CS546 Parallel and distributed Processing
Programming Assignment
Performance Evaluation

The assignment carries out Evaluation of the serial code and Parallel algorithms Pthreads and MPI. These parallel programs were capable of achieving more speedup, when compared with the speedup of the serial code hence supporting the fact that the parallel algorithms can provide great efficiency when compared to serial code.
I've evaluated the elapsed time taken to Gauss elimination without pivoting using PThreads and MPI up to 8 process on local machine and 32 processes on Jarvis.

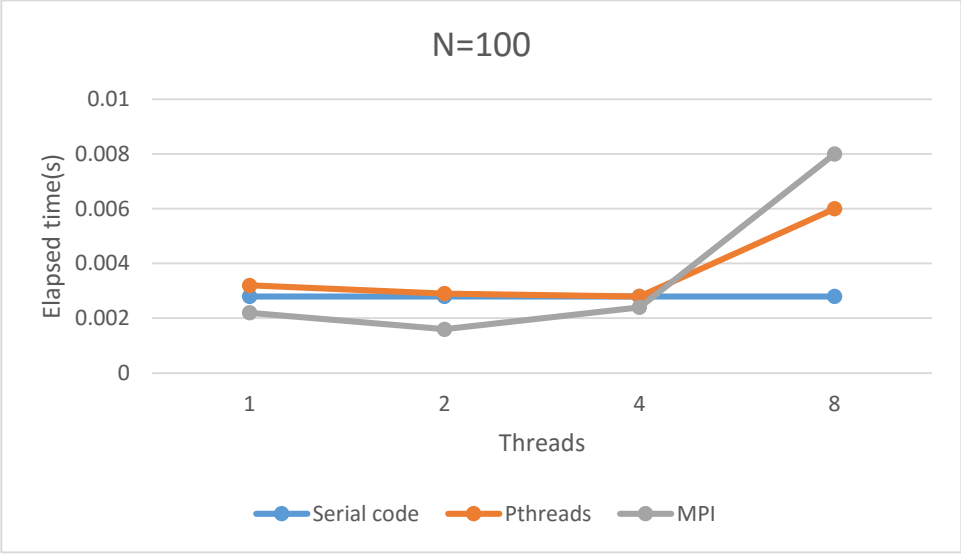
The below tables and plots for each matrix size.The plots were drawn basedon elapsed time and number of processes(threads).

1. Matrix Size =100

Local Machine:

	1	2	4	8
Serial code	0.0028	0.0028	0.0028	0.0028
Pthreads	0.0032	0.0029	0.0028	0.006
MPI	0.0022	0.0016	0.0024	0.008

Graph:



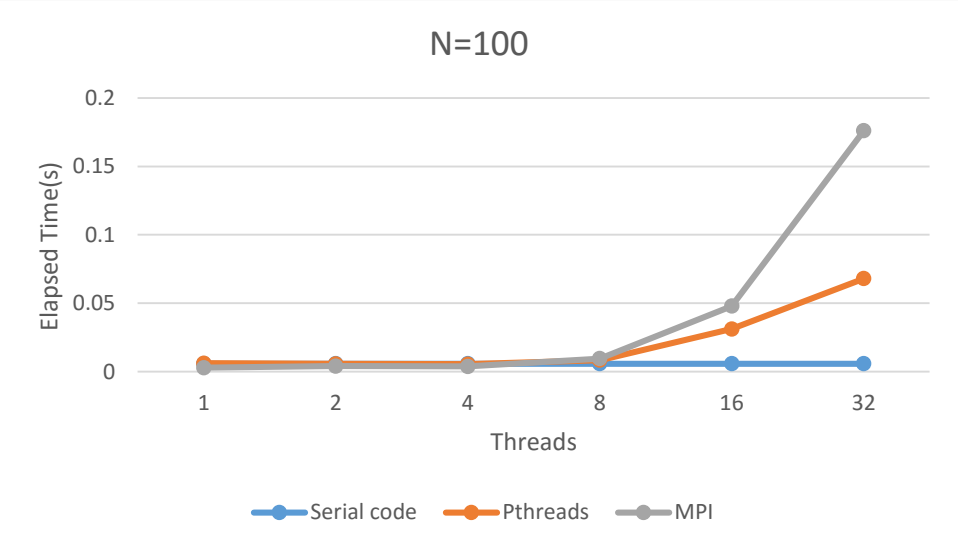
Conclusion: Based on the above data (N=100) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 4 threads. Then, it decreases because as number of threads increases, performance would decreases at certain point because of

inter thread communication on local machine since local machine have less cores and amount of time to perform the backwards substitution step is minimal when compared to the forward elimination in MPI.

Jarvis:

	1	2	4	8	16	32
Serial code	0.0058	0.0058	0.0058	0.0058	0.0058	0.0058
Pthreads	0.0062	0.0057	0.0055	0.0084	0.0312	0.068
MPI	0.0029	0.0041	0.0038	0.0095	0.0479	0.176

Graph:



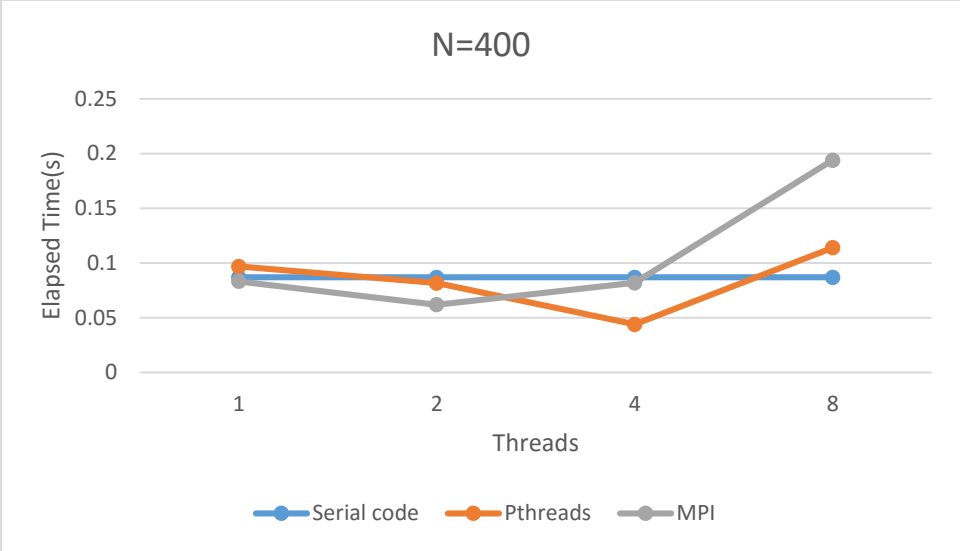
Conclusion: Based on the above data (N=100) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 8 threads. Then, it decreases because of inter process communication on Jarvis, but MPI doesn't give better performance as expected, it gives only up to 8 threads because distribution of work took more time.

2. Matrix Size =400

Local Machine:

	1	2	4	8
Serial code	0.087	0.087	0.087	0.087
Pthreads	0.097	0.0818	0.044	0.114
MPI	0.0833	0.062	0.082	0.194

Graph:

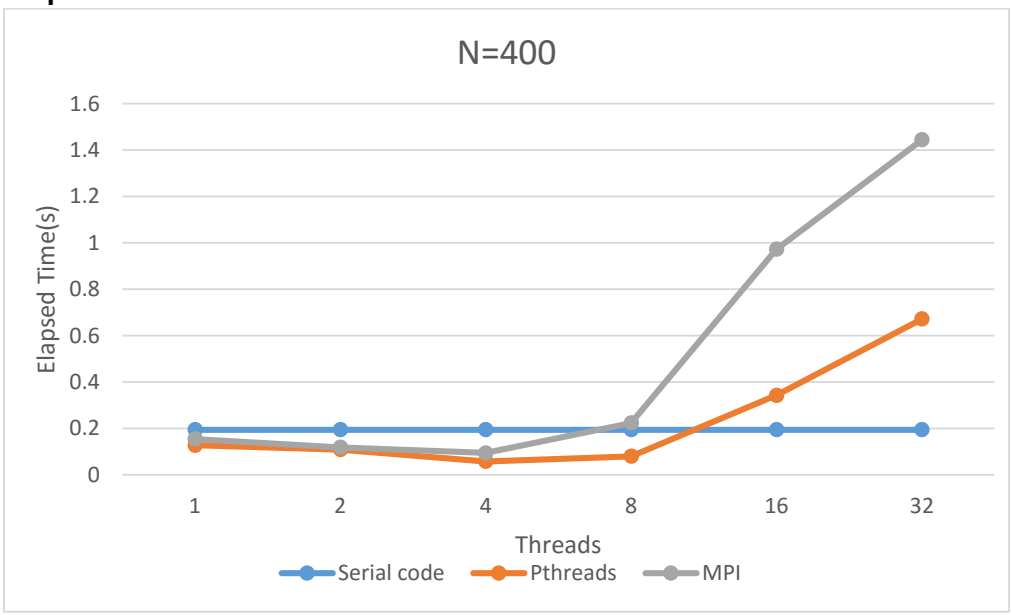


Conclusion: Based on the above data (N=400) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 4 threads. Then, it decreases because as number of threads increases, performance would decreases at certain point because of inter thread communication on local machine

Jarvis:

	1	2	4	8	16	32
Serial code	0.194	0.194	0.194	0.194	0.194	0.194
Pthreads	0.127	0.108	0.0575	0.0791	0.342	0.671
MPI	0.154	0.118	0.094	0.223	0.972	1.444

Graph:



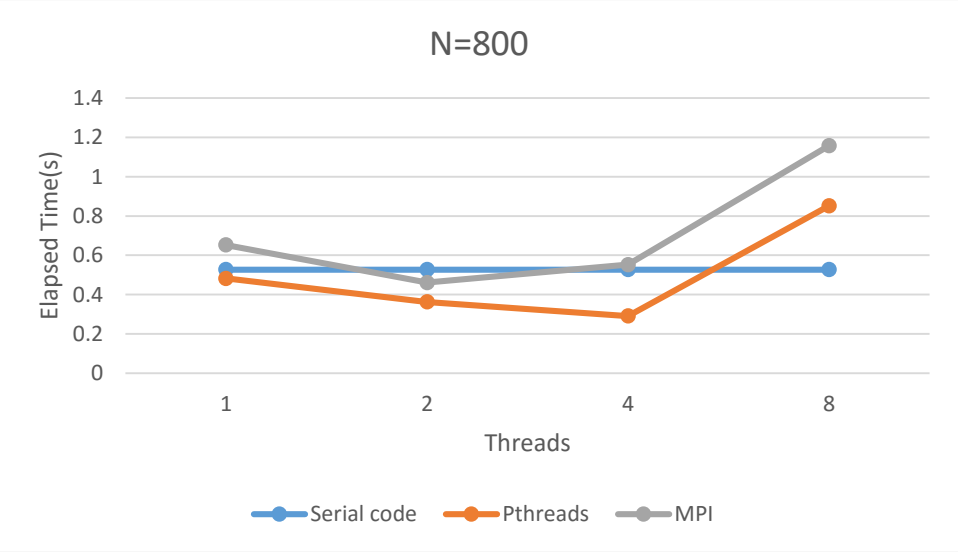
Conclusion: Based on the above data (N=400) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 8 threads. Then, it decreases because distribution of work among processes tooks more time.

3. Matrix Size =800

Local Machine:

	1	2	4	8
Serial code	0.527	0.527	0.527	0.527
Pthreads	0.482	0.363	0.291	0.852
MPI	0.652	0.461	0.552	1.158

Graph:

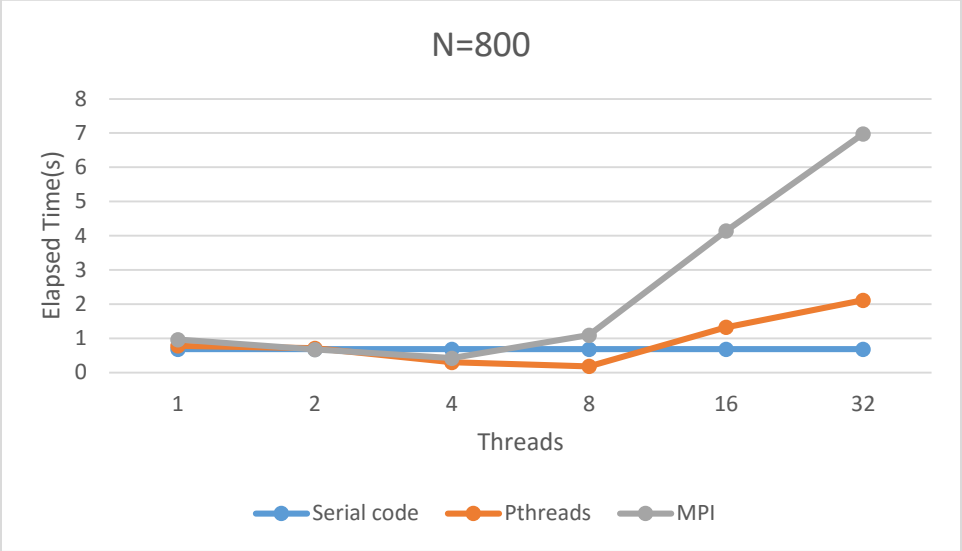


Conclusion: Based on the above data (N=800) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 4 threads. And as matrix size increases, more work distribution needed, it results decreases in performance.

Jarvis:

	1	2	4	8	16	32
Serial code	0.684	0.684	0.684	0.684	0.684	0.684
Pthreads	0.782	0.712	0.3013	0.1829	1.323	2.112
MPI	0.964	0.674	0.419	1.094	4.141	6.98

Graph:



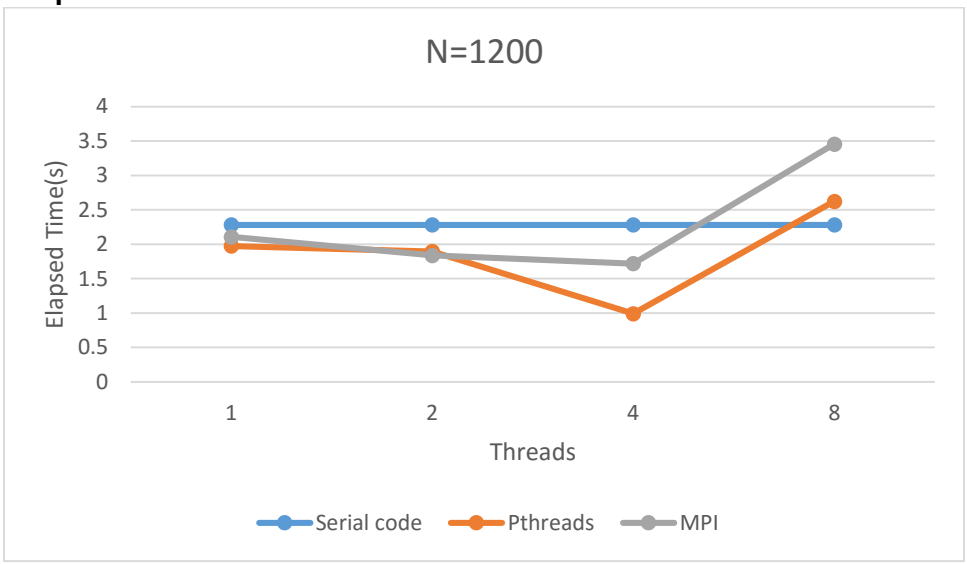
Conclusion: Based on the above data (N=800) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 8 threads. Then, it decreases because distribution of work took more time in MPI.

4. Matrix Size =1200

Local Machine:

	1	2	4	8
Serial code	2.28	2.28	2.28	2.28
Pthreads	1.975	1.895	0.99	2.622
MPI	2.106	1.836	1.719	3.456

Graph:

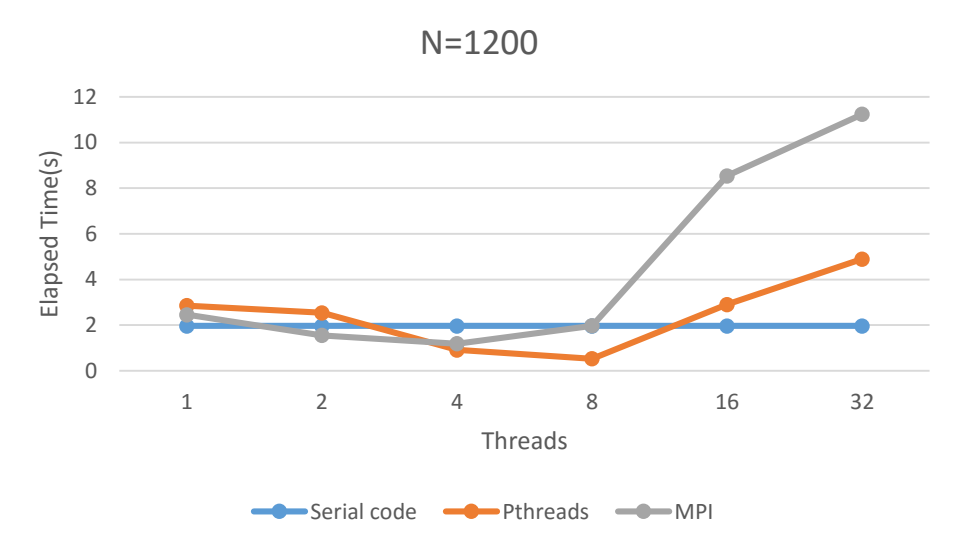


Conclusion: Based on the above data (N=1200) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 4 threads. Then, it decreases because as number of threads increases, performance would decreases at certain point because of inter thread communication on local machine.

Jarvis:

	1	2	4	8	16	32
Serial code	1.965	1.965	1.965	1.965	1.965	1.965
Pthreads	2.854	2.537	0.909	0.528	2.901	4.898
MPI	2.451	1.547	1.185	1.967	8.54	11.241

Graph:



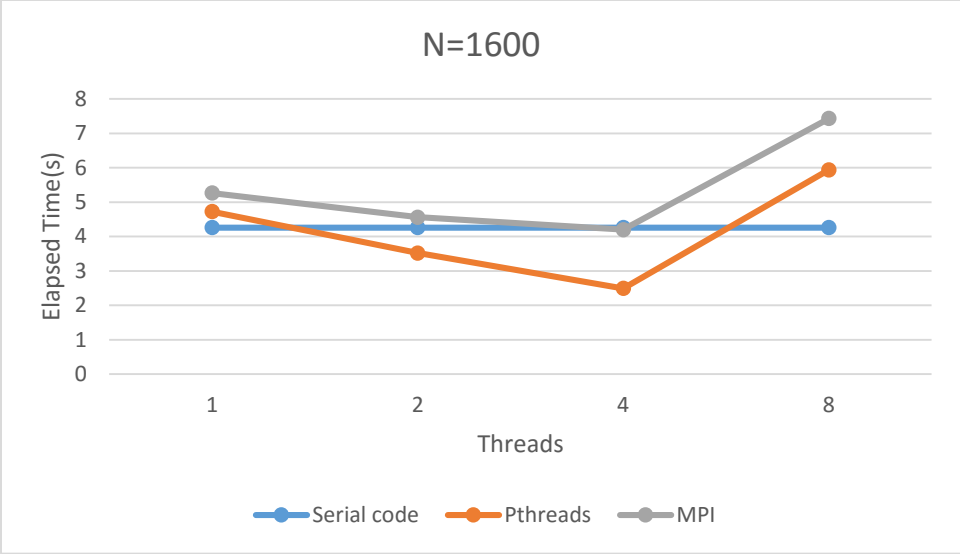
Conclusion: Based on the above data (N=1200) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 8 threads. Then, it decreases because of communication between processes, but MPI doesn't give better performance as expected.

5. Matrix Size =1600

Local Machine:

	1	2	4	8
Serial code	4.26	4.26	4.26	4.26
Pthreads	4.723	3.519	2.495	5.94
MPI	5.264	4.564	4.2	7.4314

Graph:

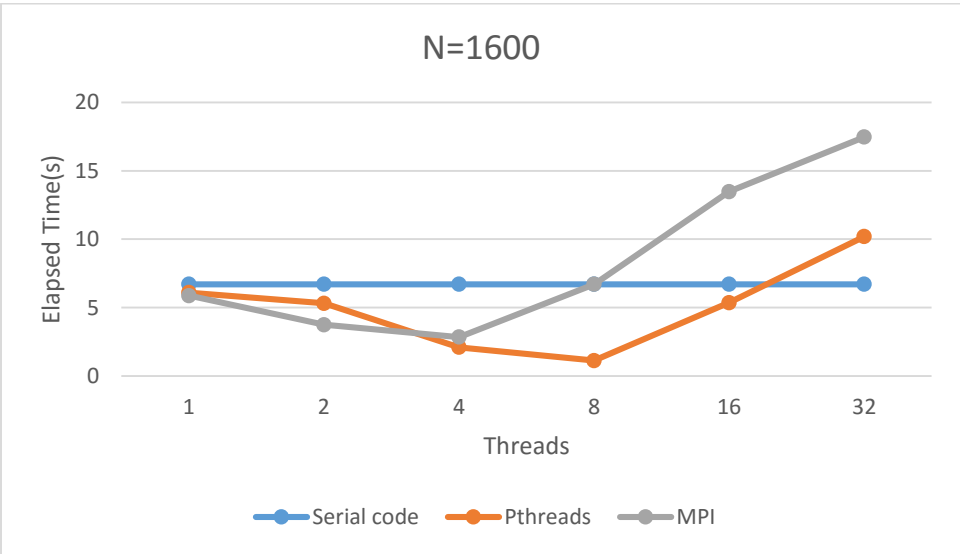


Conclusion : Based on the above data (N=1600) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 4 threads. Then, it decreases because of communication.

Jarvis:

	1	2	4	8	16	32
Serial code	6.712	6.712	6.712	6.712	6.712	6.712
Pthreads	6.104	5.32	2.103	1.131	5.36	10.186
MPI	5.873	3.745	2.843	6.712	13.475	17.458

Graph:



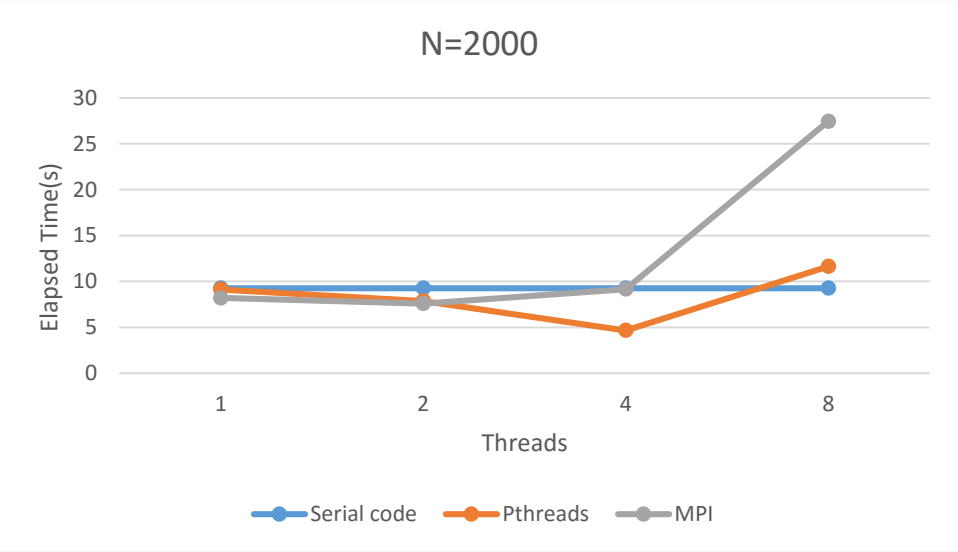
Conclusion: Based on the above data (N=1600) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 8 threads. Then, it decreases

because of communication on Jarvis, but MPI doesn't give better performance as expected, it gives only up to 8 threads.

6. Matrix Size =2000
Local Machine:

	1	2	4	8
Serial code	9.26	9.26	9.26	9.26
Pthreads	9.134	7.863	4.659	11.634
MPI	8.2	7.6	9.166	27.452

Graph:

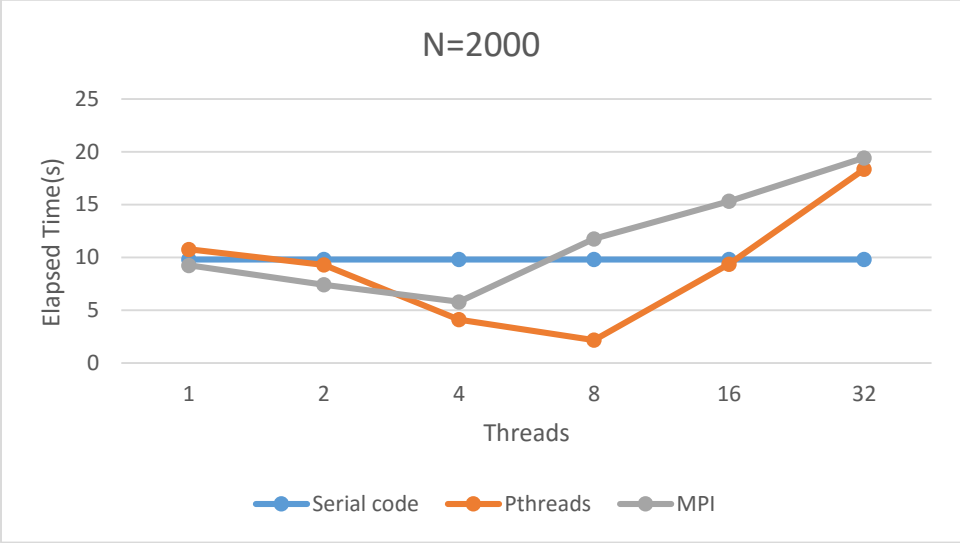


Conclusion: Based on the above data (N=2000) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 4 threads. Then, it decreases because as number of threads increases, performance would decreases at certain point because of inter thread communication on local machine since local machine have less cores.

Jarvis:

	1	2	4	8	16	32
Serial code	9.806	9.806	9.806	9.806	9.806	9.806
Pthreads	10.756	9.301	4.114	2.185	9.367	18.351
MPI	9.242	7.423	5.793	11.768	15.316	19.426

Graph:



Conclusion: Based on the above data (N=2000) above shown in the table, it is clearly seen that both the approaches performs better than the actual serial code up to 4 threads. Then, it decreases because of inter process communication on Jarvis, but MPI doesn't give better performance as expected, it gives only up to 4 threads.

Now, we need to find speedup and efficiency of these two algorithms.

$$S = \frac{T_{old}}{T_{new}}$$

Where, S - Speedup. T_{old} - Serial code execution time, T_{new} - Parallel code execution time.

And efficiency is ,

$$E_p = \frac{S_p}{p} = \frac{T_1}{pT_p}$$

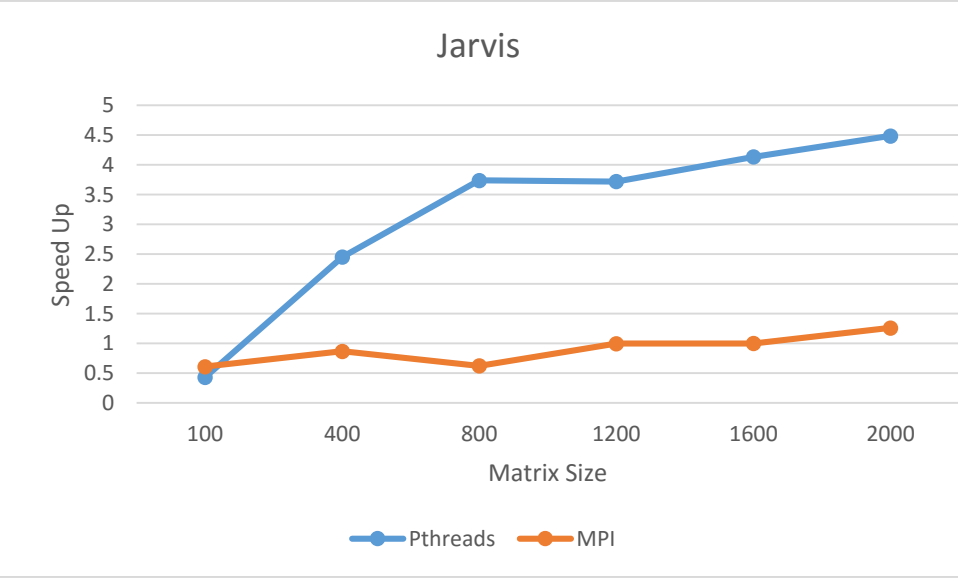
Where, S_p - SpeedUp and p - number processors/threads.

1.Jarvis : The below table shows that speedup and efficiency of Pthreads and Mpi for Jarvis,

number of processors are 8.

			Jarvis	np=8				
Serial Code	Pthreads	Speedup	Efficiency		Serial Code	MPI	Speedup	Efficiency
0.0058	0.0134	0.432836	5.410448		0.0058	0.0095	0.610526	7.631579
0.194	0.0791	2.452592	30.6574		0.194	0.223	0.869955	10.87444
0.684	0.1829	3.739748	46.74686		0.684	1.094	0.625229	7.815356
1.965	0.528	3.721591	46.51989		1.965	1.967	0.998983	12.48729
6.712	1.531	4.384063	54.80078		6.712	6.712	1	12.5
9.806	2.185	4.487872	56.0984		9.806	7.768	1.262358	15.77948

Graph:

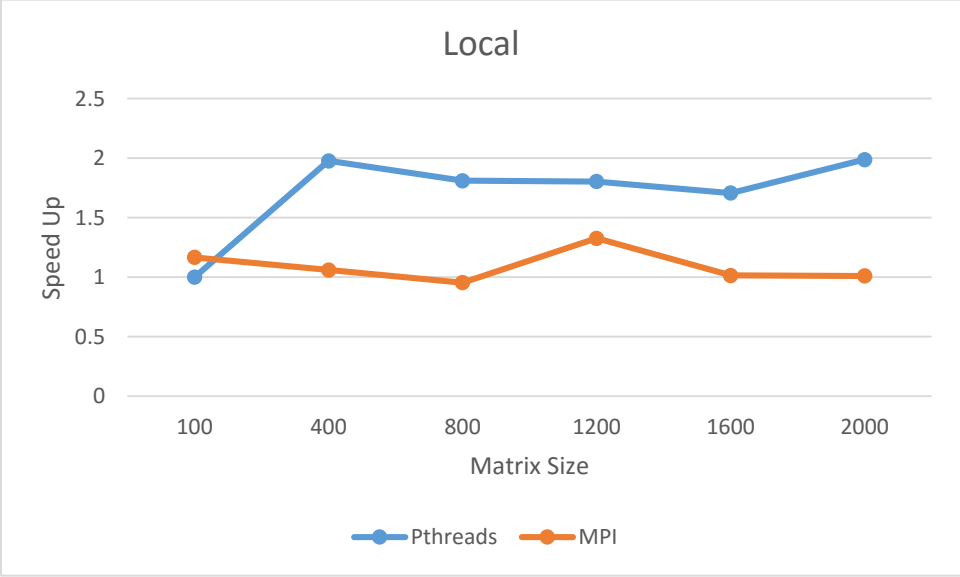


Conclusion: The speedup of the pthreads is increases when the size of the matrix increases and mpi is slowly increasing (very low) on jarvis .Out of these two parallel approaches, ptherads is better than mpi. similarly Efficiency is also increases in pthreads and mpi shows constant rise.

2. Local Machine : The below table shows that speedup and efficiency of Pthreads and Mpi for Local Machine and number of processors are 4.

			Local	np=4				
Serial Code	Pthreads	Speedup	Efficiency		Serial Code	MPI	Speedup	Efficiency
0.0028	0.0028	1	25		0.0028	0.0024	1.166667	29.16667
0.087	0.044	1.977273	49.43182		0.087	0.082	1.060976	26.52439
0.527	0.291	1.810997	45.27491		0.527	0.552	0.95471	23.86775
1.88	0.99	1.89899	47.47475		2.28	1.719	1.326353	33.15881
4.26	2.495	1.707415	42.68537		4.26	4.2	1.014286	25.35714
9.26	4.659	1.987551	49.68877		9.26	9.166	1.010255	25.25638

Graph :



Conclusion: The speedup of the pthreads and mpi is increases when the size of the matrix increases on Local machine .Out of these two parallel approaches, ptherads is better than mpi. similarly Efficiency is also increases in both pthreads and mpi .