CS633: Group 01: Assignment 2

Manasvi Jain - 210581 - manasvij21@iitk.ac.in
Sarthak Kalankar - 210935 - sarthakk21@iitk.ac.in
Saugat Kannojia - 210943 - saugatk21@iitk.ac.in

March 2024

Explanation of Code

- Initialised the data matrix with random values based on the arguments provided
- Checked for existing neighbours by making flags for them
- Reduced the number of inter-node exchanges by assigning a leader process and sending only the desired data points
- For the above communication in the case of top and bottom rows during halo exchange, the following order is maintained:
    1. MPI_Comm_split to create sub-communicator for a node to enable the operations of Gather and Scatter
    2. MPI_Pack for packing the boundary data points
    3. MPI_Gather for all processes within a node in the sub-communicator for gathering all the packed rows and sending them to the first process
    4. MPI_Isend for sending the packed data from the first process to the first processes of the neighbouring nodes if they exist
    5. MPI_Recv for receiving the packed data in the first process of the node
    6. MPI_Unpack to unpack the received packed data
    7. MPI_Scatter to send those received rows in all the processes from the first process in the sub-communicator
- The halo exchange for left and right rows, and the computations of data matrix were done in the same fashion as before.

Performance Comparison

Observation 1:

When the number of processes increases keeping the nodes fixed, the communication time increases as the number of intra-node communications increases.

Observation 2:

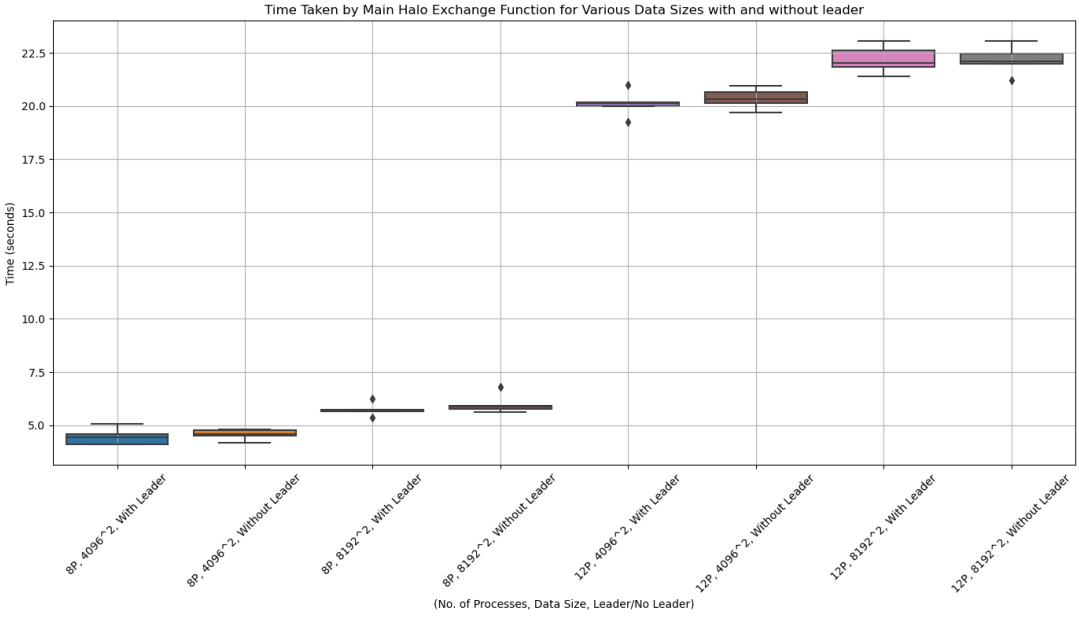
Increasing the data size increases the total time taken because more amount of data points get packed and communicated as well as more computation time.

Observation 3:

The slight increase in the time with leader for 8 and 12 processes can be explained by the fact that there are not that many processes such that the reduction in inter-node communications has an effect on the timings and the Gather-Scatter increases the time slightly. However for large number of processes, this works fine as the time would be smaller with leader process as compared to without a leader.

Optimizations

We have used Isend/Recv to ensure non-blocking message passing takes place accurately which gives better performance over blocking calls. We have used MPI_Gather/Scatter over normal Send/Recv since it has an optimised parallel algorithm. We tried to allocate the first process of every node as the leader so that the number of inter-node communications get reduced for a large number of processes.


Time Taken by Main Halo Exchange Function for Various Data Sizes with and without leader
Time (seconds)
22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
8P, 4096^2, With Leader
8P, 4096^2, Without Leader
8P, 8192^2, With Leader
8P, 8192^2, Without Leader
12P, 4096^2, With Leader
12P, 4096^2, Without Leader
12P, 8192^2, With Leader
12P, 8192^2, Without Leader
(No. of Processes, Data Size, Leader/No Leader)