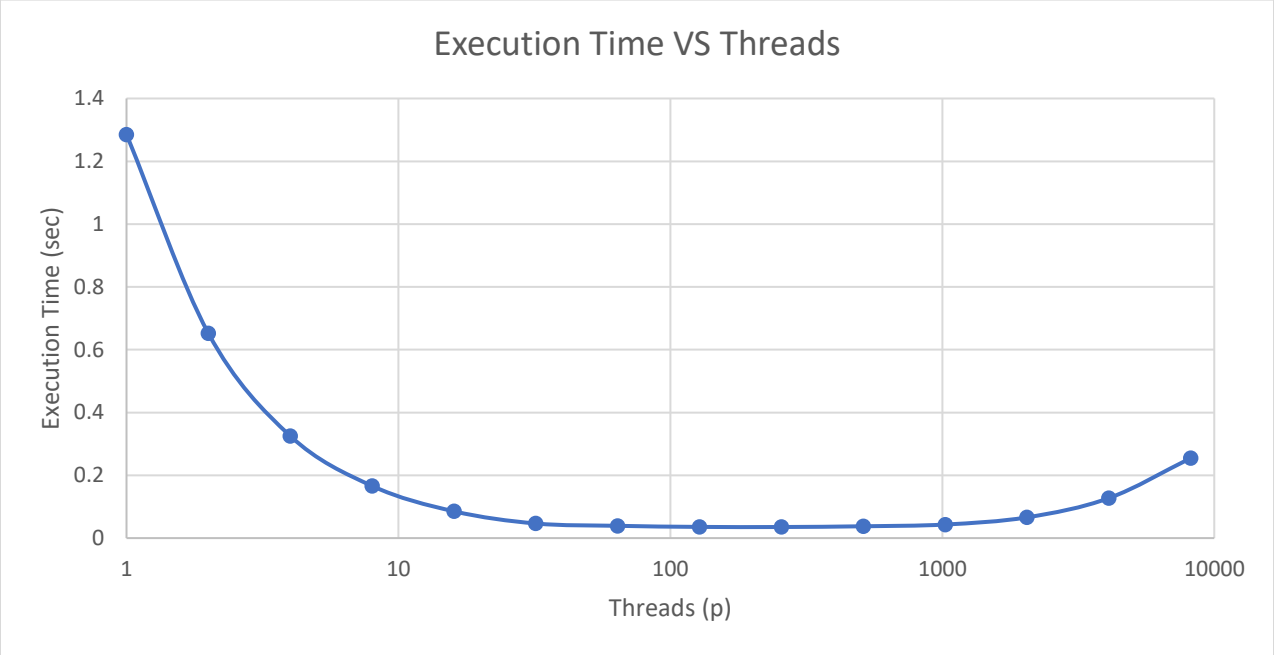
**PART 1. SHARED-MEMORY PROGRAMMING WITH THREADS**

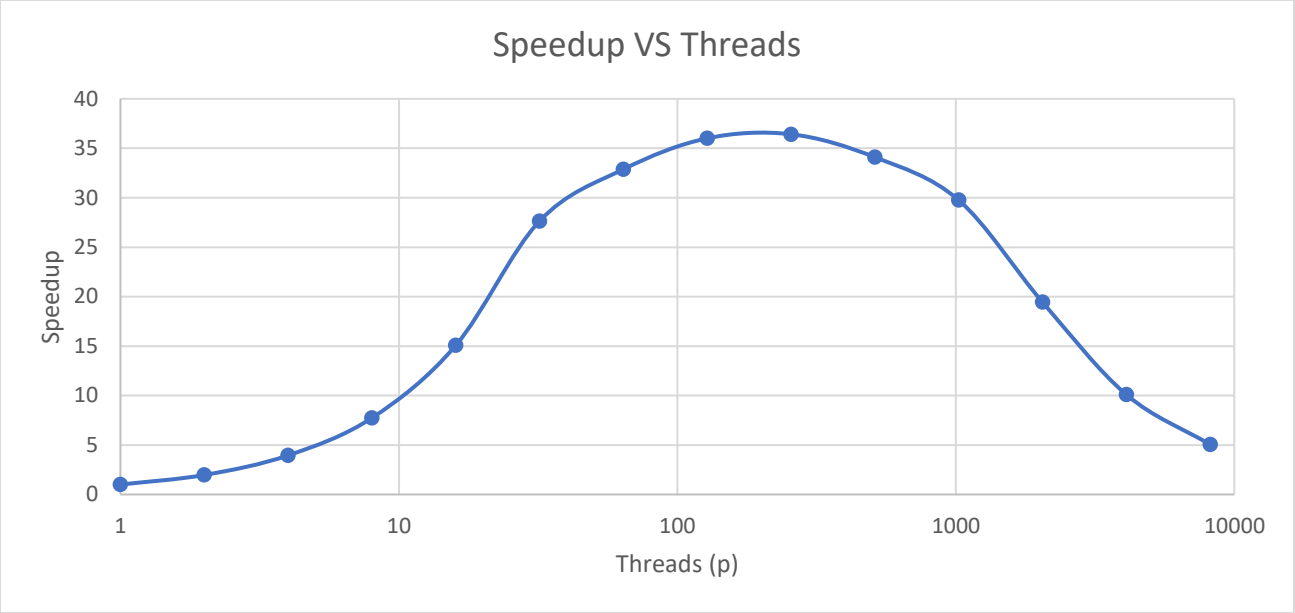
| Trials | Threads (p) | Execution Time | Speedup | Efficiency |
|---|---|---|---|---|
| 100000000 | 1 | 1.2856 | 1 | 1 |
| 100000000 | 2 | 0.6522 | 1.971174486 | 0.985587243 |
| 100000000 | 4 | 0.3253 | 3.952044267 | 0.988011067 |
| 100000000 | 8 | 0.1661 | 7.739915713 | 0.967489464 |
| 100000000 | 16 | 0.0853 | 15.07151231 | 0.941969519 |
| 100000000 | 32 | 0.0465 | 27.64731183 | 0.863978495 |
| 100000000 | 64 | 0.0391 | 32.8797954 | 0.513746803 |
| 100000000 | 128 | 0.0357 | 36.01120448 | 0.281337535 |
| 100000000 | 256 | 0.0353 | 36.41926346 | 0.142262748 |
| 100000000 | 512 | 0.0377 | 34.10079576 | 0.066603117 |
| 100000000 | 1024 | 0.0432 | 29.75925926 | 0.029061777 |
| 100000000 | 2048 | 0.0661 | 19.44931921 | 0.009496738 |
| 100000000 | 4096 | 0.1273 | 10.09897879 | 0.002465571 |
| 100000000 | 8192 | 0.2547 | 5.047506871 | 0.000616151 |

| Trials | Error |
|---|---|
| 1000 | 4.76E-02 |
| 10000 | 1.01E-02 |
| 100000 | 4.29E-03 |
| 1000000 | 1.21E-03 |
| 10000000 | 1.29E-03 |
| 100000000 | 2.08E-04 |
| 1000000000 | 8.04E-06 |

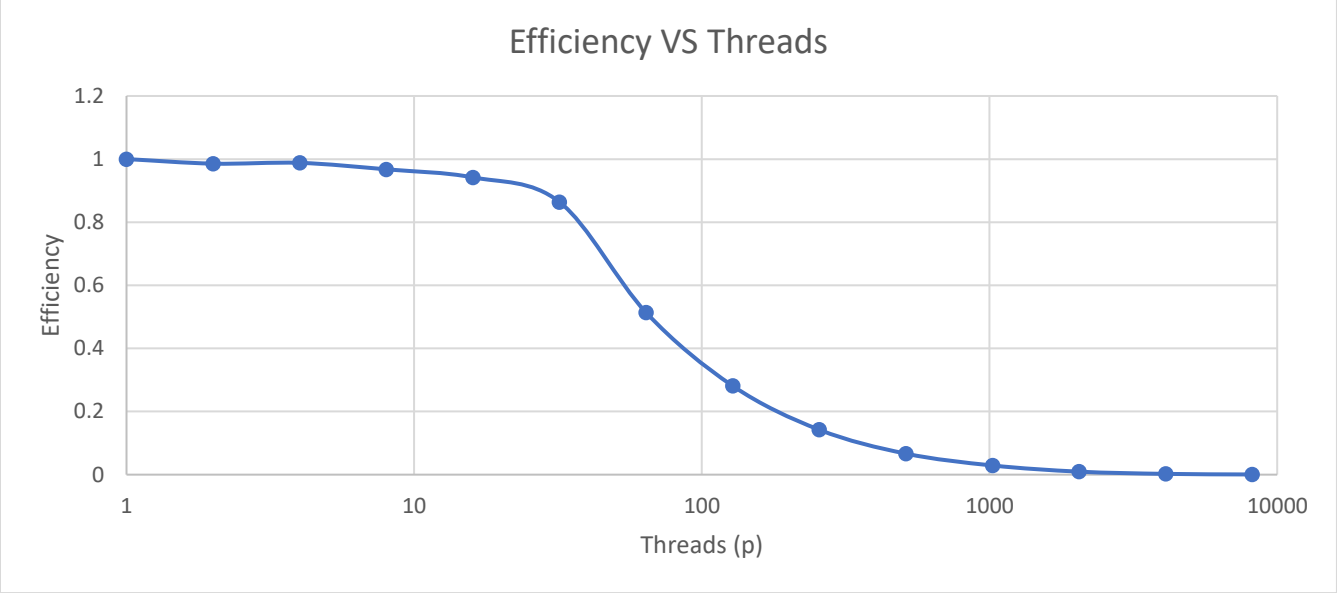
1.1:



1.2:




1.3:

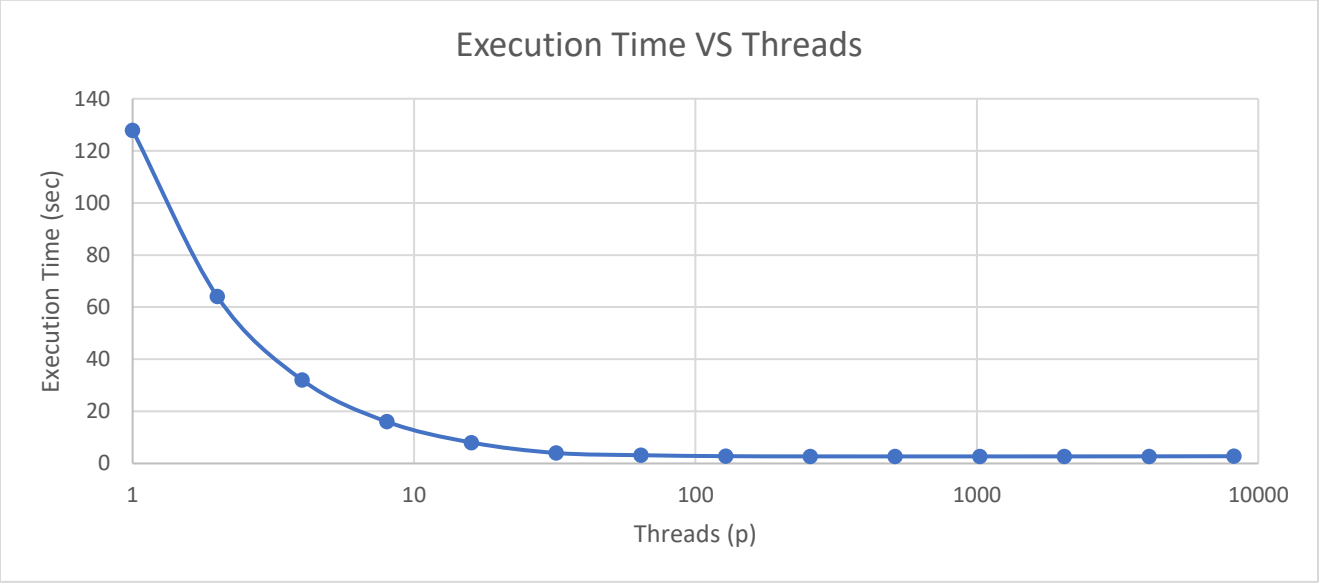


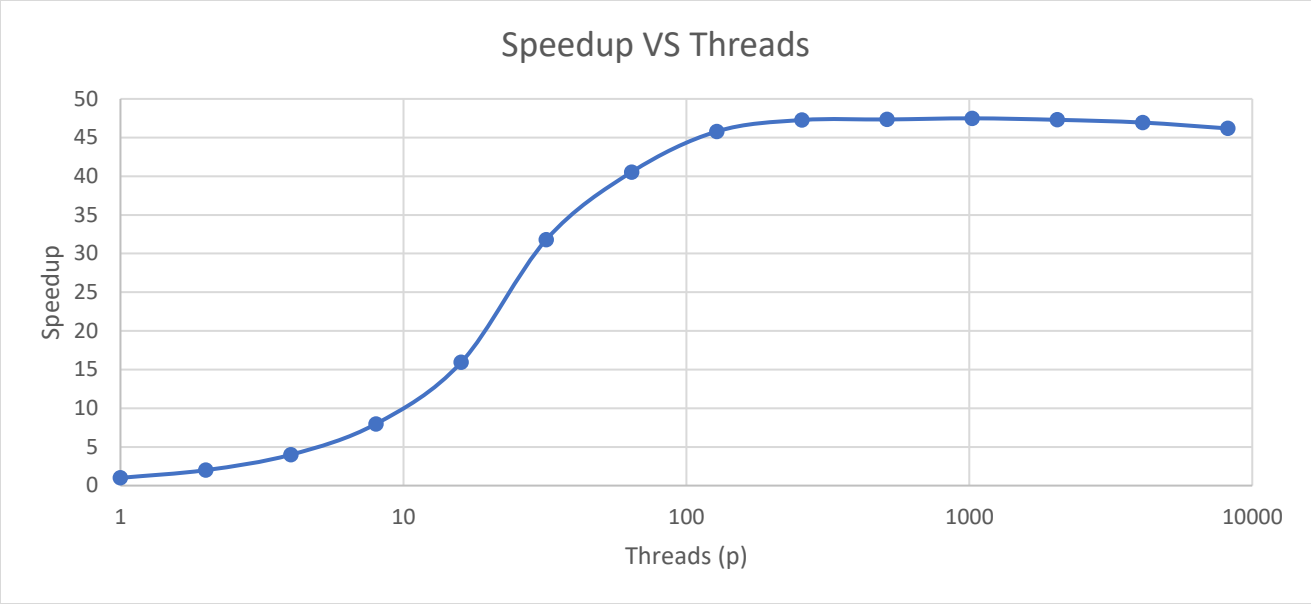

1.4:

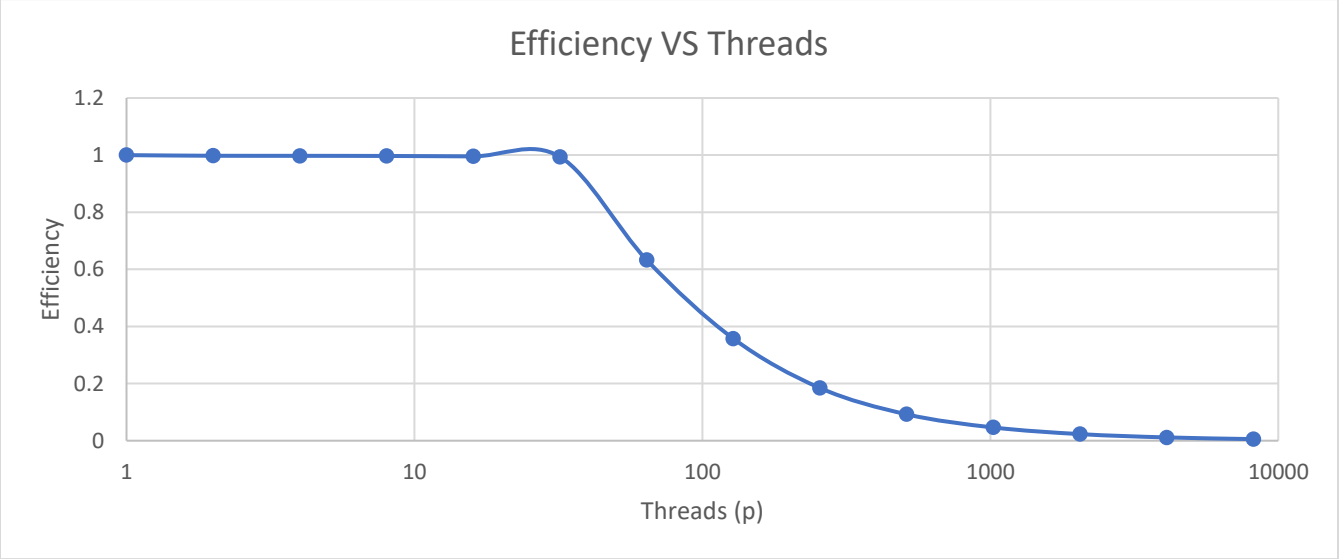
In my experiments, it was found that the parallel runtime is minimized when the number of threads (p) is 256, with the smallest runtime of 0.0353 seconds.

2:

| Trials | Threads (p) | Execution Time (sec) | Speedup | Efficiency |
|---|---|---|---|---|
| 10000000000 | 1 | 127.9034 | 1 | 1 |
| 10000000000 | 2 | 64.0943 | 1.995550306 | 0.997775153 |
| 10000000000 | 4 | 32.052 | 3.990496693 | 0.997624173 |
| 10000000000 | 8 | 16.0346 | 7.976712858 | 0.997089107 |
| 10000000000 | 16 | 8.0292 | 15.9297813 | 0.995611331 |
| 10000000000 | 32 | 4.02 | 31.81676617 | 0.994273943 |
| 10000000000 | 64 | 3.1552 | 40.53733519 | 0.633395862 |
| 10000000000 | 128 | 2.7934 | 45.7877139 | 0.357716515 |
| 10000000000 | 256 | 2.7058 | 47.27008648 | 0.184648775 |
| 10000000000 | 512 | 2.7018 | 47.34006958 | 0.092461073 |
| 10000000000 | 1024 | 2.6934 | 47.4877107 | 0.046374717 |
| 10000000000 | 2048 | 2.7044 | 47.29455702 | 0.023093045 |
| 10000000000 | 4096 | 2.7247 | 46.94219547 | 0.011460497 |
| 10000000000 | 8192 | 2.7695 | 46.18284889 | 0.005637555 |






2.1:

In my experiments, it was found that the parallel runtime is minimized when the number of threads (p) is 1024, with the smallest runtime of 2.6934 seconds.

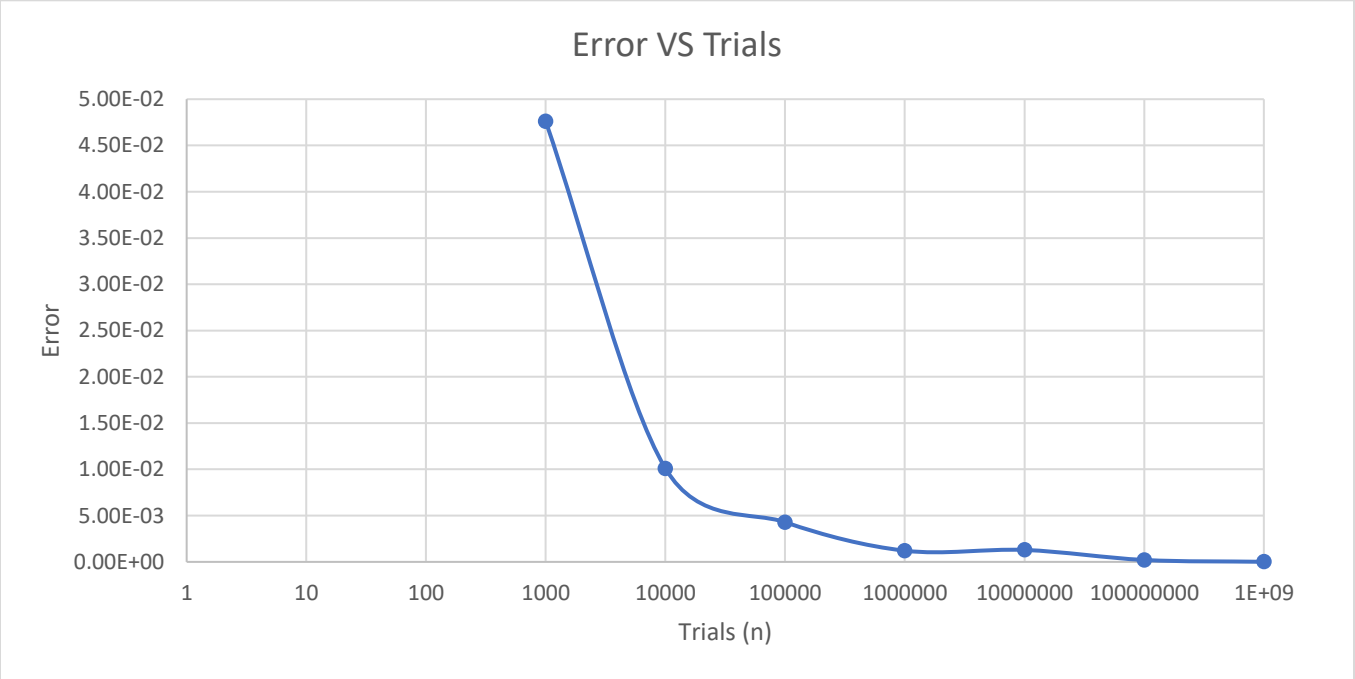
2.2:

As you increase the number of threads (p), the runtime may decrease in the beginning (when # of threads is smaller) but will immediately go back up (increase) due to the overhead by the number of threads. The reason the runtime increases is because of task scheduling, which causes the runtime to increase. This is observed in my experiment as after p was set to 1024, the runtime began to increase once again. If we had tested with a bigger number of threads (p), then we would see the runtime to continue to increase.

3:

Yes, I did expect that there would be a difference in the number of threads to get the minimum execution time. If we look at 1.4, the number of threads (p) with the minimized execution time was 256, while in 2.1, the number of threads (p) with the minimized execution time as 1024. There is a great difference between the number of threads required to get the least amount of runtime. The only difference between the two would be that the number of trials are different, with 1.4 being 100000000, and 2.1 being 10000000000. This is easily showcased within the data.

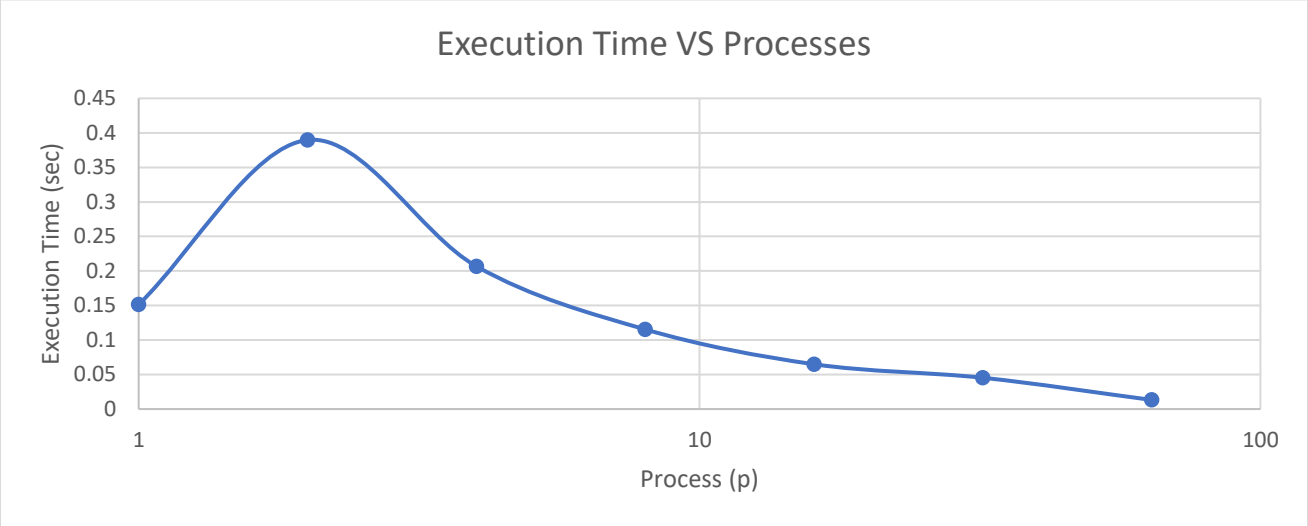
4:



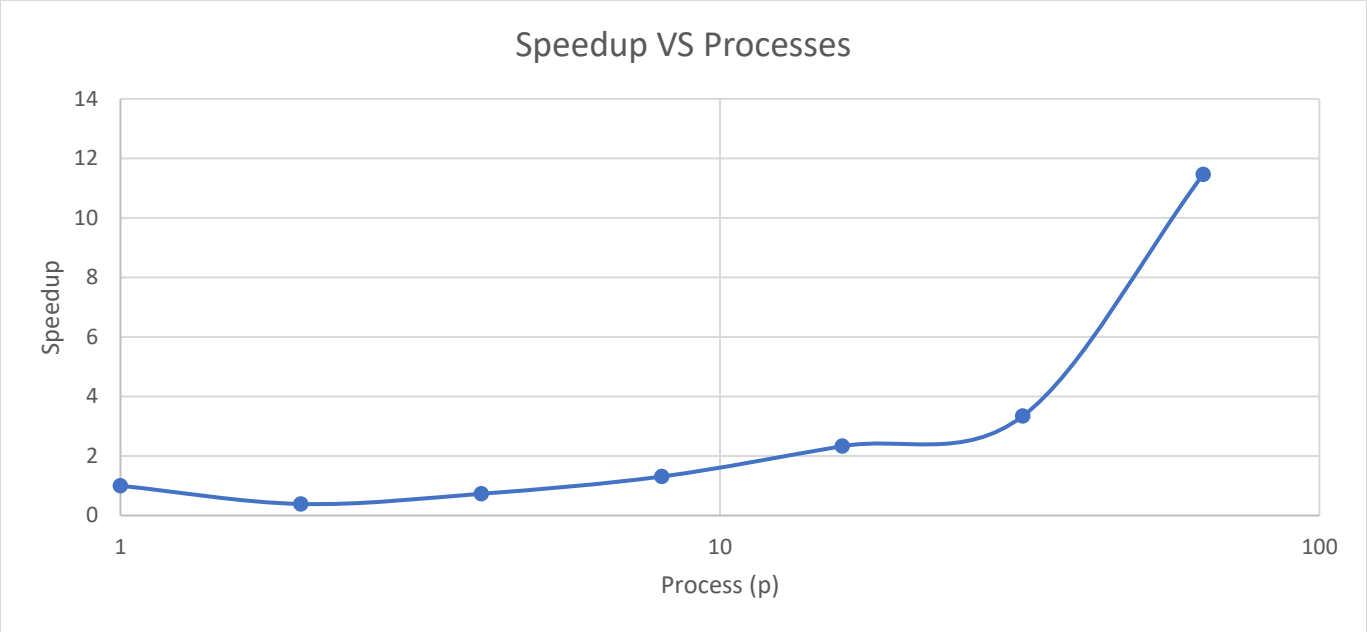
**PART 2. DISTRIBUTED-MEMORY PROGRAMMING WITH MPI**

5.1:

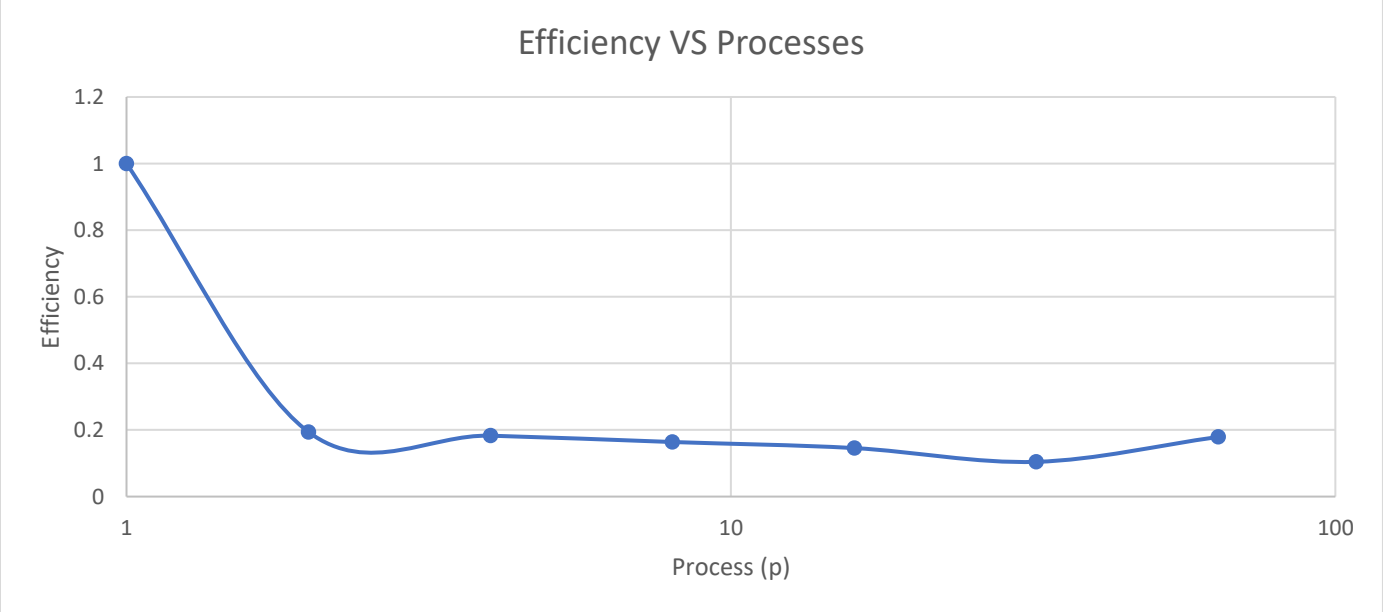
| Processes (p) | Execution Time | Speedup | Efficiency |
|---|---|---|---|
| 1 | 0.1513 | 1 | 1 |
| 2 | 0.3897 | 0.38824737 | 0.194123685 |
| 4 | 0.2068 | 0.731624758 | 0.18290619 |
| 8 | 0.1154 | 1.311091854 | 0.163886482 |
| 16 | 0.0649 | 2.331278891 | 0.145704931 |
| 32 | 0.0453 | 3.33995585 | 0.10437362 |
| 64 | 0.0132 | 11.46212121 | 0.179095644 |



5.2:




5.3:

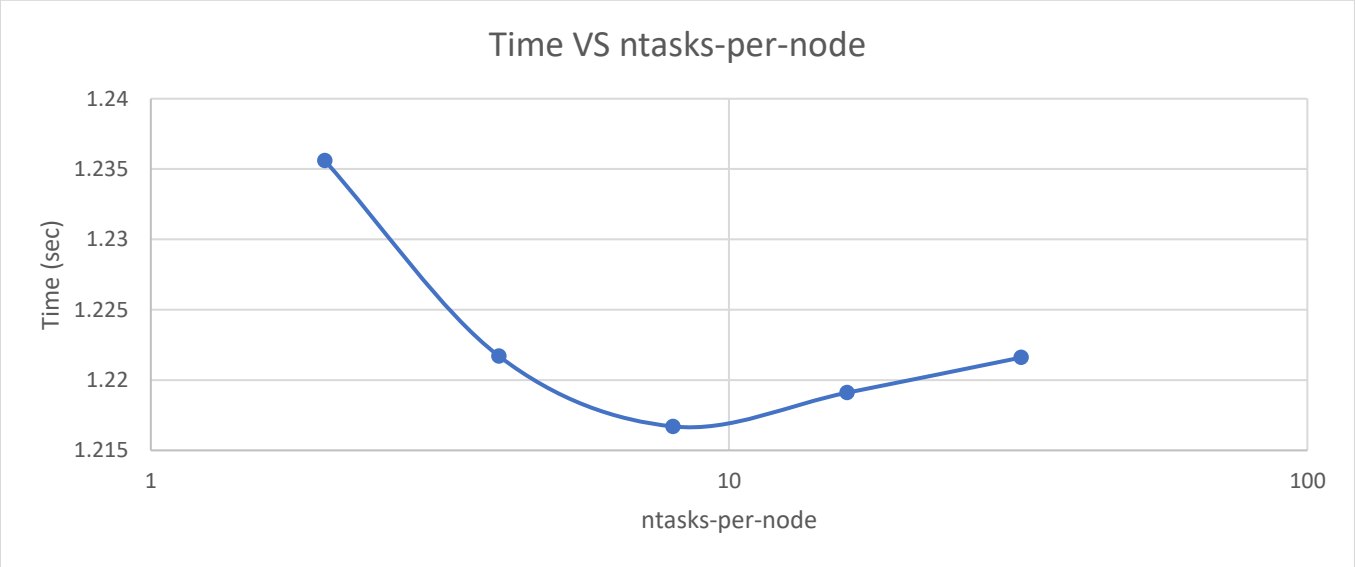



5.4:

In my experiments, it was found that the parallel runtime is minimized when the number of processes (p) is 64, with the smallest runtime of 0.0132 seconds.

6:

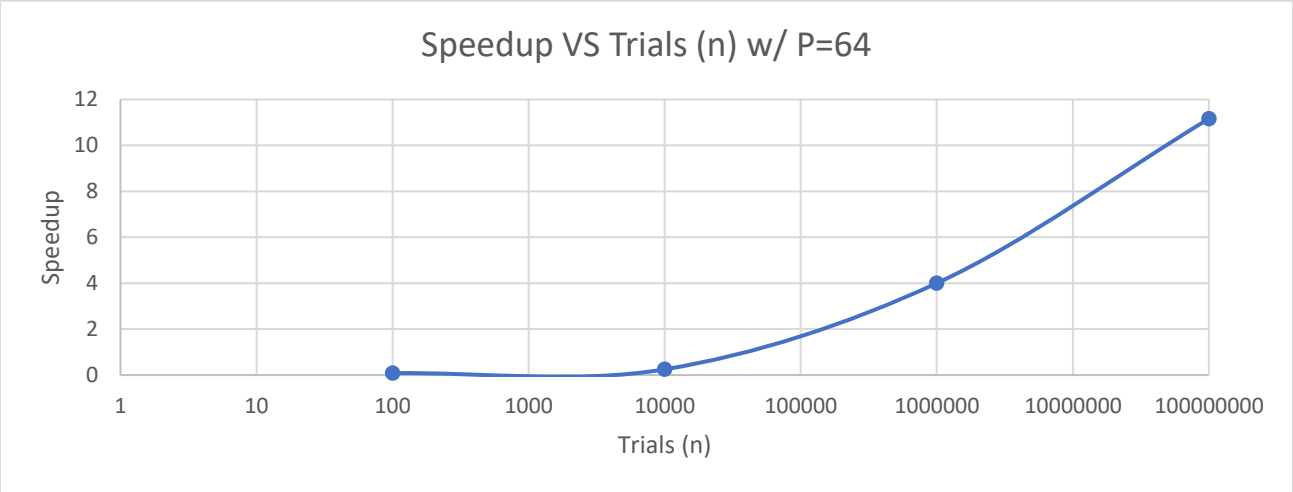
| ntasks-per-node | Execution Time |
|---|---|
| 2 | 1.2356 |
| 4 | 1.2217 |
| 8 | 1.2167 |
| 16 | 1.2191 |
| 32 | 1.2216 |




According to the graph, when ntasks-per-node = 8, the total time was minimized to 1.2167 seconds.

7.1:

| Trials (n) | Speedup | Error | Time (p = 1) | Time (p = 64) |
|---|---|---|---|---|
| 100 | 0.083333333 | 2.65E-06 | 0.0001 | 0.0012 |
| 10000 | 0.25 | 2.65E-10 | 0.0001 | 0.0004 |
| 1000000 | 4 | 2.62E-14 | 0.0016 | 0.0004 |
| 1E+08 | 11.16176471 | 2.83E-16 | 0.1518 | 0.0136 |




P is always set to 64, however we used the execution time when P is set to 1, to calculate the speedup values.

7.2: