Chapter 3

Point-to-Point Communication

3.1 Introduction

Sending and receiving of *messages* by processes is the basic MPI communication mechanism. The basic point-to-point communication operations are *send* and *receive*. Their use is illustrated in Example 3.1.

Example 3.1 A simple 'hello world' example usage of point-to-point communication.

```
#include "mpi.h"
int main(int argc, char *argv[])
{
  char message[20];
  int myrank;
  MPI_Status status;
  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
  if (myrank == 0)    /* code for process zero */
  {
      strcpy(message,"Hello, there");
      MPI_Send(message, strlen(message)+1, MPI_CHAR, 1, 99,
               MPI_COMM_WORLD);
  }
  else if (myrank == 1)  /* code for process one */
  {
      MPI_Recv(message, 20, MPI_CHAR, 0, 99, MPI_COMM_WORLD, &status);
      printf("received :%s:\n", message);
  }
  MPI_Finalize();
  return 0;
}
```

In Example 3.1, process zero (`myrank = 0`) sends a *message* to process one using the *send* operation MPI_SEND. The operation specifies a *send buffer* in the sender memory from which the *message data* is taken. In the example above, the send buffer consists of the storage containing the variable `message` in the memory of process zero. The location, size and type of the send buffer are specified by the first three parameters of the send operation. The message sent will contain the 13 characters of this variable. In addition, the send operation associates an *envelope* with the message. This *envelope* specifies the message destination and contains distinguishing information that can be used by the *receive* operation to select a particular message. The last three parameters of the send operation, along with the rank of the sender, specify the *envelope* for the message sent. Process one