**Journal of Computer Science, 9 (4): 455-462, 2013**
ISSN 1549-3636

doi:10.3844/jcssp.2013.455.462 Published Online 9 (4) 2013 (http://www.thescipub.com/jcs.toc)

# Performance Analysis of Message Passing Interface Collective Communication on Intel Xeon Quad-Core Gigabit Ethernet and Infiniband Clusters

**[1,2]Roswan Ismail, [1]Nor Asilah Wati Abdul Hamid, [1]Mohamed Othman and [1]Rohaya Latip**

[1]Department of Communication and Network Technology,
Faculty of Computer Sciences and Information Technology,
Universiti Putra Malaysia, Selangor, Malaysia
[2]Department of Computing, Faculty of Art, Computing and Creative Industry,
Universiti Pendidikan Sultan Idris, Perak, Malaysia

Received 2012-10-03, Revised 2013-04-03; Accepted 2013-05-09

## ABSTRACT

The performance of MPI implementation operations still presents critical issues for high performance computing systems, particularly for more advanced processor technology. Consequently, this study concentrates on benchmarking MPI implementation on multi-core architecture by measuring the performance of Open MPI collective communication on Intel Xeon dual quad-core Gigabit Ethernet and InfiniBand clusters using SKaMPI. It focuses on well known collective communication routines such as MPI-Bcast, MPI-AlltoAll, MPI-Scatter and MPI-Gather. From the collection of results, MPI collective communication on InfiniBand clusters had distinctly better performance in terms of latency and throughput. The analysis indicates that the algorithm used for collective communication performed very well for all message sizes except for MPI-Bcast and MPI-Alltoall operation of inter-node communication. However, InfiniBand provides the lowest latency for all operations since it provides applications with an easy to use messaging service, compared to Gigabit Ethernet, which still requests the operating system for access to one of the server communication resources with the complex dance between an application and a network.

**Keywords:** MPI Benchmark, Performance Analysis, MPI Communication, Open MPI, Gigabit, InfiniBand

## 1. INTRODUCTION

Over the past few years, clusters have become the main architecture used for high performance computing systems. The emerging trend of using cluster as High Performance Computing (HPC) has led to much research in this field, particularly the standard approach utilized for communication between nodes; Message Passing Interface (MPI) (Isaila *et al*., 2010; Balaji *et al*., 2009). MPI is a library of routines provides a portable programming paradigm for existing development environments with a fundamental message management service and standard message passing API. Since MPI is used to program parallel machines, the performance of most clusters depends critically on the performance of the communication routines provided by the MPI library.

Cluster interconnect is another important factor that can influence the communication performance of clusters. Slower interconnects may cause processes to run slowly. The ideal cluster interconnect should provide low latency high bandwidth and non-blocking interconnect architecture. Consequently, numerous protocols have been designed and proposed to maximize the MPI standard implementation in high performance clusters such as InfiniBand and Gigabit Ethernet.

**Corresponding Author:** Roswan Ismail, Department of Communication and Network Technology,
Faculty of Computer Sciences and Information Technology, Universiti Putra Malaysia, Selangor, Malaysia

Currently, InfiniBand and Gigabit Ethernet are the most popular interconnect employed in High Performance Computers (HPC). Based on statistic on November 2012 from the top 500 supercomputers site, InfiniBand came on top with 44.8% while Gigabit Ethernet was the close second with 37.8%. Gigabit Ethernet provides LAN technology with a latency range between 40-300 μs and is able to deliver up to 1 Gbit/sec (or 1000 MBytes/sec) bandwidth of full duplex communication using TCP/IP. Meanwhile, InfiniBand is able to provide lower latency and higher bandwidth than Gigabit Ethernet. It has latency range between 1-10 μs and can support network bandwidth up to 10 Gbit/sec (or 10000 MBytes/sec). InfiniBand with multi path provides much better throughput as compared to Gigabit Ethernet since latency effects throughput in HPC network. However, high speed InfiniBand network is more expensive than Gigabit Ethernet.

As most clusters use these two types of interconnect for communicating data between the nodes, It is important to implement the MPI on top of the cluster interconnect efficiently in order to achieve the optimal performance. Therefore, the analysis and evaluation of the MPI routines performance on clusters are indispensable. This study discusses the benchmarking results of Open MPI collective communication on Gigabit Ethernet and InfiniBand clusters of UPM Biruni Grid. The measurements were done using SKaMPI, one of the most commonly used MPI benchmark tools. The outcome would be beneficial for further research related to the Open MPI implementation on multi-core clusters.

## 2. RELATED WORKS

There has been considerable previous research focusing on the performance analysis of MPI implementation on different parallel machines and different interconnects. Some studies provide performance evaluation of MPI communication on clusters with ccNUMA nodes (Hamid and Coddington, 2010; Kayi *et al*., 2008) and multi-core architecture such as dual-core and quad-core nodes (Gu *et al*., 2013; Cheng and Gu, 2012; Kayi *et al*., 2009).

Other studies provide performance analysis of point to point or collective communication on different interconnects (Ismail *et al*., 2011; Rashti and Afsahi, 2007) while some provide comparison and analysis of multiple algorithms for collective communication in order to find the best solution for different parallel systems (Nanri and Kurokawa, 2011; Hamid and Coddington, 2007). Other related studies focused on optimizing the performance of MPI collective communication by proposing topology aware mechanisms (Gong *et al*., 2013; Subramoni *et al*., 2011; 2013; Kandalla *et al*., 2010) and process arrival patterns aware mechanisms (Qian and Afsahi, 2009; 2011; Patarasuk and Yuan, 2008) to achieve the best performance in terms of time.

However, there have been no studies on the comparison and measurement of MPI collective communication for Open MPI in Gigabit Ethernet and InfiniBand technology, particularly on a cluster with dual quad-core nodes. Unlike previous works, the work presented in this article provides the measurement of the MPI collective communication performance on clusters with Intel Xeon II dual quad-core processor using two different types of interconnect: Gigabit Ethernet and InfiniBand. This study discusses the results of Open MPI for collective communication. All findings are discussed and highlighted.

## 3. CLUSTER CONFIGURATION

The experiments in this study were conducted on clusters of Biruni Grid. Biruni Grid is a project that was developed and managed by the InfoComm Development Centre (iDEC) of UPM as part of the HPC clusters for A-Grid. This project was initiated in 2008 with the funding from EuAsiaGrid. Biruni Grid consists of three clusters: Khaldun, Razi and Haitham which consist of six, twenty and eight worker nodes respectively. Each cluster node uses IBM Blade HS21 Servers running Scientific Linux 5.4 64 bit Operating Systems and uses a GCC compiler to compile the benchmark programs.

Each node has dual Intel Xeon quad-core processors E5405, 2 GHz with 8 GB RAMs. For MPI implementation, Khaldun uses Open MPI-1.3.3 while Razi and Haitham Open MPI-1.4.3. The inter-node interconnects for Khaldun and Razi were done through Gigabit Ethernet with a Maximum data transfer of 2×1 Gb/sec while Haitham through high speed InfiniBand technology with a maximum of 2×10 Gb/sec of data transfer. All nodes were connected together using a 48-port switch employing star topology.

However, this study only provides a comparison of MPI collective communication conducted on the Razi and Haitham clusters as both have identical configuration except their inter-node interconnection. The different configuration of the clusters is listed in **Table 1** while **Fig. 1** represents the block diagram of Intel Xeon II dual quad-core processor E5405.

**Table 1.** Cluster configuration

| | Khaldum | Razi | Haitham |
|---|---|---|---|
| Number of nodes | 6 | 20 | 8 |
| Machines | IBM blade HS21 servers | | |
| CPU | 2×Intel Xeon Quad-Core2 GHZ Processors (8 cores per node) | | |
| RAM | 8 GB | | |
| Storage capacity | each node has 2×147 GB (only1×147 GB opend the rest reserved for future use (multilayer grid)) | | |
| O.S | Scientific Linux 5.4 64 bit | | |
| Compiler | GCC Compiler | | |
| Interconnect | Gigabit ethernet switch | Infini band switch | |
| MPI Implementation | Open-MPI-1.3.3 | Open-MPI-1.4.3 | |

## 4. COLLECTIVE COMMUNICATION

A group of processes can exchange data by collective communication. MPI-Bcast is one of the most commonly used collective communication routines. It enables the root process to broadcast the data from the buffer to all processes in the communicator. A broadcast has a specified root process and every process receives one copy of the message from the root. All processes must specify the same root. The root argument is the rank of the root process. The buffer, count and data type arguments are treated as in a point-to-point send on the root and as in a point-to-point receive elsewhere. The process for MPI-Bcast is shown in **Fig. 2**. MPI-Alltoall routine refers to the operation of sending distinct data from all processes to all other processes in the same group. In this operation, each process performs a scatter operation in order. Data distribution is to all processes of two data objects from each process. The process for MPI-Alltoall is shown in **Fig. 3**.

MPI-Scatter is used to distribute distinct data from the root process to all processes in the group including itself. In this case, each process including the root process sends the contents of its send buffer to the root process. It specifies a root process and all processes must specify the same root. The main difference from MPI-Bcast is that the send and receive details are in general different and therefore, both must be specified in the argument lists. The sendbuf, sendcount, sendtype arguments are only significant at the root. The process for MPI-Scatter is shown in **Fig. 4**.

MPI-Gather does the reverse operation of MPI-Scatter by recombining the data back from each processor into a single large data set. The argument list is the same as for MPI-Scatter. It specifies a root process and all processes must specify the same root. The recvbuf, recvcount, recvtype arguments are only significant at the root. However the data in recvbuf are held by rank order. The process for MPI-Gather is shown in **Fig. 5**.

## 5. EXPERIMENTAL RESULTS

This section describes the SKaMPI results for MPI-Bcast, MPI-Alltoall, MPI-Scatter and MPI-Gather operations on 16 and 32 cores on Gigabit Ethernet (GBE) and InfiniBand (IB) quad-core clusters as shown in **Fig. 6-9**. As expected from **Fig. 6**, the MPI-Bcast on 16 and 32 cores on GBE gave the highest result as compared to IB. The results show that the InfiniBand has the lowest latency, approximately 24.2% compared to the Gigabit Ethernet for both cores. This happened since the multi path of high speed InfiniBand allows transmission of data to be completed faster than the Gigabit Ethernet.

It is noted that the broadcast latency at 1048576 byte on GBE and IB were slightly decreased and getting closer to latency for 16 cores on IB. However, this phenomenon does not occur for latency on 16 cores in IB. It means that the change-over point of multiple algorithms used in Open MPI affected the results of inter-node communication on both technologies but not on 16 cores on IB where the results obtained were consistent for all message sizes. In this case, for future enhancement the changeover point at 1048576 bytes of multiple algorithms used in Open MPI will be highlighted to justify the results.

As predictable from **Fig. 7**, the MPI-Alltoall operation on larger core on both clusters gave the highest latency compared to the smaller number of cores. The results show that the InfiniBand has the lowest latency, approximately 8.8% compared to the Gigabit Ethernet for both sizes. InfiniBand gives every application direct to the messaging service which means that an application does not rely on the operating system to transfer data. This contrasts with Gigabit Ethernet, which must rely on the involvement of the operating system to move data.

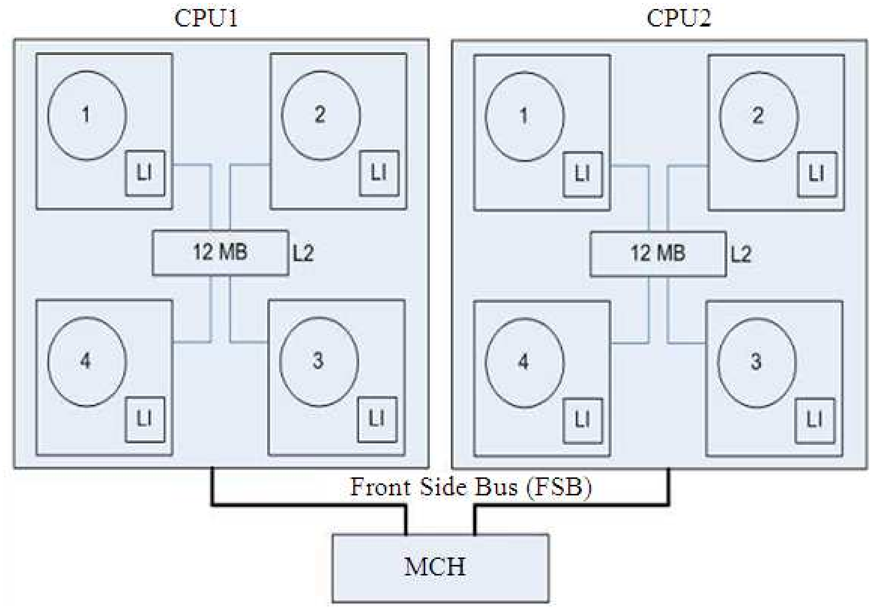


**Fig. 1.** Block diagram of intel xeon processor E5405

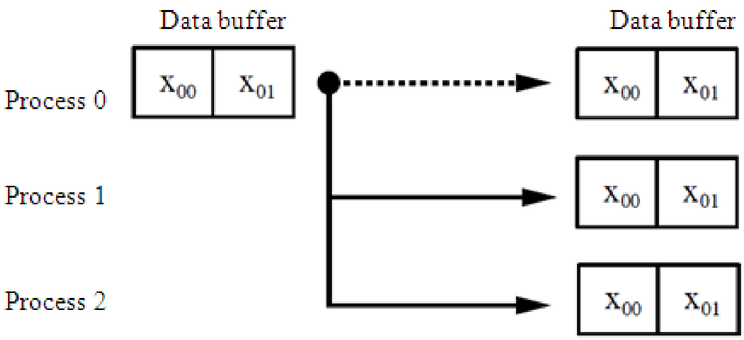


**Fig. 2.** MPI_Bcast process

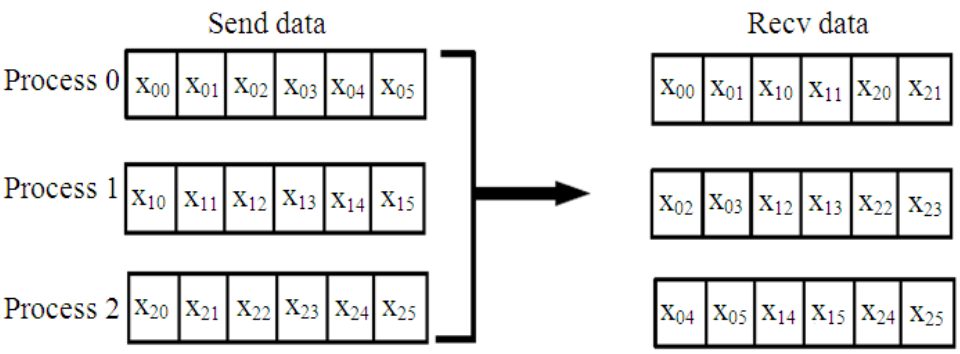


**Fig. 3.** MPI_AlltoAll process

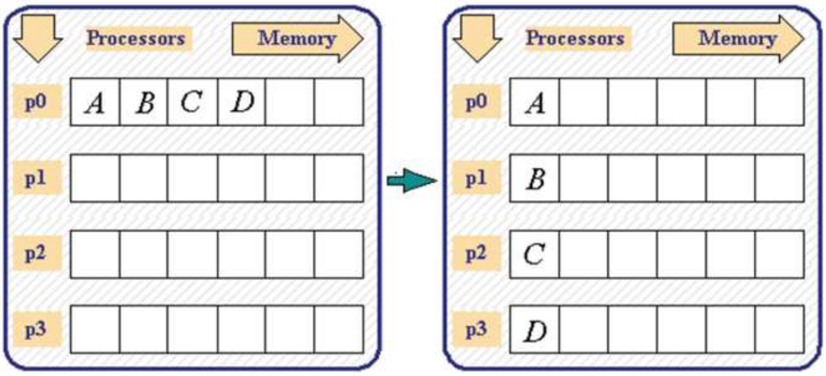


**Fig. 4.** MPI_Scatter process

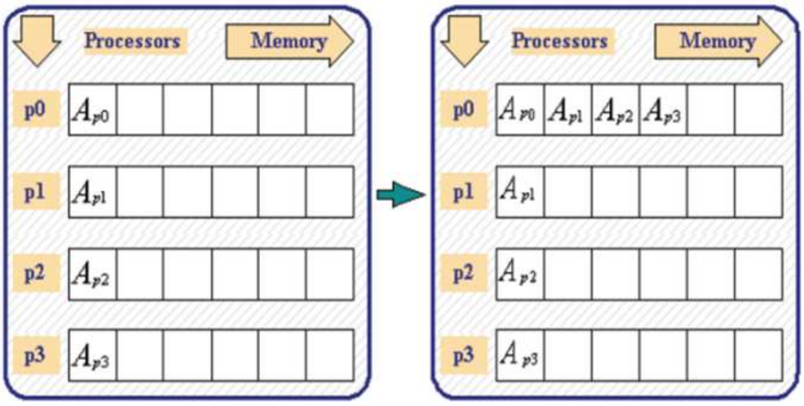


**Fig. 5.** MPI_Gather process

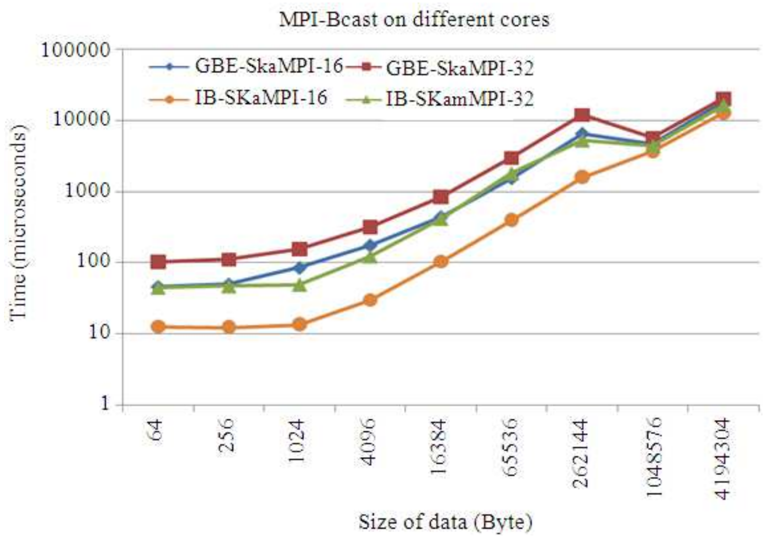


**Fig. 6.** SKaMPI results for MPI Bcast on different cores

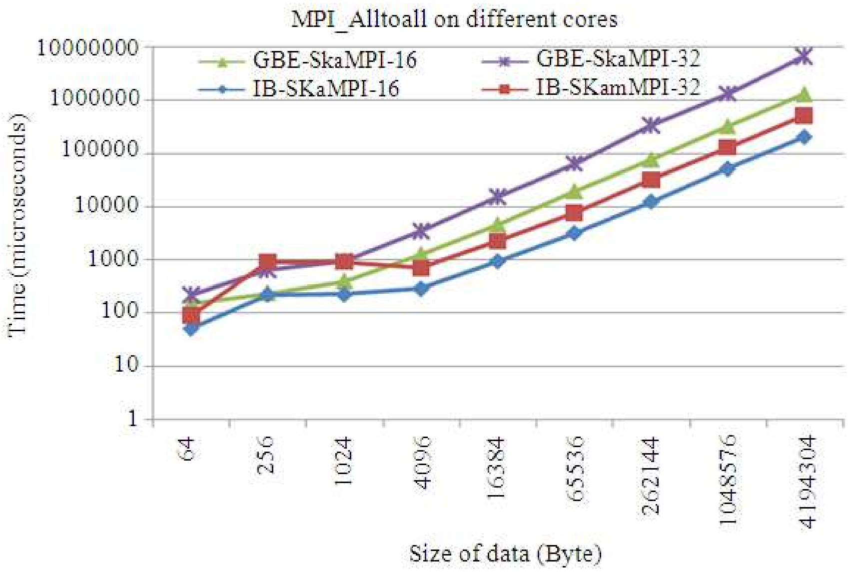


**Fig. 7.** SKaMPI results for MPI AlltoAll on different cores

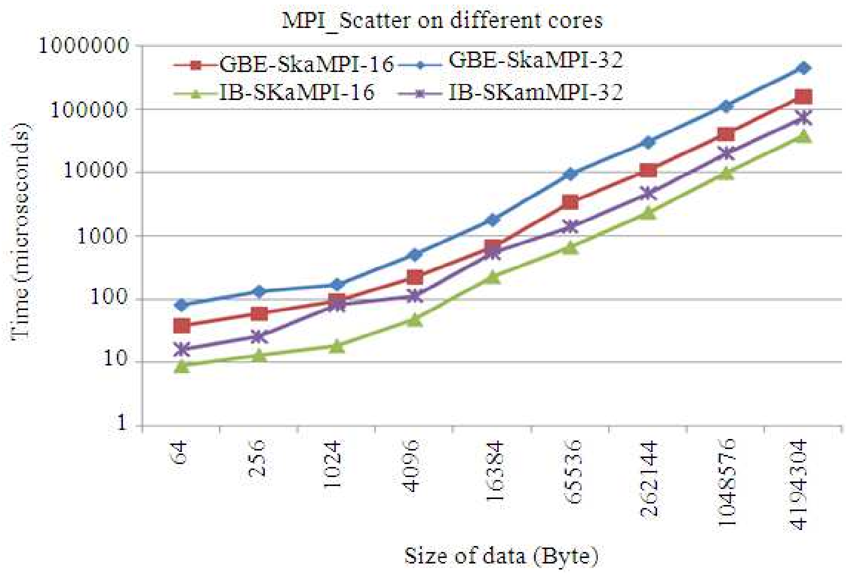


**Fig. 8.** SKaMPI results for MPI Scatter on different cores

The results on **Fig. 9** show that IB has the lowest latency, approximately 7.9% compared to GBE for both sizes. From **Fig. 8 and 9**, it can be concluded that MPI Scatter and MPI Gather with 16 and 32 cores on GBE provided higher results compared to IB as it will take a longer time to complete since it needs to distribute and gather data to/from processors using a lower bandwidth compared to IB. Consequently, it produces more overheads. It was also noted that the results trend for MPI-Scatter and MPI-Gather were consistent for both clusters, which means that the selection of multiple algorithms used to gather and scatter message performed very well for all message sizes.

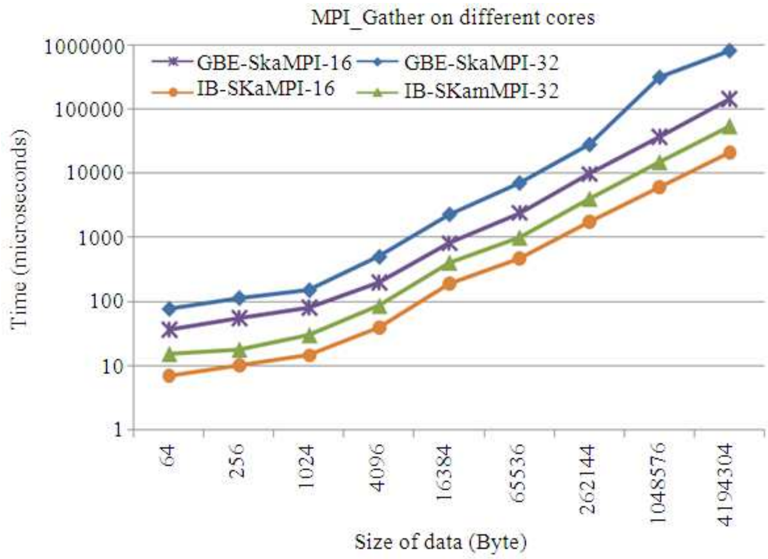


**Fig. 9.** SKaMPI results for MPI gather on different cores

# 6. ACKNOWLEDGEMENT

This study was supported by the Malaysian Ministry of Higher Education (MOHE). Thanks to infoComm Development Center (iDEC) of Universiti Putra Malaysia for granting access to the Biruni Grid. Special thanks to Muhammad Farhan Sjaugi from iDEC for testing support and his useful feedback.

# 7. REFERENCES

Balaji, P., A. Chan, R. Thakur, W. Gropp and E. Lusk, 2009. Toward message passing for a million processes: Characterizing MPI on a massive scale blue gene/p. Comput. Sci. Res. Dev., 24: 11-19. DOI: 10.1007/s00450-009-0095-3

Cheng, P. and Y. Gu, 2012. An analysis of multicore specific optimization in MPI implementations. Proceedings of the IEEE 26th International Parallel and Distributed Processing Symposium Workshops and PhD Forum, May 21-25, IEEE Xplore Press, Shanghai, pp: 1874-1878. DOI: 10.1109/IPDPSW.2012.231

Gong, Y., B. He and J. Zhong, 2013. Network performance Aware MPI collective communication operations in the cloud. IEEE Trans. Parall. Distrib. Syst. DOI: 10.1109/TPDS.2013.96

Gu, Z., M. Small, X. Yuan, A. Marathe and D.K. Lowenthal, 2013. Protocol customization for improving MPI performance on RDMA-enabled clusters. Int. J. Parallel Programm. DOI: 10.1007/s10766-013-0242-0

Hamid, N.A.W.A and P. Coddington, 2010. Comparison of MPI benchmark programs on shared memory and distributed memory machines (point-to-point communication). Int. J. High Perform. Comput. Applic., 24: 469-483. DOI: 10.1177/1094342010371106

Hamid, N.A.W.A. and P. Coddington, 2007. Analysis of algorithm selection for optimizing collective communication with mpich for ethernet and myrinet networks. Proceedings of the 8th International Conference on Parallel and Distributed Computing, Applications and Technologies, Dec. 3-6, IEEE Xplore Press, Adelaide, SA., pp: 133-140. DOI: 10.1109/PDCAT.2007.65

Isaila, F., F.J.G. Blas, J. Carretero, W.K. Liao and A. Choudhary, 2010. A scalable message passing interface implementation of an ad-hoc parallel I/O system. Int. J. High Perform. Comput. Applic., 24: 164-184. DOI: 10.1177/1094342009347890

Ismail, R., N.A.W.A. Hamid, M. Othman, R. Latip and M.A. Sanwani, 2011. Point-to-point communication on gigabit ethernet and InfiniBand networks. Proceedings of the International Conference on Informatics Engineering and Information Science, Nov. 14-16, Springer Berlin Heidelberg, Kuala Lumpur, pp: 369-382. DOI: 10.1007/978-3-642-25483-3_30

Kandalla, K., H. Subramoni, A. Vishnu and D.K. Panda, 2010. Designing topology-aware collective communication algorithms for large scale InfiniBand clusters: Case studies with scatter and gather. Proceedings of the IEEE International Symposium on Parallel and Distributed Processing, Workshops and Phd Forum, Apr. 19-23, IEEE Xplore Press, Atlanta, GA., pp: 1-8. DOI: 10.1109/IPDPSW.2010.5470853

Kayi, A., E. Kornkven, T. El-Ghazawi and G. Newby, 2008. Application performance tuning for clusters with ccnuma nodes. Proceedings of the 11th IEEE International Conference on Computational Science and Engineering, Jul. 16-18, IEEE Xplore Press, Sao Paulo, pp: 245-252. DOI: 10.1109/CSE.2008.46

Kayi, A., T. El-Ghazawi and G.B. Newby, 2009. Performance issues in emerging homogeneous multi-core architectures. Simulat. Modell. Pract. Theory, 17: 1485-1499. DOI: 10.1016/j.simpat.2009.06.014

Nanri, T. and M. Kurokawa, 2011. Effect of dynamic algorithm selection of Alltoall communication on environments with unstable network speed. Proceedings of the International Conference on High Performance Computing and Simulation, Jul. 4-8, IEEE Xplore Press, Instanbul, pp: 693-698. DOI: 10.1109/HPCSim.2011.5999894

Patarasuk, P. and X. Yuan, 2008. Efficient MPI bcast across different process arrival patterns. Proceedings of the IEEE International Symposium on Parallel and Distributed Processing, Apr. 14-18, IEEE Xplore Press, Miami, FL., pp: 1-11. DOI: 10.1109/IPDPS.2008.4536308

Qian, Y. and A. Afsahi, 2009. Process arrival pattern and shared memory aware alltoall on infiniband. Proceedings of the 16th European PVM/MPI Users' Group Meeting on Recent Advances in Parallel Virtual Machine and Message Passing Interface, Sept. 7-10, Springer Berlin Heidelberg, Espoo, Finland, pp: 250-260. DOI: 10.1007/978-3-642-03770-2_31

Qian, Y. and A. Afsahi, 2011. Process arrival pattern aware alltoall and allgather on InfiniBand clusters. Int. J. Parallel Programm., 39: 473-493. DOI: 10.1007/s10766-010-0152-3

Rashti, M.J. and A. Afsahi, 2007. 10-gigabit iwarp ethernet: Comparative performance analysis with InfiniBand and myrinet-10G. Proceedings of the IEEE International Parallel and Distributed Processing Symposium, Mar. 26-30, IEEE Xplore Press, Long Beach, CA., pp: 1-8. DOI: 10.1109/IPDPS.2007.370480

Subramoni, H., J. Vienne and D.K. Panda, 2013. A scalable InfiniBand network topology-aware performance analysis tool for MPI. Proceedings of the 18th International Conference on Parallel Processing Workshops, Aug. 27-31, Springer Berlin Heidelberg, Rhodes Islands, Greece, pp: 439-450. DOI: 10.1007/978-3-642-36949-0_49

Subramoni, H., K. Kandalla, J. Vienne, S. Sur and B. Barth, *et al*., 2011. Design and evaluation of network topology-/speed-aware broadcast algorithms for InfiniBand clusters. Proceedings of the IEEE International Conference on Cluster Computing, Sept. 26-30, IEEE Xplore Press, Austin, TX., pp: 317-325). DOI: 10.1109/CLUSTER.2011.43