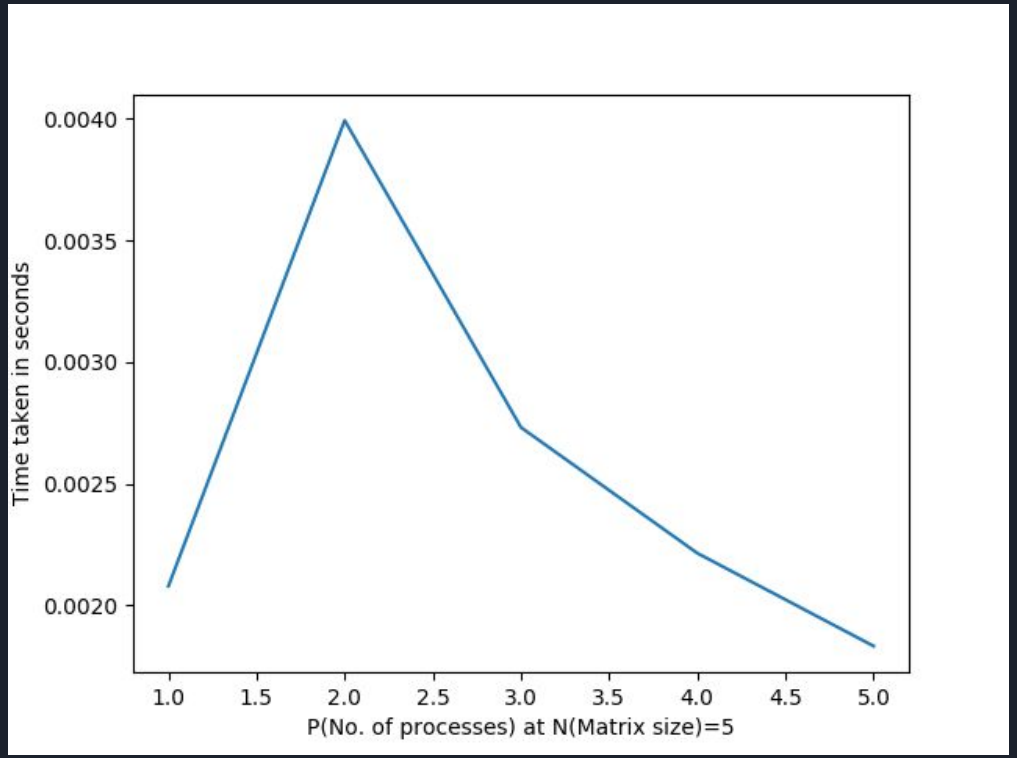
Assignment 2(OS)

KANISH GARG
2016EE10446

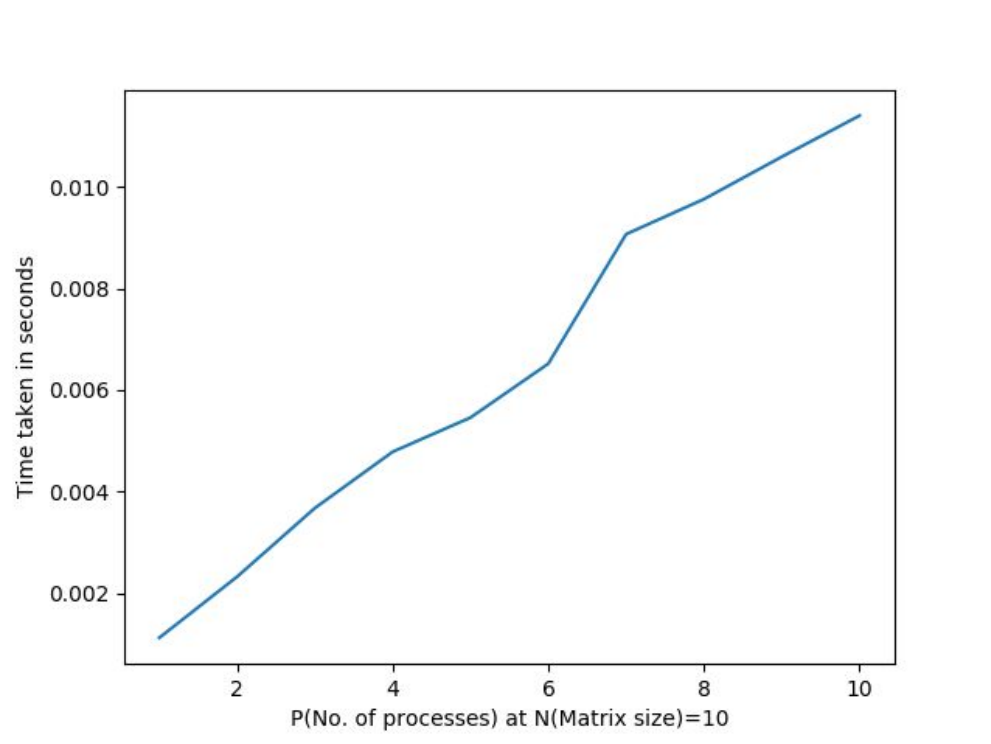
Part3 Observations

1. Plots of performance of Jacobi vs the number of processes at different Matrix sizes

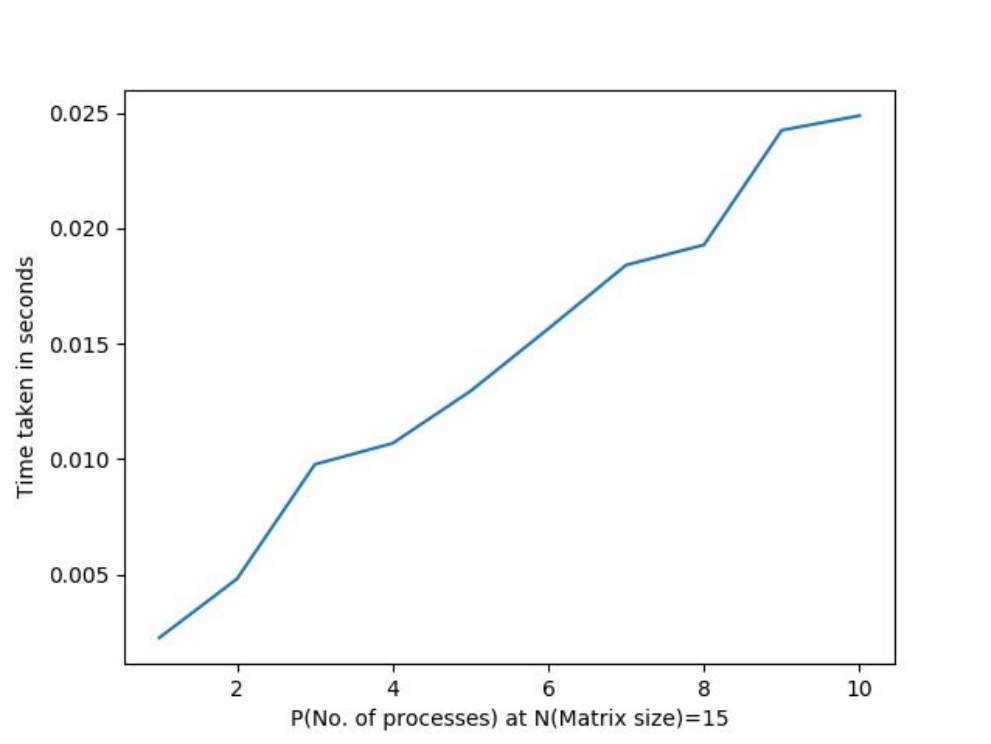
N=5


0.0040
0.0035
0.0030
0.0025
0.0020
Time taken in seconds
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
P(No. of processes) at N(Matrix size)=5

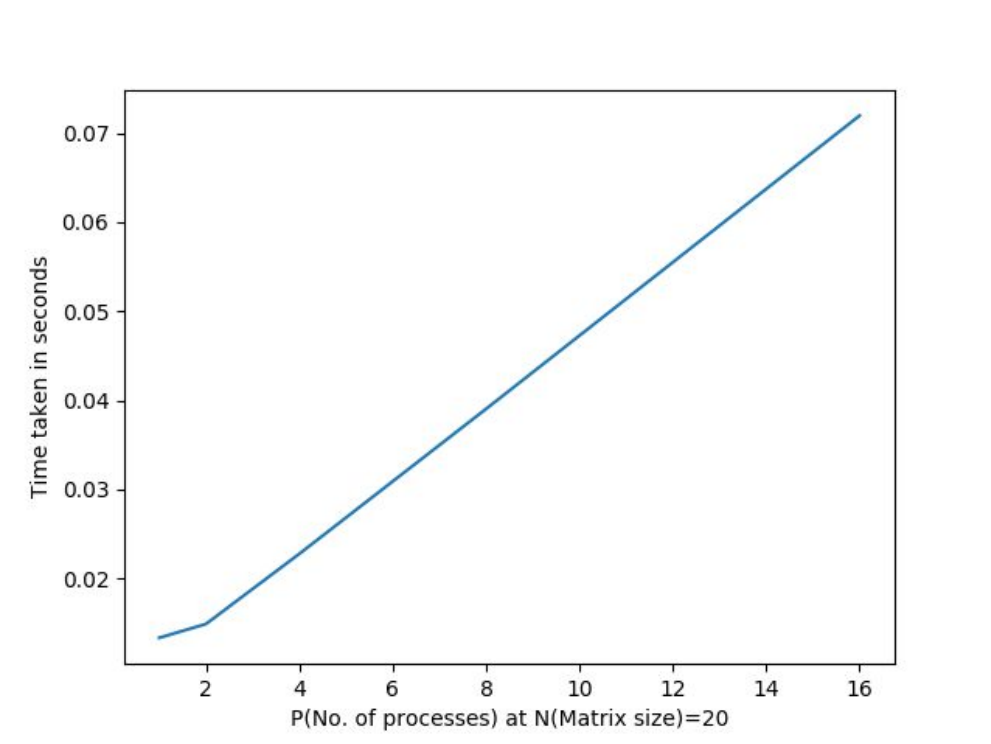
N=10


0.010
0.008
0.006
0.004
0.002
Time taken in seconds
2
4
6
8
10
P(No. of processes) at N(Matrix size)=10

N=15


0.025
0.020
0.015
0.010
0.005
Time taken in seconds
2
4
6
8
10
P(No. of processes) at N(Matrix size)=15

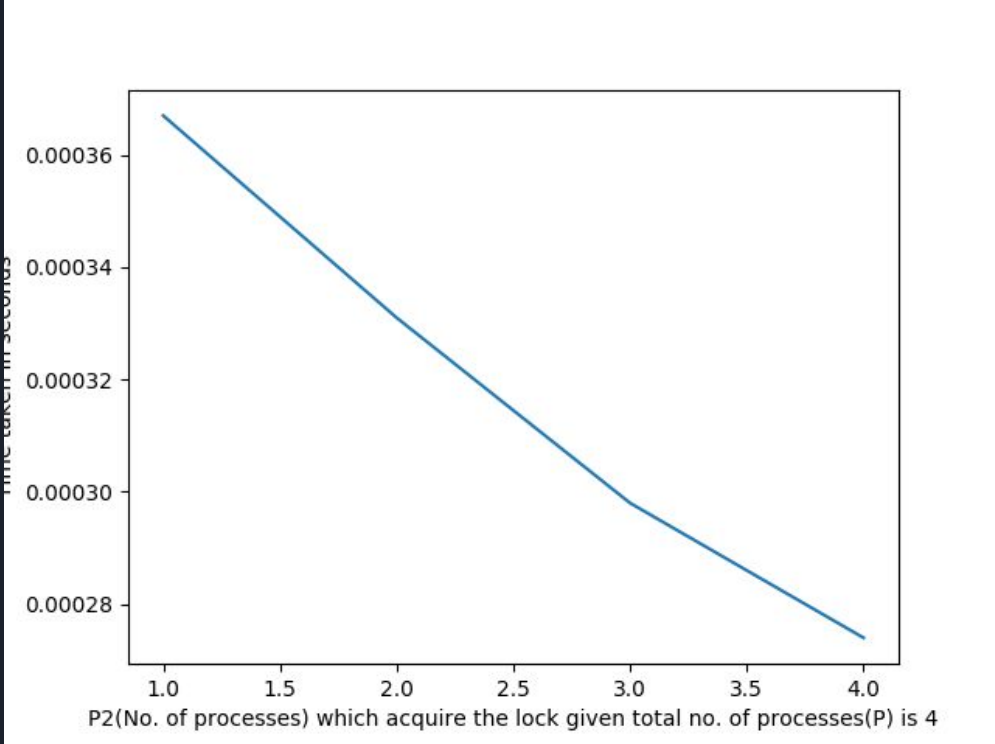
N=20


0.07
0.06
0.05
0.04
0.03
0.02
Time taken in seconds
2
4
6
8
10
12
14
16
P(No. of processes) at N(Matrix size)=20

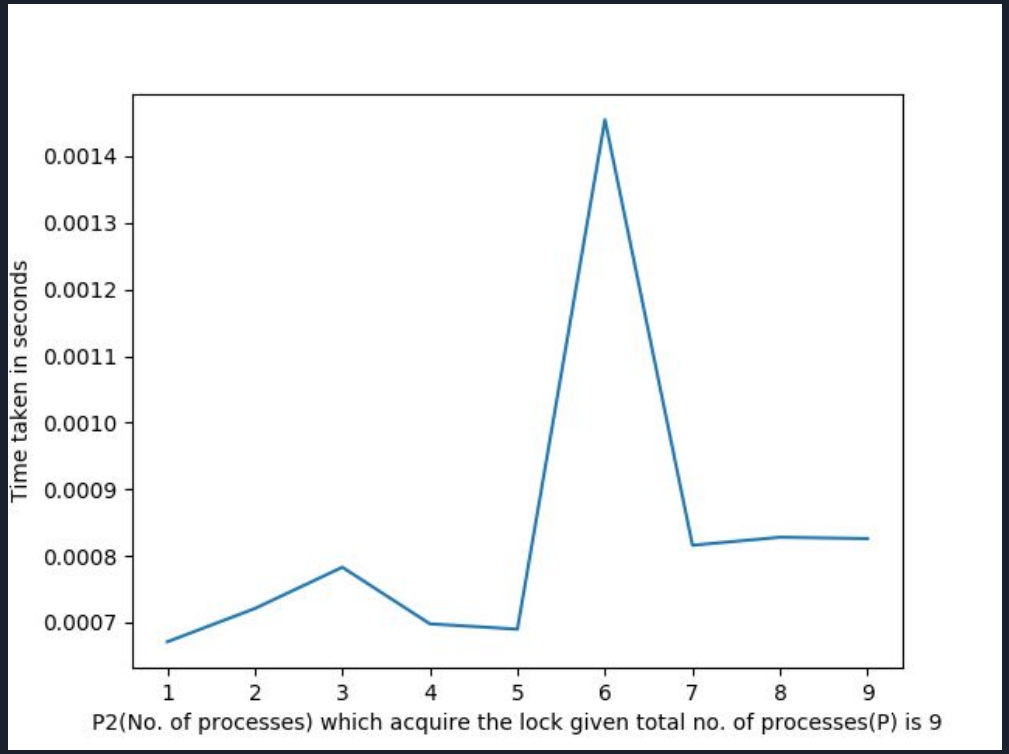
- For large values on N as we increase the number of processes, time taken by program increases hence performance decreases because interprocess communication increases.
- Overall performance of program depends both on IPC time and computation time of each iteration of jacobi.
- For N=5 we can can see that performance increases with increase in no. of processes. Here Computation time matters. IPC time is relatively low because of small N.

2. Plots of the total time taken by the program by varying the number of processes which acquire the lock(P2) out of total no. of processes(P) at different P.

P=4


0.00036
0.00034
0.00032
0.00030
0.00028
1.0
1.5
2.0
2.5
3.0
3.5
4.0
P2(No. of processes) which acquire the lock given total no. of processes(P) is 4

P=9


Time taken in seconds
0.0014
0.0013
0.0012
0.0011
0.0010
0.0009
0.0008
0.0007
1
2
3
4
5
6
7
8
9
P2(No. of processes) which acquire the lock given total no. of processes(P) is 9

Correctness of program

- For Jacobi I compared the output of my programs both in linux and xv6 with that of the given non parallel jacob.c file for different values of inputs.
- I stored both the outputs in a file and wrote a small python script to compare them.
- For maekawa, In the critical section I put a system call which increments the global int declared in proc.c by 1 and return it. I then verified that whether every process gets the correct value of that variable in the order in which they acquired locks.