Class: Final Year (Computer Science and Engineering)

Year: 2022-23 **Semester:** 1

Course: High Performance Computing Lab

Practical No. 11

Exam Seat No: 2019BTECS00110

Name: Pavan Krishnat Shinde

Q1. 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

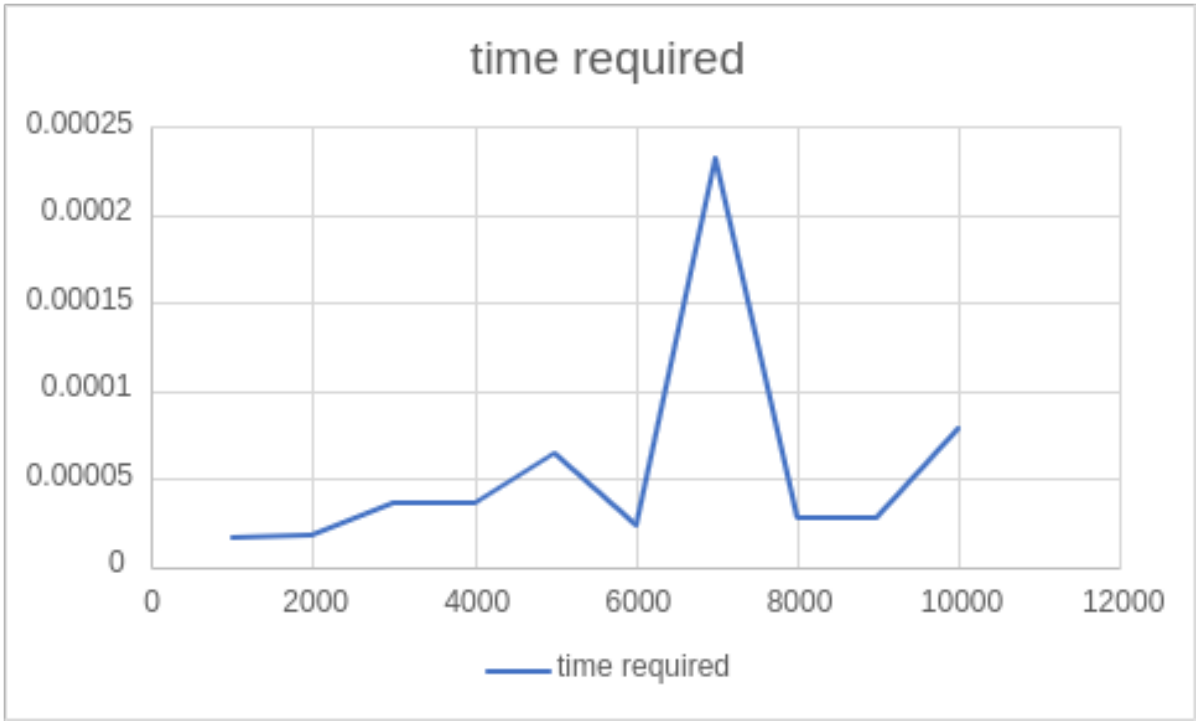
constant number of processes, 8). Explain the performance observed.

Code:

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <mpi.h>

int main(int argc, char *argv[])
{
    if (argc != 2)
    {
        printf("Usage : alltoall message_size\n");
        return 1;
    }
    int rank;
    int num_procs;
    int size = atoi(argv[1]);
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &num_procs);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    int i;
    char input_buffer[size * num_procs];
    char recv_buffer[size * num_procs];
    srand(time(NULL));
    for (i = 0; i < size; i++)
        input_buffer[i] = rand() % 256;
    int j;
    for (j = 1; j < num_procs; j++)
    {
        int k = 0;
        for (i = j * size; i < j * size + size; i++)
        {
            input_buffer[i] = input_buffer[k];
            k++;
        }
    }
    double total_time = 0.0;
    double start_time = 0.0;
    for (i = 0; i < 100; i++)
    {
        MPI_Barrier(MPI_COMM_WORLD);
        start_time = MPI_Wtime();
        MPI_Alltoall(input_buffer, size, MPI_CHAR, recv_buffer, size, MPI_CHAR, MPI_COMM_WORLD);
        MPI_Barrier(MPI_COMM_WORLD);
        total_time += (MPI_Wtime() - start_time);
    }
    if (rank == 0)
    {
        printf("Average time for alltoall : %f secs\n", total_time / 100);
    }
    MPI_Finalize();
}

```

Output:

```
PROBLEMS 2   OUTPUT   DEBUG CONSOLE   TERMINAL   JUPYTER

abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpicc c.c -o c
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 1000
Average time for alltoall : 0.000016 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 2000
Average time for alltoall : 0.000018 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 3000
Average time for alltoall : 0.000036 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 4000
Average time for alltoall : 0.000036 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 5000
Average time for alltoall : 0.000065 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 6000
Average time for alltoall : 0.000023 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 7000
Average time for alltoall : 0.000232 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 8000
Average time for alltoall : 0.000028 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 9000
Average time for alltoall : 0.000027 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./c 10000
Average time for alltoall : 0.000078 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$
```



Conclusion:

So, as we can see when we increase the number of Processors for execution of the code the execution time also increases. This observation is not because of the computation time but as we increase the number of processes the communication overhead between the processors increases and thus there is increase in execution time.

Q2. 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

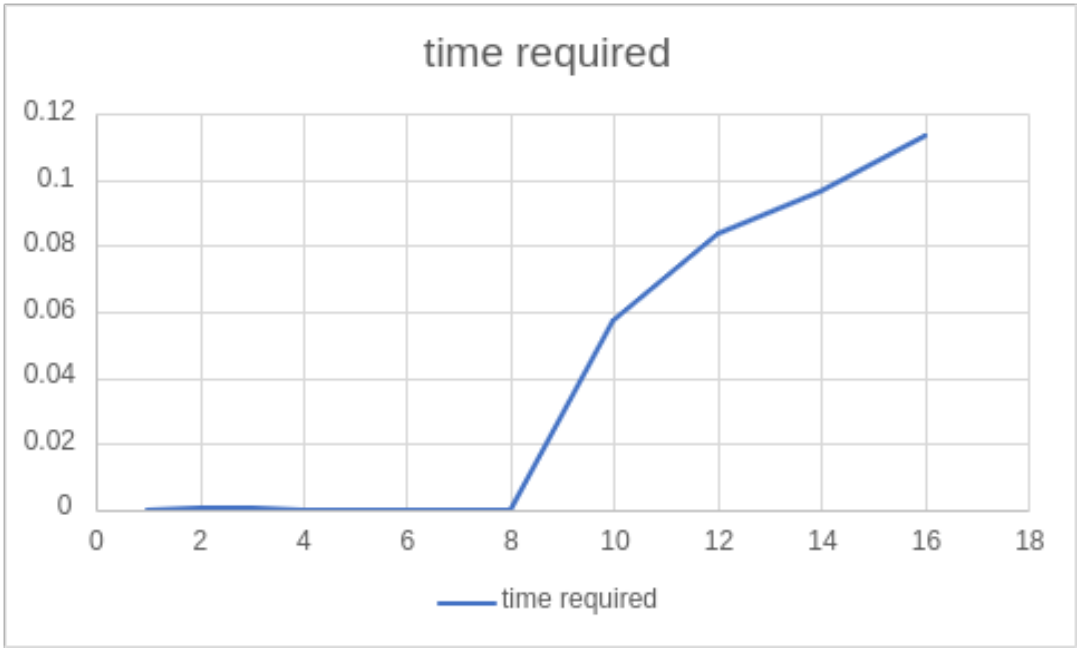
observed.

Code:

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <mpi.h>

int main(int argc, char *argv[])
{
    if (argc != 2)
    {
        printf("Usage : allreduce message_size\n");
        return 1;
    }
    int rank;
    int size = atoi(argv[1]);
    char input_buffer[size];
    char recv_buffer[size];
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    int i;
    srand(time(NULL));
    for (i = 0; i < size; i++)
        input_buffer[i] = rand() % 256;
    double total_time = 0.0;
    double start_time = 0.0;
    for (i = 0; i < 100; i++)
    {
        MPI_Barrier(MPI_COMM_WORLD);
        start_time = MPI_Wtime();
        MPI_Allreduce(input_buffer, recv_buffer, size, MPI_BYTE, MPI_BOR, MPI_COMM_ WORLD);
        MPI_Barrier(MPI_COMM_WORLD);
        total_time += (MPI_Wtime() - start_time);
    }
    if (rank == 0)
    {
        printf("Average time for allreduce : %f secs\n", total_time / 100);
    }
    MPI_Finalize();
}
```

Output:

```
PROBLEMS 4   OUTPUT   DEBUG CONSOLE   TERMINAL   JUPYTER

abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpicc d.c -o d
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 1 ./d 10000
Average time for allreduce : 0.000000 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 2 ./d 10000
Average time for allreduce : 0.000139 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 3 ./d 10000
Average time for allreduce : 0.000107 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 4 ./d 10000
Average time for allreduce : 0.000025 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 6 ./d 10000
Average time for allreduce : 0.000035 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 8 ./d 10000
Average time for allreduce : 0.000030 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 10 ./d 10000
Average time for allreduce : 0.057176 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 12 ./d 10000
Average time for allreduce : 0.083971 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 14 ./d 10000
Average time for allreduce : 0.096794 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$ mpirun -np 16 ./d 10000
Average time for allreduce : 0.113726 secs
abhi@kingsman:~/Documents/Sem7/Assignment/HPC/Assignment11$
```



Conclusion:

So, as we can see when we increase the number of Processors for execution of the code the execution time also increases. This observation is not because of the computation time but as we increase the number of processes the communication overhead between the processors increases and thus there is increase in execution time.

Github Link : https://github.com/pavanshinde7494/HPC-Assignment