CS633A

Parallel Computing

Assignment I

Halo Exchange

Rohit Raj
20111051

Mani Kant Kumar
20111030

Instructor:
Dr. Preeti Malakar

1 Overview

In this assignment we compare the performance of halo exchange with neighbouring processes using three methods:

- Multiple MPI_Send/MPI_Recv are used, each MPI_Send transmits only 1 element (1 double here).
- MPI_Pack/MPI_Unpack and MPI_Send/MPI_Recv to transmit multiple elements at a time
- MPI_Send/Recv using MPI derived datatypes (vector)

Every process exchange data with its neighbors. In case of non-existing neighbours the boundary processes need not exchange data. In many cases, the individual sub-domains logically overlap along their boundaries and cells in these overlapping regions (called halo cells) and they need to be iteratively updated with data from neighboring processes; a communication pattern which we refer to as halo-exchange. We performed the experiment for the following configurations:

- P (number of processes) = 16 36, 49, 64
- N (data points per process) = 16^2, 32^2, 64^2, 128^2, 256^2, 512^2, 1024^2
- ppn(process per node) = 8
- numtimesteps = 50

2 Experimental Setup

We need buffers to store the exchanged data from neighboring processes so we created an array of (N+2)*(N+2) dimension instead of N*N. The additional rows and columns will act as buffer to that process. For instance if we have N=16 data points then we dynamically allocated 6*6 matrix so that we can store the exchanged data at boundaries instead of creating buffers for each. We initialised the matrix with randomly generated positive numbers and assigned -1 to outer boundaries to differentiate between them as shown below:

X	-1	-1	-1	-1	X
-1	151165686	140526222	6884861	6510410	-1
-1	2926850	2532135	246654	79627	-1
-1	10864143	9623814	656336	107325	-1
-1	1214763	1065497	131963	70584	-1
X	-1	-1	-1	-1	X

- To generate boxplot we used python preferably(python3) and its libraries like numpy, matplotlib.pyplot, statistics.

3 Instructions to run the code

- chmod +x run.sh (in case root access is required)
- ./run.sh

4 Explanation of code

After initializing N data points at time_step 0 and assigning boundary cells to -1, we divided the set of processes into three categories:

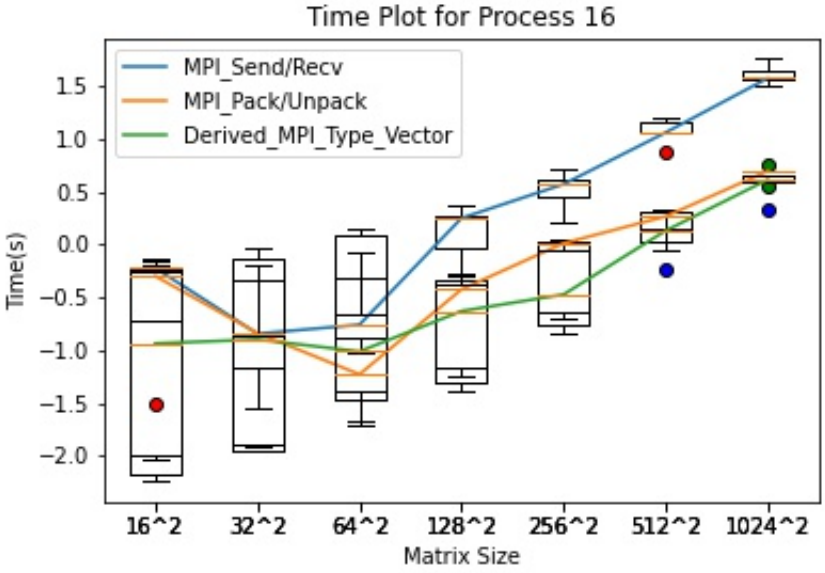
- corner process that exchange with 2 neighbors i.e one column and one row. The corner processess are named as "tl" - top left, "tr" - top right, "bl" - bottom left, "br" - bottom right
- process(boundary process other than corner) that exchange with 3 neighbors
- center process that exchange with all 4 neighbors.

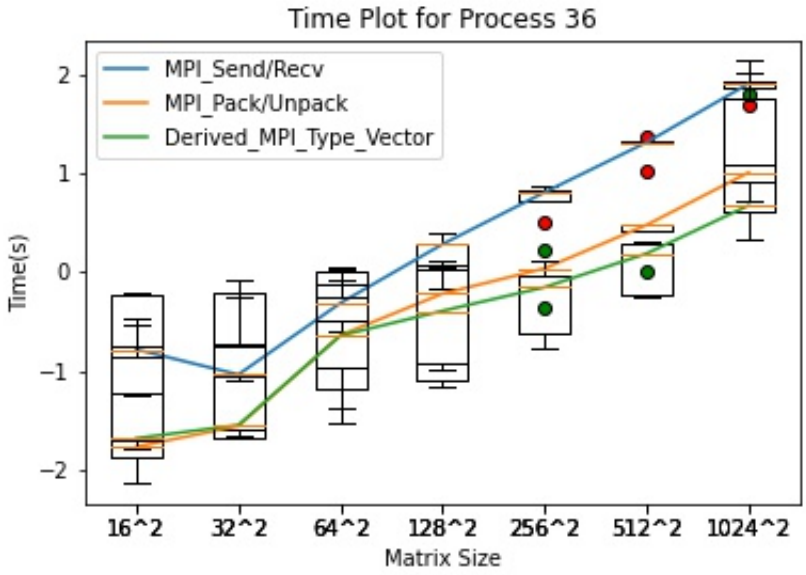
Every process performs stencil computation followed by communication for the boundary points/cells. Next time_step value at a point/cell P is computed as the average of the neighbouring cells of P (left, right, top, bottom). We also created a temporary matrix to differentiate the computed value and previous value of matrix during computation.

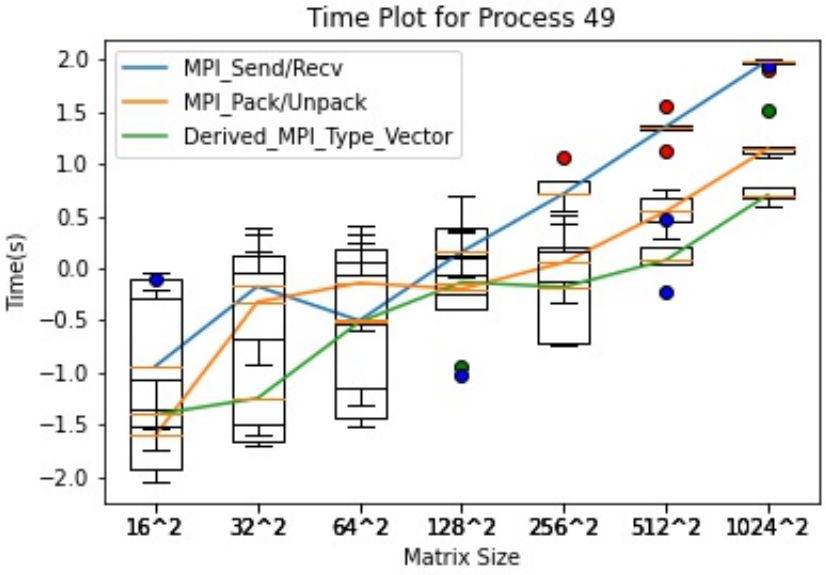
The functionality of individual files submitted:

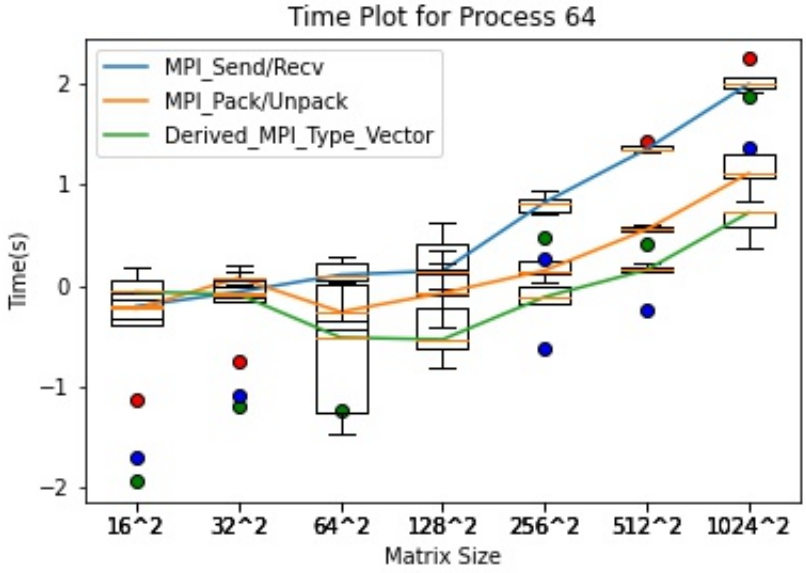
- **run.sh** : This job script runs all the configurations mentioned in the assignment. It first compiles the code using the "make" command and then generates machinefile / hostfile on-the-fly based on the node status so that jobs never fail. After generation of hostfile, the script performs the Halo exchange experiments for all the mentioned configurations. The generated data* files are used to generate boxplot for each P(number of processes).
- **Makefile** : compile the code named "src.c" using the "make" command
- **hostfile.sh** : search out all the online nodes from the allhost files and store it in the hosts file.
- **src.c** : contains the code in C to perform the given experiment and generate 4 data files.
- **plot.py** : generates 4 boxplot (named as plotP.jpg) using the generated data files.

5 Plots


Time Plot for Process 16
MPI_Send/Recv
MPI_Pack/Unpack
Derived_MPI_Type_Vector
Time(s)
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
16^2
32^2
64^2
128^2
256^2
512^2
1024^2
Matrix Size


Time Plot for Process 36
MPI_Send/Recv
MPI_Pack/Unpack
Derived_MPI_Type_Vector
Time(s)
2
1
0
−1
−2
16^2
32^2
64^2
128^2
256^2
512^2
1024^2
Matrix Size


Time Plot for Process 49
MPI_Send/Recv
MPI_Pack/Unpack
Derived_MPI_Type_Vector
Time(s)
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
16^2
32^2
64^2
128^2
256^2
512^2
1024^2
Matrix Size


Time Plot for Process 64
MPI_Send/Recv
MPI_Pack/Unpack
Derived_MPI_Type_Vector
Time(s)
2
1
0
−1
−2
16^2
32^2
64^2
128^2
256^2
512^2
1024^2
Matrix Size

6 Observations

The time taken for Halo exchange for a particular N data points decreases as the number of processes increase because by increasing number of processes the workload gets divided and hence reduces the processing and execution time.
In general the order of execution time for 3 configurations are:

- MPI_Send/MPI_Recv < MPI_Pack/MPI_Unpack < Derived_MPI_Type_Vector