Class: Final Year (Computer Science and Engineering)

Year: 2021-22 **Semester:** 1

Course: High Performance Computing Lab

Practical 6

Exam Seat No:2018BTECS00005

Name : Rahul Sanjay Naravadkar

Problem Statement 1 : MPI Scatter and Gather

MPI scatter:

MPI_Scatter is a collective routine that is very similar to MPI_Bcast MPI_Scatter involves a designated root process sending data to all processes in a communicator. The primary difference between MPI_Bcast and MPI_Scatter is small but important. MPI_Bcast sends the same piece of data to all processes while MPI_Scatter sends chunks of an array to different processes.

Here is what the function prototype of MPI_Scatter looks like.

```
MPI_Scatter(
      void* send_data,int
      send_count,
      MPI_Datatype send_datatype,
      void* recv_data,
      int  recv_count, MPI_Datatype
      recv_datatype,int root,
      MPI_Comm communicator
)
```

MPI gather:

MPI_Gather is the inverse of MPI_Scatter. Instead of spreading elements from one process to many processes, MPI_Gather takes elements from many processes and gather them to one single process. This routine is highly useful to many parallel algorithms, such as parallel sorting and searching. Similar to MPI_Scatter, MPI_Gather takes elements from each process and gathers them to the root process. The elements are ordered by the rank of the process from which they were received. The function prototype for MPI_Gather is identical to that of MPI_Scatter.

```
MPI_Gather(
      void* send_data,int
      send_count,
      MPI_Datatype send_datatype,
      void* recv_data,
      int recv_count,
```

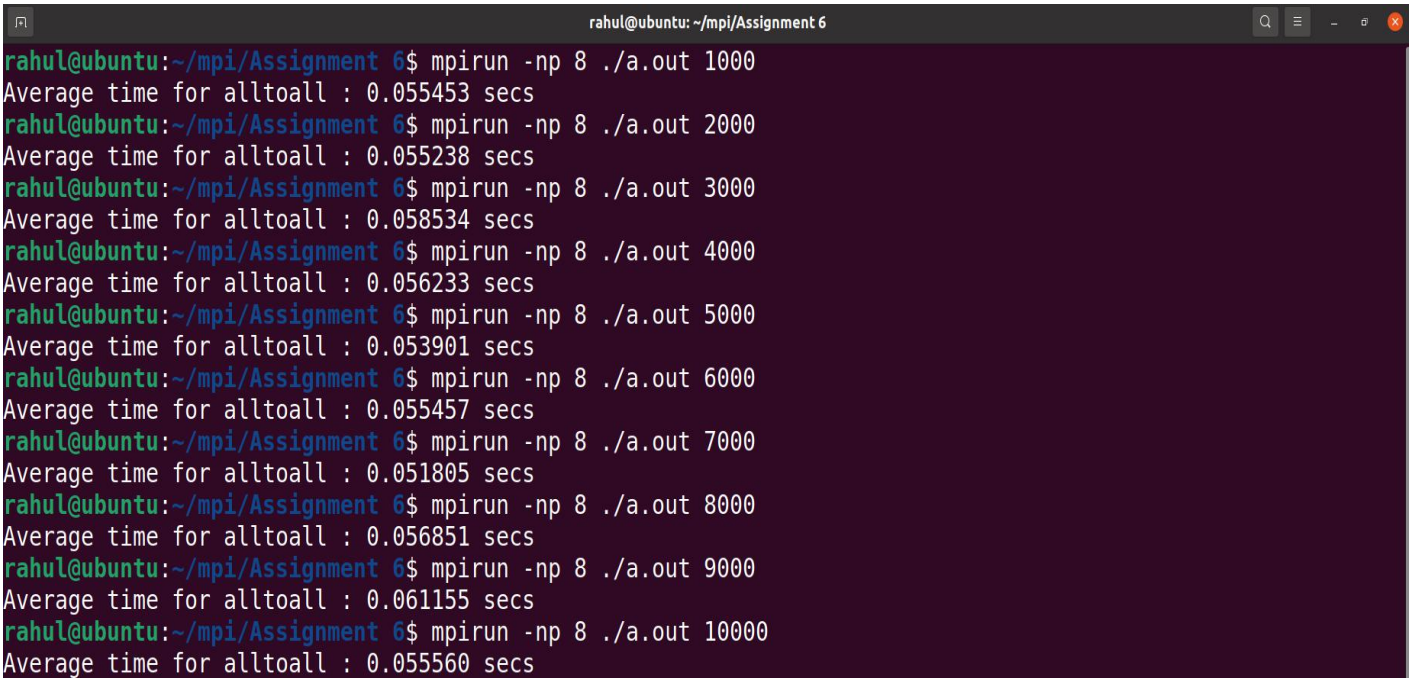
```
    MPI_Datatype  recv_datatype,
    int root,
    MPI_Comm  communicator
)
```

Problem Statement 2 : Execute the all-to-all broadcast operation (Program C) with varying message sizes.

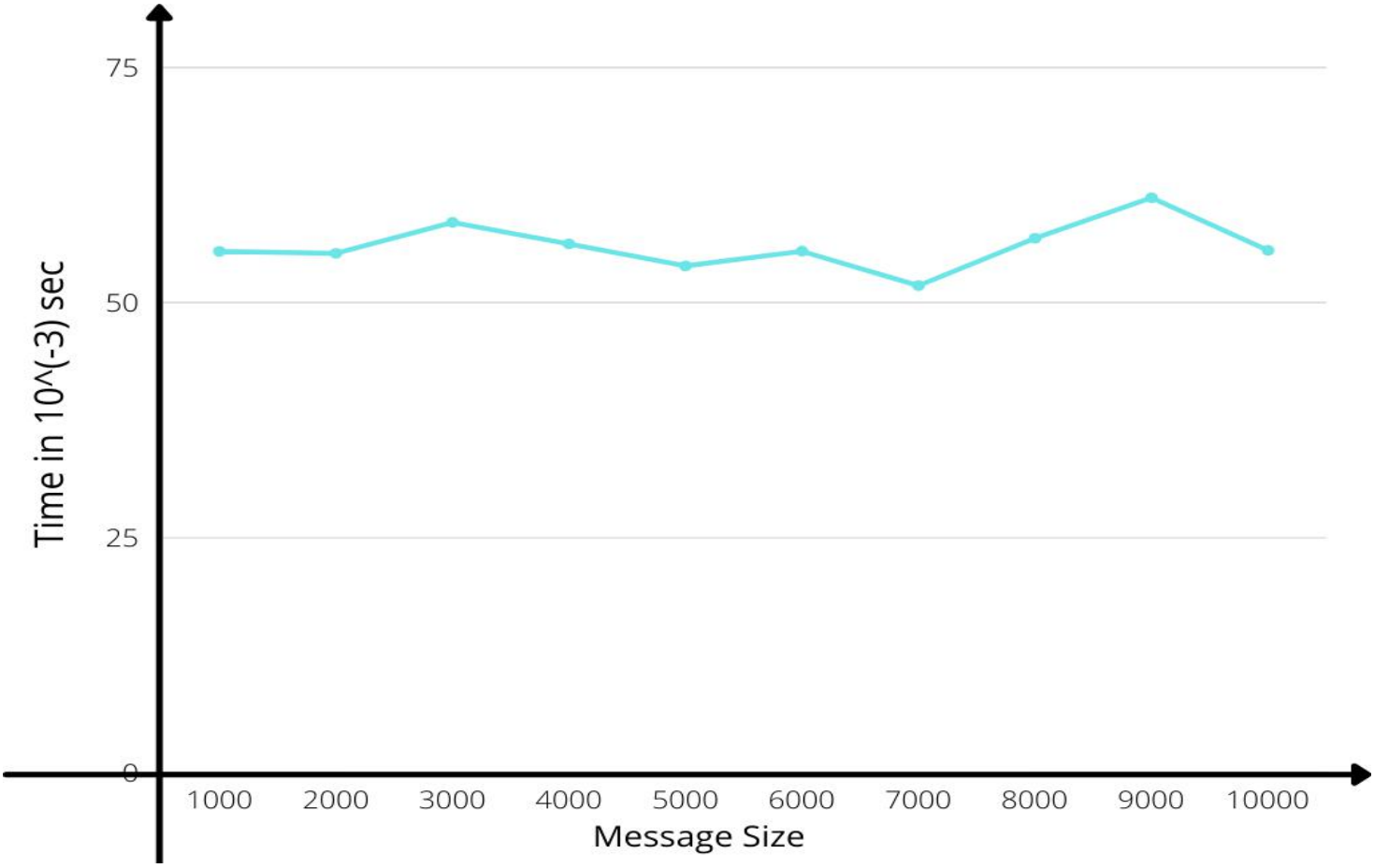
Plot the performance of the operation with varying message sizes from 1K to 10K (with

constant number of processes. Explain the performance observed.

Screenshot :



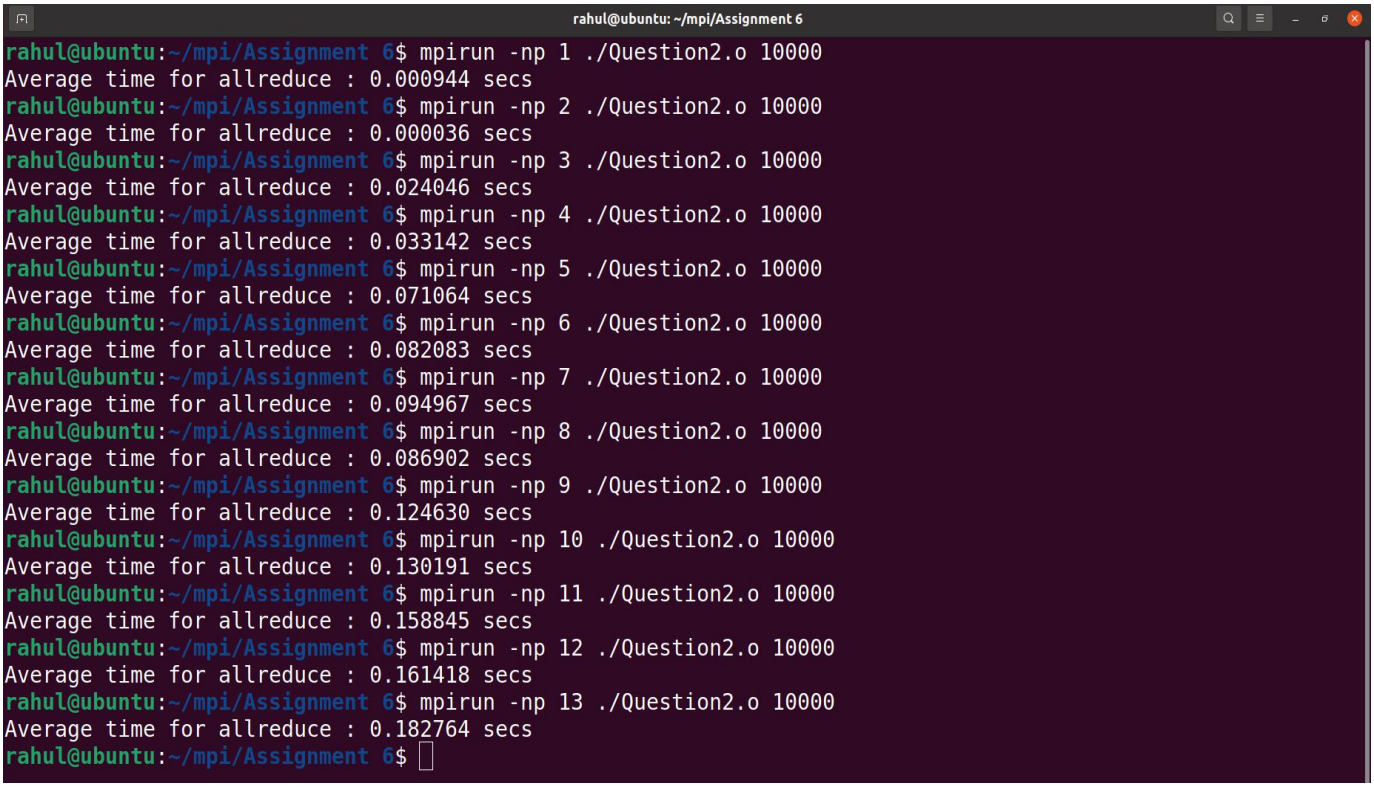
Graph Analysis:

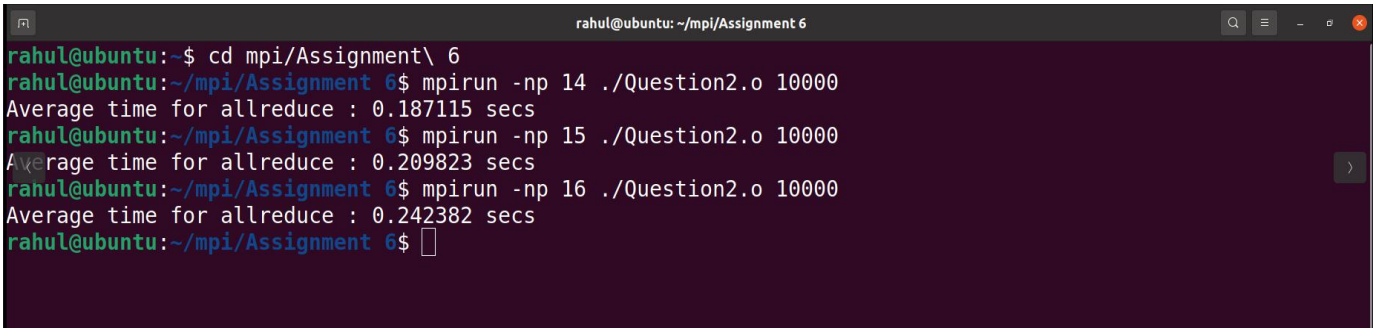


Information : Average time for broadcast increases as size of message increases.

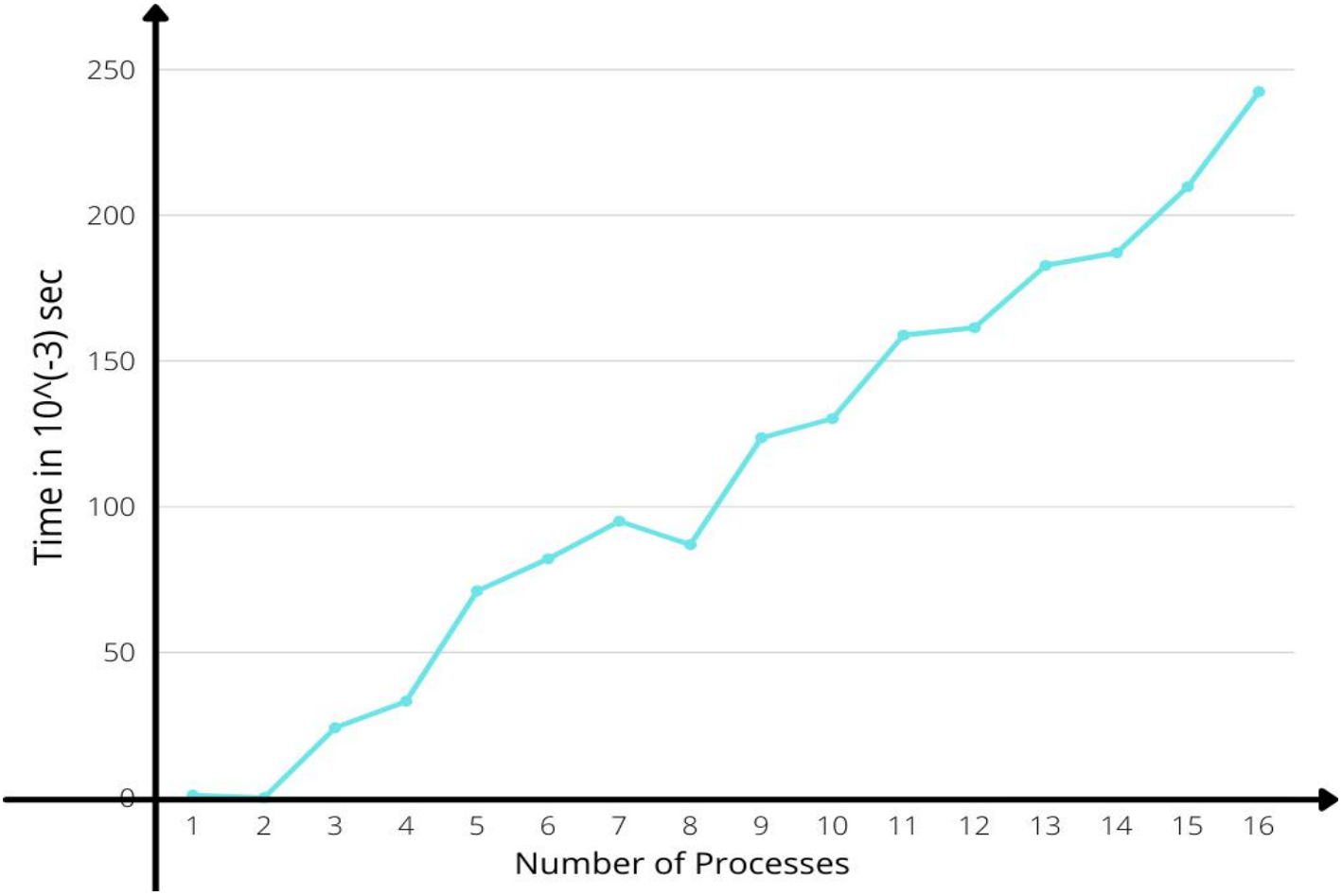
Problem Statement 3 : Execute the all-reduce operation (Program D) with varying number of processes (1 to 16) and fixed message size of 10K words. Plot the performance of the operation with varying number of processes (with constant message size). Explain the performance observed.

Screenshot :





Graph Analysis:



Information : Average time for all-reduce increases as number of processes increases withfixed message size.

Github Link