README

23 February 2021

1 Code Explanation:

The code performs Halo exchange where the domain is divided into P subdomains. The communication between the P process is performed using three methods,

1) Multiple MPI-Isends, each MPI-Isend transmits only 1 element.
2) MPI-Pack/MPI-Unpack and MPI-Irecv.
3) MPI Derived Data types.

- The processes can be categorised into 3 types depending upon their position i.e.

1) Corner 2) Edges 3) Center.

Also, each of the directions UP, RIGHT, BOTTOM and LEFT is represented by numbers 0,1,2,3 respectively, which is used as tag parameter to determine from which neighbour, data was received. Considering each process has N^2 datapoints, we maintained a receive buffer of size 4 x N for each process. Thus , the element recvbuffer[i][j] denotes the j^{th} datapoint received from neighbour in the i^{th} direction.In addition to this, as integar array neighbour[4] of size 1 x 4 is maintained for each process. Here, neighbour[i]=1 denotes that a neighbour exists in the i^{th} direction. Each process uses MPI_Irecv to receive data from its neighbours.

- direct_communication():
This function calls multiple MPI_Isend to communicate with its neigbours. The receiver process uses MPI_Irecv to receive the data and stores it in the receive buffer.

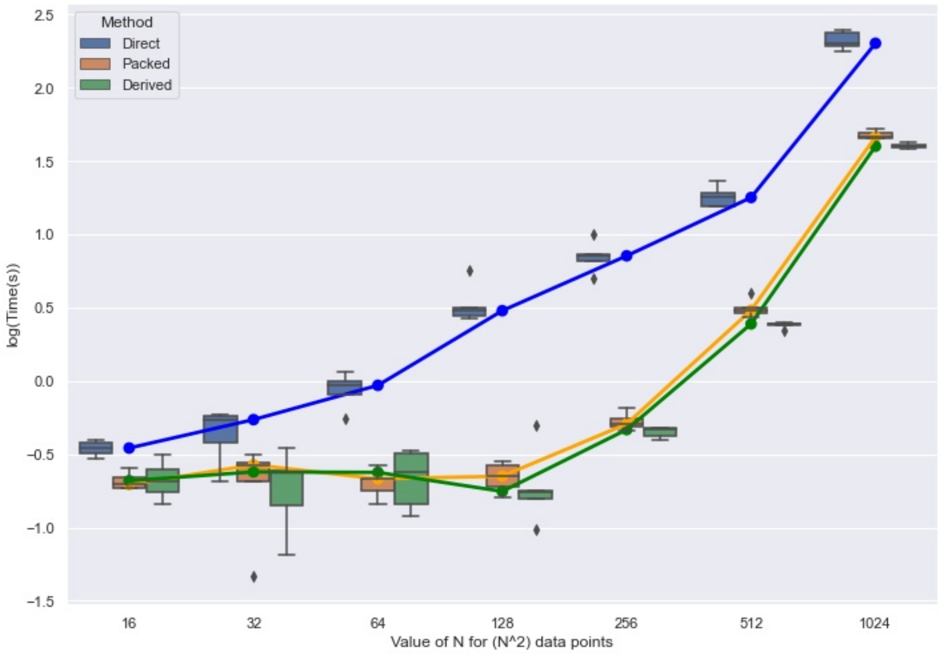
- packed_communication():
This function uses MPI_PACK to pack elements in a row/column together and then sends them using a single MPI_Isend.

-derived_communication():
This function uses MPI_Type_vector() datatype to send non contiguous datapoints in a single MPI_Isend.

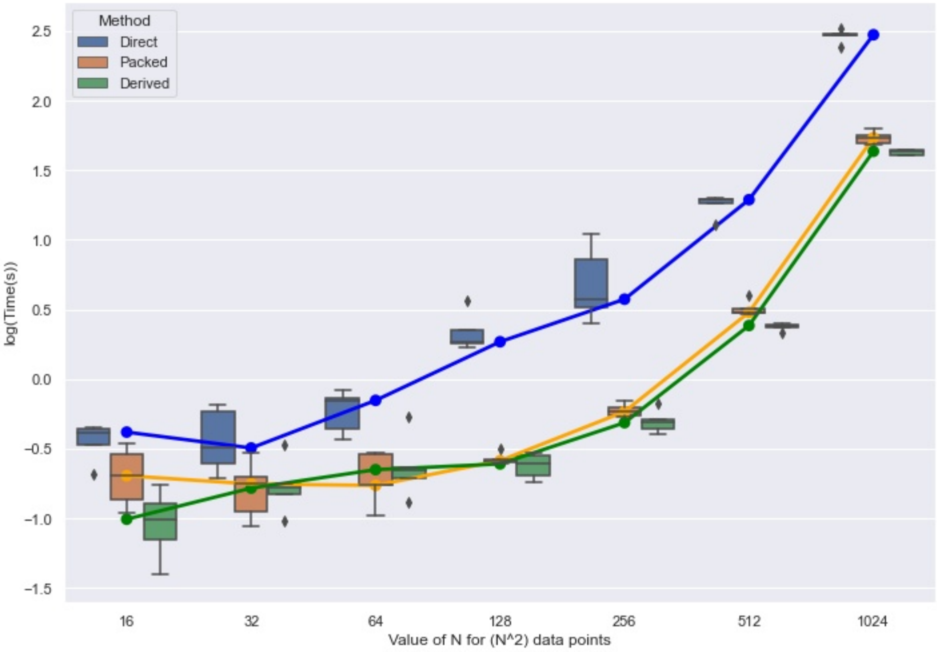
-computation():
After the processes have completed the communication and received the datapoints from their neighbours, a call to this function does the stencil computation. The communication() functions call the function count() to obtain the number of valid neighbours of a data point.

Each process performs communication and computation alternately, for 50 iterations, for given values of P and N. We record the time taken by each method and write it into a file for the purpose of plotting.

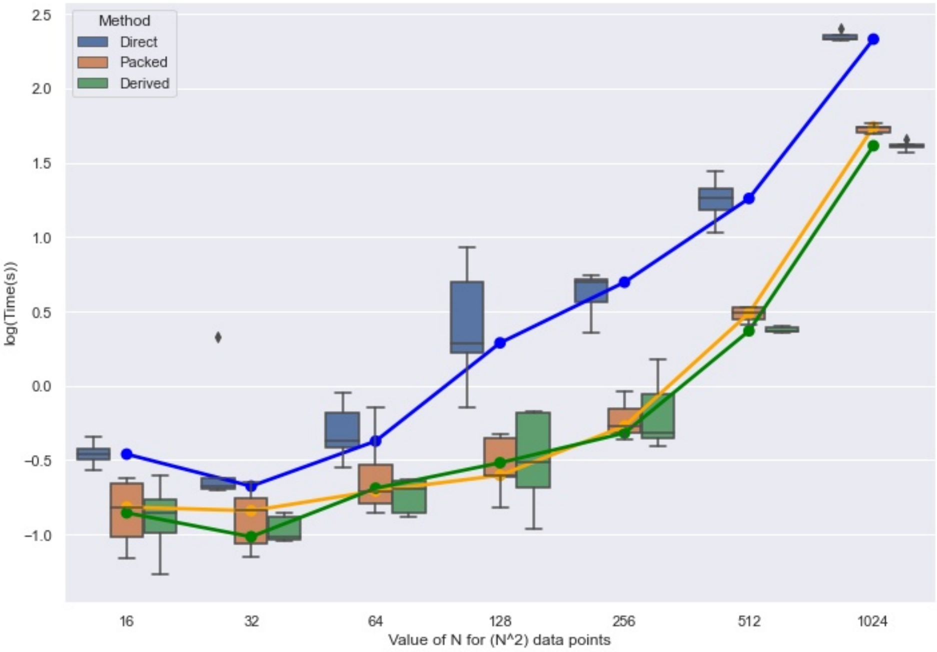
1.0.1 Plot for P=16



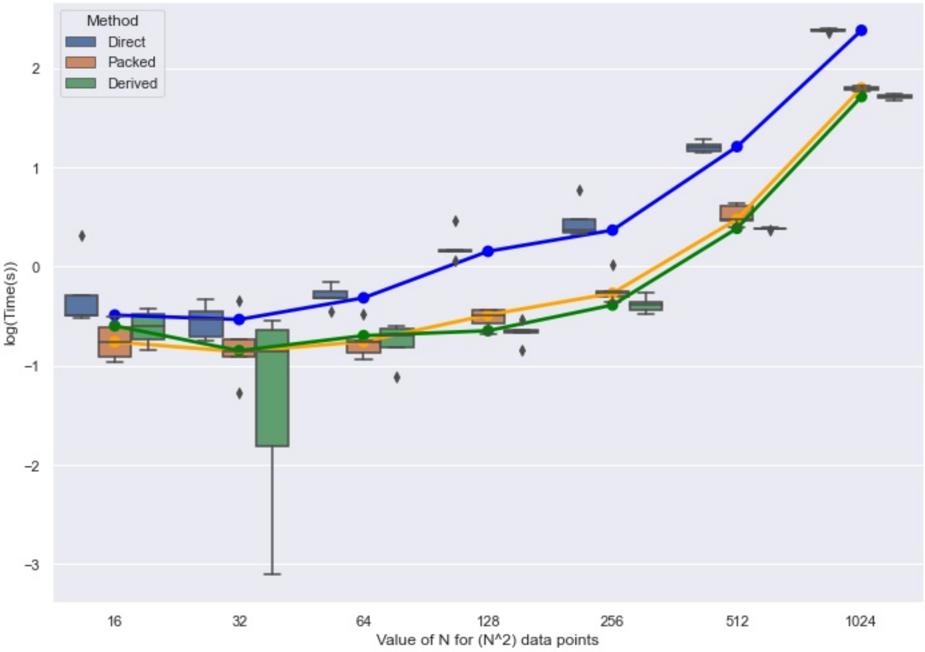
1.0.2 Plot for P=36



1.0.3 Plot for P=49



1.0.4 Plot for P=64



1.1 Observation:

(1) All methods would do same amount of computations. Thus, the difference between the 3 methods, is in the amount of time needed for communication done.

(2) The first method(multiple MPI_Isends) takes more time than the other two methods for all 4 cases. It implies that, it spends major part of its time communicating with its neighbours. We can infer that method 1 is less efficient as compared to other 2 methods.

(3) Since method 2 and 3 both use a single MPI_Isend to send a row/column, both of their plots appear to be identical. Thus, not much difference is observed in their performances.