ARYAMAN MISHRA

19BCE1027

LAB 8

1. **Simple MPI Send and Recv**

```
#include <mpi.h>

#include <stdio.h>

#include <stdlib.h>


int main(int argc, char** argv) {

  // Initialize the MPI environment

  MPI_Init(NULL, NULL);

  // Find out rank, size

  int world_rank;

  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

  int world_size;

  MPI_Comm_size(MPI_COMM_WORLD, &world_size);


  // We are assuming at least 2 processes for this task

  if (world_size < 2) {

    fprintf(stderr, "World size must be greater than 1 for %s\n", argv[0]);

    MPI_Abort(MPI_COMM_WORLD, 1);

  }


  int number;

  if (world_rank == 0) {
```

```
    // If we are rank 0, set the number to -1 and send it to process 1
    number = -1;
    MPI_Send(
      /* data        = */ &number,
      /* count       = */ 1,
      /* datatype    = */ MPI_INT,
      /* destination = */ 1,
      /* tag         = */ 0,
      /* communicator = */ MPI_COMM_WORLD);
  } else if (world_rank == 1) {
    MPI_Recv(
      /* data        = */ &number,
      /* count       = */ 1,
      /* datatype    = */ MPI_INT,
      /* source      = */ 0,
      /* tag         = */ 0,
      /* communicator = */ MPI_COMM_WORLD,
      /* status      = */ MPI_STATUS_IGNORE);
    printf("Process 1 received number %d from process 0\n", number);
  }
  MPI_Finalize();
}
```

```
19/10/2021  16:00.05  /home/mobaxterm/Desktop/19BCE1027_PCD  mpicc -o lab81 lab81.c

19/10/2021  16:01.27  /home/mobaxterm/Desktop/19BCE1027_PCD  mpirun
-n 2 ./lab81
Process 1 received number -1 from process 0
```

```
#include <mpi.h>

#include <stdio.h>

#include <stdlib.h>


int main(int argc, char** argv) {

  const int PING_PONG_LIMIT = 10;


  // Initialize the MPI environment

  MPI_Init(NULL, NULL);

  // Find out rank, size

  int world_rank;

  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

  int world_size;

  MPI_Comm_size(MPI_COMM_WORLD, &world_size);


  // We are assuming 2 processes for this task

  if (world_size != 2) {

    fprintf(stderr, "World size must be two for %s\n", argv[0]);

    MPI_Abort(MPI_COMM_WORLD, 1);

  }


  int ping_pong_count = 0;

  int partner_rank = (world_rank + 1) % 2;

  while (ping_pong_count < PING_PONG_LIMIT) {

    if (world_rank == ping_pong_count % 2) {
```

```
    // Increment the ping pong count before you send it
    ping_pong_count++;
    MPI_Send(&ping_pong_count, 1, MPI_INT, partner_rank, 0, MPI_COMM_WORLD);
    printf("%d sent and incremented ping_pong_count %d to %d\n",
           world_rank, ping_pong_count, partner_rank);
  } else {
    MPI_Recv(&ping_pong_count, 1, MPI_INT, partner_rank, 0, MPI_COMM_WORLD,
             MPI_STATUS_IGNORE);
    printf("%d received ping_pong_count %d from %d\n",
           world_rank, ping_pong_count, partner_rank);
  }
}
MPI_Finalize();
}
```

```
19/10/2021  16:02.39  /home/mobaxterm/Desktop/19BCE1027_PCD  mpicc -o lab811 lab811.c

19/10/2021  16:03.25  /home/mobaxterm/Desktop/19BCE1027_PCD  mpirun -n 2 ./lab811
0 sent and incremented ping_pong_count 1 to 1
0 received ping_pong_count 2 from 1
0 sent and incremented ping_pong_count 3 to 1
0 received ping_pong_count 4 from 1
0 sent and incremented ping_pong_count 5 to 1
0 received ping_pong_count 6 from 1
0 sent and1 received ping_pong_count 1 from 0
1 sent and incremented ping_pong_count 2 to 0
1 received ping_pong_count 3 from 0
1 sent and incremented ping_pong_count 4 to 0
1 received ping_pong_count 5 from 0
1 sent and incremented ping_pong_count 6 to 0
1 received incremented ping_pong_count 7 to 1
0 received ping_pong_count 8 from 1
0 sent and incremented ping_pong_count 9 to 1
0 received ping_pong_count 10 from 1
 ping_pong_count 7 from 0
1 sent and incremented ping_pong_count 8 to 0
1 received ping_pong_count 9 from 0
1 sent and incremented ping_pong_count 10 to 0
```

2. Simple MPI iSend and iRecv

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

```
#include<stdlib.h>

#include"mpi.h"

//int MPI_Isend(void *buf,int count , MPI_Datatype datatype,int dest , int tag ,MPI_Comm
comm,MPI_Request *request);

//int MPI_Irecv(void *buf,int count , MPI_Datatype datatype,int source , int tag ,
MPI_Comm comm,MPI Request *request)

//int MPI_Wait(MPI_Request *request , MPI_Status *status)



int main(int argc,char* argv[])

{

int numtasks,rank,next,prev,buf[2],tag1=1,tag2=2;

tag1=tag2=0;

MPI_Request reqs[4];

MPI_Status stats[4];

MPI_Init(&argc,&argv);

MPI_Comm_size(MPI_COMM_WORLD,&numtasks);

MPI_Comm_rank(MPI_COMM_WORLD,&rank);

prev=rank-1;

next=rank+1;

if(rank==0)

prev=numtasks-1;

if(rank==numtasks-1)

next=0;

MPI_Irecv(&buf[0],1,MPI_INT,prev,tag1,MPI_COMM_WORLD,&reqs[0]);

MPI_Irecv(&buf[1],1,MPI_INT,prev,tag2,MPI_COMM_WORLD,&reqs[1]);

MPI_Isend(&rank,1,MPI_INT,next,tag2,MPI_COMM_WORLD,&reqs[2]);
```

```
MPI_Isend(&rank,1,MPI_INT,next,tag1,MPI_COMM_WORLD,&reqs[3]);

MPI_Waitall(4,reqs,stats);

printf("Task %d communicated with tasks %d & %d\n",rank,prev,next);

MPI_Finalize();

return 0;

}
```

```
19/10/2021  16:20.07  /home/mobaxterm/Desktop/19BCE1027_PCD  mpicc -o lab82 lab82.c

19/10/2021  16:20.26  /home/mobaxterm/Desktop/19BCE1027_PCD  mpirun -n 3 ./lab82
Task 0 communicated with tasks 2 & 1
Task 1 communicated with tasks 0 & 2
Task 2 communicated with tasks 1 & 0
```

3. **Simple MPI Bcast**

```
#include <mpi.h>

#include <stdio.h>


int main(int argc, char** argv) {

    int rank;

    int buf;

    const int root=0;


    MPI_Init(&argc, &argv);

    MPI_Comm_rank(MPI_COMM_WORLD, &rank);


    if(rank == root) {

      buf = 777;

    }
```

```
    printf("[%d]: Before Bcast, buf is %d\n", rank, buf);


    /* everyone calls bcast, data is taken from root and ends up in everyone's buf */
    MPI_Bcast(&buf, 1, MPI_INT, root, MPI_COMM_WORLD);


    printf("[%d]: After Bcast, buf is %d\n", rank, buf);


    MPI_Finalize();
    return 0;
}
```

```
19/10/2021  16:04.26  /home/mobaxterm/Desktop/19BCE1027_PCD  mpicc -o lab83 lab83.c

19/10/2021  16:10.51  /home/mobaxterm/Desktop/19BCE1027_PCD  mpirun -n 3 ./lab83
[0]: Before Bcast, buf is 777
[0]: After Bcast, buf is 777
[1]: Before Bcast, buf is 6475108
[1]: After Bcast, buf is 777
[2]: Before Bcast, buf is 6475108
[2]: After Bcast, buf is 777
```

4. Implementation of Bcast using MPI send and recv

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


void my_bcast(void* data, int count, MPI_Datatype datatype, int root,
            MPI_Comm communicator) {
  int world_rank;
  MPI_Comm_rank(communicator, &world_rank);
  int world_size;
  MPI_Comm_size(communicator, &world_size);
```

```
  if (world_rank == root) {
    // If we are the root process, send our data to everyone
    int i;
    for (i = 0; i < world_size; i++) {
      if (i != world_rank) {
        MPI_Send(data, count, datatype, i, 0, communicator);
      }
    }
  } else {
    // If we are a receiver process, receive the data from the root
    MPI_Recv(data, count, datatype, root, 0, communicator, MPI_STATUS_IGNORE);
  }
}

int main(int argc, char** argv) {
  MPI_Init(NULL, NULL);

  int world_rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

  int data;
  if (world_rank == 0) {
    data = 100;
    printf("Process 0 broadcasting data %d\n", data);
```

```
    my_bcast(&data, 1, MPI_INT, 0, MPI_COMM_WORLD);
  } else {
    my_bcast(&data, 1, MPI_INT, 0, MPI_COMM_WORLD);
    printf("Process %d received data %d from root process\n", world_rank, data);
  }

  MPI_Finalize();
}
```

```
19/10/2021  16:11.21  /home/mobaxterm/Desktop/19BCE1027_PCD  mpicc -o lab84 lab84.c

19/10/2021  16:11.45  /home/mobaxterm/Desktop/19BCE1027_PCD  mpirun -n 3 ./lab84
Process 0 broadcasting data 100
Process 1 received data 100 from root process
Process 2 received data 100 from root process
```