Name: Raghav Maheshwari
RollNO: 53
Panel: A

Lab Assignment - 4 (PP)

* Write a parallel code for calculation of pi (Area under curve) using MPI.

* Aim:
To write C program to calculate value of Pi up to 20 decimal digit and find error % in actual value of Pi.

* Theory:

1. Explain MPI Scatter and Gather operations.

MPI Scatter and Gather.

Group functions that can manage data distribution and collecting more easily.

Scatter: divides big array into a number of smaller parts equal to no. of processes & sends each process a piece of array in rank order.

Gather: OPP. receives data stored in small arrays from all processes and concatenates it in receive array in Rank order.

Syntax :

int MPI_Scatter(void *sendbuf, int sendcnt, MPI_Datatype sendtype, void* recvbuf, int recvcnt, MPI_Datatype recvtype, int root, MPI_Comm comm);

the above syntax applied for MPI_Scatter and MPI_Gather.

2. How value of pi can be calculated using area under the curve.?

Ans We will estimate value of π using area formula for a circle $\pi = r^2$. We can compute π by finding area of a unit circle, i.e. a circle with radius $r = 1$

Formula for unit circle is $x^2 + y^2 = 1$ Solving for y we get $y = \pm\sqrt{1-x^2}$, NOTE: that each value of x between and $x = -1$ and $x = 1$ yields two values of y according to this formula, corresponding to upper and lower semi-circle in figure.

area of rectangle is $\sqrt{1-x^2} \times h$ we must add up these that sum to get approximation of π

$$\left[\int_{-1} \sqrt{1-x^2} \times h\right]$$

More rectangle we use, greater accuracy we expect in our rectangle approximation to exact area under semi-circle curve $y = f(x)$

FAQ

(1) What are the different ways to calculate the value of Pi?

Ans → Calculating Pi using measurements of circle.
→ Calculating Pi using and infinite series.
→ Calculating Pi using Buffon's needle problem.
→ Calculating Pi using a limit.
→ Using Arcsine function / Inverse sine function.

(2) What does ssh-copy-id command do?

Ans ssh-copy-id command is a simple tool that allows you to install an SSH key on a remote server's authorized keys. This command facilitates SSH key login, which removes need for a password for each login, thus ensuring password less automatic login process. ssh-copy-id command is part of open SSH, tool for performing remote system administrations using encrypted SSH connections.

(3) What does ssh-keygen command do?

Ans ssh-keygen is able to generate a key using one of three different digital signature algorithms. With the help of ssh-keygen tool, a user can create passphrase keys for any of these key types. To provide for unattended operation, passphrase can be left empty albeit at increased risk.

Since entire circle area is π, we uses approximate area of shaded semi-circle i.e. area under the curve $1-x^2$ then multiply by 2. to approximate that area under the curve, we will add up areas of rectangle that approximately cover area of that semi-circle.

Cluster Formation steps:

1. Server Installation
$ sudo apt-get install openssh-server

2. Find hostname.
$ hostname

3. Find IP Address
$ ifconfig

4. make one file of the host name and ip
$ sudo gedit /etc/hosts

5. Make the required changes by adding IP and host name and save the file

6. Generate Key
$ ssh-keygen

7. copy id from each node to cluster.
$ ssh-copy-id

8. Test the connection.

Pi Calculation program:

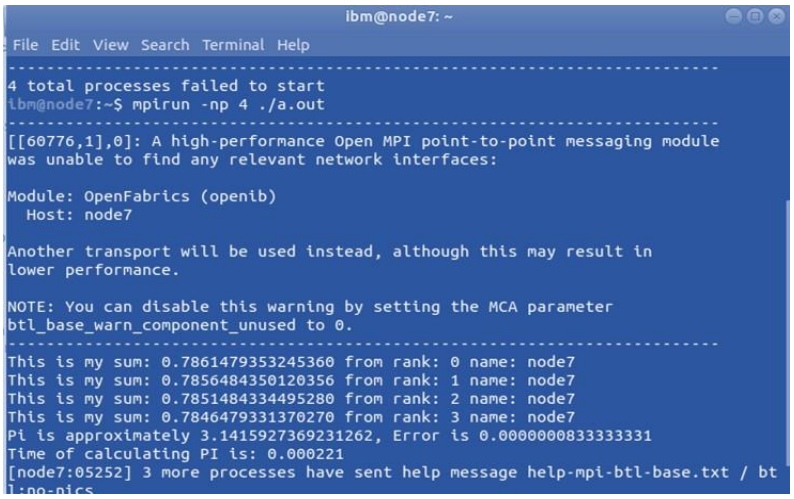
On a single machine:

Code:

```
/*************************************************************
This Program for Pi calculation depicts the usage of
MPI_Wtime, MPI_Bcast and MPI_Reduce
This exercise presents a simple program to determine the value of pi. The algorithm
suggested here is chosen for its simplicity. The method evaluates the integral of 4/(1+x*x)
between 0 and 1. The method is simple: the integral is approximated by a sum of n intervals;
the approximation to the integral in each interval is (1/n)*4/(1+x*x). The master process
(rank 0) asks the user for the number of intervals; the master should then broadcast this
number to all of the other processes. Each process then adds up every n'th interval (x =
rank/n, rank/n+size/n,...). Finally, the sums computed by each process are added together
using a reduction.
*************************************************************/

#include "mpi.h"
#include <math.h>
#include <stdio.h>

#define MAX_NAME 80   /* length of characters for naming a process */
#define MASTER 0      /* rank of the master */

int main(int argc, char *argv[])
{
   int rank,                          /* rank variable to identify the process */
      nprocs,                         /* number of processes */
      i,
      len;                           /* variable for storing name of processes */

   int n = 1000;                       /* the number of bins */
   double PI25DT = 3.14159265358979323846264643;        /* 25-digit-PI*/
   double mypi,                        /* value from each process */
         pi,                          /* value of PI in total*/
         step,                        /* the step */
         sum,                         /* sum of area under the curve */
         x;

   char name[MAX_NAME];       /* char array for storing the name of each process */

   double start_time,         /* starting time */
```

```
        end_time,           /* ending time */
        computation_time;    /* time for computing value of PI */

    /*Initialize MPI execution environment */
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &nprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    MPI_Get_processor_name(name, &len);

    start_time = MPI_Wtime();

    /* Broadcast the number of bins to all processes */
    /* This broadcasts an integer which is n, from the master to all processes
     * and
     */

    MPI_Bcast(&n, 1, MPI_INT, MASTER, MPI_COMM_WORLD);

    /* Calculating for each process */
    step = 1.0 / (double) n;
    sum = 0.0;
    for (i = rank + 1; i <= n; i += nprocs)
    {
       x = step * ((double)i-.5);
       sum += (4.0/(1.0 + x*x));
    }

    mypi = step * sum;

    printf("This is my sum: %.16f from rank: %d name: %s\n", mypi, rank, name);

    /* Now we can reduce all those sums to one value which is Pi */
    MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);

    if (rank == 0)
    {
       printf("Pi is approximately %.16f, Error is %.16f\n", pi, fabs(pi - PI25DT));
       end_time = MPI_Wtime();
       computation_time = end_time - start_time;
       printf("Time of calculating PI is: %f\n", computation_time);
    }
    /* Terminate MPI execution environment */
    MPI_Finalize();
}
```

OUTPUT ON A SINGLE MACHINE:



Text Format

[[60776,1],0]: A high-performance Open MPI point-to-point messaging module
was unable to find any relevant network interfaces:

Module: OpenFabrics (openib)
  Host: node7

Another transport will be used instead, although this may result in
lower performance.

NOTE: You can disable this warning by setting the MCA parameter
btl_base_warn_component_unused to 0.

This is my sum: 0.7861479353245360 from rank: 0 name: node7
This is my sum: 0.7856484350120356 from rank: 1 name: node7
This is my sum: 0.7851484334495280 from rank: 2 name: node7
This is my sum: 0.7846479331370270 from rank: 3 name: node7
Pi is approximately 3.1415927369231262, Error is 0.0000000833333331
Time of calculating PI is: 0.000221
[node7:05252] 3 more processes have sent help message help-mpi-btl-base.txt / btl:no-nics
[node7:05252] Set MCA parameter "orte_base_help_aggregate" to 0 to see all help / error
messages

Output Image on a Cluster of nodes: