Fast adjacent communication among with RDMA, MPI-RMA, and Double buffering

Kota Yoshimoto, Akihiro Fujii, Teruo Tanaka
Kogakuin University

Introduction

In large-scale scientific computing programs, parallelization by MPI communication is generally used. MPI is convenient because it can be executed on many computers. However, inter-process communication often becomes a bottleneck in highly parallel computers. Therefore, research is being conducted on optimizing communication[1]. In this research, we measured and compared the performance of MPI, MPI_RMA, RDMA, and these method with double buffering for adjacent communication on the supercomputer Wisteria-O.

Adjacent Communication

Large-scale scientific and engineering calculations frequently require the solution of the simultaneous equation Ax=b. When solving Ax=b in parallel by domain decomposition, it is necessary to obtain the elements in different domains. Adjacency communication is employed to communicate the elements.

➢ Double Buffering

- This method is used for one-sided communication.
- It does not require synchronization by using two buffers for communication.

➢ MPI

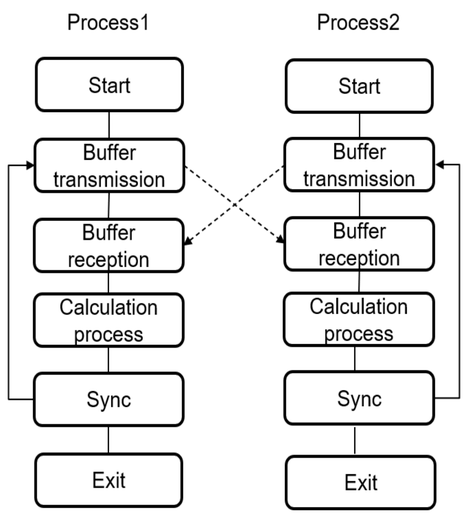
- Two-way communication using MPI_Isend/Irecv, MPI_Waitall (MPI_Isend/Irecv)
- One-sided communication using MPI_Put and MPI_Win_fence (MPI_RMA)
- Double buffering is also evaluated with one-sided communication using MPI_Put and MPI_Win_lock/unlock (MPI_RMA_DB)

➢ RDMA

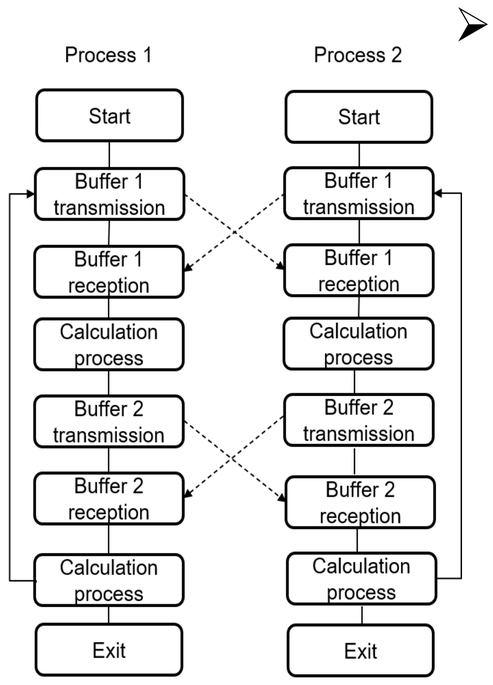
RDMA communication can read and write data without the intervention of the program of the destination node by using a dedicated memory. RDMA is one-sided communication.

In RDMA communication, it is necessary to synchronize so that the data will not be overwritten while it is being used.

- RDMA is evaluated by uTofu_put and MPI_Barrier (RDMA)
- RDMA using double buffering is also evaluated (RDMA_DB)

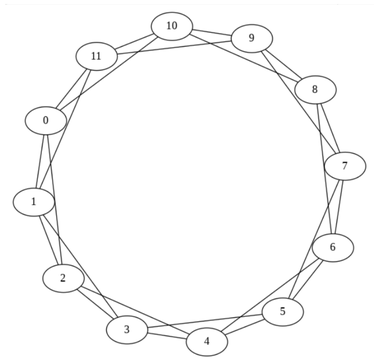


One-sided communication (MPI_RMA, RDMA)

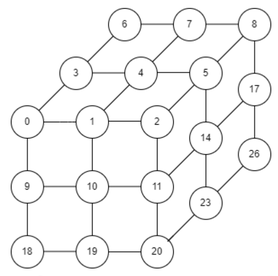


One-sided communication with double buffering (MPI_RMA_DB, RDMA_DB)

➢ Communication setting



The figure is1D topology.



The figure is 3D topology.

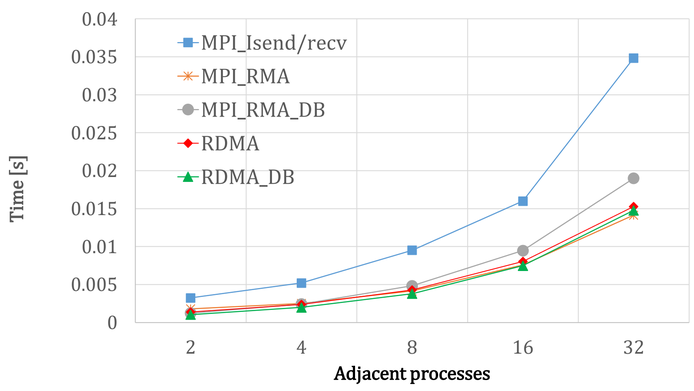
Numerical experiment

In this experiment, we used a supercomputer Wisteria-O at University. It is equipped with a system called FX1000.

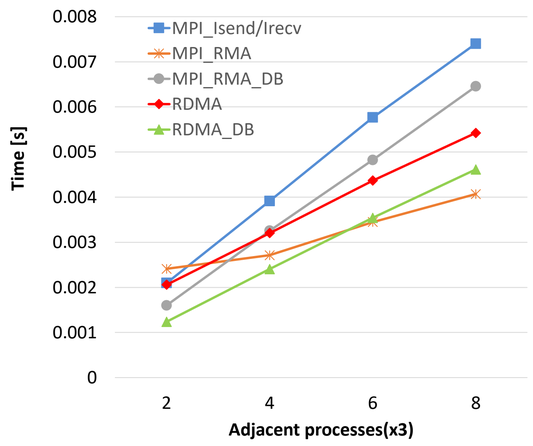
When changing the number of adjacent processes in adjacent communication with 1D topology

Experiment Condition

- The number of processes is 128 with 32 nodes.
- 100 double precision data was communicated between processes.



As a result, all one-sided communication methods had similar performance and were faster than MPI_Isend/Irecv.

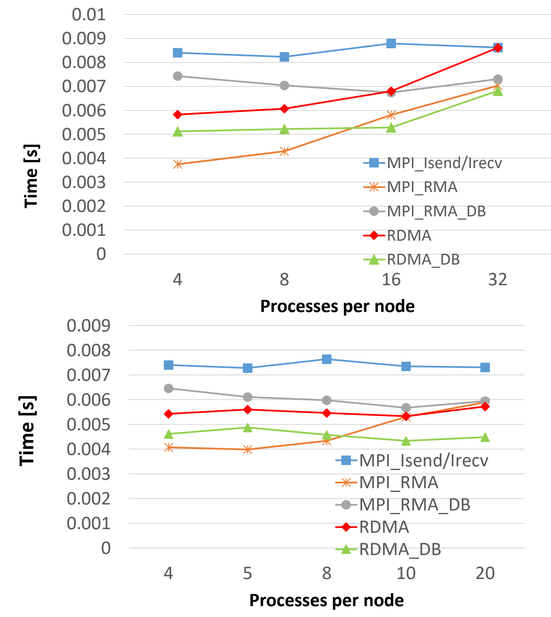


When changing the number of adjacent processes in adjacent communication with 3D topology

Experiment Condition

- The number of processes is 1000 with 250 nodes.
- The data size was set random from 10 to 100 double precision data.

When the number of adjacent processes is small, RDMA_DB is the fastest, and when the number of adjacent processes is large, MPI_RMA is the fastest.



The results for different number of processes per node

The figure above shows communication when using 1D topology.

1D topology experiment Condition

- The number of adjacent processes is 32.
- The data size was set random from 10 to 100 double precision data.

The figure below shows communication when using 3D topology.

3D topology experiment Condition

- The number of adjacent processes is 24 (3*8).
- The data size was set random from 10 to 100 double precision data.

In both cases, MPI_RMA was the fastest when processes/node was small, and RDMA_DB was fastest when processes/node was large.

Conclusion

On the FX1000, we evaluated the performance of two-way communication using MPI, one-sided communication, and one-sided communication using RDMA. Under our experimental conditions, the one-sided communication method was faster than MPI_Isend/Irecv in most cases. When the number of processes per node was four and the number of adjacent processes was large, RMA using MPI_Put and MPI_Win_fence was found to be the most efficient. However, when the number of adjacent processes was small or when the number of processes per node was large, RDMA using double buffering are also effective.

Acknowledgments

This work is supported by “Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures” and “High Performance Computing Infrastructure” in Japan (Project ID: jh230060-NAH).

Reference

[1]金森逸作，中村宜文，似鳥啓吾，辻美和子，向井優太，三吉郁夫，松古栄夫，石川健一，低レイテンシuTofuインターフェースを用いたQCD計算における通信の高速化，情報処理学会研究報告，Vol.2020-HPC-177 No.22 (2020).