RESULT REPORT

Let us first look at the result produced in serial execution.

```
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:43 im1
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 im2
-rw-r--r-- 1 kraman1 iit104    7694 Dec  7 15:38 MPI_collcomm2.c
-rw-r--r-- 1 kraman1 iit104    8915 Dec  7 17:16 MPI_sendrecv1.c
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 out
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  7 17:27 outputMatrix
-rwxr-xr-x 1 kraman1 iit104   13744 Dec  7 17:27 sequentialprogram
-rw-r--r-- 1 kraman1 iit104    4813 Dec  7 17:27 sequentialprogram.c
-rw-r--r-- 1 kraman1 iit104   13452 Dec  7 15:38 TaskDataparallel.c
[kraman1@comet-ln3 project]$ rm MPI_collcomm2.c
[kraman1@comet-ln3 project]$ ll
total 14100
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:43 im1
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 im2
-rw-r--r-- 1 kraman1 iit104    8915 Dec  7 17:16 MPI_sendrecv1.c
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 out
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  7 17:27 outputMatrix
-rwxr-xr-x 1 kraman1 iit104   13744 Dec  7 17:27 sequentialprogram
-rw-r--r-- 1 kraman1 iit104    4813 Dec  7 17:27 sequentialprogram.c
-rw-r--r-- 1 kraman1 iit104   13452 Dec  7 15:38 TaskDataparallel.c
[kraman1@comet-ln3 project]$ rm MPI_sendrecv1.c
[kraman1@comet-ln3 project]$ rm TaskDataparallel.c
[kraman1@comet-ln3 project]$ ll
total 14062
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:43 im1
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 im2
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 out
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  7 17:27 outputMatrix
-rwxr-xr-x 1 kraman1 iit104   13744 Dec  7 17:27 sequentialprogram
-rw-r--r-- 1 kraman1 iit104    4813 Dec  7 17:27 sequentialprogram.c
[kraman1@comet-ln3 project]$ gcc sequentialprogram.c -o sequentialprogram -lm
[kraman1@comet-ln3 project]$ ./sequentialprogram


CS546 Project - Koushik Raman A20388858


Sequential program


Output matrix has been generated and stored in outputMatrix file
Time taken for the serial computation = 0.130000 sec
[kraman1@comet-ln3 project]$
```

Computation time = 0.13 sec = 130 ms

Now let us look at the execution of the various types mentioned in the question.

a. MPI send and recv:

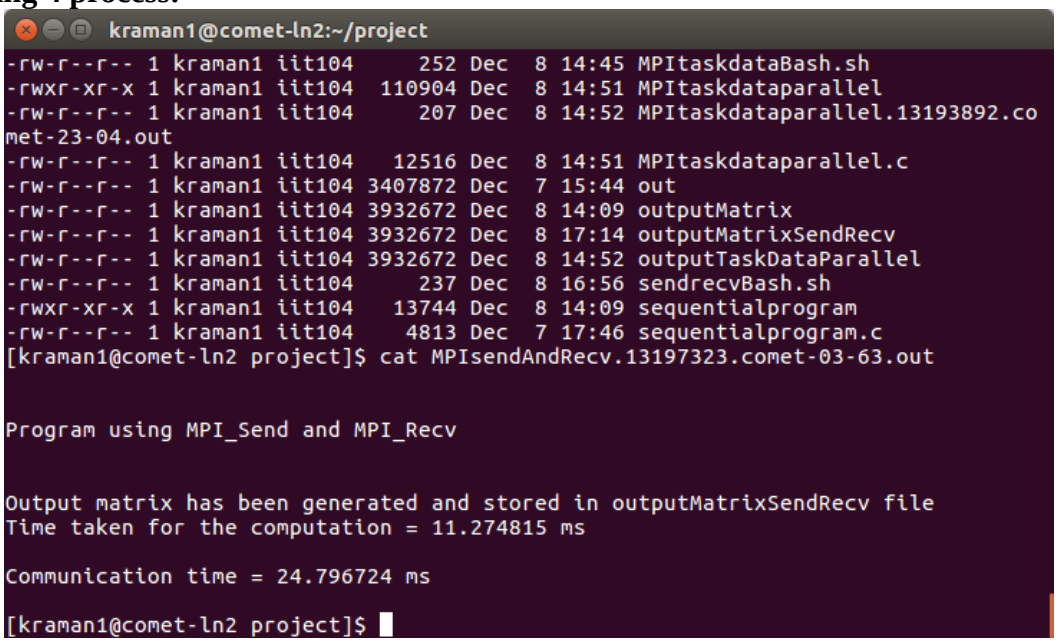
Using 8 process:

```
kraman1@comet-ln2:~/project
-rw-r--r-- 1 kraman1 iit104     252 Dec  8 14:45 MPItaskdataBash.sh
-rwxr-xr-x 1 kraman1 iit104  110904 Dec  8 14:51 MPItaskdataparallel
-rw-r--r-- 1 kraman1 iit104     207 Dec  8 14:52 MPItaskdataparallel.13193892.co
met-23-04.out
-rw-r--r-- 1 kraman1 iit104   12516 Dec  8 14:51 MPItaskdataparallel.c
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 out
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 14:09 outputMatrix
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 16:50 outputMatrixSendRecv
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 14:52 outputTaskDataParallel
-rw-r--r-- 1 kraman1 iit104     237 Dec  8 16:42 sendrecvBash.sh
-rwxr-xr-x 1 kraman1 iit104   13744 Dec  8 14:09 sequentialprogram
-rw-r--r-- 1 kraman1 iit104    4813 Dec  7 17:46 sequentialprogram.c
[kraman1@comet-ln2 project]$ cat MPIsendAndRecv.13196265.comet-11-30.out


Program using MPI_Send and MPI_Recv


Output matrix has been generated and stored in outputMatrixSendRecv file
Time taken for the computation = 7.927179 ms

Communication time = 33.164024 ms

[kraman1@comet-ln2 project]$
```

Speedup = sequential time/ parallel execution time
= 130ms / 7.927ms
= 16.4

Computation time = 7.927ms
Communication time = 33.164ms

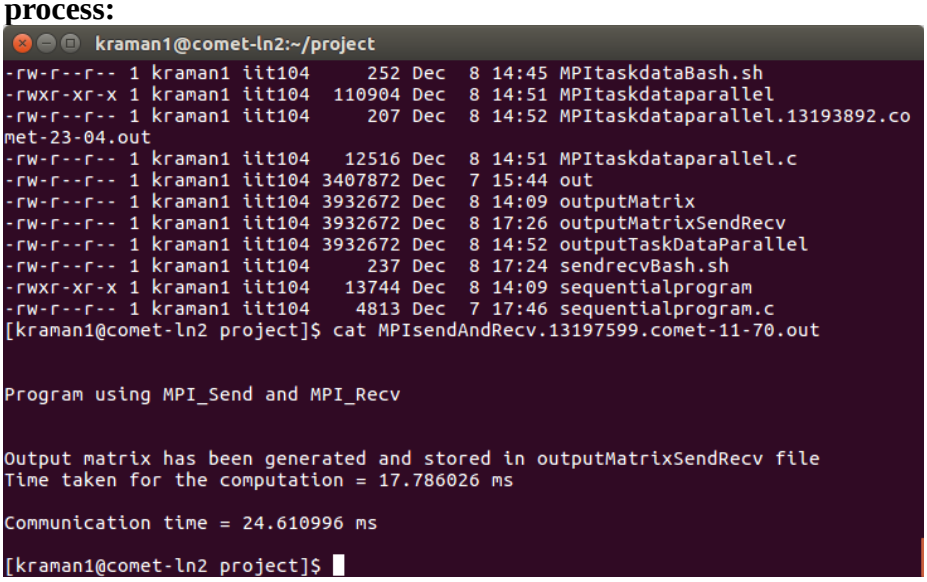
Using 4 process:



Speedup = Serial execution time / parallel execution time
= 130ms / 11.27ms
= 11.54

Computation time = 11.27ms
Communication time = 24.8ms

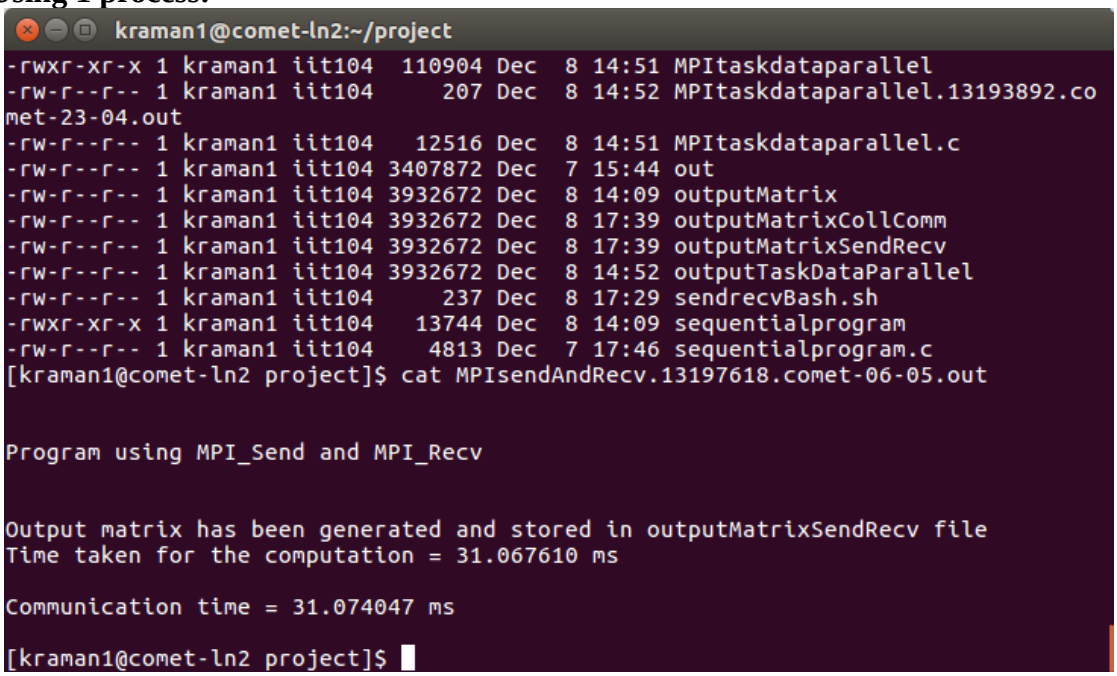
Using 2 process:



Speedup = Serial time / Parallel time
= 130ms / 17.8ms
= 7.3

Computation time = 17.8ms
Communication time = 24.61ms

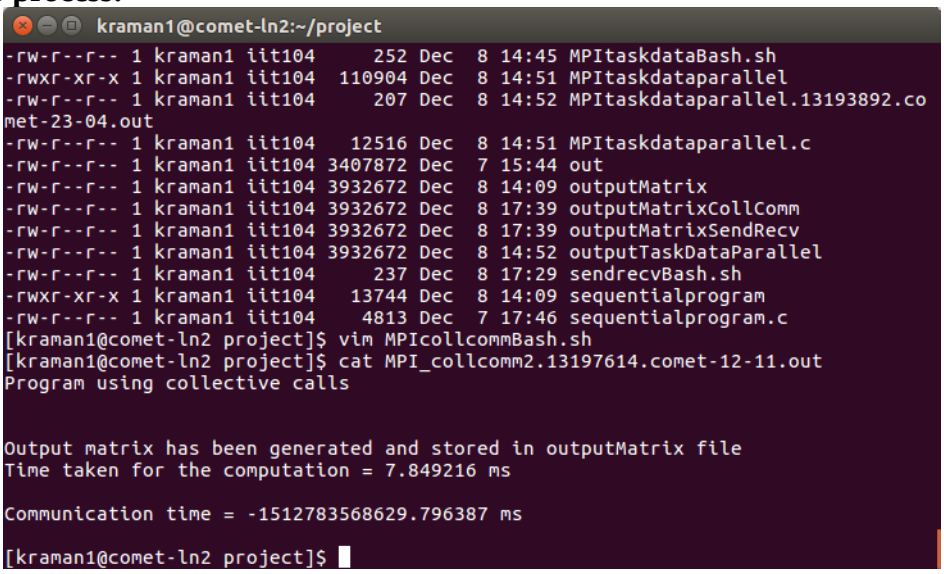
Using 1 process:



Speedup = Serial execution time / Parallel execution time
= 130ms / 31.1ms
= 4.18

b. MPI collective calls:

Using 8 process:



Speedup = Serial execution time / Parallel execution time
= 130ms / 7.85ms
= 16.56

Computation time = 7.85ms

Using 4 process:

```
kraman1@comet-ln2:~/project
-rw-r--r-- 1 kraman1 iit104    8248 Dec  8 16:31 MPIsendAndRecv.c
-rw-r--r-- 1 kraman1 iit104     252 Dec  8 14:45 MPItaskdataBash.sh
-rwxr-xr-x 1 kraman1 iit104  110904 Dec  8 14:51 MPItaskdataparallel
-rw-r--r-- 1 kraman1 iit104     207 Dec  8 14:52 MPItaskdataparallel.13193892.co
met-23-04.out
-rw-r--r-- 1 kraman1 iit104   12516 Dec  8 14:51 MPItaskdataparallel.c
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 out
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 14:09 outputMatrix
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 17:51 outputMatrixCollComm
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 17:39 outputMatrixSendRecv
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 14:52 outputTaskDataParallel
-rw-r--r-- 1 kraman1 iit104     237 Dec  8 17:29 sendrecvBash.sh
-rwxr-xr-x 1 kraman1 iit104   13744 Dec  8 14:09 sequentialprogram
-rw-r--r-- 1 kraman1 iit104    4813 Dec  7 17:46 sequentialprogram.c
[kraman1@comet-ln2 project]$ cat MPIcollcomm.13197735.comet-13-38.out
Program using collective calls


Output matrix has been generated and stored in outputMatrix file
Time taken for the computation = 11.277914 ms

Communication time = -1512784290959.070068 ms

[kraman1@comet-ln2 project]$
```

Speedup = Serial execution time / parallel execution time
= 130ms / 11.28ms
= 11.52

Computation time = 11.28ms

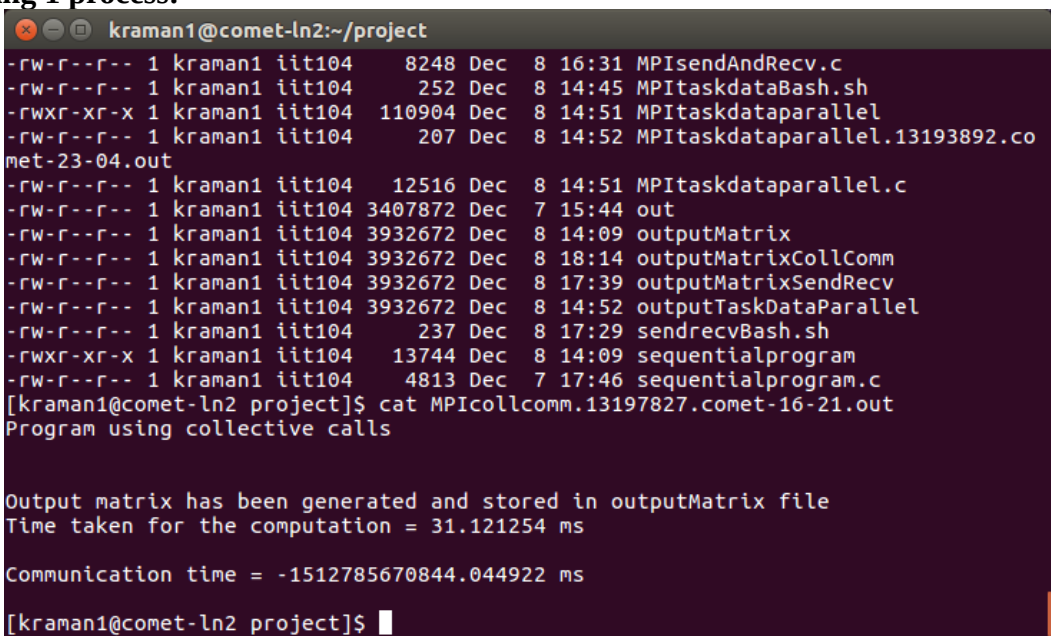
Using 2 process:

```
kraman1@comet-ln2:~/project
-rw-r--r-- 1 kraman1 iit104    8248 Dec  8 16:31 MPIsendAndRecv.c
-rw-r--r-- 1 kraman1 iit104     252 Dec  8 14:45 MPItaskdataBash.sh
-rwxr-xr-x 1 kraman1 iit104  110904 Dec  8 14:51 MPItaskdataparallel
-rw-r--r-- 1 kraman1 iit104     207 Dec  8 14:52 MPItaskdataparallel.13193892.co
met-23-04.out
-rw-r--r-- 1 kraman1 iit104   12516 Dec  8 14:51 MPItaskdataparallel.c
-rw-r--r-- 1 kraman1 iit104 3407872 Dec  7 15:44 out
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 14:09 outputMatrix
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 18:02 outputMatrixCollComm
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 17:39 outputMatrixSendRecv
-rw-r--r-- 1 kraman1 iit104 3932672 Dec  8 14:52 outputTaskDataParallel
-rw-r--r-- 1 kraman1 iit104     237 Dec  8 17:29 sendrecvBash.sh
-rwxr-xr-x 1 kraman1 iit104   13744 Dec  8 14:09 sequentialprogram
-rw-r--r-- 1 kraman1 iit104    4813 Dec  7 17:46 sequentialprogram.c
[kraman1@comet-ln2 project]$ cat MPIcollcomm.13197810.comet-05-18.out
Program using collective calls


Output matrix has been generated and stored in outputMatrix file
Time taken for the computation = 17.837048 ms

Communication time = -1512784950794.129150 ms

[kraman1@comet-ln2 project]$
```

Speedup = Serial execution time / Parallel execution time
= 130ms / 17.84ms
= 7.29

Computation time = 17.84ms

Using 1 process:

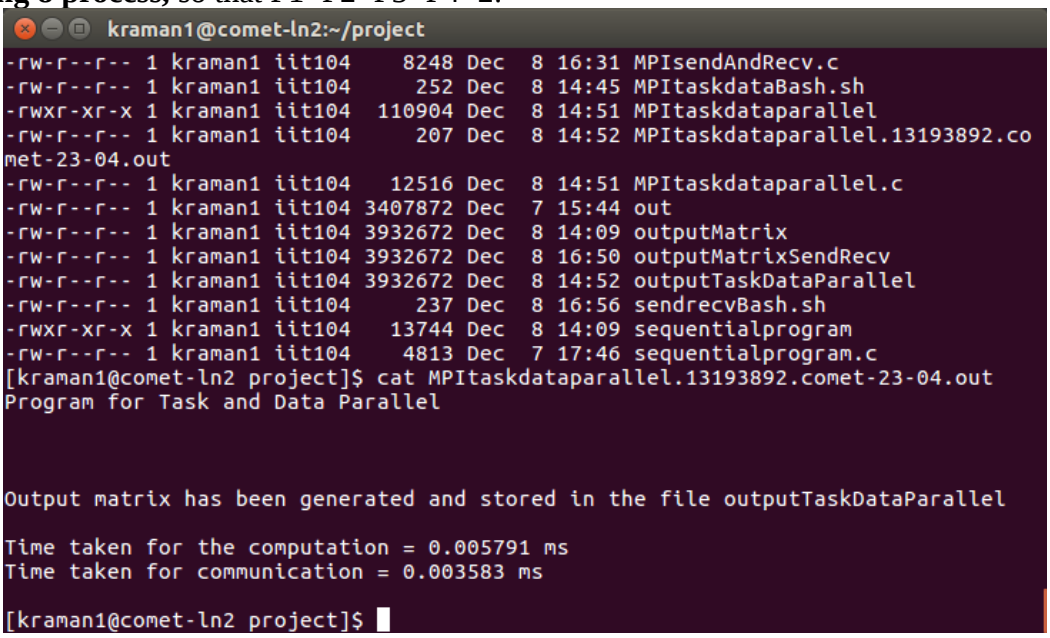


Speedup = Serial execution time / Parallel execution time
= 130ms / 31.12ms
= 4.177

Computation time = 31.12ms

c. MPI task and data parallelism:

Using 8 process, so that P1=P2=P3=P4=2:



Speedup = Serial execution time / parallel execution time
= 130ms / 0.005791ms
= 22413.8

Computation time = 0.005791ms
Communication time = 0.003583ms

d. Comparing the results in a-c for the case of 8 process:

METHOD	COMPUTATION TIME (IN ms)
MPI send and recv	7.927
MPI collective calls	7.85
MPI task & data parallel	0.005791

Comparing the results in the above table, we can clearly see that MPI Task and Data parallelism gives the output in the shortest time. The speedup is extremely good in this case. This is due to the fact that the tasks as well as data is split up among groups of processors, thus making it easy for the individual processors to perform their designated operation on their designated data. Since all these are done in parallel, the total computation time becomes too low, thus making Task & Data Parallelism the fastest method among the 3 methods described above.