHY_1/p1.c for comments related to this question. For info on how to run p1.c, see the README.md file.

Question 2

1.

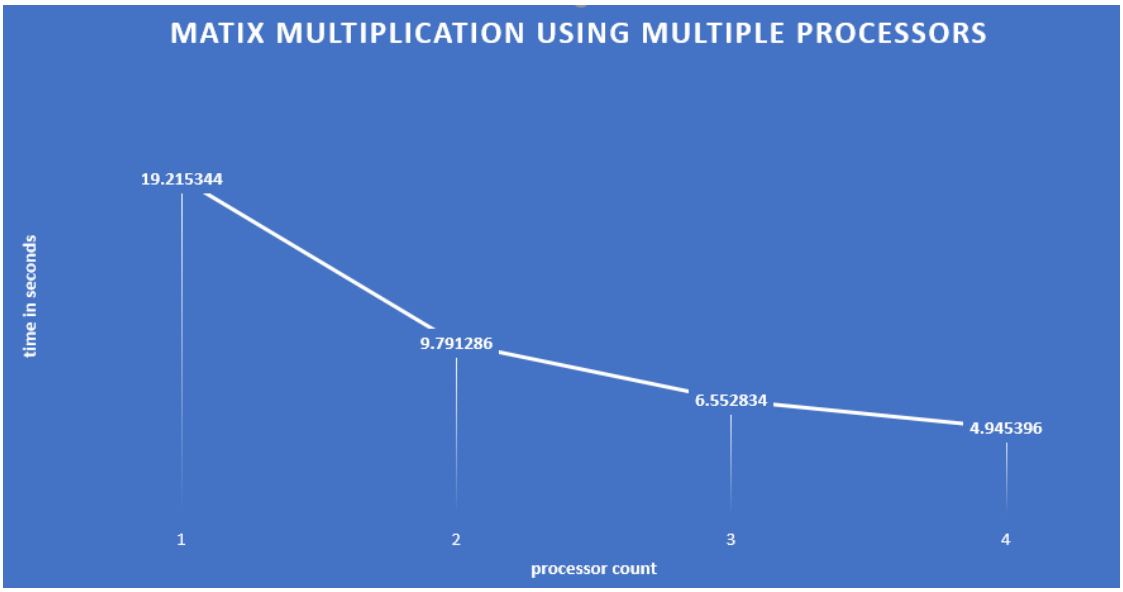
Matrices MatA and MatB are initialized to a SIZExSIZE dimension and populated with randomly generated integers. To achieve parallel matrix multiplication, each processor calculates a subset of the final result.

MatA is transposed to optimize memory locality and each processor multiplies its corresponding sub-set of MatA against the entirety of MatB, saving the results to a local one-dimensional array MatFlat. Each process' MatFlat is then gathered to the root processes MatC.

The one-dimensional matrix MatFlat was used to simplify the datatype used in the MPI_Gather operation.

As can be seen below, there is a performance benefit to multi-processing. One can note that the biggest improvement is seen between the single processor serial calculation, and the parallel calculation using 2 processors. The slightly diminishing returns in performance improvements might be explained by the overhead of sharing memory between the different processes.

2.



3.

An example of the input and output for a small matrix size. For the output of a large matrix, see the file MatC.txt

```
dooms@doomsnack:/media/sf_ParallelAndSciComp-6454/hw/hw3$ mpicc -g p2.c -o p2
dooms@doomsnack:/media/sf_ParallelAndSciComp-6454/hw/hw3$ mpirun -np 4 p2

INPUT:
MatA: ----------------------------------------------------       MatB ----------------------------------------------------
7       8       0       1       5       1       4       5       5       4       7       4       1       3       5       4
1       0       5       3       1       0       7       2       8       8       0       2       7       5       5       8
8       2       7       1       3       2       3       1       7       7       3       7       4       0       2       2
0       8       7       7       6       8       2       1       8       2       0       0       8       0       5       8
3       0       8       7       3       6       3       5       0       8       2       4       2       6       0       0
1       8       5       5       0       8       4       3       6       0       4       1       3       4       3       2
6       8       1       4       0       2       0       5       8       6       3       2       7       1       2       6
3       4       0       5       2       6       3       1       8       1       6       7       7       2       2       8

OUTPUT:-----------------------------------------------------
185     163     105     108     147     109     101     166
136     97      57      71      110     20      48      96
157     142     106     112     98      65      83      100
241     188     77      108     197     112     119     170
227     129     114     127     171     64      100     154
248     140     84      92      190     85      118     182
185     108     83      84      139     76      108     166
155     89      64      47      121     70      86      122

time taken 0.000084
```

These results are printed to the console automatically for matrix sizes of 9x9 and less.