

Point to Point Communications in MPI

Name: Satyam Singh

Reg no: 230905256

CSE B2 - 37

Lab Exercises:

Q1) Write a MPI program using synchronous send. The sender process sends a word to the receiver. The second process receives the word, toggles each letter of the word and sends it back to the first process. Both processes use synchronous send operations.

Ans)

```
#include<stdio.h>
#include<string.h>
#include<mpi.h>

int main(int argc, char* argv[]){
int size,rank;
char str[100];
MPI_Init(&argc,&argv);
MPI_Comm_size(MPI_COMM_WORLD,&size);
MPI_Comm_rank(MPI_COMM_WORLD,&rank);
MPI_Status status;

if(rank == 0){
printf("Enter string: ");
scanf("%s",str);
printf("Process %d: Sending string %s\n", rank, str);
MPI_Ssend(&str,sizeof(str),MPI_CHAR,1,0,MPI_COMM_WORLD);
MPI_Recv(&str,sizeof(str),MPI_CHAR,1,1,MPI_COMM_WORLD,&status);
printf("Process %d: Received string %s\n",rank, str);
}
else if(rank == 1){
```

```

MPI_Recv(&str,sizeof(str),MPI_CHAR,0,0,MPI_COMM_WORLD,&status);
printf("Process %d: Received string %s\n",rank, str);
for (int i = 0; i <= strlen(str); i++) {
if (str[i] >= 'A' && str[i] <= 'Z') str[i] += 32;
else if (str[i] >= 'a' && str[i] <= 'z') str[i] -= 32;
}
printf("Process %d: Toggled string %s\n",rank, str);
printf("Process %d: Sending string %s\n", rank, str);
MPI_Ssend(&str,sizeof(str),MPI_CHAR,0,1,MPI_COMM_WORLD);
}
MPI_Finalize();
return 0;
}

```

Output:

```

Rank 0: Toggling. Received 27
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpicc -o q1 q1.c
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpirun -n 4 ./q1
Enter string: SaTYaM
Process 0: Sending string SaTYaM
Process 0: Received string sAtyAm
Process 1: Received string SaTYaM
Process 1: Toggled string sAtyAm
Process 1: Sending string sAtyAm

```

Q2) Write a MPI program where the master process (process 0) sends a number to each of the slaves and the slave processes receive the number and prints it. Use standard send.

Ans)

```
#include<mpi.h>
#include<stdio.h>

int main(int argc, char* argv[]){
int size,rank,n;
MPI_Init(&argc,&argv);
MPI_Comm_size(MPI_COMM_WORLD,&size);
MPI_Comm_rank(MPI_COMM_WORLD,&rank);
MPI_Status status;

if(rank == 0){
printf("Enter number: ");
scanf("%d",&n);
printf("Process %d: Sending number %d\n",rank, n);
for(int i=1;i<size;i++) MPI_Send(&n,1,MPI_INT,i,0,MPI_COMM_WORLD);
}
else{
MPI_Recv(&n,1,MPI_INT,0,0,MPI_COMM_WORLD,&status);
printf("Process %d: Receiving number %d\n",rank,n);
}
MPI_Finalize();
return 0;
}
```

Output:

```
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpicc -o q2 q2.c
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpirun -n 4 ./q2
Enter number: 04
Process 0: Sending number 4
Process 1: Receiving number 4
Process 2: Receiving number 4
Process 3: Receiving number 4
```

Q3) Write a MPI program to read N elements of the array in the root process (process 0) where N is equal to the total number of processes. The root process sends one value to each of the slaves. Let even ranked process finds square of the received element and odd ranked process finds cube of received element. Use Buffered send.

Ans)

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

int main(int argc, char *argv[])
{
int size, rank, arr[100], num;
MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &size);
MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Status status;

if (rank == 0)
{
printf("Enter elements of the array of size %d : ", size - 1);
for (int i = 0; i < size - 1; i++)
scanf("%d", &arr[i]);
int bSize = sizeof(arr);
int *buf = (int *)malloc(bSize);
MPI_Buffer_attach(buf, bSize);
for (int i = 0; i < size - 1; i++)
MPI_Bsend(&arr[i], 1, MPI_INT, i + 1, 0, MPI_COMM_WORLD);
MPI_Buffer_detach(&buf, &bSize);
}
else
{
MPI_Recv(&num, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, &status);
if (rank % 2 == 0)
printf("Rank %d squaring: Received %d\n", rank, num * num);
else
printf("Rank %d cubing: Received %d\n", rank, num * num * num);
}
```

```
MPI_Finalize();  
return 0;  
}
```

Output:

```
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpicc -o q2 q3.c  
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpirun -n 4 ./q3  
Enter elements of the array of size 3 : 3 7 4  
Rank 1 cubing: Received 27  
Rank 2 squaring: Received 49  
Rank 3 cubing: Received 64
```

Q4) Write a MPI program to read an integer value in the root process. Root process sends this value to Process1, Process1 sends this value to Process 2 and so on. Last process sends the value back to root process. When sending the value each process will first increment the received value one by one. Write the program using point to point communication routines.

Ans)

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

int main(int argc, char *argv[]){
int rank, size, x;
MPI_Init(&argc,&argv);
MPI_Comm_rank(MPI_COMM_WORLD,&rank);
MPI_Comm_size(MPI_COMM_WORLD, &size);
MPI_Status status;

if(rank==0){
printf("Enter integer : ");
scanf("%d", &x);
MPI_Ssend(&x, 1, MPI_INT, rank+1, 1, MPI_COMM_WORLD);
MPI_Recv(&x,1,MPI_INT,(size-1),1,MPI_COMM_WORLD,&status);
printf("Process %d: Received %d from process %d\n",rank, x, size-1);
}
else{
MPI_Recv(&x,1,MPI_INT,(rank-1),1,MPI_COMM_WORLD,&status);
printf("Process %d: Received %d from process %d\n",rank, x, rank-1);
x += 1;
if(rank == (size-1)) MPI_Ssend(&x, 1, MPI_INT, 0, 1, MPI_COMM_WORLD);
else MPI_Ssend(&x, 1, MPI_INT, rank+1, 1, MPI_COMM_WORLD);
}
MPI_Finalize();
return 0;
}
```

Output:

```
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpicc -o q4 q4.c
STUDENT@MIT-ICT-LAB5-03:~/230905256/Lab2$ mpirun -n 8 ./q4
Enter integer : 15
Process 1: Received 15 from process 0
Process 2: Received 16 from process 1
Process 3: Received 17 from process 2
Process 4: Received 18 from process 3
Process 5: Received 19 from process 4
Process 6: Received 20 from process 5
Process 7: Received 21 from process 6
Process 0: Received 22 from process 7
```