Parallel Implementations of Matrix Multiplication Using MPI

Hardik Khichi
2016CS50404

Saransh Verma
2016CS10326

24th January, 2020

COL380: Introduction to Parallel and Distributed Programming - Assignment 2 Part 1

1 Algorithm Overview

The algorithm is very simple, to find a row in the product of two matrix A and B a row needs to be multiplied with each column of B, therefore our algorithm is to send a process all the columns of B and N/P rows of A, where N is the number of rows and P is the number of processes, then a process computes N/P of rows of the product say C, and sends it back to the parent process which assembles all such messages and generates the product of the matrices A and B, C. If A is of shape x x y and B is of shape y x z then the complexity of this algorithm is O(xyz) but here n is 32 and x and y are n so the complexity is O(n^2).

2 Parallel Algorithms

2.1 Sequential Program:

Matrix multiplication function: This is a nested loop which calculates C[i][j] by calculating the dot product of ith row of A and jth column of B;

2.2 Blocking P2P Communication

Blocking communication is done using MPI_Send() and MPI_Recv(). These function do not return (i.e they block) until the communication is complete, i.e. if MPI_Recv() is called first then it does not return till MPI_Send() is called and sends the message similarly if MPI_send() is called first it does not return until MPI_Recv() is called and it does not recieve the message. We send rows of A to every other process using MPI_Send(), which receive the data using MPI_Recv(). Up until this communication is done it blocks the process. Then similarly we send B. Then computation is done to computer rows of C and then C is sent to the original node again using above functions.

2.3 Non-Blocking P2P Communication

Non-Blocking communication is done using MPI_Isend() and MPI_Irecv(). These function return immediately, so when we need to insure that all the communication is complete we call MPI_Wait(), which waits till the communication is complete. The algorithm is similar we just change the function from MPI_Send() to MPI_Isend() and from MPI_Recv() to MPI_Irecv(). Then we use MPI_Wait() to ensure that the communication is complete and all the necessary data is received.

2.4 Collective Communication

Collective communication here uses three functions

- MPI_Bcast(): Broadcasts a message from the process with rank ”root” to all other processes of the communicator
- MPI_Scatter(): Sends chunks of an array to different processes.
- MPI_Gather(): Takes elements from many processes and gathers them to one single process.

We use MPI_Bcast() to send matrix B to all processes, then we use MPI_scatter to send chunks of matrix A to all processes, then we compute chunks of C and then we use MPI_Gather to collect those chunks into C.

3 Restrictions on N

- There is no restriction on N for blocking and Non-blocking communication programs. We have divided the workload effectively, such that, if N is not divisible by number of processors, last worker thread gets the work of remaining rows of the matrix multiplication. This ensure correctness of program.
- However in collective communication we need N to be divisible by the number of processes as MPI_Gather and MPI_Scatter use chunk size which needs to be the same which is only possible if N is divisible by P, if we want to use general N we will need to send extra messages using MPI_Send and MPI_Recv.

4 Running Times

4.1 Running times(in ms) for serial, blocking P2P, non-blocking P2P and collective communication for P = 2

Size(N)	Serial	Blocking P2P	Non Blocking P2P	Collective Communication
100	1.26	0.79	0.66	0.83
200	5.23	2.89	2.57	2.93
500	25.59	13.34	16.99	12.68
1000	96.60	48.66	47.76	50.31
2000	412.56	197.30	211.43	205.85
4000	1689.01	806.52	831.70	835.86
5000	2614.32	1379.85	1363.15	1419.03
7500	5709.35	3000.71	3049.05	3044.86
10000	11098.91	5514.55	5245.49	5714.92

4.2 Running times(in ms) for serial, blocking P2P, non-blocking P2P and collective communication for P = 4

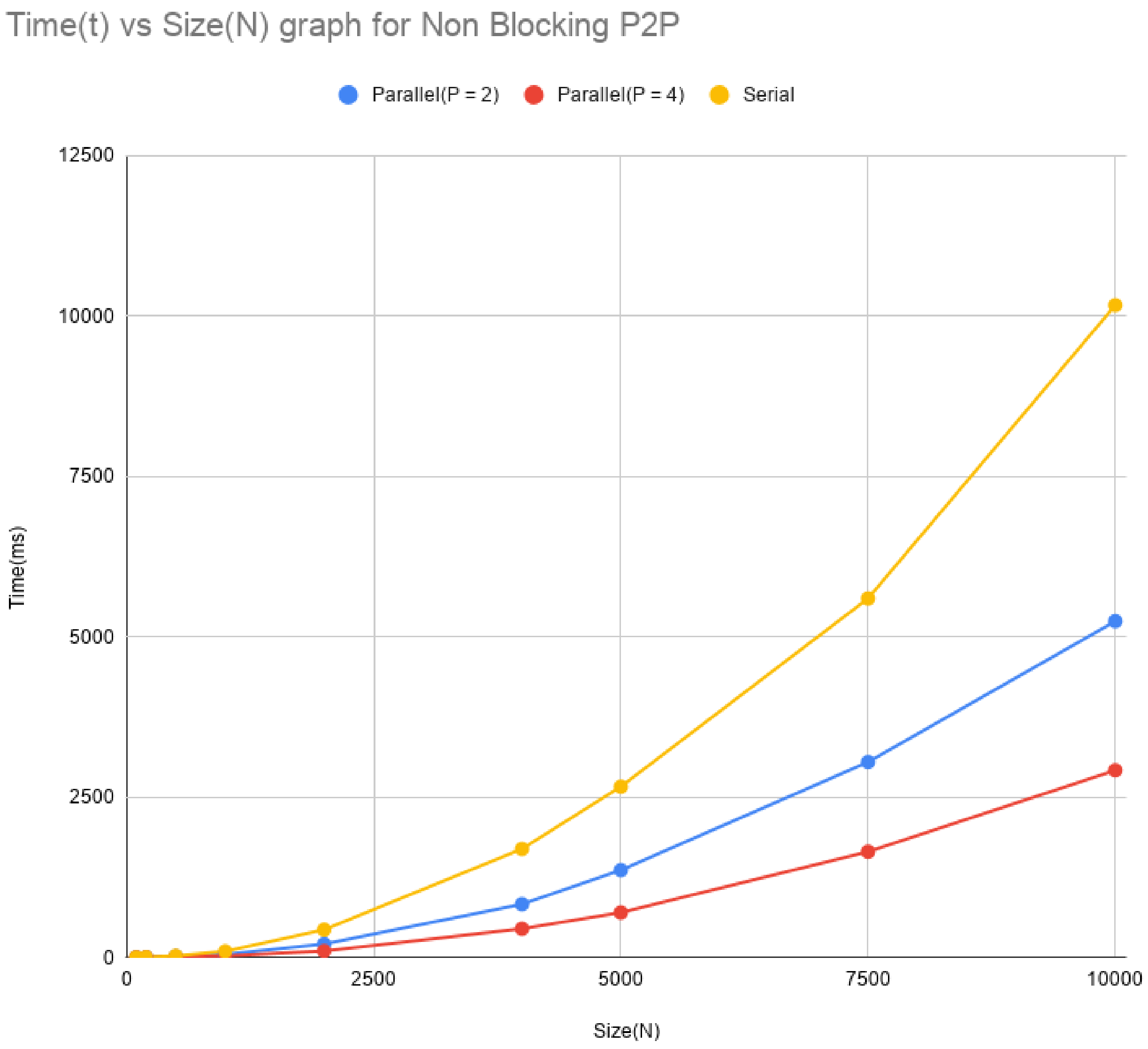
Size(N)	Serial	Blocking P2P	Non Blocking P2P	Collective Communication
100	1.26	0.52	0.38	0.69
200	5.23	1.69	2.59	1.88
500	25.59	6.82	11.80	7.24
1000	96.60	25.85	24.76	26.34
2000	412.56	102.09	101.35	102.43
4000	1689.01	445.00	449.41	434.51
5000	2614.32	729.12	702.16	721.12
7500	5709.35	1683.88	1650.31	1686.80
10000	11098.91	2983.39	2920.18	2949.97

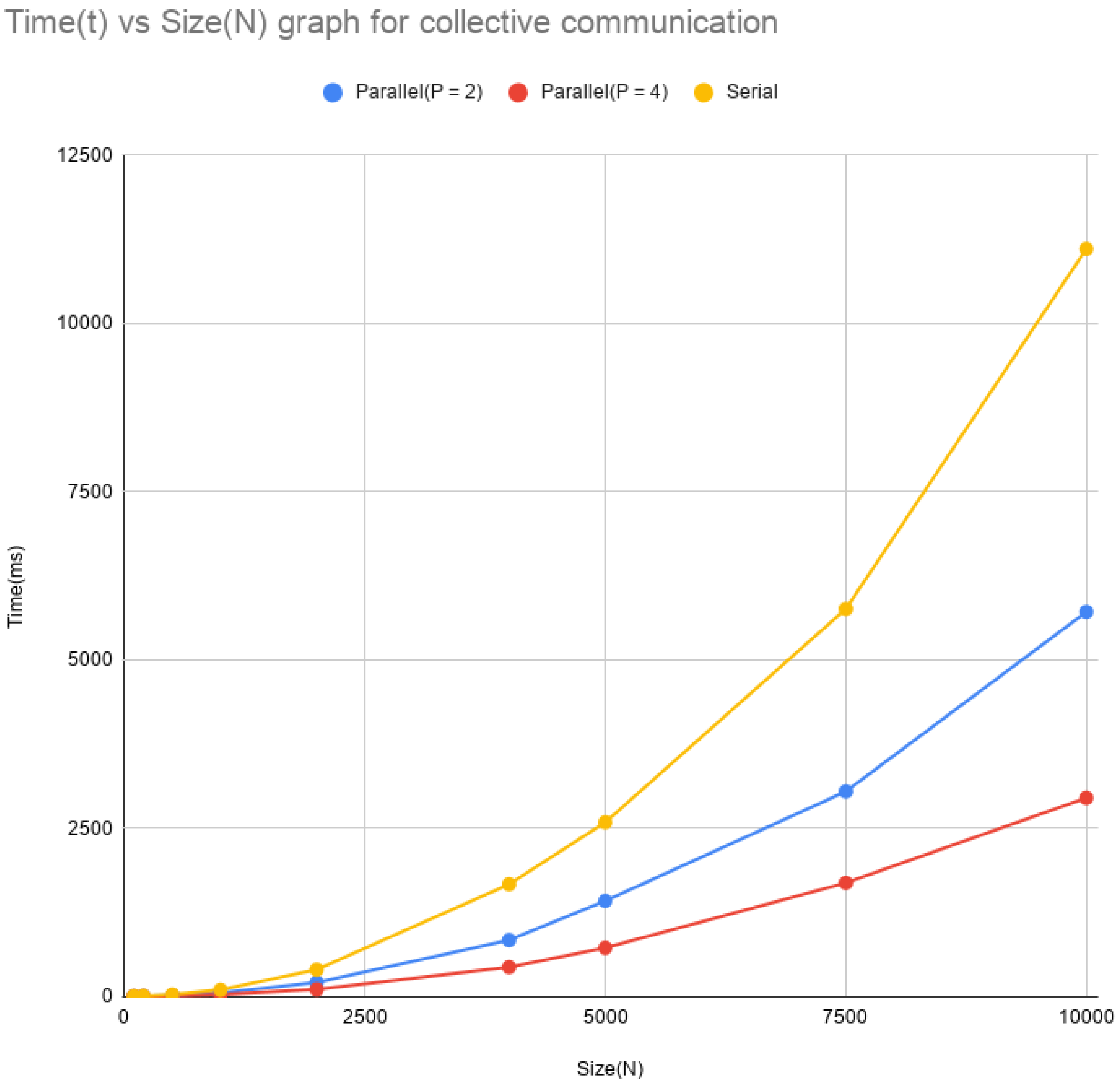
5 Obervations

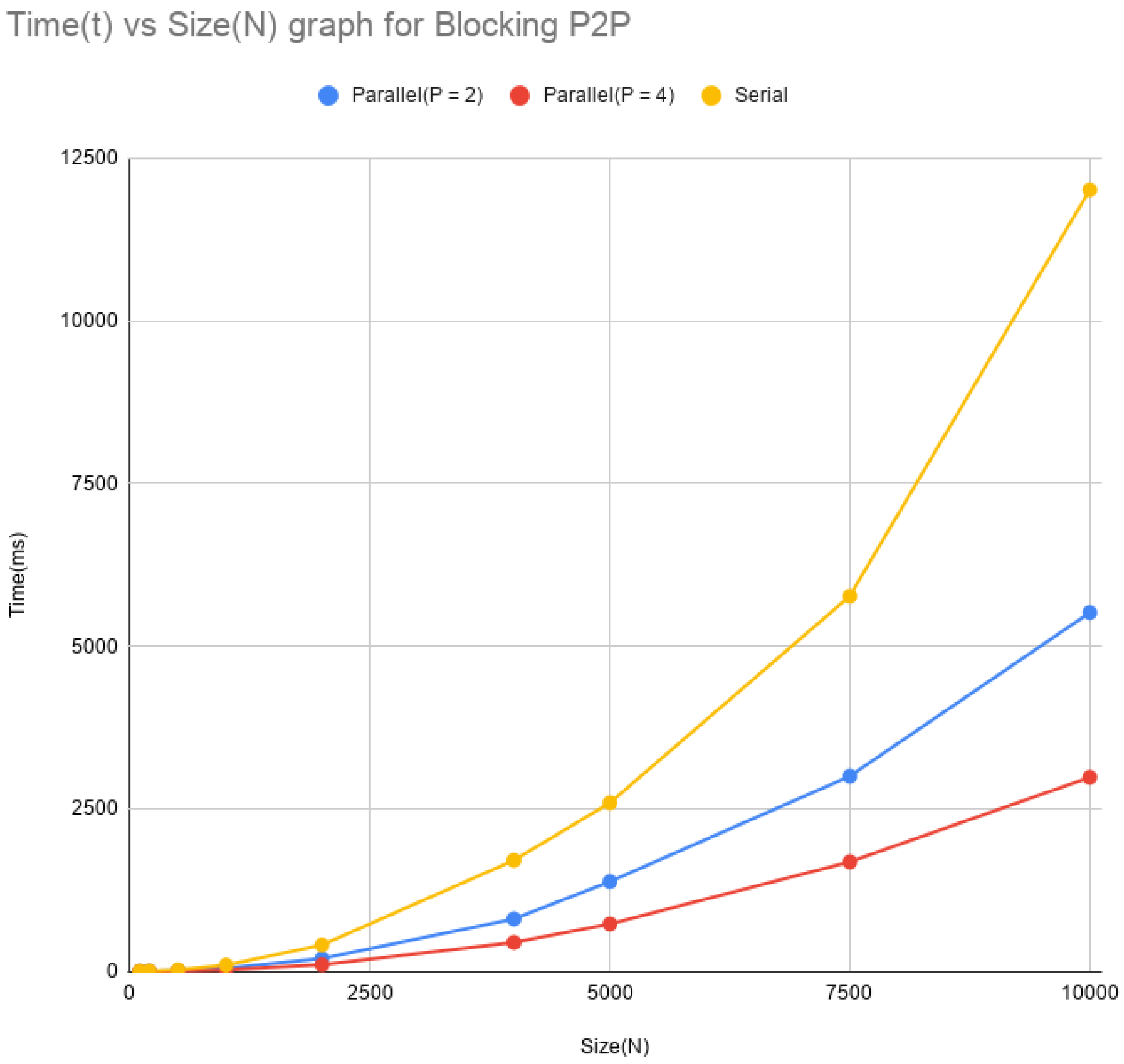
The following things were observed

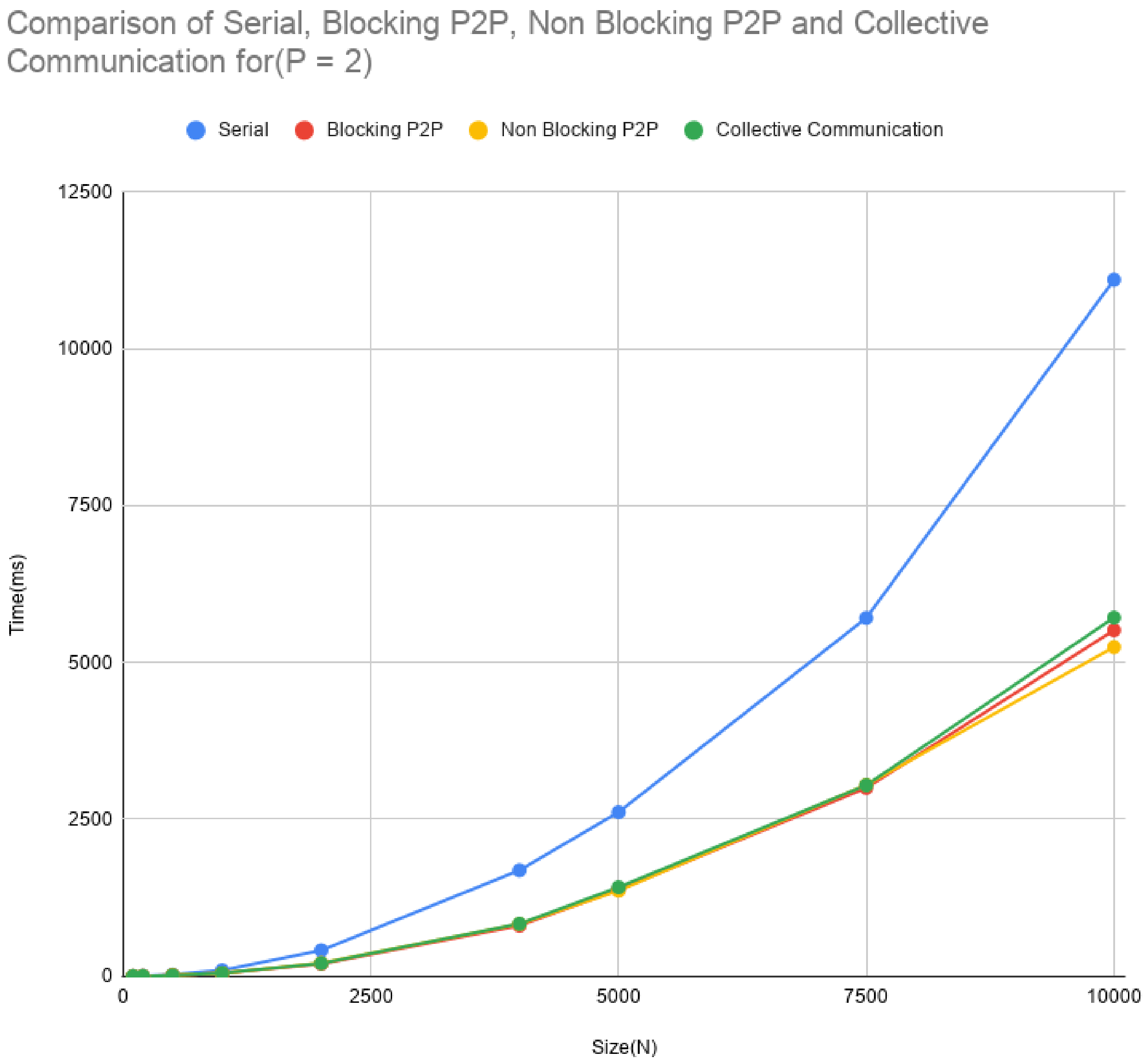
- Time vs size graph indicates O(n^2) complexity, as explained above.
- Non-Blocking P2P communication was the fastest because it required only one waiting as compared to many waits in blocing P2P.
- Trend between blocking P2P and collective communication is not clear. For example for problem size 10,000 blocking P2P performed better when P=2 but for P=4 collective communication was better.

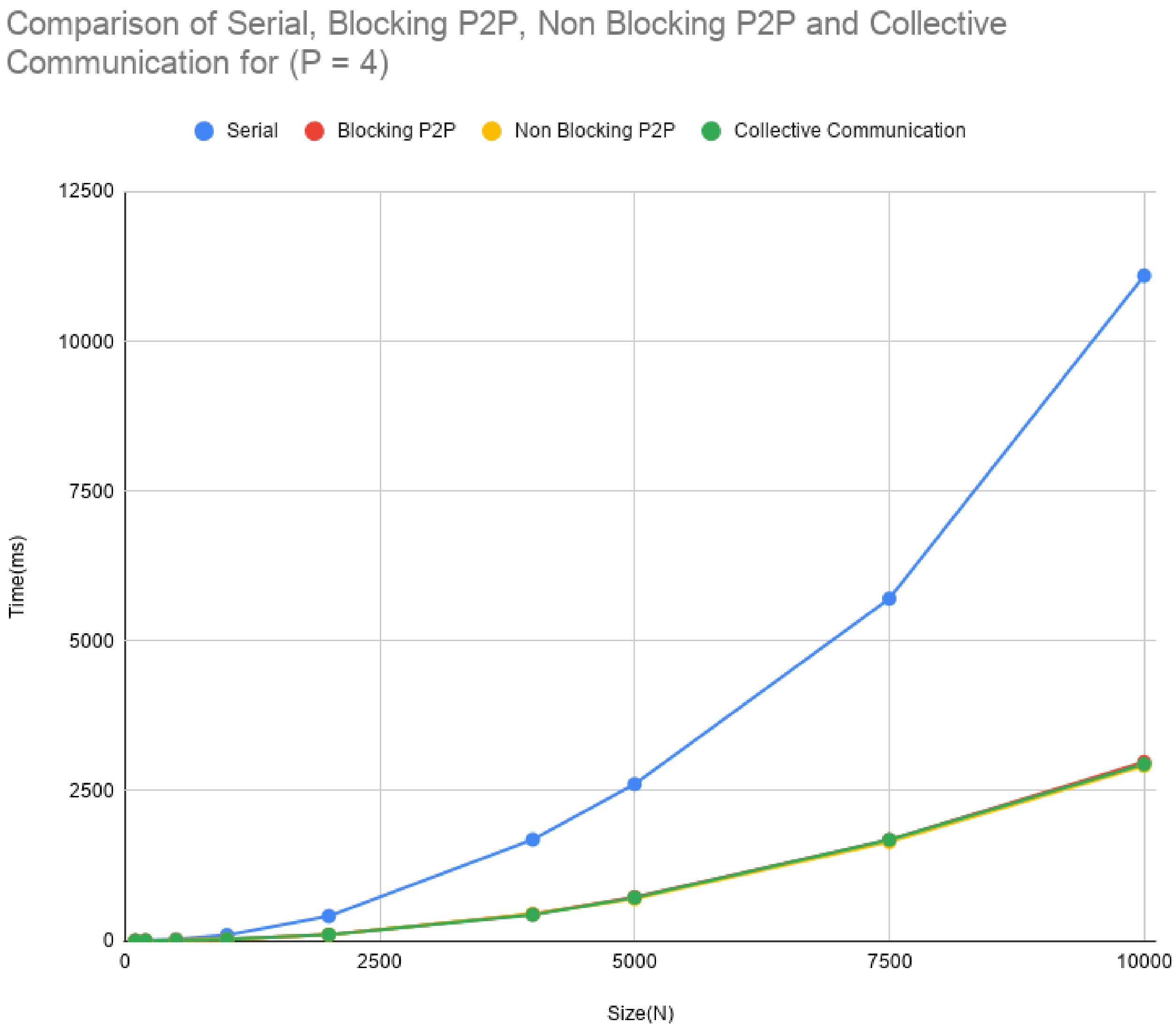
6 Plots

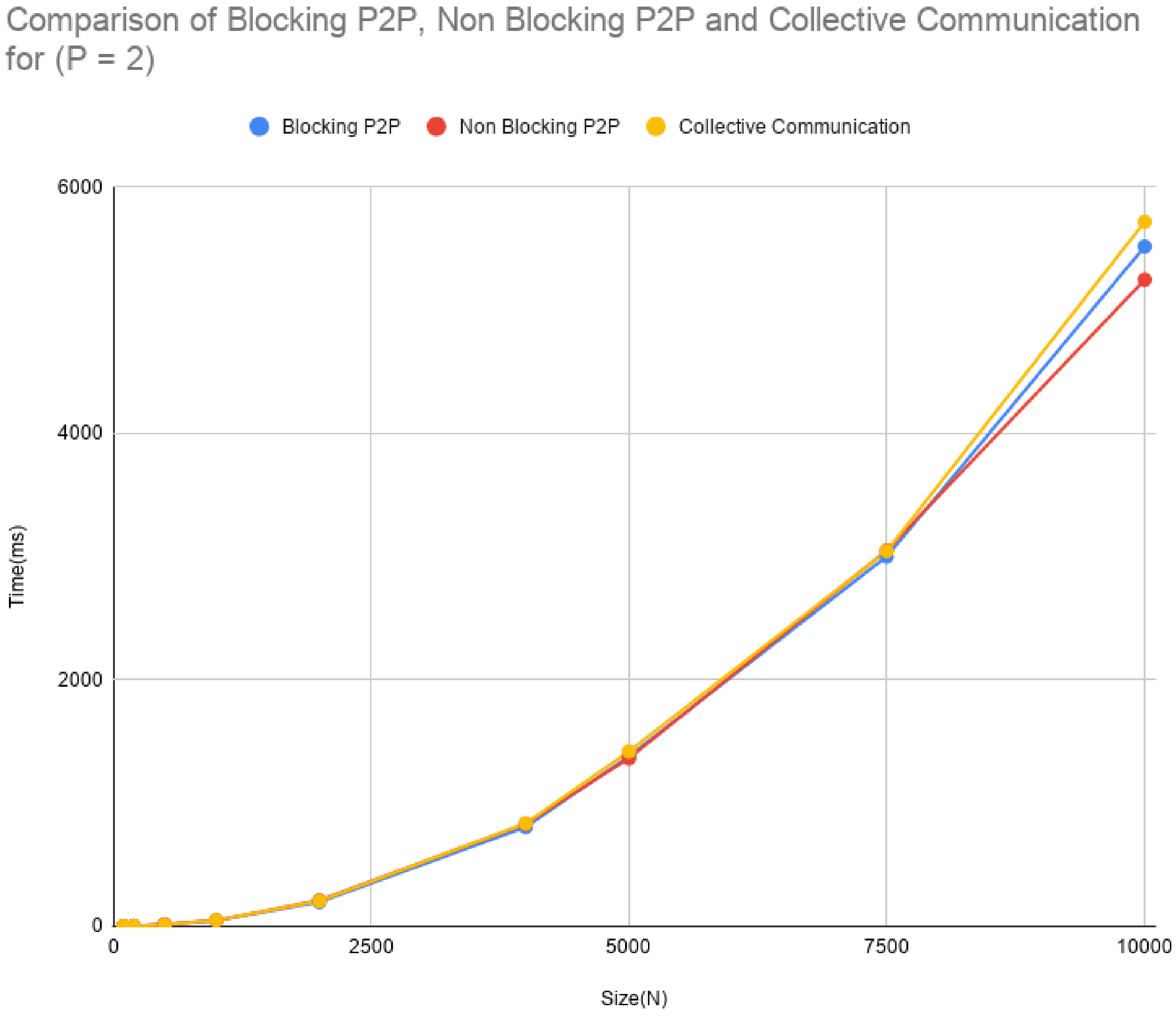

Time(t) vs Size(N) graph for Non Blocking P2P
● Parallel(P = 2)
● Parallel(P = 4)
● Serial
Time(ms)
12500
10000
7500
5000
2500
0
0
2500
5000
7500
10000
Size(N)

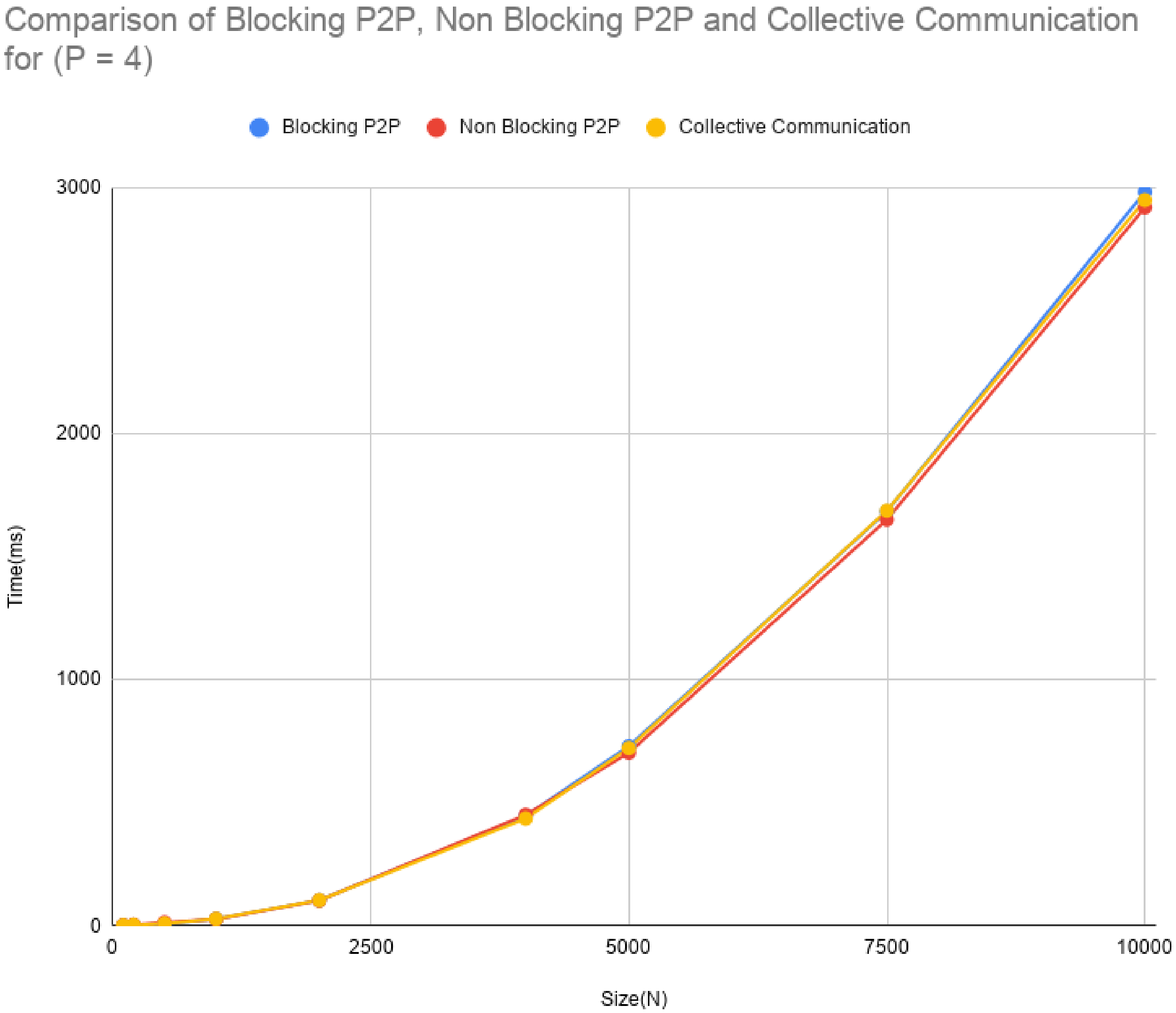

Time(t) vs Size(N) graph for collective communication
● Parallel(P = 2)
● Parallel(P = 4)
● Serial
12500
10000
7500
5000
2500
0
Time(ms)
0
2500
5000
7500
10000
Size(N)


Time(t) vs Size(N) graph for Blocking P2P
● Parallel(P = 2)
● Parallel(P = 4)
● Serial
12500
10000
7500
5000
2500
0
Time(ms)
0
2500
5000
7500
10000
Size(N)


Comparison of Serial, Blocking P2P, Non Blocking P2P and Collective Communication for(P = 2)
● Serial
● Blocking P2P
● Non Blocking P2P
● Collective Communication
Time(ms)
12500
10000
7500
5000
2500
0
0
2500
5000
7500
10000
Size(N)


Comparison of Serial, Blocking P2P, Non Blocking P2P and Collective Communication for (P = 4)
● Serial
● Blocking P2P
● Non Blocking P2P
● Collective Communication
12500
10000
7500
5000
2500
0
Time(ms)
0
2500
5000
7500
10000
Size(N)


Comparison of Blocking P2P, Non Blocking P2P and Collective Communication for (P = 2)
● Blocking P2P
● Non Blocking P2P
● Collective Communication
Time(ms)
6000
4000
2000
0
0
2500
5000
7500
10000
Size(N)


Comparison of Blocking P2P, Non Blocking P2P and Collective Communication for (P = 4)
● Blocking P2P
● Non Blocking P2P
● Collective Communication
3000
2000
1000
0
Time(ms)
0
2500
5000
7500
10000
Size(N)