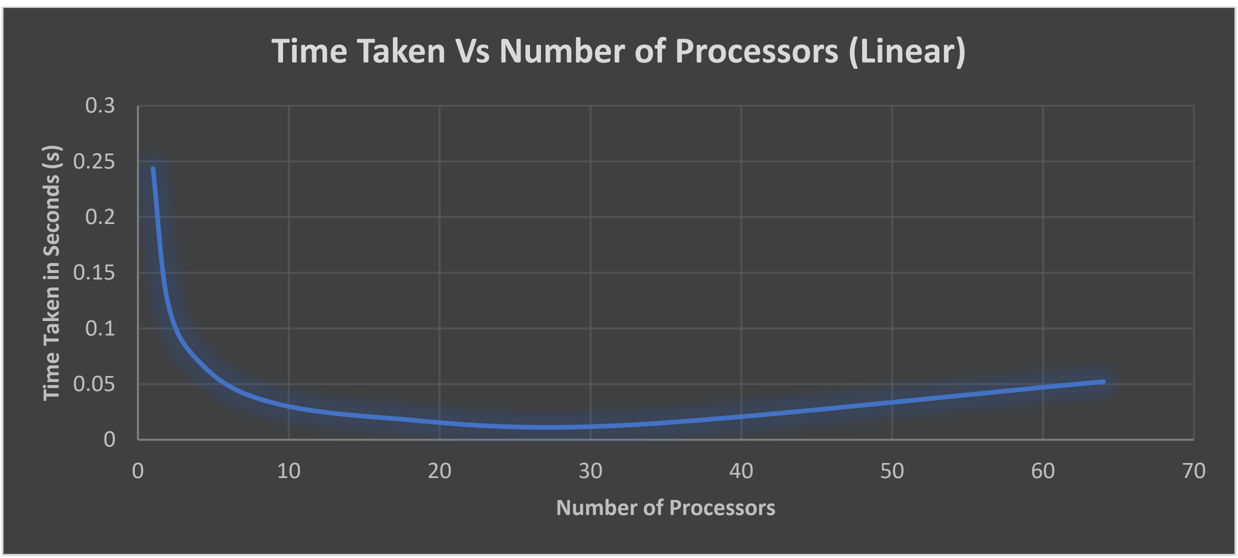
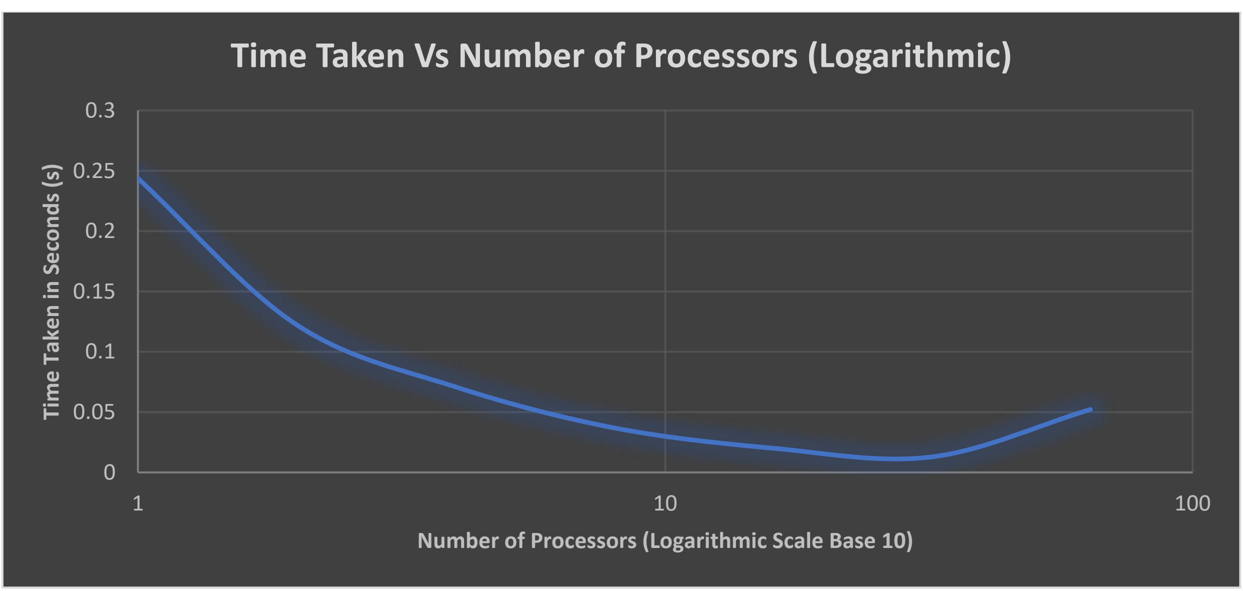
Alireza Safdari Ghandhari | Homework1 | 09/13/2020
UIN: 728009403 | CSCE-735

Q1) Plot execution time versus p to demonstrate how time varies with the number of processes. Use a logarithmic scale for the x-axis.

The number of intervals is 100,000,000

Number of Processes	Test 1 (s)	Test 2 (s)	Test 3 (s)	Test 4 (s)	Test 5 (s)	Average Time (s)
1	0.244	0.2426	0.2434	0.2427	0.2457	0.24368
2	0.122	0.1217	0.1221	0.1217	0.1242	0.12234
4	0.0619	0.0614	0.0611	0.0614	0.1088	0.07092
8	0.0324	0.0332	0.0319	0.0319	0.0551	0.0369
16	0.0163	0.0165	0.0245	0.0175	0.0255	0.02006
32	0.0088	0.019	0.0088	0.0105	0.0171	0.01284
64	0.0515	0.055	0.0525	0.0508	0.051	0.05216

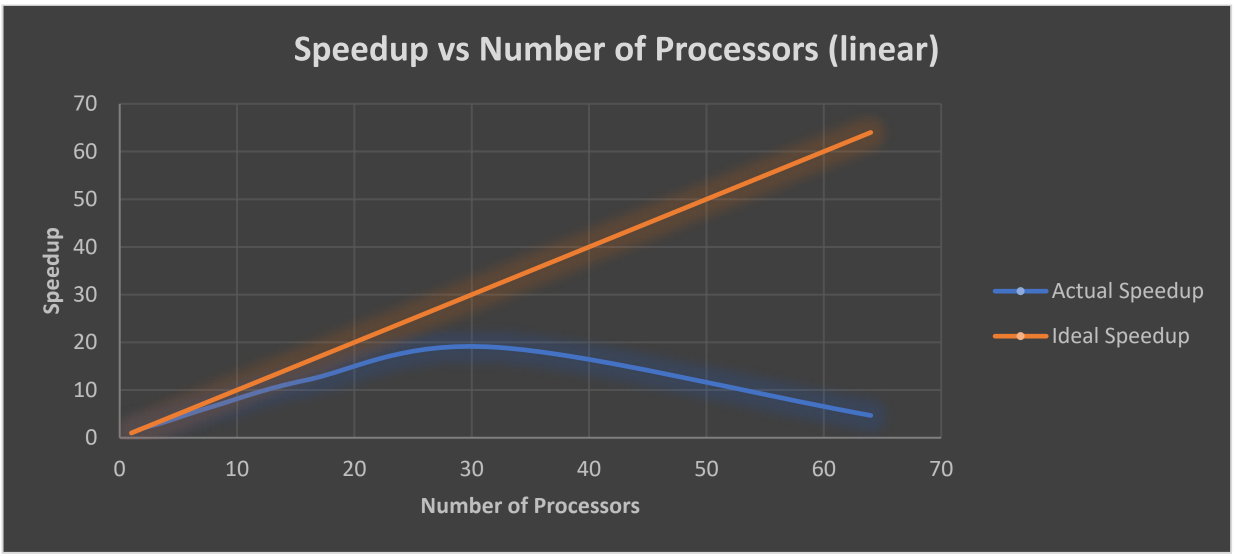


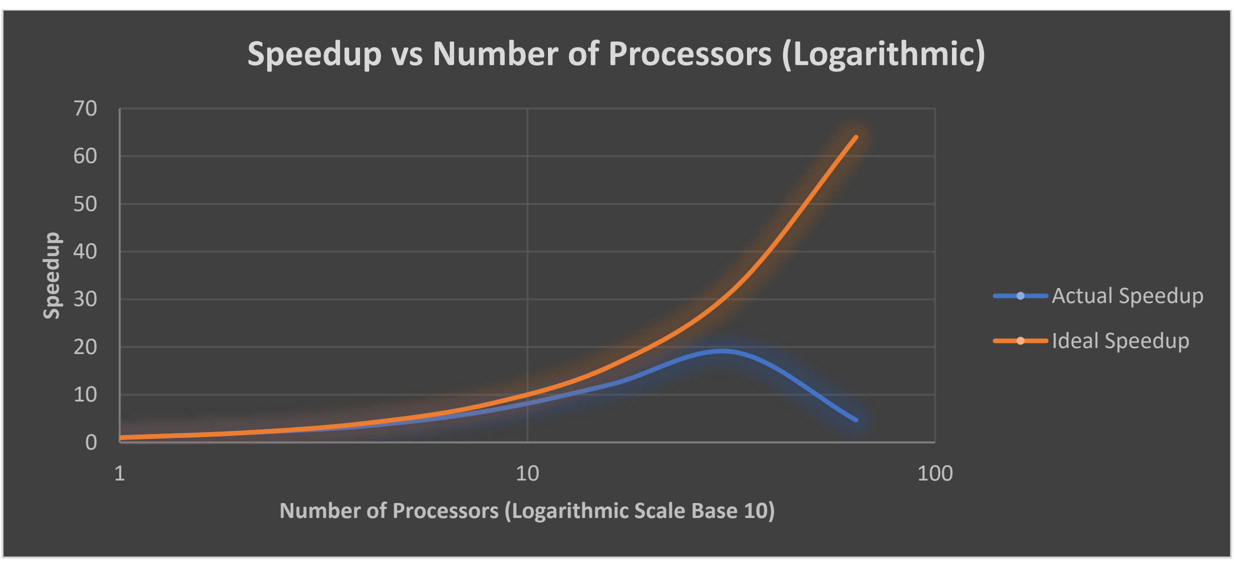


Alireza Safdari Ghandhari | Homework1 | 09/13/2020
UIN: 728009403 | CSCE-735

Q2) Plot speedup versus p to demonstrate the change in speedup with p.

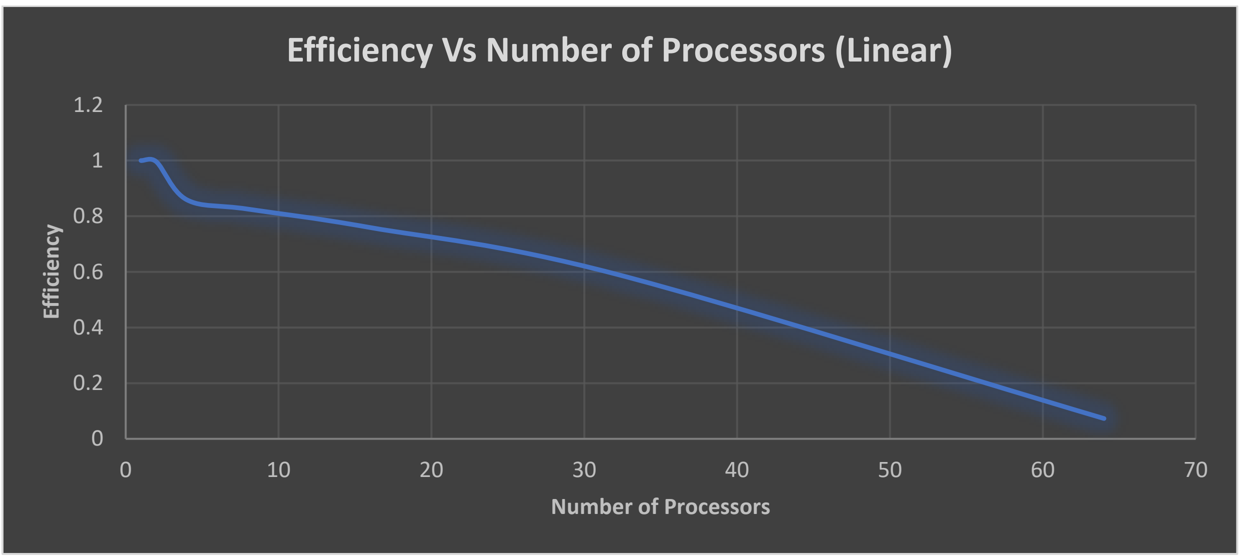
Number of Processes	Average Time (s)	Speedup	Ideal Speedup
1	0.24368	1	1
2	0.12234	1.991826	2
4	0.07092	3.435984	4
8	0.0369	6.603794	8
16	0.02006	12.14756	16
32	0.01284	18.97819	32
64	0.05216	4.671779	64

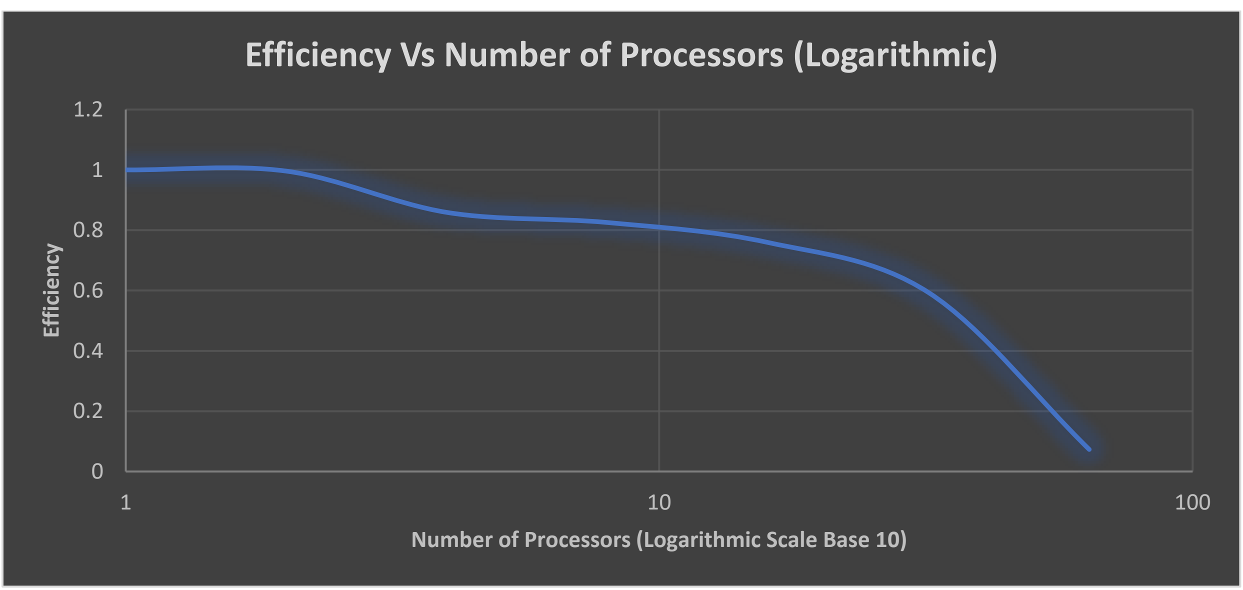




Q3) Using the definition: efficiency = speedup/p, plot efficiency versus p to demonstrate how efficiency changes as the number of processes is increased.

Number of Processes	Average Time (s)	Speedup	Ideal Speedup	Efficiency
1	0.24368	1	1	1
2	0.12234	1.991826	2	0.99591303
4	0.07092	3.435984	4	0.85899605
8	0.0369	6.603794	8	0.82547425
16	0.02006	12.14756	16	0.75922233
32	0.01284	18.97819	32	0.59306854
64	0.05216	4.671779	64	0.07299655





Alireza Safdari Ghandhari
UIN: 728009403

Homework1
CSCE-735

09/13/2020

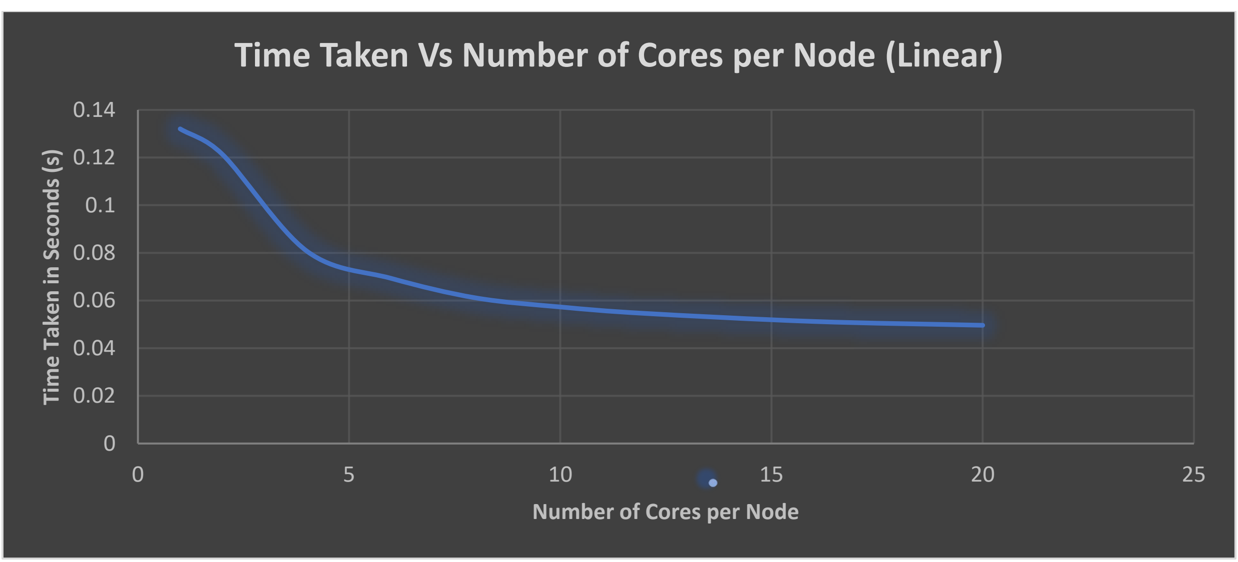
Q4) What value of p minimizes the parallel runtime?

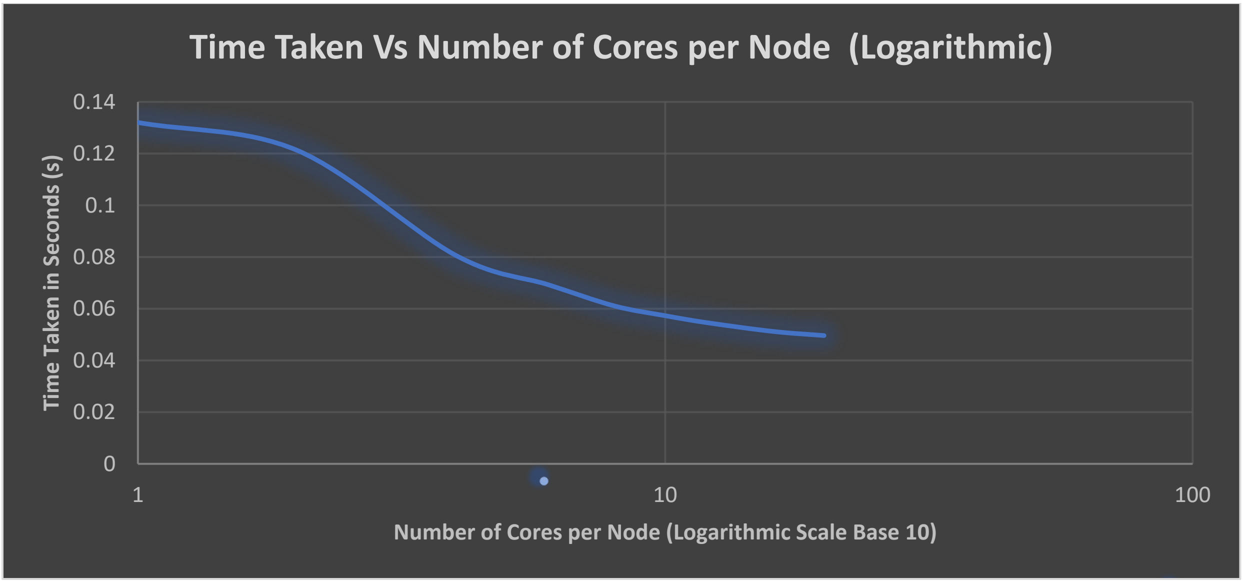
P of 32 seems to give us the best possible run time at 0.01284 seconds. Surprisingly 64 processors do not give us a better time, this could be mainly due to interconnect communication which results in slower overall performance. Efficiency once drop after 2 processors and then it drops at 16 and 32 again.

Q5) With n=10^9 and p=64, determine the value of ptile that minimizes the total_time. Plot time versus ptile to illustrate your experimental results for this question.

Ptile	Test 1 (s)	Test 2 (s)	Test 3 (s)	Test 4 (s)	Test 5 (s)	Average Time (s)
1	0.131	0.1362	0.1305	0.1309	0.1316	0.13204
2	0.1212	0.1214	0.1207	0.1214	0.122	0.12134
4	0.0805	0.0809	0.081	0.0804	0.0808	0.08072
6	0.0722	0.0679	0.0709	0.0677	0.0677	0.06928
8	0.0615	0.0602	0.0614	0.0612	0.0612	0.0611
10	0.057	0.0573	0.0574	0.0574	0.0574	0.0573
12	0.0549	0.0546	0.0543	0.0544	0.0548	0.0546
16	0.0517	0.0511	0.0516	0.052	0.0497	0.05122
20	0.0513	0.0491	0.0494	0.0492	0.0491	0.04962
24	Job 12630279 Still Pending					
40	Job 12630327 Still Pending					

Ptile values were selected based on Ada Hardware, there are multiple computers with 24 and 40 cores per node that is why 24 and 40 are in the list but since there are only small number of such nodes the job has been on pending for a long time. Beside 24 and 40 cores per nodes, 20 cores per node takes the list amount of time

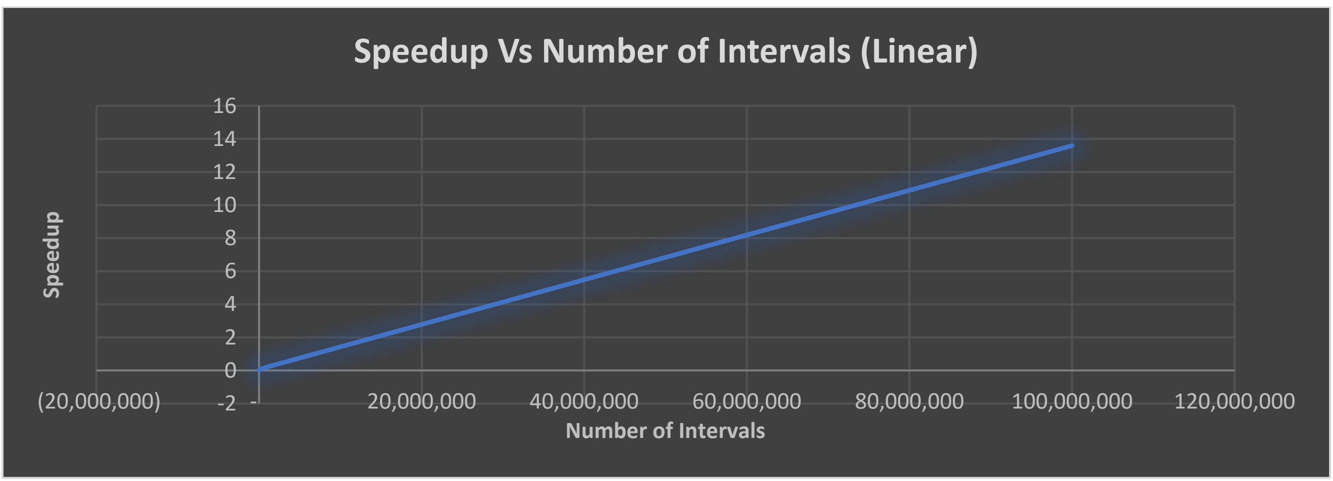


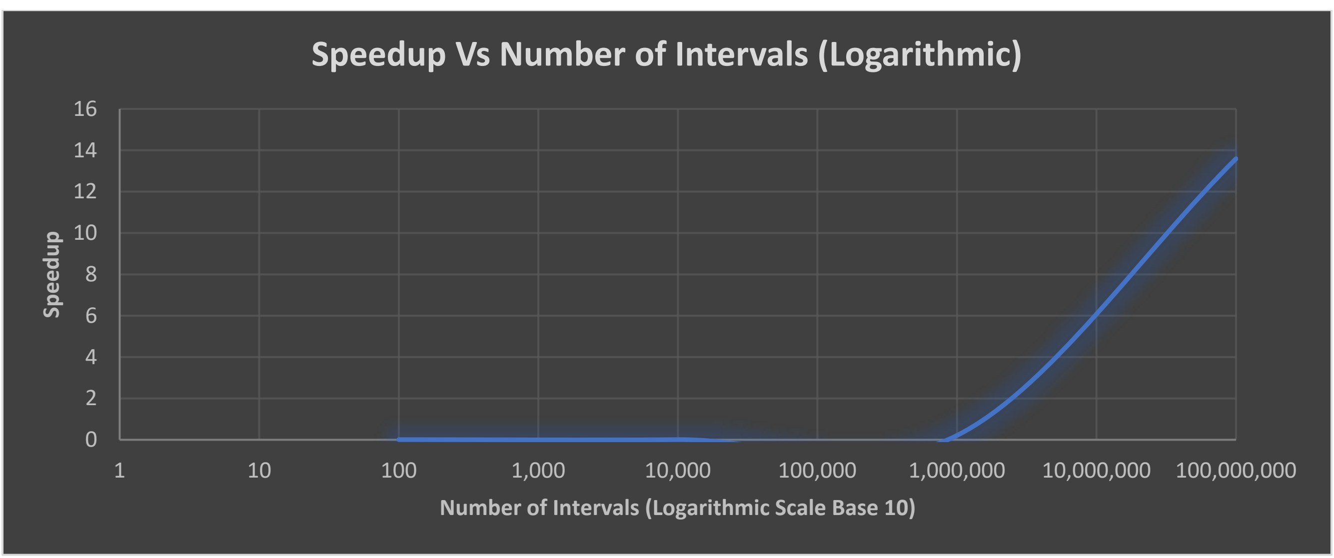

Time Taken Vs Number of Cores per Node (Logarithmic)
Time Taken in Seconds (s)
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
1
10
100
Number of Cores per Node (Logarithmic Scale Base 10)

Q6) Repeat the experiments with p=64 for n=10^2, 10^4, 10^6 and 10^8.

a) Plot the speedup observed w.r.t. p=1 versus n.
   The 64 processor only become faster when we have 10^8 intervals, prior to that the overhead of 64 processors working in parallel outweigh the benefits of running the task in parallel.

# of Processors	Number of Intervals	Test 1 (s)	Test 2 (s)	Test 3 (s)	Test 4 (s)	Test 5 (s)	Average Time (s)	Speedup
1	100	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	1
64	100	0.0144	0.0126	0.0117	0.0113	0.0115	0.0123	0.00813
1	10,000	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	1
64	10,000	0.0133	0.0115	0.0143	0.012	0.0134	0.0129	0.007752
1	1,000,000	0.0025	0.0026	0.0025	0.0025	0.0025	0.00252	1
64	1,000,000	0.0133	0.0115	0.0113	0.0133	0.0112	0.01212	0.207921
1	100,000,000	0.2427	0.2427	0.2429	0.2427	0.2428	0.24276	1
64	100,000,000	0.0245	0.0155	0.017	0.0172	0.0151	0.01786	13.59239





b) Plot the relative error versus n to illustrate the accuracy of the algorithm as a function of n. The error for 1 processor increases at 10^8 intervals but that is just a coincidence due to random numbers generated for that round, otherwise the number of processors should not make a difference in the general trend of pf the error. To make the graphs easier to understand the error is in logarithmic scale in the second graph. Only error for 64 processors are considered in the graph to avoid the anomaly in 1 processor case.

# of Processors	Number of Intervals	Test 1 Error	Test 2 Error	Test 3 Error	Test 4 Error	Test 5 Error	Average Error
1	100	2.65E-06	2.65E-06	2.65E-06	2.65E-06	2.65E-06	2.65E-06
64	100	2.65E-06	2.65E-06	2.65E-06	2.65E-06	2.65E-06	2.65E-06
1	10,000	2.65E-10	2.65E-10	2.65E-10	2.65E-10	2.65E-10	2.65E-10
64	10,000	2.65E-10	2.65E-10	2.65E-10	2.65E-10	2.65E-10	2.65E-10
1	1,000,000	3.51E-14	3.51E-14	3.51E-14	3.51E-14	3.51E-14	3.51E-14
64	1,000,000	2.63E-14	2.63E-14	2.63E-14	2.63E-14	2.63E-14	2.63E-14
1	100,000,000	1.35E-13	1.35E-13	1.35E-13	1.35E-13	1.35E-13	1.35E-13
64	100,000,000	7.07E-16	7.07E-16	7.07E-16	7.07E-16	7.07E-16	7.07E-16