Operating Systems

Assignment 2

Group 17

By

Vishwas Baya 2019AAPS0224H

Abdul Azeem Shaik 2019AAPS1234H

Haarika Manda 2019AAPS1226H

Mohit Krishna 2019AAPS0343H

Pavan Sreekireddy 2019AAPS1225H

Surya Garikipati 2019AAPS1221H

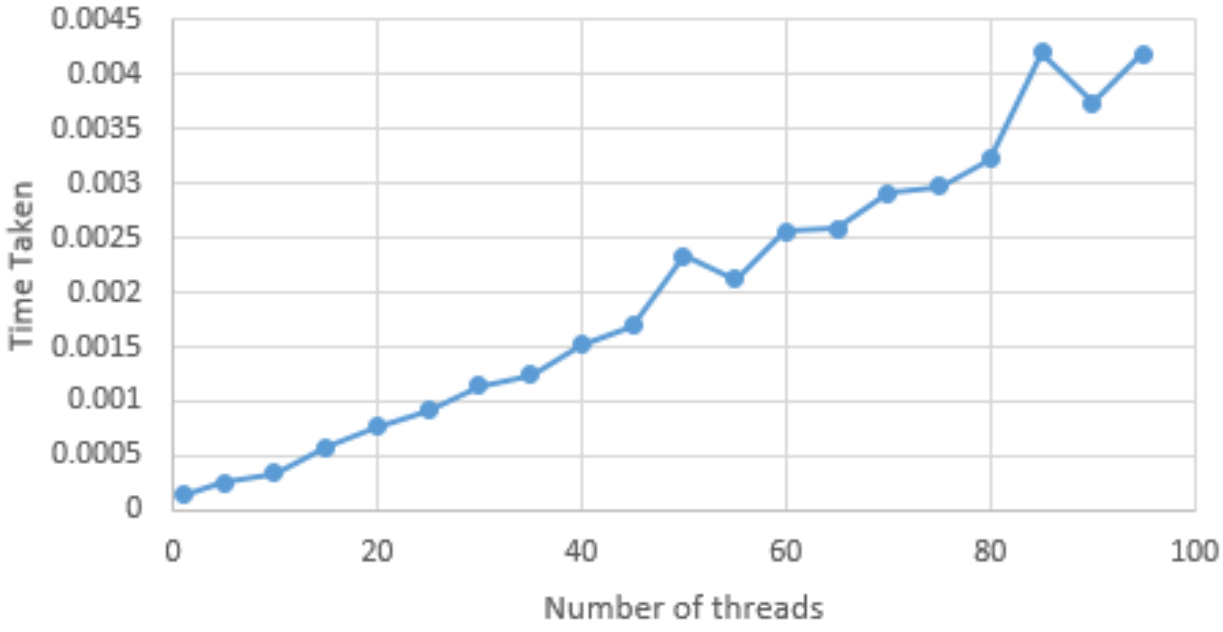
Nilay Arya 2019AAPS1230H

Pradeep Alapati 2019AAPS0283H

Plots for Process P1

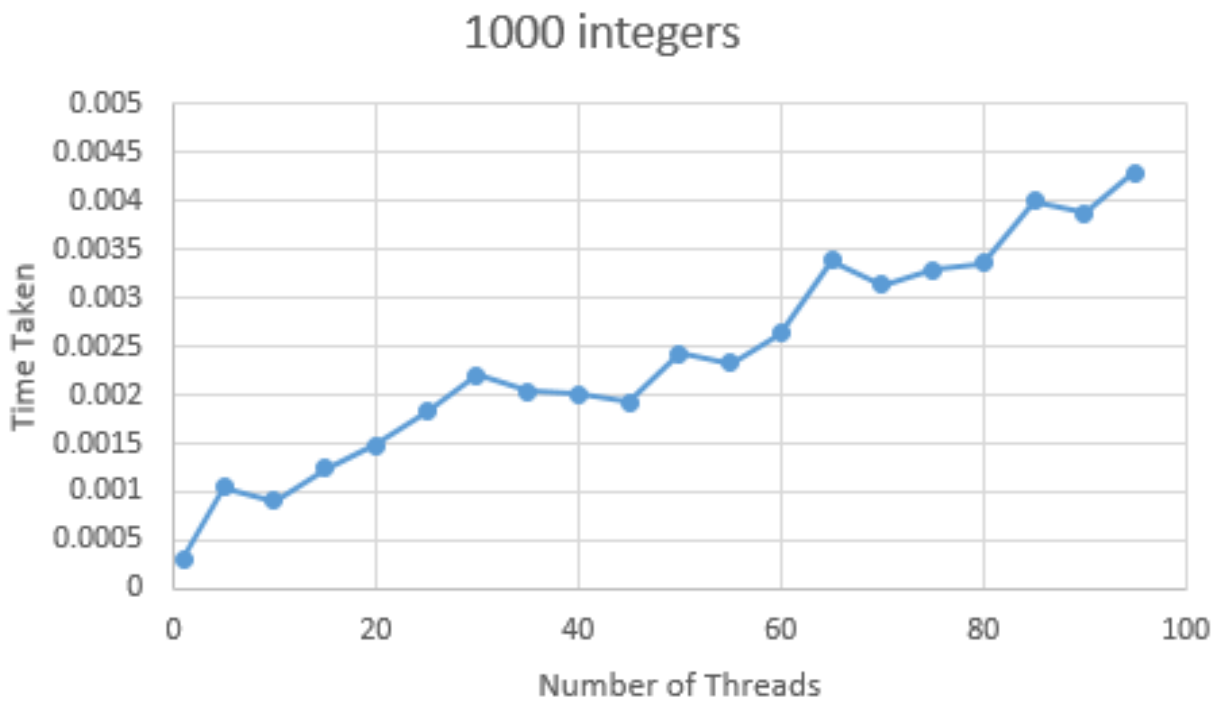
100 Numbers

Number of Threads	Time taken
1	0.000137
5	0.000252
10	0.000344
15	0.000574
20	0.000761
25	0.000918
30	0.001149
35	0.001249
40	0.001516
45	0.001692
50	0.002338
55	0.002124
60	0.002564
65	0.002585
70	0.00291
75	0.002974
80	0.003226
85	0.004203
90	0.003742
95	0.004191



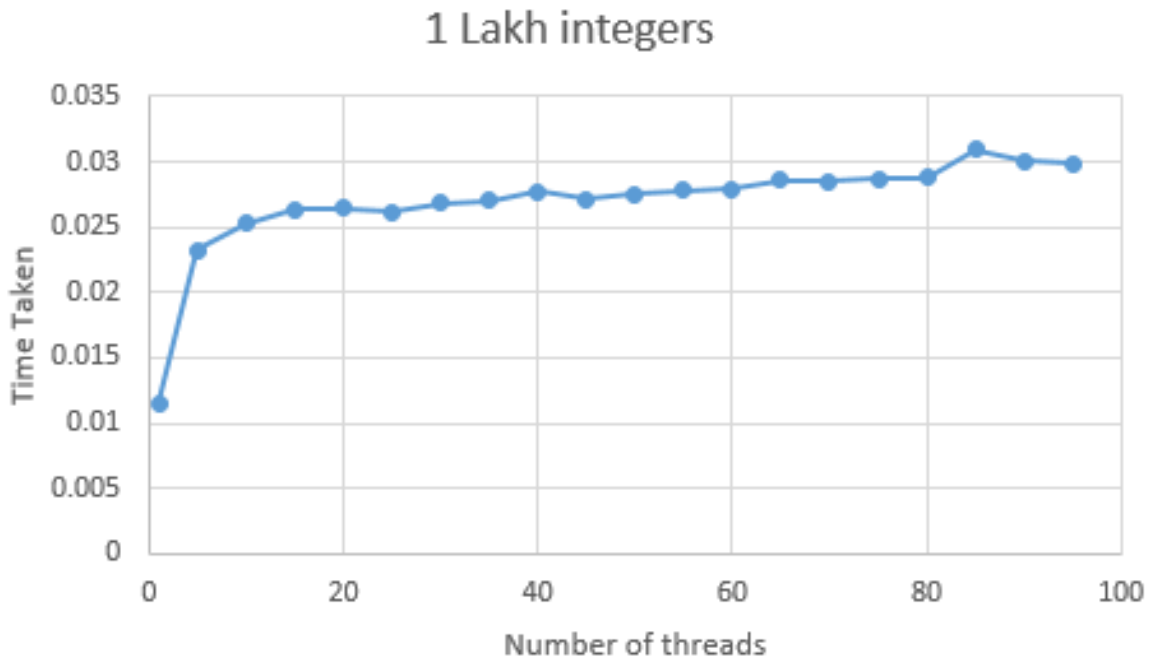
1000 Numbers

Threads	Time Taken
1	0.000305
5	0.001051
10	0.000913
15	0.001238
20	0.001483
25	0.001825
30	0.002206
35	0.002038
40	0.002014
45	0.001924
50	0.00243
55	0.002328
60	0.00263
65	0.003384
70	0.003143
75	0.003298
80	0.003364
85	0.004002
90	0.00388
95	0.00429



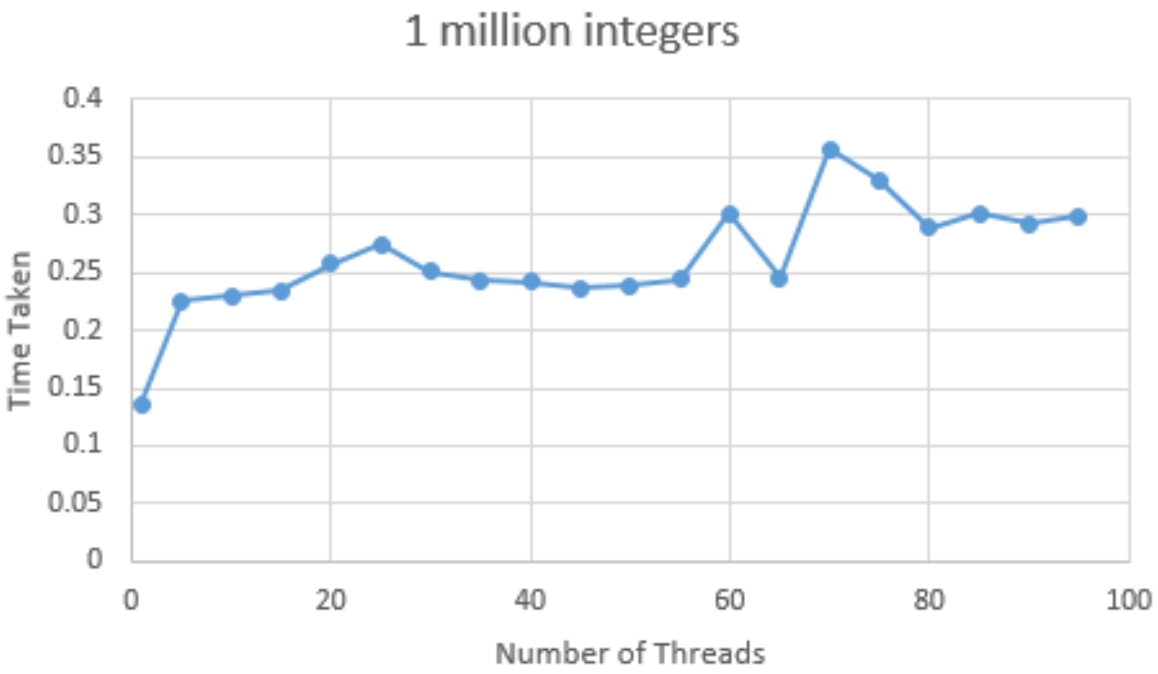
1 Lakh Numbers

Threads	Time Taken
1	0.01157
5	0.023244
10	0.025249
15	0.026314
20	0.026498
25	0.026149
30	0.02684
35	0.027068
40	0.027712
45	0.027135
50	0.027524
55	0.027846
60	0.027924
65	0.028564
70	0.028521
75	0.028702
80	0.028826
85	0.030977
90	0.030071
95	0.029905



1 Million Numbers

Threads	Time Taken
1	0.136134
5	0.225059
10	0.229567
15	0.234575
20	0.257966
25	0.274126
30	0.251031
35	0.243655
40	0.24256
45	0.236152
50	0.239141
55	0.244059
60	0.301475
65	0.245425
70	0.356761
75	0.329704
80	0.289032
85	0.300812
90	0.291859
95	0.299305



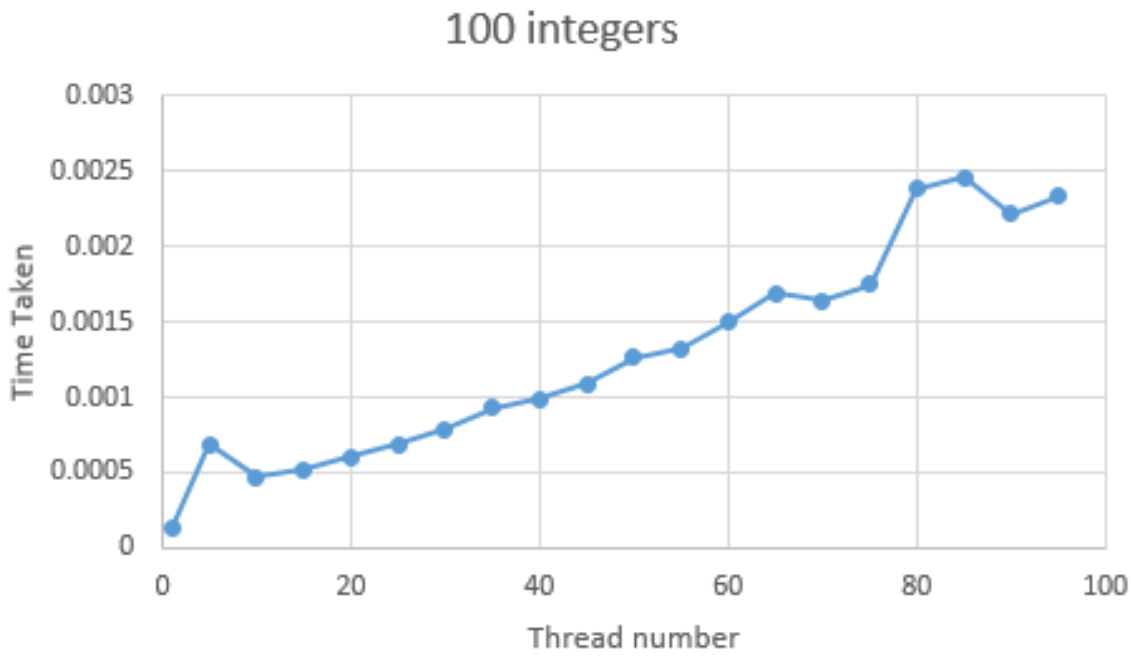
Analysis:

As we can see the write time to a shared memory increases as the number of threads increase, although it can be seen that the relation is not linear as is somewhat erratic. For example, we can see for smaller numbers (like 100, 1000) the relation is close to linear while for larger numbers (like 1 Lakh and 1 Million) there seems to be a general increasing trend of time taken per increase in threads, but as can be seen from the graph higher values are reaching towards a plateau where the time taken either flat-lines, or decreases.

Plots for Process P2

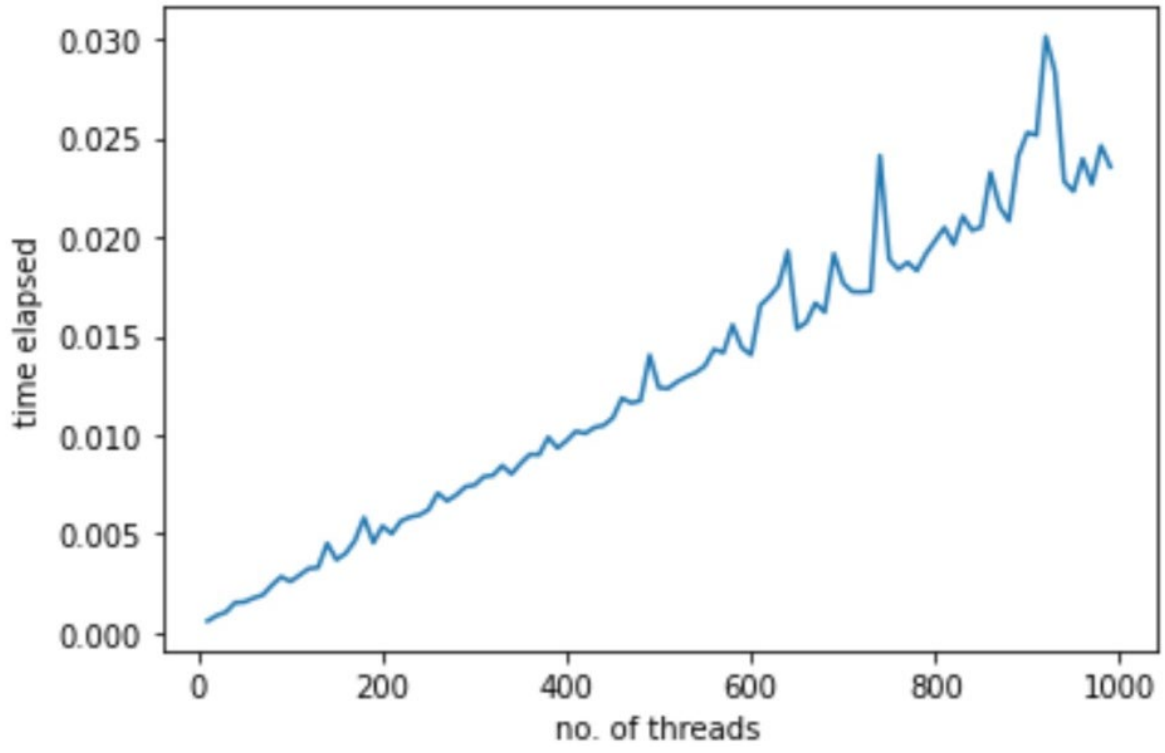
100 Numbers

Number of Threads	Time Taken
1	0.000128
5	0.000684
10	0.000465
15	0.000516
20	0.0006
25	0.000689
30	0.000787
35	0.000925
40	0.000987
45	0.00109
50	0.001264
55	0.001325
60	0.001498
65	0.001687
70	0.001639
75	0.001747
80	0.002382
85	0.002458
90	0.002215
95	0.002331



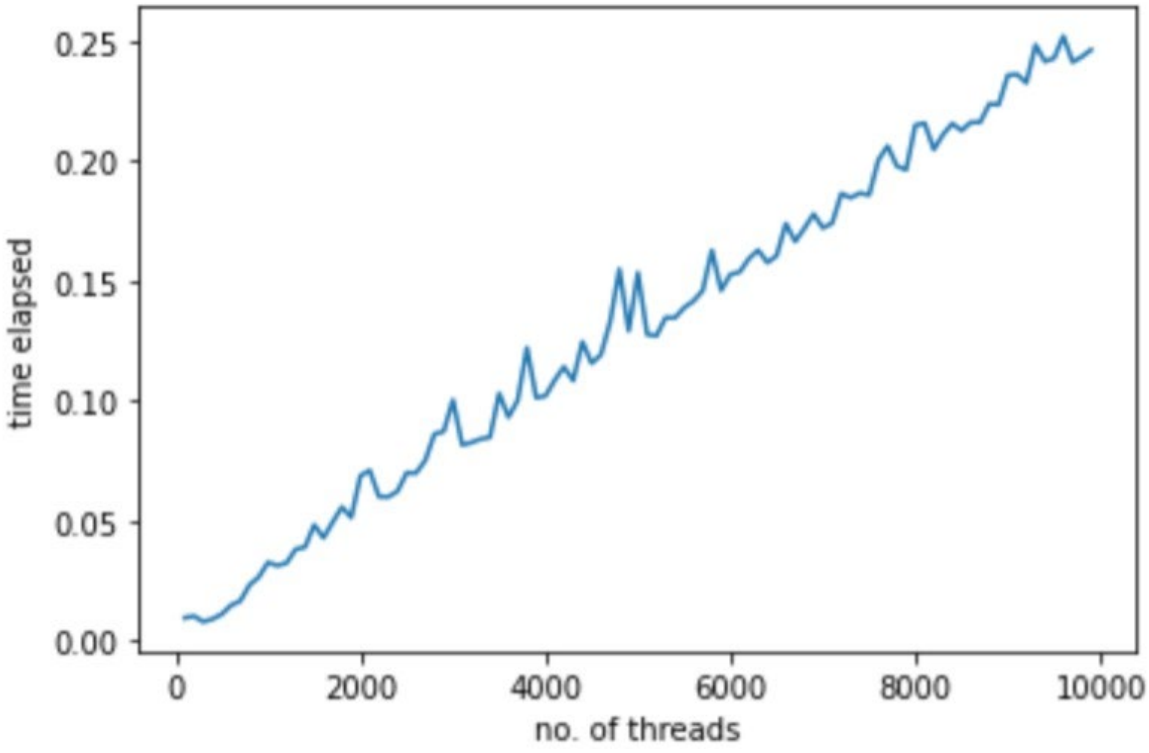
1000 Numbers

1	0.000122
10	0.000603
20	0.000882
30	0.001055
40	0.001533
50	0.001565
60	0.001767
70	0.001922
80	0.002415
90	0.002843
100	0.002601
110	0.002924
120	0.003258
130	0.003309
140	0.004528
150	0.003699
160	0.004025
170	0.004658
180	0.005828
190	0.00456
200	0.005399



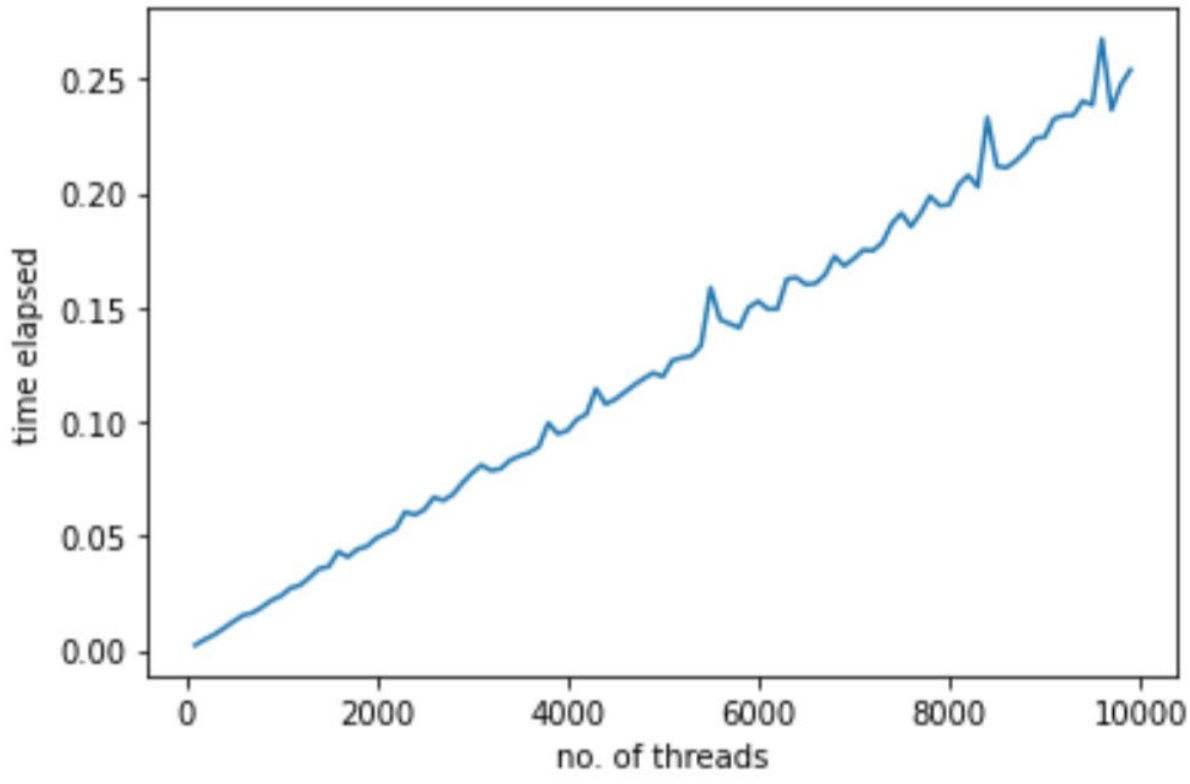
1 Lakh Numbers

1	0.000215
100	0.009687
200	0.010389
300	0.008037
400	0.009195
500	0.0112
600	0.014902
700	0.016678
800	0.02337
900	0.02666
1000	0.032748
1100	0.031444
1200	0.032495
1300	0.038255
1400	0.039118
1500	0.048424
1600	0.042959
1700	0.049579
1800	0.055831
1900	0.051466
2000	0.06888



1 Million Numbers

1	0.000129
100	0.002243
200	0.00472
300	0.00685
400	0.009604
500	0.012528
600	0.015253
700	0.016346
800	0.018865
900	0.021862
1000	0.023807
1100	0.027192
1200	0.028499
1300	0.031861
1400	0.035737
1500	0.036619
1600	0.043128
1700	0.040801
1800	0.044162
1900	0.045483
2000	0.049115

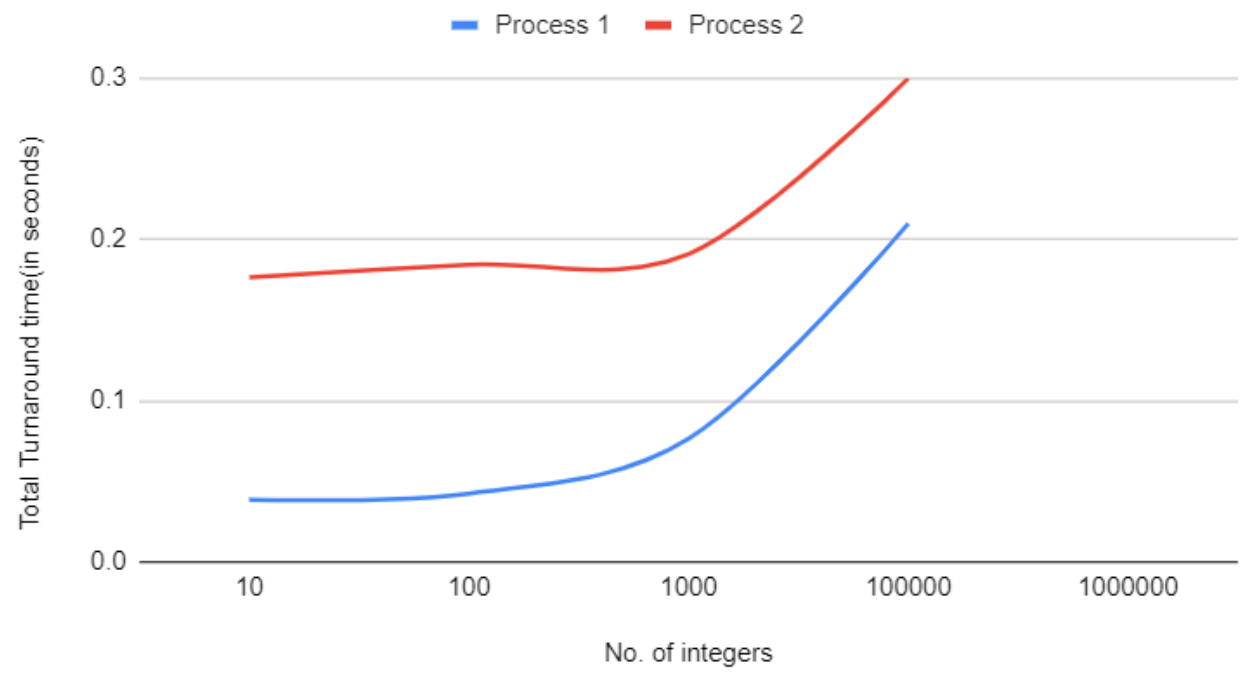


Analysis:

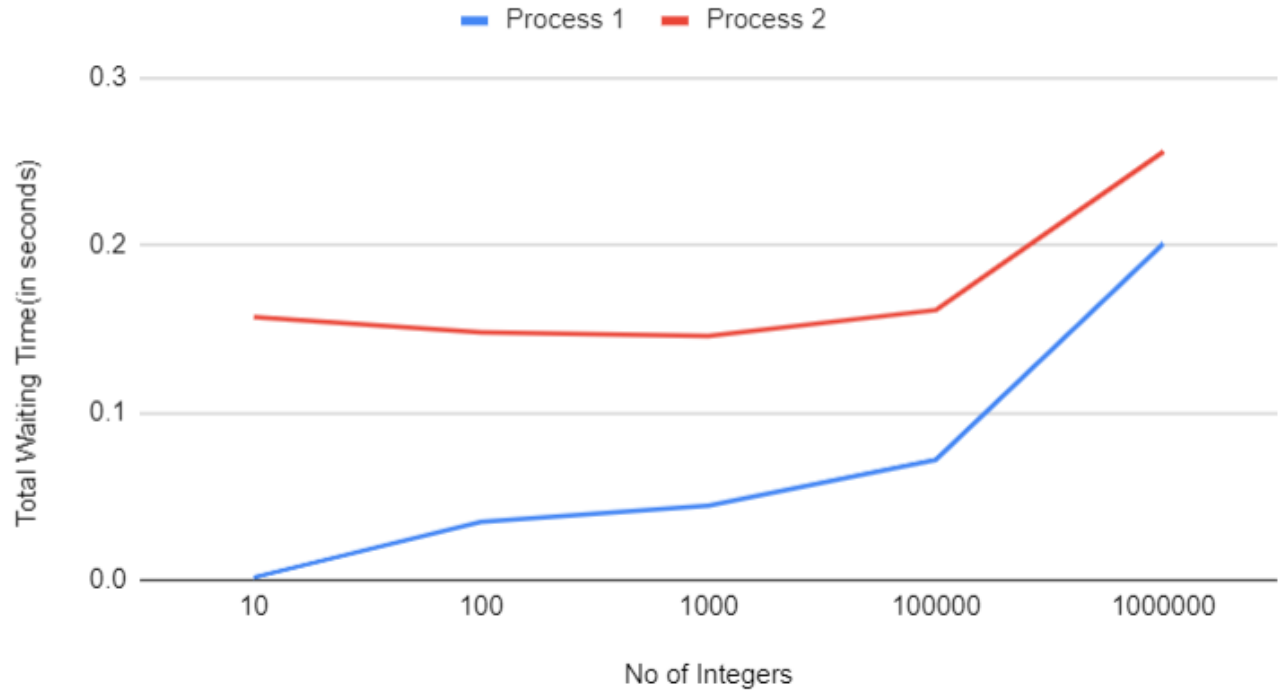
As we can see the read time from the shared memory increases as the number of threads increase, the time taken to complete the task also increases. The relation is very close to linear, and it only deviates at certain points which are depicted as spikes in time in the general increasing trend. The plots also show that if number of threads stays constant, the time decreases with an increase in numbers to sum per thread.

Plots for Process P3: Pre-emptive Priority Scheduling

Total Turnaround Time vs No of Integers



Waiting Time vs No of Integers



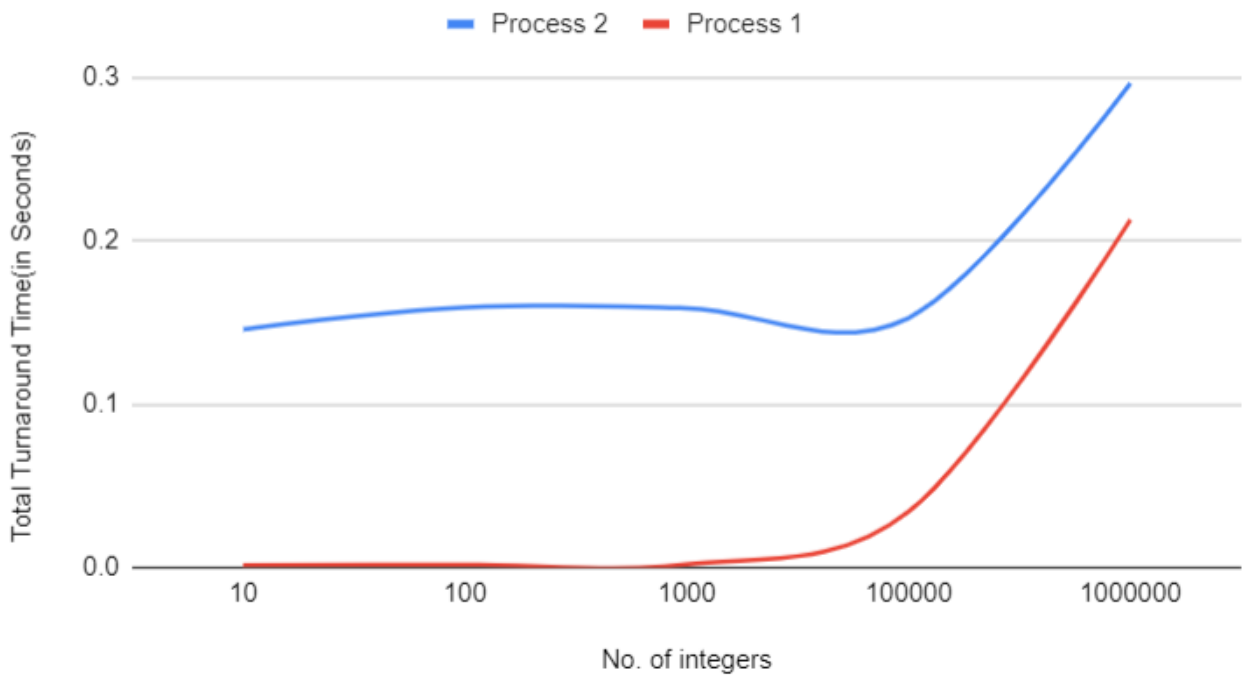
Analysis:

As we can see the “Turn Around Time” of Process 1 is always lesser than Process 2 assuming the number of integers remain the same. We can also notice that the “Turn Around Time” till about 800 integers remains plateaued and then increases as the number of integers increase. This relation is very close to linear. We can also notice a dip in the “Turn Around Time” taken for about 800 numbers.

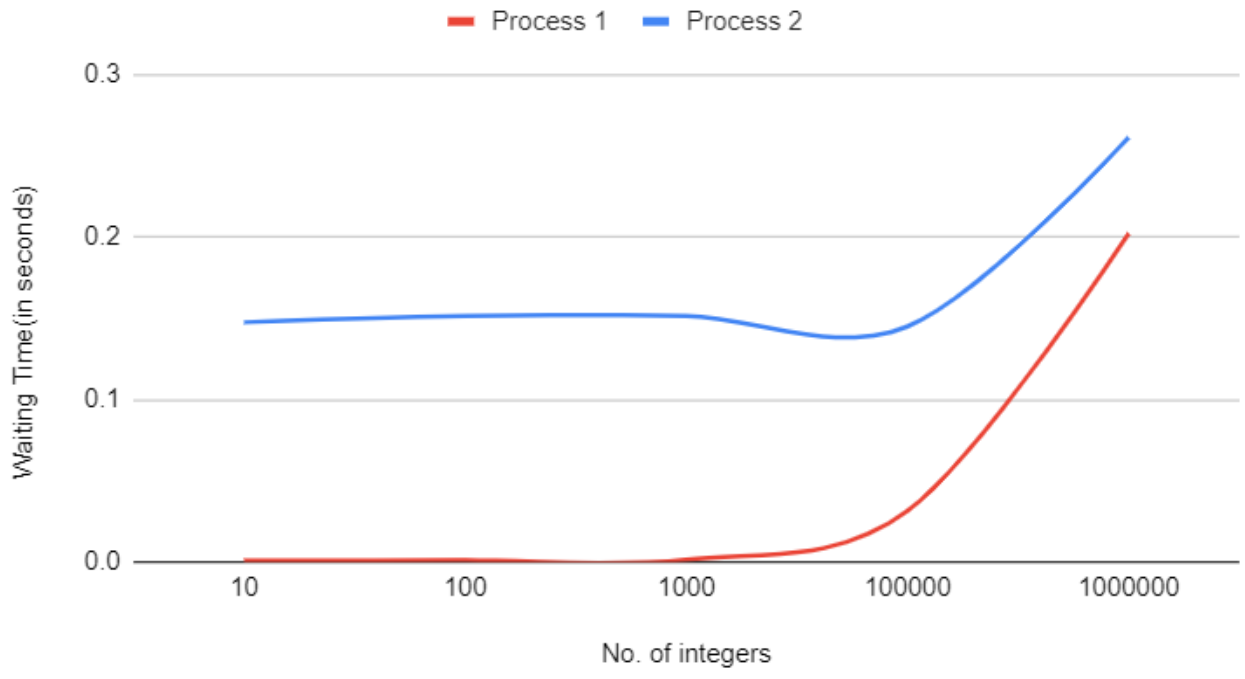
As we can see the “Waiting Time” of Process 1 is also less than that of Process 2 assuming the number of integers remain the same. We can also notice that the graph is very semi-linear as it is always increasing when the amount of integers increases, but it also shows erratic slopes. For example, Process 1 at 100 integers it almost plateaus but it shoots up again at 100000 integers, whereas Process 2 till about a 100000 integers it remains about constant, and then increases linearly.

Plots for Process P3: Round Robin Scheduling

Total Turnaround Time vs No of Integers



Waiting Time vs No of Integers



Analysis:

As we can see the “Turn Around Time” of Process 1 is always lesser than Process 2 assuming the number of integers remains the same. We can also notice that the “Turn Around Time” till about 8000 integers remains plateaued and then increases as the number of integers increase. This relation is very close to linear. We can also notice a dip in the “Turn Around Time” taken for about 8000 numbers.

As we can see the “Waiting Time” of Process 1 is always lesser than Process 2 assuming the number of integers remains the same. We can also notice that the “Waiting Time” till about 8000 integers remains plateaued and then increases as the number of integers increase. This relation is very close to linear. We can also notice a dip in the “Waiting Time” taken for about 8000 numbers.