Performance analysis of MPI on Cray XC40 Xeon Phi System

Extended Abstract

Sudheer Chunduri
Argonne National Laboratory
Lemont, IL 60439
sudheer@anl.gov

Paul Coffman
Argonne National Laboratory
Lemont, IL 60439
pcoffman@anl.gov

Scott Parker
Argonne National Laboratory
Lemont, IL 60439
sparker@anl.gov

Kalyan Kumaran
Argonne National Laboratory
Lemont, IL 60439
kumaran@anl.gov

ABSTRACT

As the scale of the computing systems and applications is increasing, the performance of the interconnect has become key for application performance. An efficient implementation of the MPI layer is essential for realizing optimal communication performance. This work presents different performance aspects of the MPI layer implementation on the Cray XC system. Specifically, a set of guidelines are defined for the expected performance of the MPI collective algorithms. We analyze how these guidelines were adhered to and also highlight where there is scope for improvement. We expect our analysis can be used as a feedback to the MPI developers to further tune the MPI on Cray XC, and as a guide to the application developers for their application performance projections and analysis.

KEYWORDS

Performance consistency, guideline violation, scaling

1 INTRODUCTION

The performance and scaling of parallel applications on large-scale systems is highly dependent on the effective communication performance achieved by the MPI primitives. Performance guidelines for the expected behaviour of MPI collectives have been defined in the following papers [4], [1] and [3]. A performance guideline usually defines a common-sense performance expectation based on sematic functionality of the collectives, for example, one performance guideline states that a call to MPI_Allgather on n data elements should "not be slower" than a combination of a call to MPI_Gather with n data elements followed by a call to MPI_Broadcast with n data elements.

2 STUDY OF MPI PERFORMANCE ON CRAY XC

Cray XC is one of the prominent system architecture used in the latest top 500 systems. We use the Theta system at ANL for the experiments, Theta is a Cray XC40 system with 3624 Intel KNL 7230 compute nodes interconnected using the Cray Aries interconnect. The compute nodes are connected with the Cray Aries high-speed network in the Cray dragonfly topology. The Intel KNL [2] node used in Theta has 64 CPU cores with 16 GiB of MCDRAM and 192 GiB of DDR4 2400. The interconnection network used in Theta is a three-level dragonfly topology interconnected using the Cray Aries routers.

The KNL MCDRAM can be configured on boot into one of several combinations of memory and cluster modes. We used the flat memory mode with the quadrant cluster mode for the best performance versus ease-of-use trade-off. The numactl -p 1 option was used to explicitly load the data into MCDRAM. We use Cray MPI implementation on the Cray XC system for this study.

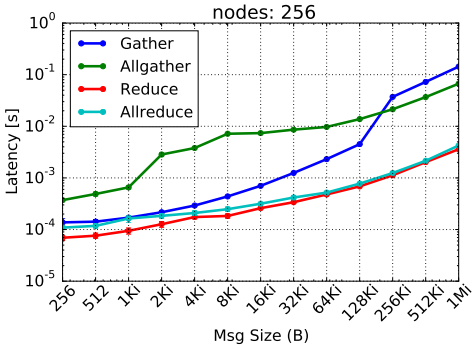

Figure 1: Scaling of Gather, Allgather, Reduce and Allreduce collectives with different message sizes for the 256 node size

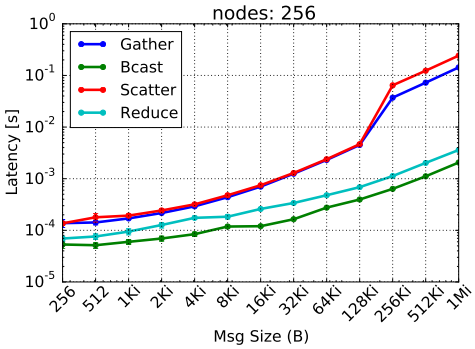

Figure 2: Scaling of Gather, Broadcast, Scatter and Reduce collectives with different message sizes for the 256 node size

The figure 1 compares the performance of many-to-one collectives with their related many-to-many counterparts across different message sizes using 256 processes (1 process per node). We can notice few performance guideline violations; MPI_Gather takes more time than MPI_Allgather for certain message sizes and MPI_Allgather performance scaling with increasing the node size is inconsistent with latency hikes at certain message sizes.

The figure 2 compares the performance of many-to-one with their related one-to-many counterparts. Broadcast is semantically more costly than Scatter and Reduce is semantically costlier than Gather [1], both these guidelines can be seen violated here.

	1	2	4	8	16	32	64	128	256	512	1024	2048	4096	8192	16384	32768	65536	131072	262144	524288	1048576
MPI_Allgather ≤ Alltoall												✗	✗	✗	✗	✗	✗	✗		✗	
MPI_Allgather ≤ Allreduce						✗						✗									
MPI_Allgather ≤ Gather + Bcast	✗	✗	✗			✗	✗	✗				✗	✗	✗	✗						

Table 1: Performance guideline violations for Allgather in Cray MPI using 256 processes (1 process per node)

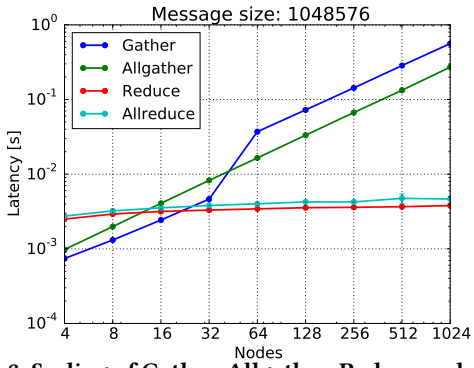

Figure 3: Scaling of Gather, Allgather, Reduce and Allreduce collectives with different node sizes for the 1MB message size

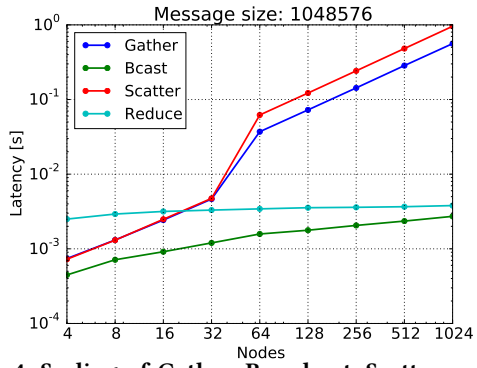

Figure 4: Scaling of Gather, Broadcast, Scatter and Reduce collectives with different node sizes for the 1MB message size

The performance guideline violations for the Allgather across different message sizes on 256 nodes are presented in the table 1. The performance of the Allgather is compared with the different collective combinations that essentially achieve the same functionality. Allgather has higher latency than Alltoall. This is true across all the node sizes tested for messages sizes between 4KB and 512KB.

Figure 3 shows the performance scaling of the collectives for a 1 MB message size as the number of nodes used increased. While the scaling curve for Allgather has a consistent slope, the scaling curve for Gather is inconsistent with a just at 64 node size especially with the larger messages sizes.

Figure 4 shows a few other violations. Gather and Scatter are slower than Reduce for message sizes greater than 16KB, whereas for smaller message sizes, Reduce is faster than Gather and Scatter. Broadcast is consistently faster than Gather and Scatter, and the slopes of the scaling curves of Broadcast and Gather/Scatter are different. There is sudden jump in the scaling curve for Gather/Scatter at specific node sizes.

While we highlight guideline violations here, we eventually wish to isolate the sources of this inconsistencies and come up with ways to address them.

REFERENCES

[1] Sascha Hunold, Alexandra Carpen-Amarie, Felix Donatus Lübbe, and Jesper Larsson Träff. 2016. Automatic Verification of Self-consistent MPI Performance Guidelines.. In *Euro-Par*. 433–446.

[2] Avinash Sodani. 2015. Knights landing (KNL): 2nd Generation Intel® Xeon Phi processor. In *Hot Chips 27 Symposium (HCS), 2015 IEEE*. IEEE, 1–24.

[3] Jesper Larsson Träff. 2012. mpicroscope: Towards an MPI benchmark tool for performance guideline verification. In *European MPI Users' Group Meeting*. Springer, 100–109.

[4] Jesper Larsson Träff, William D Gropp, and Rajeev Thakur. 2010. Self-consistent MPI performance guidelines. *IEEE Transactions on Parallel and Distributed Systems* 21, 5 (2010), 698–709.