CS 633A Assignment 2

Gautam Chauhan, Sreejit Bose (20111020, 20111065)

gautamc20, sreejit20@iitk.ac.in

March 27, 2021

Code takes 4 arguments as follow:

1. data size
2. No of groups
3. Nodes per group
4. Cores per node

This help code understand the physical topology of network and optimize the code using it. Code create multiple communicative, start from bottom:

- **Inter-node**: a communicative for all the processes inside a node and create a *node leader*.
- **Inter-group**: create a leader(group leader) from each inter-node with-in a group and create a communicative for them.
- **Intra-group**: create a leader(intra-group leader) from each inter-group and create a communicative for them.

This creates an hierarchy of processes, mapping the physical network topology.

Section 1.

Bcast

To optimize the Bcast we used the knowledge of physical topology as follows:

- The root node broadcasts the message to all the leaders of the groups using a communicator which includes only leaders of groups.
- These group leaders then communicateb the message to all node leaders in its group.
- Node leaders then communicate the message to all the cores in that node.

This gives improvements with respect to the default MPI Bcast.

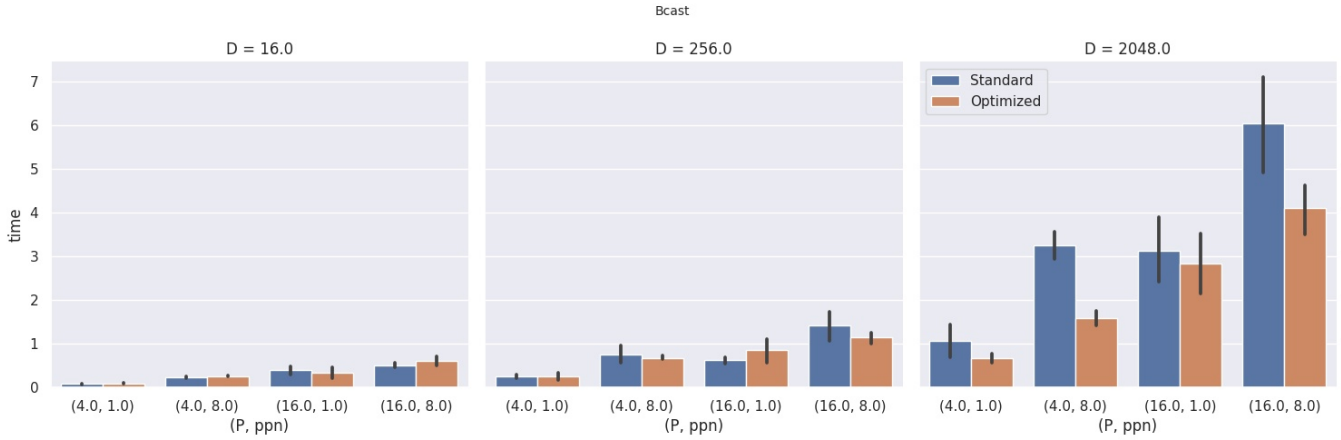

Figure 1: Bcast

Section 2.

Reduce

To optimize the Reduce we used the knowledge of physical topology as follows:

- Each node used default $MPIReduce$ to make the data available to node leaders .
- Each node leaders used default $MPIReduce$ to make the data available to group leaders.
- Each group leaders used default $MPIReduce$ to make the data available to the root node.

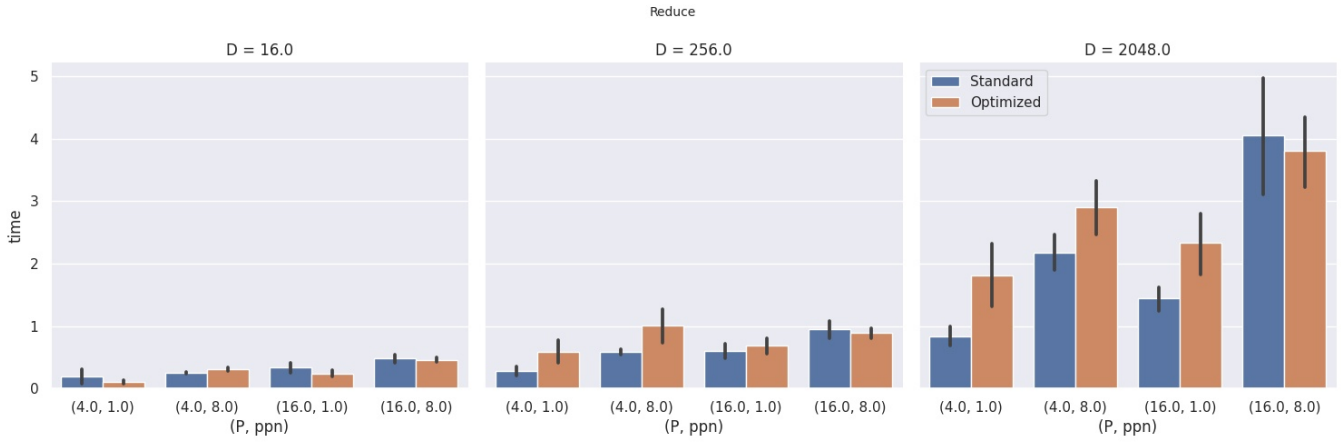

Figure 2: Reduce

Section 3.

Gather

To optimize the Reduce we used the knowledge of physical topology as follows:

- Each node used default *MPIGather* to make the data available to node leaders .
- Each node leaders used default *MPIGather* to make the data available to group leaders.
- Each group leaders used default *MPIGather* to make the data available to the root node.

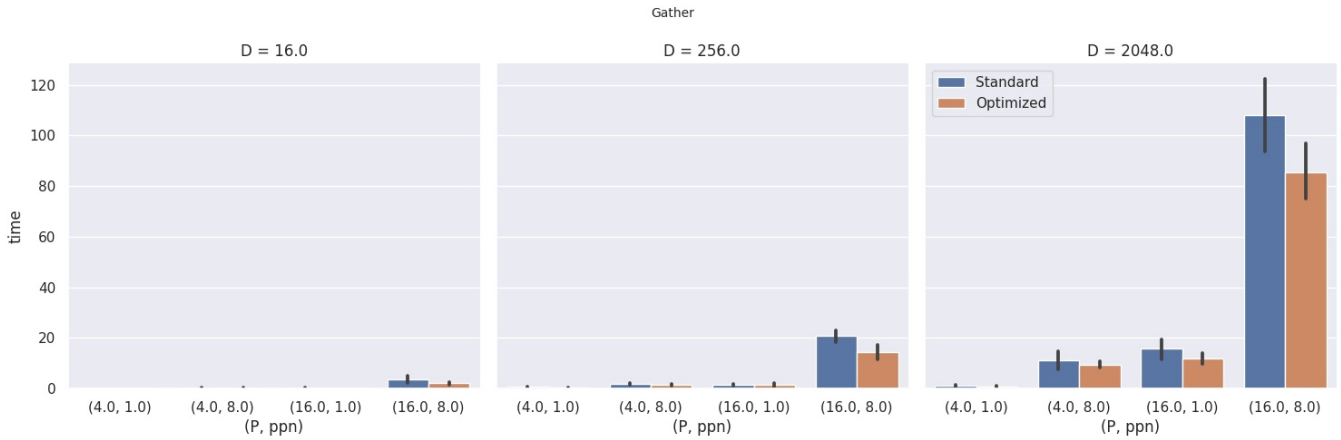

Figure 3: Gather

This gives improvements with respect to the default MPI Gather.

Section 4.

Alltoallv

To optimize the Alltoallv and use physical topology, intialily we created mechanism, in which alltoallv was done in following steps:

- All Node leader use *MPI Gatherv* to gather *sendbuf* from all the core to single core(node leader).
- All group leader use *MPI Gatherv* to gather *sendbuf* from all the node leader to single group leader.
- Group leaders use *MPI Allgatherv*, so all group leaders have all the sendbuf from both groups.
- Now all Group leaders need to do multiple *MPI Scatterv* to send relevant buffers to node leaders.
- Then node leaders do multiple *MPI Scatterv* to send relevant buffers to cores(end processes).

Problems: We implemented the above the above approach but it was slower by a lot. Problems are as follows:

- Due to complexity of alltoallv and makes it very complicated to multiple buffers.
- While initial gatherv, leader have to store buff from multiple process and higher the leader more data it need to store. This creates an problem for groups leaders.

We decided to use simpler approach by using multiple MPI Scatterv. Fig 4 shows it's performance comparison.

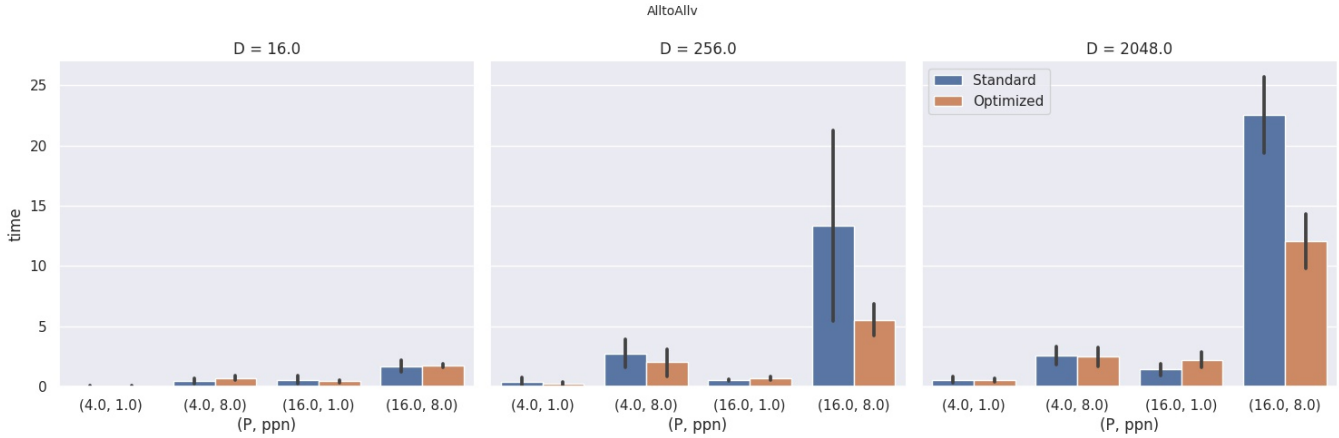

Figure 4: Alltoallv