# CS633:Assignment1

## Performance Analysis of Halo Exchange Function

**GROUP 1**

March 21, 2024

## 1 Code Explanation

This document explains the parallel implementation of stencil calculation on large matrices using 12 processes using halo exchange. The program performs halo exchange operations followed by stencil computation across multiple processes.

1. **Initialization:** Each process initializes its data points randomly according to the problem statement using a seed value provided as a command-line argument.
2. **Halo Exchange:** Processes exchange boundary data points with neighbouring processes using `MPI_Isend` and `MPI_Irecv`. Only existing neighbours participate in the halo exchange. `MPI_Pack` and `MPI_Unpack` are used for adding data to sending and receiving buffers.
3. **Stencil Computation:** After halo exchange, each process performs stencil computation to update its data points. For data points lying in the halo regions, corresponding buffers with data from neighbouring processes are used.
4. **Time Measurement:** The program measures the time taken for halo exchange operation and stencil calculation using `MPI_Wtime`. This measurement excludes the data initialization step.

## 2 Halo Exchange

We focus on how the halo exchange program is implemented in our program.

- We use four 2D buffers of 4*N dimensions corresponding to the top, bottom, left and right halo region data points.
- In the case of 5 stencil size, only 1st and 3rd rows are used, while the entire buffer is used for 9 stencil size.
- The first half of each buffer acts as `send_buf` and second half of each buffer acts as `recv_buf`.

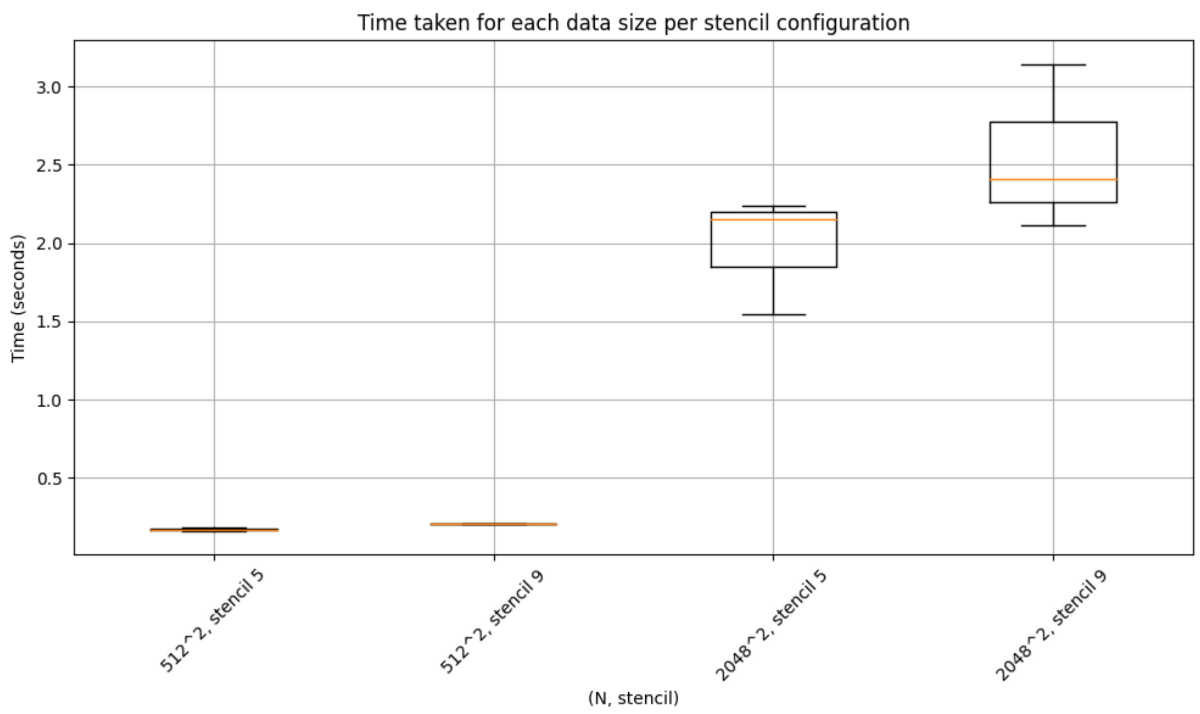
The following steps give an overview of the communication:

1. Pack the values in the halo region into the first half of each buffer.
2. Using `MPI_Isend` and `MPI_Irecv` for communication, the first half of the buffers act as a sending buffer while the second half receives data points.
3. The received data in second half of buffers is unpacked in the first half.

## 3 Performance Observations

From the boxplot of timings, following observations can be made regarding the performance of the halo exchange function:

- The execution time increases with increased data size (N) and stencil size.
- Variability in execution time is observed due to other processes running on the `csews` machines and the differing times taken for communication between the nodes.
- The data communicated is doubled when the stencil size is increased from 5 to 9, but the time taken isn't doubled for both the data sizes.
- The time taken for $2048^2$ data size is 8-14 times of $512^2$ data size.



## 4 Optimizations

The following optimizations have been used within the code to improve performance:

- We use non-blocking communication to allow overlapping communications.
- The send and recv buffers are reused at each timestamp and also for packing and unpacking of datapoints.

## 5 Group Information

- **Group Number:** 1
- **Group Members:**
  - Om Shivam verma (Roll Number: 210684, Email: omsv21@iitk.ac.in)
  - Piyush Bhatia (Roll Number: 210720, Email: piyushb21@iitk.ac.in)
  - Lakshvant Balachandran (Roll Number:210557, Email:lakshvant21@iitk.ac.in)