Laboratory 6

In this project, the procedures will be made locally because of an unresolved error in executing mpirun that orted shows despite the successful configuration and connection by ssh of 2 virtual machines.

_
The number of processors will be 2, so the flag -np of mpirun will be specified with 2.

1. **Description of measurement methodology.**
   The following variables will be created to measure.

```
 double initTimeBegin, endInitTime; //to calculate initiation time
 double initSend, endSend; //to calculate bandwith when sending
double initReceive, endReceive; //to calculate bandwith when receiving.
```

   - **Initiation time:** this is assumed to meant of the time that takes to initialize all needed variables and MPI fields:

```
initTimeBegin = MPI_Wtime();
int rank,ranksent,size,source,dest,tag,i,len;
char name[20];
len=20;
MPI_Status status;
MPI_Init(&argc,&argv);
MPI_Comm_rank(MPI_COMM_WORLD,&rank);
MPI_Comm_size(MPI_COMM_WORLD,&size);
MPI_Get_processor_name(name,&len);

endInitTime = MPI_Wtime(); // end of the initialization of all required fiels of MPI
```

   - **Bandwidth**: it can be measured in MB/sec so in this case, it is measured both bandwidth of sending and receiving:

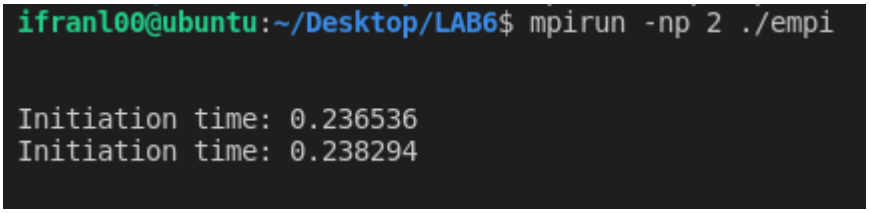
```
initSend = MPI_Wtime(); //begins sending
MPI_Send(&rank,1,MPI_INT,i,tag,MPI_COMM_WORLD);
endSend = MPI_Wtime(); //ends sending
printf("\n Time of sending: %lf\n",endSend-initSend);
```

```
initReceive = MPI_Wtime(); //begins receiving
MPI_Recv(&ranksent,1,MPI_INT,0,MPI_ANY_TAG,MPI_COMM_WORLD,&status);
printf("\nData to process %d recieved from %d on %s\n",rank,status.MPI_SOURCE,name);
endReceive = MPI_Wtime(); //ends receiving
printf("\n Time of receiving: %lf\n",endReceive-initReceive);
```

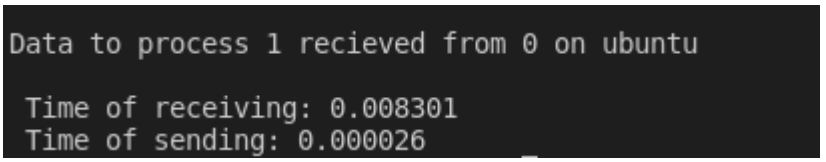
2. **Test program code.**

In order to test the explained measurements, as can be seen in the images detailed before, I used printf() to show by console in a easy and fast way the values of the involved times that we can know thanks to MPI_Wtime.

Using the variables initTimeBegin and endInitTime, we can use MPI_Wtime to subtract this times and obtain the initiation time that may differ because of two processes involved:



Also, using the function of MPI_Wtime() we can subtract and obtain the following times:



And knowing that in the functions of receive and send of MPI there is a field that tells the amount that is going to be transmitted, we can know what is send and received in the buffer, the rank:

*int MPI_Comm_rank(MPI_Comm comm, int *rank)*

Because what is transmitted is an int, it can be stored in 2 or 4 bytes, we are going to assume that is 4 bytes.

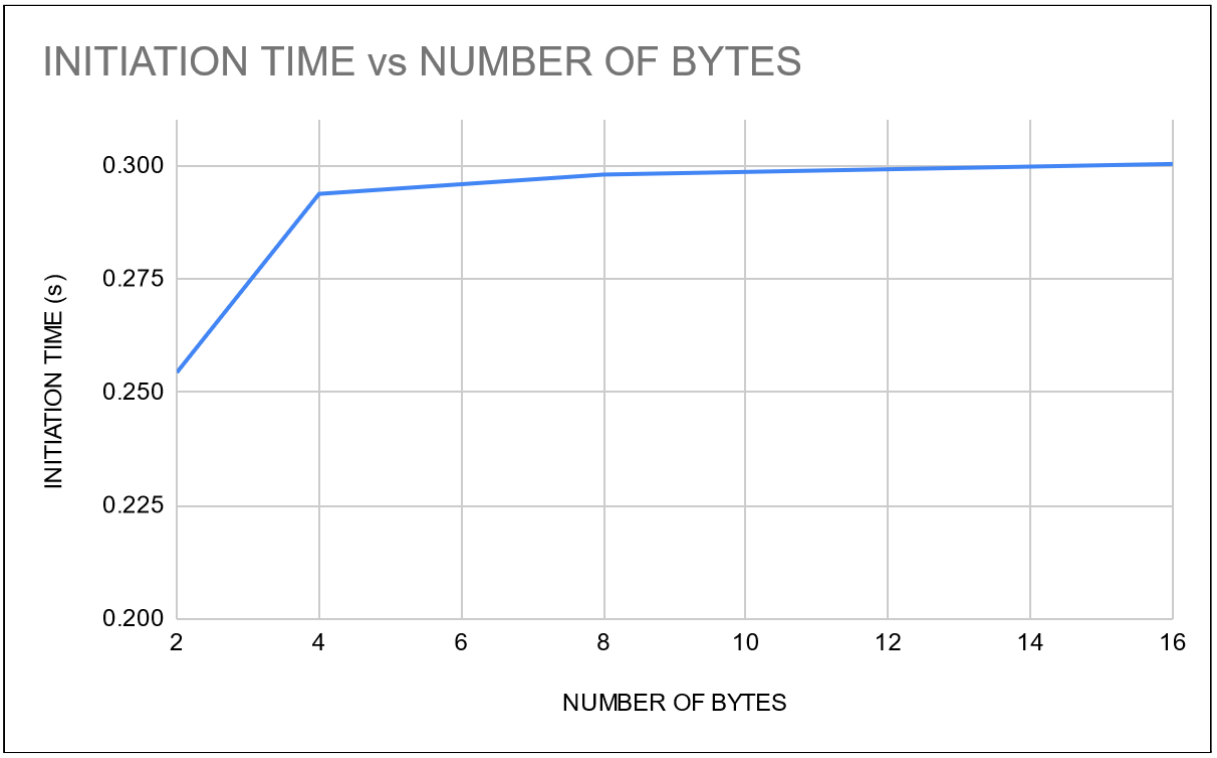
So as bandwidth of sending: 0.000026 s/4 bytes.
Bandwidth of receiving: 0.0008301 s/4 bytes.

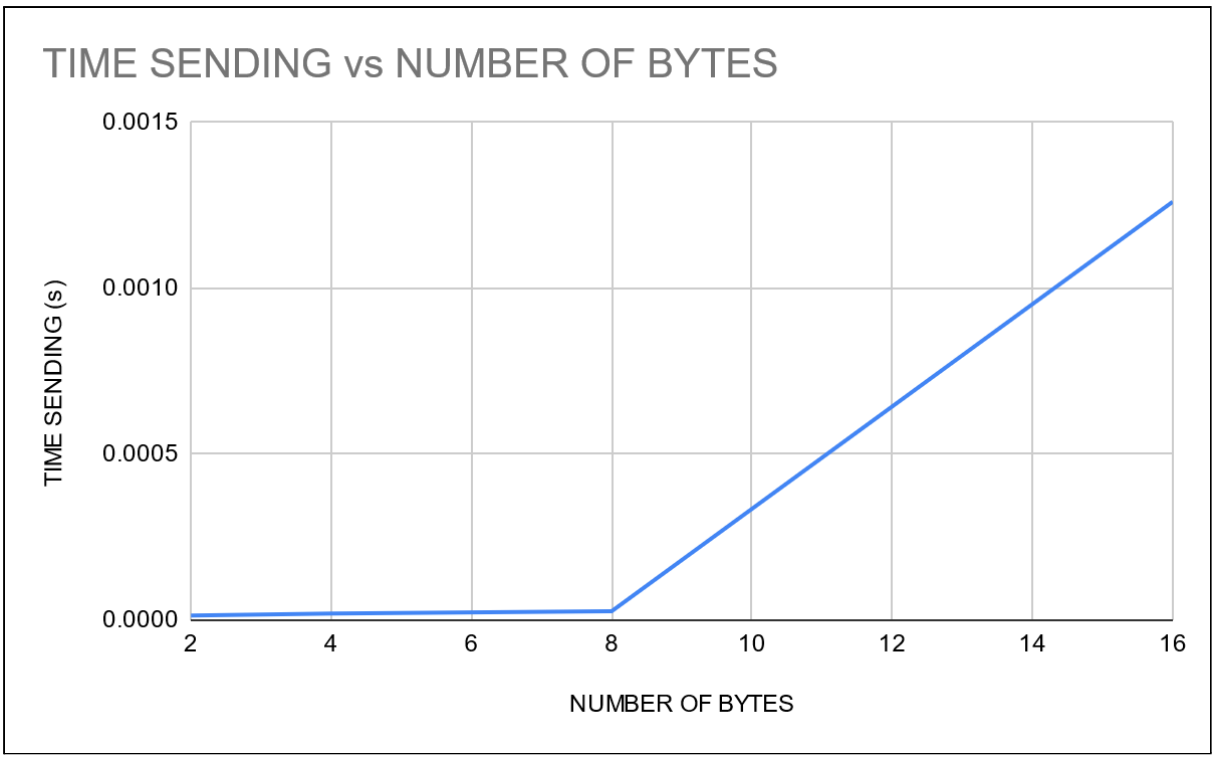
In this assignment, also was required to test bandwidth with different message sizes, so let's change rank and ranksent to the following ones -also setting MPI_DOUBLE or MPI_INT in the type in MPI_Send and MPI_Receive:

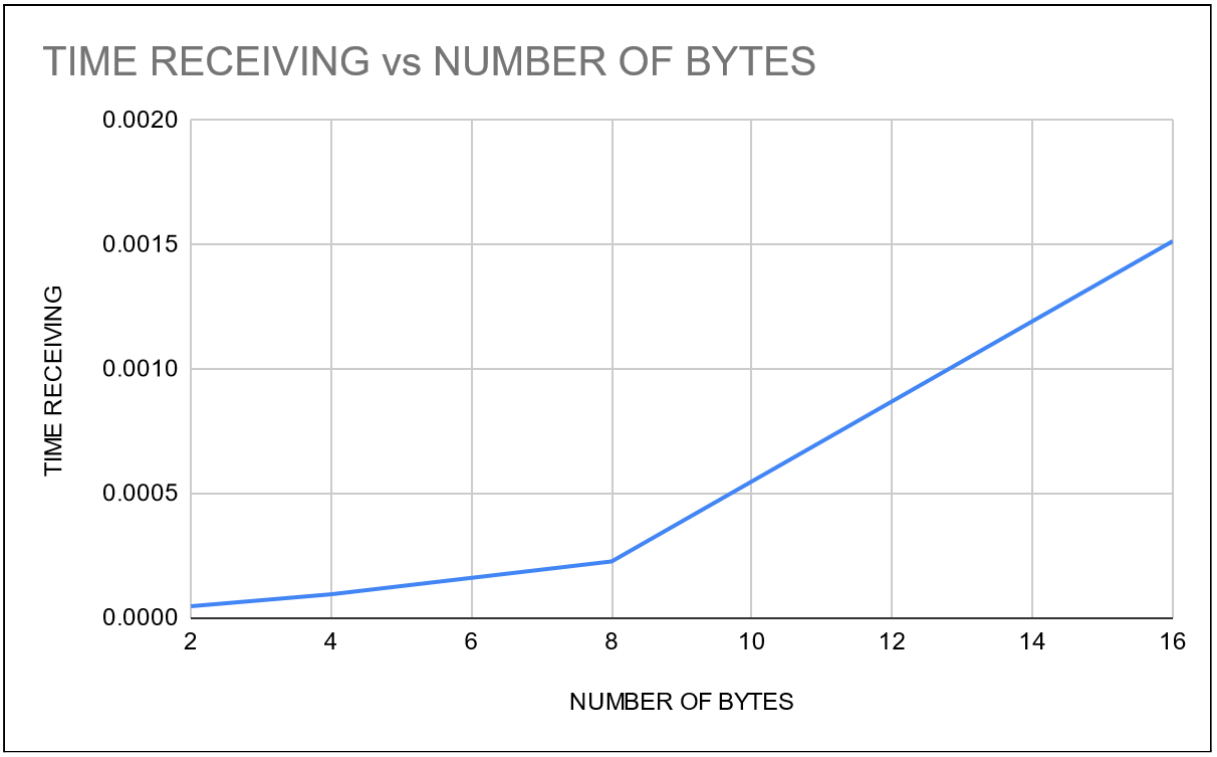
```
/* CHANGING SIZE OF MESSAGE
 double temp,tempSent; //8 bytes
short int temp,tempSent; // 2 bytes
int temp,tempSent; // 4 bytes
long double temp,tempSent; //16 bytes
*/
```

3. Test results in the form of tables and graphs.

NUMBER OF BYTES	INITIATION TIME	TIME SENDING	TIME RECEIVING
2	0.254402	0.000014	0.000048
4	0.293829	0.00002	0.000096
8	0.298088	0.000027	0.000228
16	0.300434	0.001261	0.001515







4. Analysis of the results and conclusions.

As we can see and it could be logical to think, time of sending and receiving will be bigger if the message size is bigger.

Also we can appreciate that the time of sending is faster than receiving because it takes time to take the information from the buffer and save it in the corresponding variable and setting the status of the operation.

In fact, even the initiation time may be bigger depending on the type of variables that we are using.

Because of the specifications of computers and the small size of the message, it is done really fast that even taking the bandwidth simplified just as the sum of sending and receiving time, it will not reach a single second.

The processor involved is an i7-8750H with a theoretical speed to read or write of 41.800 MB/s. In this experiment we test the time in seconds vs bytes ¿is it actually related with the theoretical value given in the specifications of our processor?

Let's calculate it to the sending and receiving of a short int that only involves 2 bytes.

Given 41.800 Mb per second, what time would it take for 2 bytes?

- 0.0000478 sec.

If we add the sending and receiving time obtained 0.000062 sec approximately, that is not really far from the theoretical result.