CS633: Parallel Computing
Assignment 1

Group 14 : Akshat Sharma (190090), Yash Gupta (190997)

Code

1. RUN_ME.sh : executes the files (run.py) and (make_plots.py)
2. code.c : Contains the MPI code to transmit data between two processes using send-receive. Transmission time is computed using the MIP_Reduce() function and saved to the (times.txt) file.
3. run.py : This python file creates mpi processes for each configuration, i.e. for transmission between each host pair and for each data size.
4. make_plots.py : Reads the (times.txt) file and computes the transmission time averaged over multiple runs and pairs of nodes. Plots are written to the directory (plots/)

Produced Files

1. plots/ : Plots from the experiments. The two types of plots produced are:
    a. Plot with the data size as the x-axis : This plot shows the transmission time for each data size, averaged over multiple runs and across different servers. The vertical bars indicate the standard deviation in the transmission time collected for that data size. The plot shows an approximately linear trend between transmission time and data size. However, the transmission time for the data sizes (0.25KB, 25KB and 0.25MB) are imperceptible.
    b. Plot with scaled x-axis : The figure shows the transmission time with the x-axis scaled to highlight the trend for smaller data sizes.
2. times.txt : The transmission times are written to this file.
3. code : The ELF binary produced upon compiling (code.c)
4. logs.txt : Debug info for (run.py)

Run Script

`RUN_ME.sh` is the main script:

```
#!/bin/bash

python3 ./run.py
python3 ./make_plots.py
```

How to run

1. Add the following line to `~/.bashrc` and `~/.profile` : `ulimit -s unlimited`
2. Mention 3 pairs of hosts (6 hosts) in the file `hosts.txt`.
3. Make `RUN_ME.sh` executable using the command : `chmod +x RUN_ME.sh`
4. Run it using the command : `./RUN_ME.sh`

The transmission time for each data size will be saved in `times.txt` and the plots are saved at (`plots/`)

Experimentation Methodology

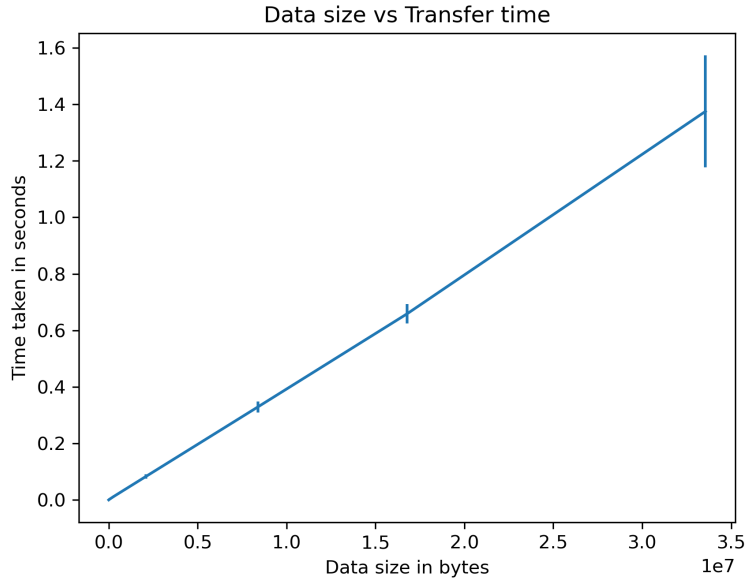
1. code.c contains a standard MPI_Send, MPI_Recv transmission pair, where one process sends to the other. The transmission segment is timed using MPI_Wtime() and is reduced using MPI_Reduce().

```
sTime = MPI_Wtime();
if(!myrank){
    MPI_Send(buf, count, MPI_INT, 1, 1, MPI_COMM_WORLD);
}
else if(myrank == 1){
    MPI_Recv(buf, count, MPI_INT, 0, 1, MPI_COMM_WORLD, &status);
}
eTime = MPI_Wtime();
time = eTime - sTime;
```

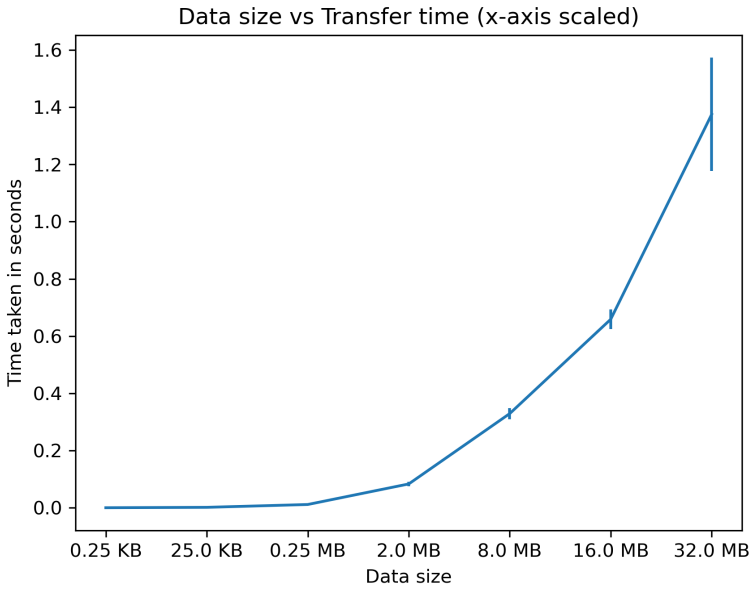
2. Each configuration, i.e. a host pair and data size, is run 10 times and the transmission times are recorded in the file `times.txt`.
3. As an example, we choose three host pairs and seven data sizes ranging from 0.25KB to 32MB. Hence there are 3 X 7 configurations, run 10 times each, resulting in 210 values for transmission time. The transmission time for each data size is averaged over the 30 values obtained across host servers and multiple runs.
4. To show the variance in the values obtained we plot the mean and standard deviation of the transmission time. The standard deviation is shown as a vertical line around the mean observation.
5. Two versions of the plot are saved to `plots/` as (data_vs_time.png) and (data_vs_time_scaled.png). The second plot shows the transmission time with the x-axis scaled to highlight the trend for smaller data sizes.

Plots

1. Data size vs Transmission time : Note the linear trend between the data size and transmission time for moderate data sizes. Transmission time is expected to scale linearly with data size below the bandwidth bound.



2. Data size vs Transmission time (x-axis scaled) : Note that the transmission time for small data sizes (0.25KB - 0.25MB) are roughly the same. This is due to the transmission being latency bound for smaller message sizes.



Observations

1. Transmission time statistics :

S.no.	Message size	Average transmission time (sec)	Standard deviation (sec)
1.	0.25 KB	0.0004	0.0001
2.	25.0 KB	0.0018	0.0008
3.	0.25 MB	0.0117	0.0017
4.	2.0 MB	0.0834	0.0079
5.	8.0 MB	0.3292	0.0194
6.	16.0 MB	0.6587	0.0347
7.	32.0 MB	1.3747	0.1979

2. Note the linear trend between the data size and transmission time for moderate data sizes. For moderate data sizes the transmission time is expected to scale linearly with data size as the transmission is neither latency nor bandwidth bound.
3. Note that the transmission time for small data sizes (0.25KB - 0.25MB) are roughly the same. This is due to the transmission being latency bound for smaller message sizes. This is apparent in the transmission time not increasing in proportion to the data size in the range (0.25KB - 0.25MB). A 100X increase in the message size leads to only a 4.5X increase in transmission time.
4. Fitting the Hockney model to the data we can obtain the latency of transmission (L) and bandwidth (B). L = 6.424 milliseconds, B = 25.429 MB/s