Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186-5140
Volume 8, Number 2, pages 139–142, July 2019

A Two-Opt Collective-Communication Method for Low-Latency Random Network Topologies

Ke Cui
School of Multidisciplinary Sciences
The Graduate University for Advanced Studies
2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, Japan 101-8430
cuike@nii.ac.jp

Michihiro Koibuchi
Information Systems Architecture Science Research Division
National Institute of Informatics
2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, Japan 101-8430
koibuchi@nii.ac.jp

***Abstract*—Random network topologies have been proposed as low-latency network for parallel-computing systems. Although their unicast routing algorithms have been well researched, collective communication methods that consist of a large number of unicasts are not well optimized for random network topologies. In this study, we apply a two-opt method to collective communication on random network topologies. It attempts to minimize the execution time of collective communication. Simulation results interestingly show that our collective communication using the two-opt operation outperforms by up to 15% in terms of network latency the existing topology-agnostic collective communications that attempt to minimize the number of network contentions and those used in MPI implementations.**

***Index Terms*—Collective communication, random network topology, interconnection network, high performance computing (HPC)**

I. Introduction

The communication latency is one of performance bottlenecks for large parallel systems. Today's large supercomputers have hundreds of thousands of compute nodes, and the latency in the communication affects the overall performance of the system. To reduce the communication latency, the network topology of a parallel system should have low diameter and low average shortest path length (ASPL). However, a few network topologies, i.e. fat tree, torus, Dragonfly [1], [2], [3], have been used to interconnect compute nodes in parallel-computing systems. They have large ASPLs than the theoretical lower bounds for a given pair of nodes and degree [4]. In this context, random network topology has been considered for obtaining low diameter and low ASPL [5], so applying random network topology to parallel systems would improve system performance.

There are a large number of studies on unicast routing algorithms on arbitrary network topologies that include random network topologies [6]. However, collective communications are not well considered for random network topologies. Since collective communications are frequently used in parallel programs such as FFT(Fast Fourier Transformation) and CG(Conjugate Gradient) [7], they sometimes dominate the execution times of the parallel programs.

In this study, we apply the two-opt method to the collective-communication operation — broadcast. The broadcast operation has been defined in the message passing interface (MPI) standard. According to the definition of MPI, broadcast operation MPI_Bcast(void* buffer, int count, MPI_Datatype datatype, int root, MPI_Comm comm) broadcasts a message from the rank-root process to all processes of the group, itself included. [8]

There are many studies to implement those collective communication operations to improve the performance by considering the message length [9] or network topologies [10]. In the real product, MPICH2 [11] implements the broadcast and reduce operation by binomial tree algorithm for short messages. Through this work, we follow MPICH2 implementation as default: broadcast operation relies on the binomial tree algorithm. For binomial tree algorithm of broadcast, the root sends data to node $(root + (N/2))$, N is the number of nodes. This node and the root then act as new roots within their own subtrees and recursively continue this algorithm. The total communication takes $log\lceil N \rceil$ steps [9]. However, they are not well optimized in terms of ASPLs of unicasts that form a binomial tree.

Our main concern is to minimize the low ASPLs of unicasts that form a binomial tree on random network topologies. In this context, we apply the two-opt method to minimize the ASPLs of the unicasts. Then, we perform the broadcast operation and compare the execution time, if the execution time of broadcast operation which applied 2-opt method is smaller, it shows that 2-opt method can minimize the ASPLs. We pick up two unicasts that form the multicast, then swap the endpoints of two unicast source-destination pairs if the multicast algorithm can be completed and if the ASPLs become lower in this operation. Repeat the procedure until the ASPLs of unicasts cannot be smaller in the moderate number of attempts.

Our main contribution in this work is:

- The SimGrid event-discrete simulation results show that the collective communication operations optimized by the two-opt method reduced by 15% the execution time when compared to those used in MPICH2.

The rest of this paper is organized as follows. Background information and related work are discussed in Section II. Section III describes the details apply two-opt method to optimize collective communication. Section IV presents our evaluation

including simulation settings and experiment results. Section V concludes with a summary of our findings in this paper.

II. Existing Collective Communications

A. Hardware-, path- and unicast-based Multicast Techniques

Collective communications are implemented by multicast techniques. Hardware-, path-, and unicast-based multicast techniques are typical methods for multicasts in interconnection networks [12]. Hardware multicasts duplicate packets at an intermediate switch for a multicast. Since it reduces the aggregate packet hop counts in a multicast, it efficiently sends data to multiple destinations. A path-based multicast sends data along a path that includes all destinations, and thus requires an efficient multicast-path search, e.g., Hamiltonian cycle for broadcast.

Current conventional network products, such as InfiniBand, do not always support hardware- and path-based multicast techniques. In this work, we assume to use unicast-based multicasts for collective communications.

B. Optimization for Collective Communications

There are a large number of optimization methods for collective communication. Some optimization methods of collective communication were proposed by given a specific network topology. For example, an optimal broadcasting was proposed over hypercube network [13], and contention-free multicast algorithms were proposed on nCUBE-2 hypercube and 2-D mesh network [14]. The study in [12] proved that it is impossible to avoid contention channels in arbitrary irregular networks, and it proposed multicast algorithms to reduce the number of channel contentions. The optimization also had been done on supercomputer BlueGene/L with 3-D torus topology [15]. There are also many optimization methods of collective communication for heterogeneous HPC platform. The study in [10], [16] proposed a topology-aware algorithm to improve the performance of collective communication operations for large scale InfiniBand clusters. An optimization of collective communication was proposed by considering message length or number of nodes, in popular MPI implementation MPICH2 [11], available algorithm will be chosen by message length [10].

However, the existing optimization of collective communication is not optimized for random topologies in terms of ASPLs of unicasts that form a collective communication.

III. Two-Opt Collective-Communication Method

The 2-opt method was first proposed at 1958 to solve traveling salesman problem [17]. We apply the two-opt method for optimization of collective communications on random network topologies as follows.

We use the notation $x \to y, x \neq y$ to denote a unicast from node x to node y. Let $U = \{n_i \to n_j | n_i \neq n_j, n_i, n_j \in N\}$ stand for the set consisting of all of the unicasts that form the target collective communication operation.

The objective function is the total execution time of the target collective communication operation. Another candidate of the objective function is the total number of path hops of the unicasts in U, however, in this study we use more "precise" measure: the total execution time.

The following procedure attempts to minimize its value. The method proceeds in three steps. In the first step, start with a conventional unicast-based multicast technique that provide U, and compute the value of the objective function for U. In the second step, randomly pick two unicasts, $n_a \to n_i, n_i \to n_b | n_a \neq n_i \neq n_b, n_a, n_b \in S$, from N. Then, swap the endpoints of the two unicasts in U, $n_c \to n_j, n_j \to n_d | n_c \neq n_j \neq n_d, n_c, n_d \in N$. Compute the objective function for the new U. If its value becomes larger, then cancel this swapping. Repeat the second step until the objective function cannot be smaller in the moderate number of attempts.

To improve the efficiency of optimization, we can select more than two unicasts, then randomly swap the endpoints of those unicasts simultaneously, then compare the elapsed time of broadcast operation rather than the number of hops. In the first step, do the broadcast operation and then we can get the elapsed time. In the second step, select more than two unicasts, randomly swap the endpoints of those unicasts simultaneously, do broadcast operation and record the elapsed time, if the elapsed time becomes lager, then cancel those swapping. Repeat the second step until the elapsed time cannot be smaller.

Figure 1 shows the example of the behavior of the two-opt method on broadcast operation. We randomly pick up 2 nodes n_i and n_j, then inverse the index of nodes between node n_i and n_j, according to the binomial algorithm of MPICH2, this operation swap several endpoints of unicasts, so it can improve the speed to reduce U. Then do broadcast operation, if the elapsed time becomes smaller, it shows 2-opt method reduce U. Repeat the last step 1000 times until the elapsed time can not be smaller.

IV. Performance Evaluations

In this section we use discrete-event simulation to evaluate the performance of test MPI programs using the collective communication.

A. Methodology

We use the SimGrid simulation framework (v3.12) [18]. SimGrid implements validated simulation models, is scalable, and makes it possible to simulate the execution of unmodified parallel applications that use the Message Passing Interface (MPI) [19].

We use a shortest-path routing, i.e. Dijkstra's algorithm. Each switch has a 100 nsec delay. Switches and hosts are interconnected together via links with 100 Gbps bandwidth. Each host has 100 GFlops. We configure simgrid to utilize its built-in version of the MPICH2 implementation of MPI collective communications [11].

In the evaluation, we attempt to swap endpoints of unicast 1,000 times in the two-opt collective communication.

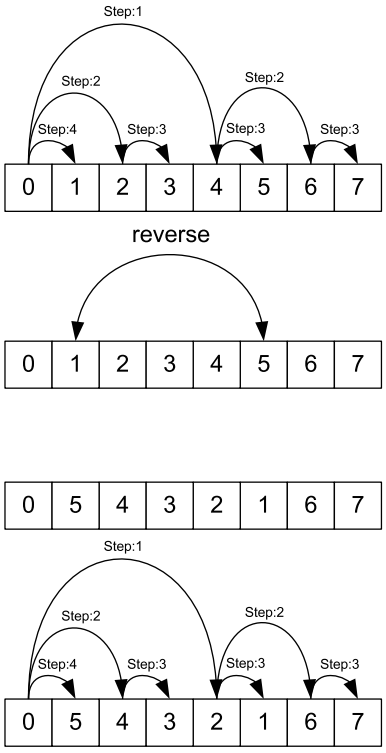

Figure 1. 2-opt method on binomial tree algorithm of Broadcast

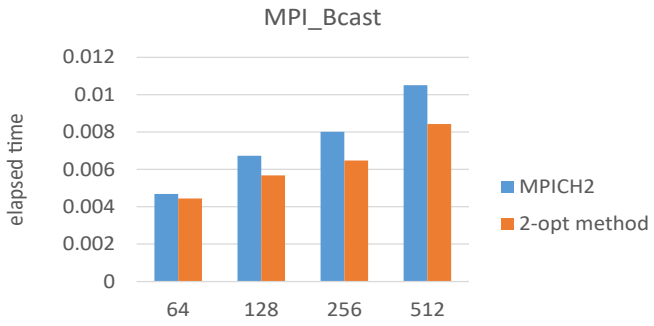

Figure 2. MPI_Bcast (message size = 64KB).

We apply the two-opt collective communication to broadcast operation, and we measure their execution time.

We simulate the programs that simply replace each MPI function 100 times, and we assume the flat MPI model. To implement the two-opt, we replace an MPI collective-communication functions with a number of MPI unicasts in the program considered in the evaluation.

B. Results

1) Network Size: The Figure 2 shows the elapsed time of broadcast operation for the MPICH2 and 2-opt method on random network. The y-axis represents the elapsed time and x-axis represents the network size. The network size is the number of switches, each switch has 1 host, and each host has 1 cores, the degree of network size is 8. We set the network size as 64, 128, 256, 512 separately and always set the message length as 64KB. As Figure 2 shows, the elapsed time of broadcast which applied 2-opt method has better performance than MPICH2. When network size is 64, the performance has been improved about 5%, but when networks is 512, the performance has been improved about 20%.

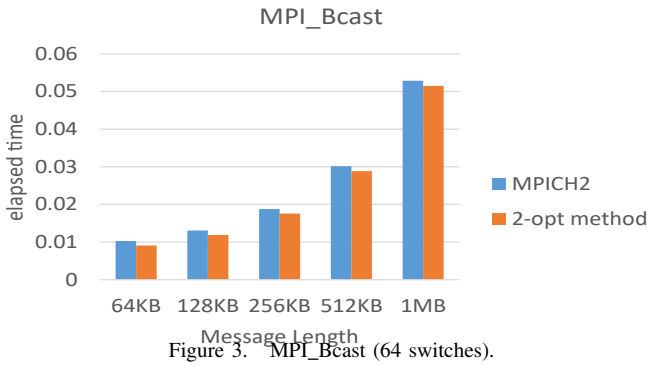

Figure 3. MPI_Bcast (64 switches).

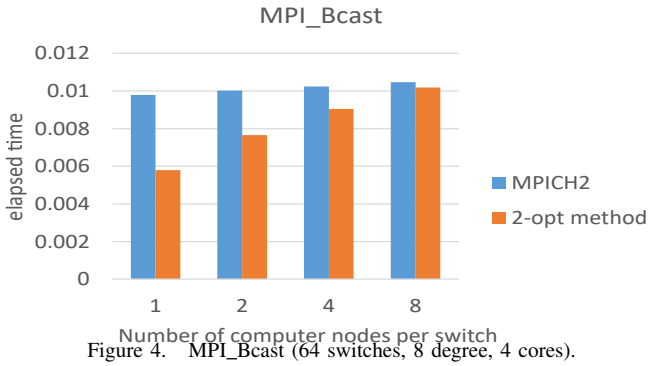

Figure 4. MPI_Bcast (64 switches, 8 degree, 4 cores).

2) Message Length: The Figure 3 shows the elapsed time of broadcast operation for the MPICH2 and 2-opt method on random network. The y-axis represents the elapsed time and x-axis represents the message length. We set the network size is 64, each switch has 4 host, and each host has 4 cores, the degree of network size is 8. We set the message length as 64KB, 128KB, 256KB, 512KB and 1MB. As Figure 3 shows, the elapsed time of broadcast which applied 2-opt method also has better performance than MPICH2. Performance has increased by an average of 4%.

3) Number of Compute Nodes per Switch and Cores per Node: The Figure 4 shows the elapsed time of broadcast operation for the MPICH2 and 2-opt method on random network. The y-axis represents the elapsed time and x-axis represents the number of compute nodes per switch. We set the network size is 64 and each host has 4 cores, the degree of network size is 8. We set the compute nodes per switch as 1, 2, 4 and 8. As Figure 4 shows, when the number of computer nodes per switch is 1, the performance has been improved about 40%, but when number of computer nodes per switch is 8, the performance has been improved about 2%.

The Figure 5 shows the relationship between cores per node and the elapsed time of broadcast. The y-axis represents the

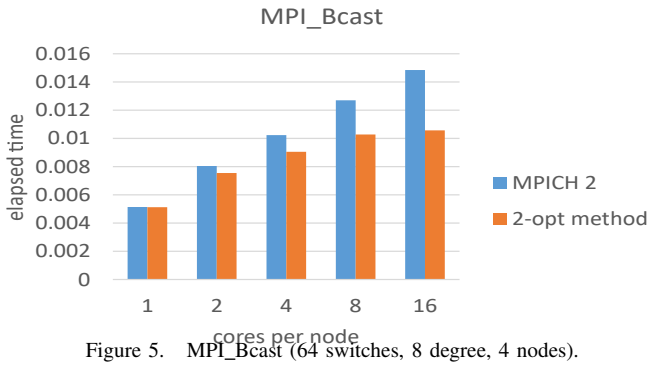

Figure 5. MPI_Bcast (64 switches, 8 degree, 4 nodes).

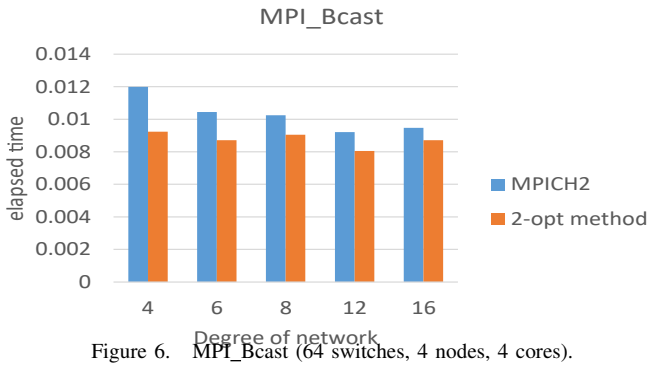

Figure 6. MPI_Bcast (64 switches, 4 nodes, 4 cores).

elapsed time and x-axis represents the number of cores per node. We set the network size is 64, each switch has 4 host, the degree of network size is 8. We set the number of cores as 1, 2, 4, 8, 16. As Figure 5 shows, when the number of cores per node increased, the elapsed time becomes lager, but broadcast applied 2-opt method has huge performance improvement.

4) Switch Degree: The Figure 6 shows the relationship between the degree of network and the elapsed time. The y-axis represents the elapsed time and x-axis represents the degree. We set the network size is 64, each switch has 4 host, and each host has 4 cores, the message length is 64KB. We set the number of degree as 4, 6, 8, 12, 16. As Figure 6 shows, when the degree increased, the elapsed time becomes smaller, and the broadcast applied 2-opt method always has better performance.

V. Conclusions

In this paper we evaluated the performance of broadcast operation of MPICH2 and the broadcast operation which applied 2-opt method. The results shows that the 2-opt method can dramatically improve the performance of broadcast operation.

Acknowledgments

This work was supported by KAKENHI 19H01106.

References

[1] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Transactions on Computers*, vol. C-34, no. 10, pp. 892–901, Oct 1985.

[2] N. R. Adiga, M. A. Blumrich, D. Chen, P. Coteus, A. Gara, M. E. Giampapa, P. Heidelberger, S. Singh, B. D. Steinmacher-Burow, T. Takken, M. Tsao, and P. Vranas, "Blue gene/l torus interconnection network," *IBM Journal of Research and Development*, vol. 49, no. 2.3, pp. 265–276, March 2005.

[3] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-Driven, Highly-Scalable Dragonfly Topology," in *ISCA*, 2008, pp. 77–88.

[4] GraphGolf: The Order/degree Problem Competition., "GraphGolf: The Order/degree Problem Competition." http://research.nii.ac.jp/graphgolf/.

[5] M. Koibuchi, H. Matsutani, H. Amano, D. F. Hsu, and H. Casanova, "A case for random shortcut topologies for hpc interconnects," in *ACM Sigarch Computer Architecture News*, vol. 40, no. 3. IEEE Computer Society, 2012, pp. 177–188.

[6] R. Kawano, H. Nakahara, I. Fujiwara, H. Matsutani, M. Koibuchi, and H. Amano, "Loren: A scalable routing method for layout-conscious random topologies," in *2016 Fourth International Symposium on Computing and Networking (CANDAR)*, Nov 2016, pp. 9–18.

[7] Nas Parallel Benchmarks, https://www.nas.nasa.gov/publications/npb.html.

[8] MPI Forum, http://mpi-forum.org/.

[9] R. Thakur, R. Rabenseifner, and W. Gropp, "Optimization of collective communication operations in mpich," *The International Journal of High Performance Computing Applications*, vol. 19, no. 1, pp. 49–66, 2005.

[10] K. Kandalla, H. Subramoni, A. Vishnu, and D. K. Panda, "Designing topology-aware collective communication algorithms for large scale infiniband clusters: Case studies with scatter and gather," in *Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*. IEEE, 2010, pp. 1–8.

[11] MPICH — High-Performance Portable MPI, https://www.mpich.org/,.

[12] R. Kesavan and D. K. Panda, "Efficient multicast on irregular switch-based cut-through networks with up-down routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 8, pp. 808–828, Aug 2001.

[13] S. L. Johnsson and C. . Ho, "Optimum broadcasting and personalized communication in hypercubes," *IEEE Transactions on Computers*, vol. 38, no. 9, pp. 1249–1268, Sep. 1989.

[14] P. K. McKinley, H. Xu, A. . Esfahanian, and L. M. Ni, "Unicast-based multicast communication in wormhole-routed networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, no. 12, pp. 1252–1265, Dec 1994.

[15] A. Faraj, S. Kumar, B. Smith, A. Mamidala, and J. Gunnels, "Mpi collective communications on the blue gene/p supercomputer: Algorithms and optimizations," in *2009 17th IEEE Symposium on High Performance Interconnects*, Aug 2009, pp. 63–72.

[16] H. Subramoni, K. Kandalla, J. Vienne, S. Sur, B. Barth, K. Tomko, R. Mclay, K. Schulz, and D. K. Panda, "Design and evaluation of network topology-/speed- aware broadcast algorithms for infiniband clusters," in *2011 IEEE International Conference on Cluster Computing*, Sep. 2011, pp. 317–325.

[17] G. A. Croes, "A method for solving traveling-salesman problems," *Operations Research*, vol. 6, no. 6, pp. 791–812, 1958. [Online]. Available: https://doi.org/10.1287/opre.6.6.791

[18] SimGrid: Versatile Simulation of Distributed Systems, http://simgrid.gforge.inria.fr/.

[19] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, Scalable, and Accurate Simulation of Distributed Appl ications and Platforms," *Journal of Parallel and Distributed Computing*, vol. 74, no. 10, pp. 2899–2917, 2014.