Towards SDN-enhanced Message Passing Interface

Cybermedia Center, Osaka University, Japan

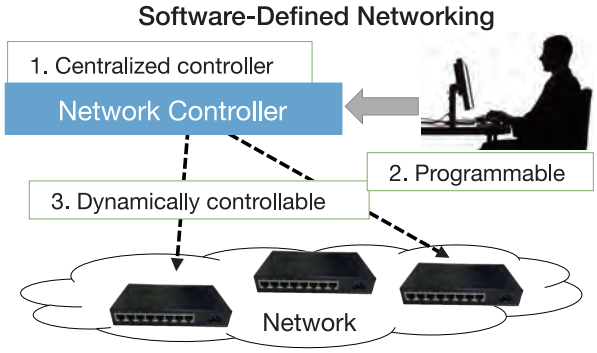
Towards SDN-enhanced Message Passing Interface

Background

Nowadays, parallel computing technique is an inseparable technology for High Performance Computing (HPC). Message Passing Interface (MPI) has been a *de facto* standard programming model in parallel computing for around two decades. MPI offers two types of communication; one is *one-to-one communication*, which is for low-level description of communication pattern, and the other one is *collective communication*, which is for high-level and human-friendly description.

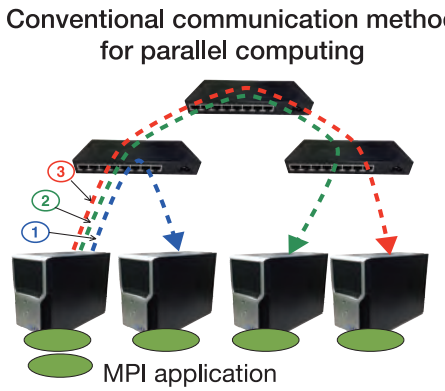
Software-Defined Networking

Software-Defined Networking (SDN) is a new concept of network architecture that decouples conventional networking function into a programmable control plane (responsible for deciding how to control the packets) and a data plane (responsible for the actual packet delivery). Currently, OpenFlow is the most common implementation of SDN, which enables to dynamically control the forwarding functionality of network devices from a centralized controller.



Problem

However, collective communication of MPI often suffers from performance degradation when it is used on HPC environment based on commodity hardware, such as Gigabit Ethernet. One of the main cause is that most of the MPI implementations are not optimized for such hardware. For example, when a process wants to broadcast some data to other processes (MPI_Bcast : a process sends data to all other processes), multiple times of transmission will happen.



Research Goal

Integrate the *dynamic* controlling ability of Software-Defined Networking into MPI in order to optimize *collective communication* by overturning the assumption that network is a static resource. Ultimately, we'd like to implement a new MPI library, which cuts down communication latency and traffic amount by programmatically controlling the underlying network.

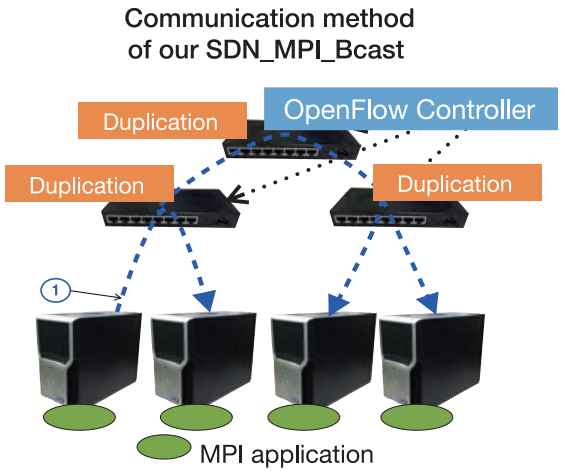
Progress Report

MPI_Bcast [1]

Design:

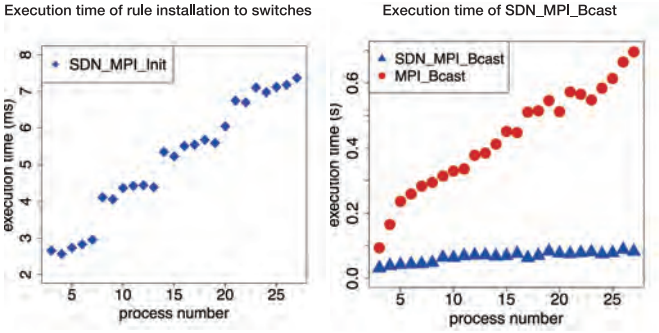
1. OpenFlow controller obtains MPI-cluster network's topology with LLDP.
2. IP addresses of the node which will broadcast and all other receiving nodes are sent to the controller.
3. Controller builds a broadcast tree with the information gathered in step 1 and 2. An algorithm based on Floyd-Warshall's method is used in this process.
4. A Set of packet duplication rules are generated, which instructs to flow packets from a source process to other processes on the broadcast tree in step 3. These rules are deployed to OpenFlow compatible network switches.
5. Broadcast node sends packets to its network.

Every packet being broadcasted includes a specific ID, which are unique to the combination of source node and destination nodes. Packet copy rules will only match if the incoming packet includes the corresponding ID



Current Result:

Reduced execution time of MPI_Bcast.



MPI_Reduce

Design:

1. Controller analyzes network topology with LLDP and traffic usage with OpenFlow's flow statistics. Network link usage information is used to select one route from multiple redundant routes in step 3.
2. Size of the array being reduced is examined, so that an optimal reducing algorithm can be selected from flat-tree, binary-tree, etc.
3. Using these data, an optimized "Reduction tree" is built.
4. Packet forwarding rules are generated and installed in the same way as MPI_Bcast.

[1] Khureltulga Dashdavaa, Susumu Date, Hiroaki Yamanaka, Eiji Kawai, Yasuhiro Watashiba, Kohei Ichikawa, Hirotake Abe and Shinji Shimojo. Architecture of a High-speed MPI_Bcast Leveraging Software-Defined Network. In UCHPC2013: The 6th Workshop on UnConventional High Performance Computing 2013, Aachen, Germany, August 27, 2013.

Contact: Khureltulga Dashdavaa, huchka@ais.cmc.osaka-u.ac.jp
Keichi Takahashi, takahashi.keichi@ais.cmc.osaka-u.ac.jp