

GLEX-Alltoall: gather-scatter-based multi-leader All-to-all Communication on Multi-core Supercomputer

Jintao Peng¹, Jie Liu^{2,*}, Min Xie³

Changsha, China

Abstract

All-to-all communication is commonly used in parallel applications like FFT. In modern supercomputers, there are multiple cores, NUMAs and network endpoints. These features bring much parallelism. However, there is no method which makes use of the parallelism to improve the all-to-all communication. In this paper, we introduce an optimized NUMA-aware multi-leader all-to-all library which explores the parallelism on network, CPU cores and overlaps the intra- and inter-node communication. The results show that, compared to MPI, our library achieves up to 20x speedup. For application, our method achieves up to 1.75x speedup on peak performance for 16384 cores.

Keywords: Collective Communication, Multi-core processor, MPI all-to-all, RDMA, Shared Heap

2010 MSC: 00-01, 99-00

1. Introduction

Many parallel applications may suffer from global communication. Especially for communication-intensive applications, their time-to-solution and scalability may be affected by global communication. Message Passing Interface (MPI) provides a set of commonly used collective communication. MPI.Alltoall is one of the collective communication where each process will send a different message to all processes. It is broadly used in some parallel applications like Fast Fourier Transform (FFT) [1] and some graph algorithms like MapReduce [2] and Breadth-first search (BFS) [3]. However, each time we double the processes, the all-to-all communication workload is quadrupled. On modern supercomputers, network throughput has a linear relationship with the number of nodes. This brings great challenges to large-scale all-to-all communications.

For multiple-core processes, an effective way is node-aware all-to-all method [4]. It replaces a N nodes global all-to-all into $N-1$ times intra-node gather + local transpose + inter-node transpose + $N-1$ times intra-node gather. This method is very effective for small messages. Because, compared to original method, a node-aware all-to-all reduces the number of inter-node messages from $(M^N)^2$ to N^2 times (M is number of processors in each node). The size of the message is increased by M^2 times, which makes effective use of the network bandwidth. In the current supercomputer, a node has multiple CPU cores, NUMA and network endpoints. This architecture brings 4 kinds of parallelism to optimize a node-aware all-to-all method:

- (1) Multiple network endpoints can simultaneously process multiple communication requests.
- (2) Processes in different NUMA can simultaneously access its local memory without contention.
- (3) Multiple processes can simultaneously gather/scatter data and compose communication requests.
- (4) Inter-node communication can be overlapped with intra-node communication.

As we known, no methods combine these parallelism together to improve a node-aware all-to-all collective communication.

In this paper, we proposed a multi-leader node-aware all-to-all method. It uses multiple leaders on different NUMA which open the different network endpoints to gather/scatter data, compose communication requests, and transpose local matrix. It explores the parallelism existing in modern multi-core processor with NUMA memory architecture and multi-port network. For intra-node gather/scatter, we proposed a shared-heap-based remote accessible memory which is similar to intra-node MPI RMA. Inter-node communication is based on Remote Direct Memory Access (RDMA) which provides high throughput and low latency. The results show that, compared to MPI.alltoall, our implementation achieves up to 20x speedup and 4x speedup on average.

2. Related Work

From an algorithm perspective: Bruck algorithm [5] is commonly used for small message all-to-all. For mid size messages, isend-irecv algorithm is used. For large messages, linear shift exchange [6], pairwise exchange [7].

When considering the multi-core processors: Cache-oblivious MPI all-to-all (SH-NUMA-CO) based on morton order is proposed to minimize the cache miss rate [8]. For Infiniband and

*Corresponding author

¹JintaoPengCS@gmail.com

²liujie@nudt.edu.cn

³xiemin@nudt.edu.cn

Table 1: Overall design-space for all-to-all collective on mult-core processors

Approach/Metric	SH-NUMA-CO	SA-orig	elan_alltoall	PAIRWISE-SLR and BRUCK-SLR	GLEX-Alltoall (ours)
inter-node implementation	MVAPICH2	MVAPICH	RDMA	MPI-P2P	RDMA
intra-node implementation	Shared Heap	Shared Memory	Shared Memory	MPI-P2P	Shared Heap
Leader-based aggregation support	✓	✓	✓	✓	✓
Multi-leader support	✗	✓	✗	✗	✓
NUMA-aware	✓	✗	✗	✗	✓
Multiple network endpoints	✗	✓	✓	✗	✓
Overlapping Inter/intra-node	✓	✗	✗	✓	✓

Table 2: Memory Bandwidth (tested on Stream) and Point-to-Point Bandwidth (tested on RDMA) of Experimental Platforms

Memory Bandwidth tests (MB/s)	Copy (Single-thread)	Copy (Multi-threads)	NUMA Count	loopback	bi-loopback	two-node-p2p	two-node-bip2p
HPC-A (32 threads)	6481	21351	4	1366	1368	1181	1336
HPC-B (32 threads)							
HPC-C (24 threads)							
HPC-D (24 threads)	11280	46252	2	11530	11568	7089	11629

multi-core systems, a non-leader all-to-all collective (SA-orig) which based on shared memory aggregation techniques is proposed in [9]. For multi-rail QsNet SMP clusters, a shared memory and RDMA based all-to-all collectives (elan_alltoall) is proposed in [10]. For Intel Many Integrated Core (MIC) architecture, the re-routing scheme based all-to-all collective (PAIRWISE-SLR/BRUCK-SLR) is proposed in [11]. These works are direct related to our work. Table 1 shows the overall design-space comparison between these methods and our method.

Besides, to improve the intra-node communication, several kernel-assistant techniques have been proposed for multi-core processor (LIMIC [12], KNEM [13], XPMEM). Compared to shared memory, these method works well for large message intra-node communication. Because they avoid one memory copy overhead. Shared heap [14] has the same performance with these kernel-assistant techniques. Shared heap do not need extra kernel module.

When considering the network topology: A bandwidth-optimal all-to-all exchange is proposed for fat-tree network [15]. For torus network, a large scale all-to-all is proposed for Blue Gene/L Supercomputer [16]. A optimal schedule for all-to-all personalized communication is proposed for multiprocessor systems [17]. For Infiniband clusters, their is a topology aware all-to-all scheduler which lower the contention by adding extra communication steps [18].

3. Experimental Platforms

The methods in this paper are validated on four different systems:

- (1) HPC-A: Matrix-2000+ CPU, Tianhe series network.
- (2) HPC-B: FT-2000+ CPU, Tianhe series network.
- (3) HPC-C: Intel E5 CPU, Tianhe series network.
- (4) HPC-D: E5-2692 v2 * 2, Tianhe series network.

The Stream bandwidth test results are shown in the Table 2.

4. Motivation

Node-aware all-to-all is a traditional method for optimize small message all-to-all. It include four steps: intra-node gather, local transpose, inter-node all-to-all, intra-node scatter. With the increase in the number of processor cores, multiple CPU cores, NUMA, network endpoints and inter-/intra-node overlapping brings huge parallelism. However, we notice that all traditional method do not take advantage of this parallelism optimize the four steps. For a 32/64 cores CPU, each inter-node message in node-aware method will be 1024/4096 times larger than direct method. If there is one double data between each pair of processes. There are 1024/4096 double data between each pair of nodes. As the single-core memory access bandwidth may not faster than the network bandwidth. Using a single core to gather/scatter/transpose data and initialize communication request may causing communication hotspot happend on a CPU core.

Multiple CPU core can gather/scatter simultaneous using MPI-RMA. This get faster message aggregation bandwidth.

5. Front matter

The author names and affiliations could be formatted in two ways:

- (1) Group the authors per affiliation.
- (2) Use footnotes to indicate the affiliations.

See the front matter of this document for examples. You are recommended to conform your choice to the journal you are submitting to.

6. Bibliography styles

There are various bibliography styles available. You can select the style of your choice in the preamble of this document.

These styles are Elsevier styles based on standard styles like Harvard and Vancouver. Please use BibTeX to generate your bibliography and include DOIs whenever available.

Here are two sample references: [? ?].

References

References

- [1] C. Mehta, A. Karthi, V. Jetly, B. Chaudhury, Parallel fast multi-pole method accelerated fft on hpc clusters, *Parallel Computing* (2021) 102783.
- [2] S. J. Plimpton, K. D. Devine, Mapreduce in mpi for large-scale graph algorithms, *Parallel Computing* 37 (9) (2011) 610–632.
- [3] K. Ueno, T. Suzumura, Highly scalable graph search for the graph500 benchmark, in: *Proceedings of the 21st international symposium on High-Performance Parallel and Distributed Computing*, 2012, pp. 149–160.
- [4] S. Sistare, R. Vandevert, E. Loh, Optimization of mpi collectives on clusters of large-scale smp's, in: *Proceedings of the 1999 ACM/IEEE conference on Supercomputing*, 1999, pp. 23–es.
- [5] J. Bruck, C.-T. Ho, S. Kipnis, E. Upfal, D. Weathersby, Efficient algorithms for all-to-all communications in multiport message-passing systems, *IEEE Transactions on parallel and distributed systems* 8 (11) (1997) 1143–1156.
- [6] S. Ranka, J.-C. Wang, G. C. Fox, Static and run-time algorithms for all-to-many personalized communication on permutation networks, *IEEE Transactions on Parallel and Distributed Systems* 5 (12) (1994) 1266–1274.
- [7] R. Thakur, R. Rabenseifner, W. Gropp, Optimization of collective communication operations in mpich, *The International Journal of High Performance Computing Applications* 19 (1) (2005) 49–66.
- [8] S. Li, Y. Zhang, T. Hoefler, Cache-oblivious mpi all-to-all communications based on morton order, *IEEE Transactions on Parallel and Distributed Systems* 29 (3) (2017) 542–555.
- [9] R. Kumar, A. Mamidala, D. K. Panda, Scaling alltoall collective on multi-core systems, in: *2008 IEEE International Symposium on Parallel and Distributed Processing*, IEEE, 2008, pp. 1–8.
- [10] Y. Qian, A. Afsahi, Efficient shared memory and rdma based collectives on multi-rail qsnets in smp clusters, *Cluster Computing* 11 (4) (2008) 341–354.
- [11] A. Venkatesh, S. Potluri, R. Rajachandrasekar, M. Luo, K. Hamidouche, D. K. Panda, High performance alltoall and allgather designs for infiniband mic clusters, in: *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, IEEE, 2014, pp. 637–646.
- [12] H.-W. Jin, S. Sur, L. Chai, D. K. Panda, Lightweight kernel-level primitives for high-performance mpi intra-node communication over multi-core systems, in: *2007 IEEE International Conference on Cluster Computing*, IEEE, 2007, pp. 446–451.
- [13] B. Goglin, S. Moreaud, Knem: A generic and scalable kernel-assisted intra-node mpi communication framework, *Journal of Parallel and Distributed Computing* 73 (2) (2013) 176–188.
- [14] S. Li, T. Hoefler, C. Hu, M. Snir, Improved mpi collectives for mpi processes in shared address spaces, *Cluster computing* 17 (4) (2014) 1139–1155.
- [15] B. Prisacari, G. Rodriguez, C. Minkenberg, T. Hoefler, Bandwidth-optimal all-to-all exchanges in fat tree networks, in: *Proceedings of the 27th international ACM conference on International conference on supercomputing*, 2013, pp. 139–148.
- [16] S. Kumar, Y. Sabharwal, R. Garg, P. Heidelberger, Optimization of all-to-all communication on the blue gene/l supercomputer, in: *2008 37th International Conference on Parallel Processing*, IEEE, 2008, pp. 320–329.
- [17] D. Saha, K. Sinha, Optimal schedule for all-to-all personalized communication in multiprocessor systems, *ACM Transactions on Parallel Computing (TOPC)* 6 (1) (2019) 1–29.
- [18] H. Subramoni, K. Kandalla, J. Jose, K. Tomko, K. Schulz, D. Pekurovsky, D. K. Panda, Designing topology-aware communication schedules for all-toall operations in large infiniband clusters, in: *2014 43rd International Conference on Parallel Processing*, IEEE, 2014, pp. 231–240.