Computer Science & Information Systems

**Systems for Data Analytics - Lab Sheet 6**

**Message Passing Model –** to determine the value of PI using a cluster of machines of same network

1. Objective:

Students should be able to

1. Get familiarity with the Message Passing model in Parallel computing environment
2. Get hands-on experience on MPI APIs – MPI\_Bcast, MPI\_Reduce

This lab sheet provides a quick introduction to using MPI(Message passing Interface). This exercise will introduce the MPI\_Bcast, MPI\_Reduce API used in MPI in C language for parallel communication.

A *process* is (traditionally) a program counter and address space. Processes may have multiple *threads* (program counters and associated stacks) sharing a single address space.

MPI is for communication among processes, which have separate address spaces.

**Interprocess communication consists of** Synchronization and Movement of data from one process’s address space to another’s.

The message-passing approach makes the exchange of data *cooperative*. Data is explicitly *sent* by one process and *received* by another. An advantage is that any change in the receiving process’s memory is made with the receiver’s explicit participation. Communication and synchronization are combined.

MPI\_Bcast

Broadcasts a message from the process with rank "root" to all other processes of the communicator

MPI\_Reduce

Reduces values on all processes to a single value

Process 0

Process 1

**Send(data)**

**Receive(data)**

MPI’s MPI\_Bcast, MPI\_Reduce functions –

int MPI\_Bcast(void \*buffer, int count, MPI\_Datatype datatype, int root, MPI\_Comm comm)

Input Parameters

Count - number of entries in buffer (integer)

Datatype - data type of buffer (handle)

Root - rank of broadcast root (integer)

comm - communicator (handle)

int MPI\_Reduce(const void \*sendbuf, void \*recvbuf, int count, MPI\_Datatype datatype, MPI\_Op op, int root, MPI\_Comm comm)

Input Parameters

Sendbuf - address of send buffer (choice)

Count - number of elements in send buffer (integer)

Datatype - data type of elements of send buffer (handle)

Op - reduce operation (handle)

Root - rank of root process (integer)

comm - communicator (handle)

1. Steps to be performed:

Lab used – remote lab with Machines – MPI01 and MPI02

Language used – C

To compile: mpicc –o <output-file> <source-file>.c

To run: mpirun -n 4 -f host\_file ./<output-file>

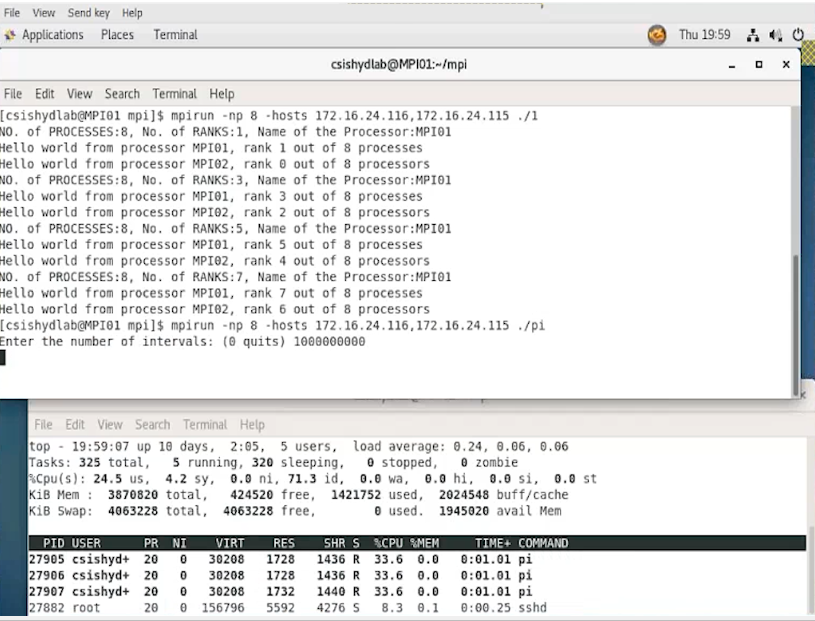
-n flag to set the number of MPI processes

-f flag to get the hosts(IP address or hostnames) listed in host\_file

If you do not have the host\_file, use –hosts flag in the following way:

mpirun –np 4 –hosts 192.168.1.2,192.168.1.3,.. ./<output-file>

1. Outputs/Results:



**Make sure the executable is present in the same path in all the machines/ hosts with the same name.**

1. Observations:

* Students to observe the output when executing with one process. Justify your observation.
* Execute and observe if the no. of process is 2, 3, ……….. 1000000