# Analysis of Topology-Dependent MPI Performance on Gemini Networks

Antonio J. Peña,[*] Ralf G. Correa Carvalho,[†] James Dinan,[*]
Pavan Balaji,[*] Rajeev Thakur,[*] and William Gropp[*]

[*] Mathematics and Computer Science Division, Argonne National Laboratory,
{apenya,dinan,balaji,thakur}@mcs.anl.gov
[†] Computation Institute, University of Chicago, ralfgunter@uchicago.edu
[*] Department of Computer Science, University of Illinois, wgropp@illinois.edu

## ABSTRACT

Current HPC systems utilize a variety of interconnection networks, with varying features and communication characteristics. MPI normalizes these interconnects with a common interface used by most HPC applications. However, network properties can have a significant impact on application performance. We explore the impact of the interconnect on application performance on the Blue Waters supercomputer. Blue Waters uses a three-dimensional, Cray Gemini torus network, which provides twice the Y-dimension bandwidth in the X and Z dimensions. Through several benchmarks, including a halo-exchange example, we demonstrate that application-level mapping to the network topology yields significant performance improvements.

## Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Network topology*

## General Terms

Design, Performance

## Keywords

MPI, Interconnection Network, Gemini, Network Topology

## 1. INTRODUCTION

A key component in high-performance computing (HPC) systems is the interconnection network, which integrates an array of computational resources into a single system and provides efficient data movement. Interconnection networks continue to be an area of high innovation, as they strive to keep pace with rapidly increasing levels of concurrency and greater demand for communication performance. As interconnects evolve, system topologies, link properties, and communication characteristics vary across systems.

Most HPC applications use the Message-Passing Interface (MPI) [10] as a portable interface for expressing data movement. MPI defines Cartesian and graph virtual process topology functions, which enable the user to indicate the communication pattern in the program and enable the implementation to map MPI processes to processors in a way that optimizes this communication.

We explore the impact of interconnect topology on application performance on the Blue Waters (BW) system at the National Center for Supercomputing Applications (NCSA), University of Illinois. BW is a Cray supercomputer that contains 237 XE6 and 32 XK7 cabinets, connected in a 3D torus using the Gemini interconnect. In this network, the X and Z dimension network links are twice those of the Y dimension. Because of its anisotropic communication characteristics, the Gemini interconnect presents interesting challenges to the efficient mapping of application-level communication patterns to the underlying network links.

Our point-to-point experiments reveal severe behavior differences among the different network dimensions. For instance, in accordance with the mentioned bandwidth disparity, we find a transfer rate difference of up to 45% when using X or Z links. Although the latency increase per traversed router is constant, the obtained transfer rates for large data payloads vary highly, depending on the number of hops.

An optimized rank-node placement to distribute the collaborative tasks according to the network particularities reveals potentially large benefits. For example, we obtained reductions in latency of up to 74% in MPI_Alltoall and 54% in MPI_Allgather collective operations when performed within Y rows of nodes in the torus, instead of within X and Z rows. Restricting these operations to planes, we see latency reductions of up to 59% and 53%, respectively, when using Y links. Moreover, we find up to 8% improvement in nearest-neighbor data exchanges when the collaborative tasks are placed according to the actual network topology instead of the system placement algorithm.

**Main contributions of this paper**: (1) Based on point-to-point microbenchmarks, we provide a characterization of the Gemini network's anisotropic behavior; (2) we prove that a Y-wise placement of the dual nodes per network Cartesian point is highly beneficial given the particular anisotropic behavior of this network; and (3) using a halo benchmark, we



*EuroMPI '13*, September 15 - 18 2013, Madrid, Spain
Copyright 2013 ACM 978-1-4503-1903-4/13/09...$15.00.
http://dx.doi.org/10.1145/2488551.2488564

demonstrate the potential gains of MPI-network topology matching versus the available node placement.

The rest of the paper is organized as follows. Section 2 presents an overview of related work. Section 3 describes the supercomputer class analyzed. Section 4 details our evaluation. Section 5 presents the conclusions of this paper.

## 2. RELATED WORK

Given the broad interest of MPI in the HPC arena, performance evaluations of communications employing this interface have been covered for the different supercomputers since MPI emerged two decades ago. Recent examples include studies of the performance behavior of MPI on the Blue Gene/P [2, 3, 4].

Performance evaluations on recent Cray XE/XK platforms have also been carried out. Early experience on the Titan system at Oak Ridge was described in [5]. Subsequent work includes an evaluation of the overhead of the Cray MPI implementation (as a justification for the design of a low-level benchmarking tool for internode, inter-GPU data transfers) [8] and an evaluation of the Partitioned Global Address Space runtime system of these platforms [12]. Prior work suggested that the placement of the nodes sharing the same Gemini application-specific integrated circuit (ASIC) in the Y direction was beneficial [1]. However, this work lacked a justificative base performance evaluation. Technical details supporting the anisotropy of this fabric were also missing in that work.

To the best of our knowledge, this is the first work focusing on the implications of the Gemini anisotropy over MPI collectives and nearest-neighbor communications.

## 3. SYSTEM

In this section we first describe the architectural view of the target system of this paper. Next, we present the details of the actual system where our experiments were performed. We then review the MPI rank ordering in these systems.

### 3.1 Cray XE6/XK7 Overview

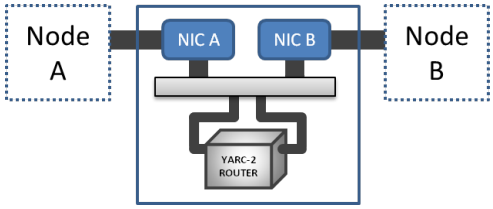
The Cray Gemini network system interconnect we target in this paper is employed by Cray XE6, XK6, and XK7 compute nodes. The system on which we performed our evaluation is composed of both XE6 and XK7 cabinets.

#### 3.1.1 The Compute Nodes

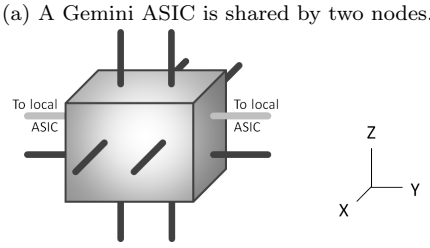
The Cray XE6 nodes are composed of two AMD Interlagos processors. The XK6/XK7 nodes contain a single processor plus an NVIDIA graphics processing unit (GPU). The XK6 version is equipped with a Fermi GPU, whereas its successor features a GPU implementing the most recent NVIDIA Kepler architecture. A detailed description of the internal architecture of these compute nodes is given in [8].

#### 3.1.2 Gemini Network

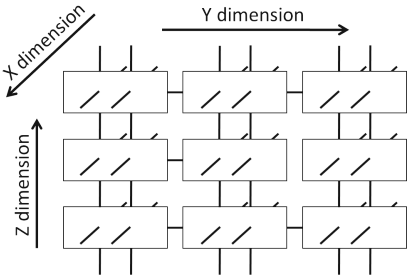
Compute nodes are connected to a Gemini fabric. Two nodes share the same Gemini ASIC, which contains the two network interface controllers (NICs) and a YARC-2 router, as shown in Figure 1a. This network is organized in a 3D torus topology, where each pair of nodes share a coordinate point in the network Cartesian topology. Each router features 10 outer links: two links per direction on the X and Z dimensions and one on the Y dimension, as depicted in Figures 1b and 1c.




(a) A Gemini ASIC is shared by two nodes.




(b) Logical view of Gemini router links.




(c) Gemini network layout.

Figure 1: Nodes and routers on the Gemini network.

The information provided by the vendor about the routing algorithm is limited and not concise, so we have preferred to directly cite it [6]: "For traffic designated as adaptive the Gemini router performs packet by packet adaptive routing, distributing traffic over lightly loaded links. (...) Hashed deterministic routing can be selected as an alternative when a sequence of operations must be performed in order."

The attainable transfer rate per node is limited by its injection rate, which in turn is that where the NIC is connected: a HyperTransport (HT) 3 link, offering up to 8 GB/s of effective transfer rate per node and direction. On the other hand, the minimum end-point latency is reported to be 1 microsecond, or 1.5 microseconds for MPI messages. Further information can be found in [6].

### 3.2 Blue Waters

Distributed among 237 XE6 and 32 XK7 cabinets, the 25,712 nodes of Blue Waters interweave a $23 \times 24 \times 24$ 3D torus. AMD 6276 processors provide 16 AMD bulldozer cores, split into two nonunified memory access nodes. Each XE6 node has two 32 GB memory banks, providing a bandwidth of roughly 50 GB/s each, whereas XK7 nodes incorporate a single memory bank. At the time of our writing this paper, the default MPI implementation is Cray MPICH [9] 5.6.1. For resource management, TORQUE 2.5.12 and Moab 6.1.9 are used in this system.

### 3.3 Job Placement and MPI Rank Ordering

Several job placement algorithms were investigated in [1]. Their final versions are currently being used by most Cray

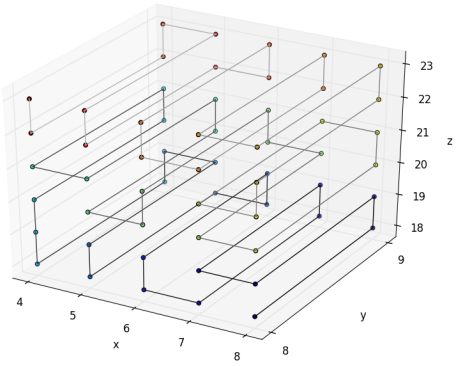



Figure 2: Example of the default rank ordering on XE/XK systems. Lines between nodes imply consecutive ranks.

systems. Ranks are ordered in a zigzag fashion (Figure 2) where the first and last ranks are adjacent, so as to decrease hop count and increase bisection bandwidth. This work started from usage data suggesting $8 \times 8 \times 8$ building blocks for reservations, or the best available approximation. Since XE6 routers contain two nodes that should be considered being in the Y direction, since Z links are faster than X links, and since every fifth Y link crosses a cabinet and is thus slower, the final choice was to use $4 \times 2 \times 8$ blocks.

# 4. EXPERIMENTAL EVALUATION

This section presents our evaluation. Basic point-to-point benchmarking is introduced first. Next, collective communications are evaluated. A *halo* benchmark [13] concludes this section. The study of the network anisotropic implications on production applications is left for future work.

## 4.1 Basic Microbenchmarks

To characterize the interconnection network, we first perform basic point-to-point benchmarking by employing the OSU Micro-Benchmarks 3.9 suite [11]. In particular, latency and uni- and bidirectional throughput are evaluated for internode transfers, focusing on the distance between peer nodes. These evaluations involve two nodes separated by a varying number of hops along all three dimensions at representative data payload sizes. The double nodes per ASIC are considered to be in the Y direction. We determined the increase in latency per router to be roughly $0.1\mu$s.

Despite both X and Z dimensions presenting the same number of links between routers, their performance behavior differs dramatically [1]. Z-wise communications are faster than their X-wise counterpart, showing up to 45% difference in the case of unidirectional throughput from 2 to 6 hops (Figure 3a). After 6 hops, the transfer rate gap between both dimensions may be considered negligible with the exception of 12 hops, where Z outperforms X in 60%.

The distance between communication peers plays an important role in this network, revealing anisotropic bounds in the number of hops leading to considerable performance differences. For example, the unidirectional transfer rate in the Z direction performs at 100% during the first 3 hops and at 70% from 4 to 6 hops, falling to less than 50% when farther nodes are involved in the data transfer (Figure 3a). In the case of bidirectional throughput (Figure 3b), the maximum

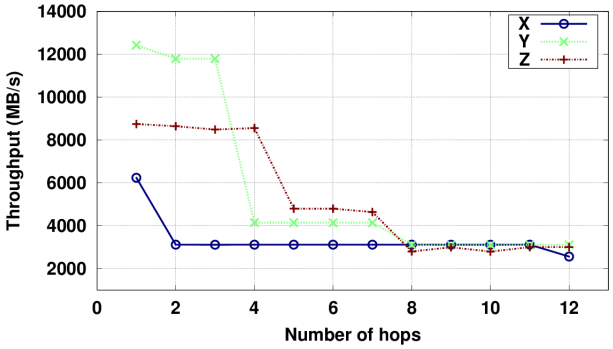



Figure 4: Internode aggregate transfer rate for 1 MB messages; 2 parallel paths concurrently transfer the data.

rate is maintained only for one-hop distances.

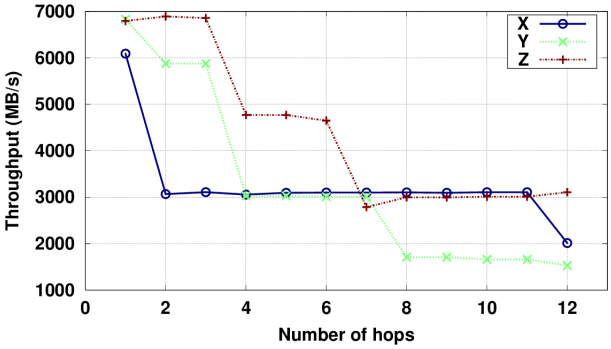
The technical details that would permit explaining the particularities exposed by this microbenchmark evaluation are not disclosed by the vendor. Here, we focus instead on the way this behavior may impact applications performing internode network communication by means of MPI.

## 4.2 Collective Communications

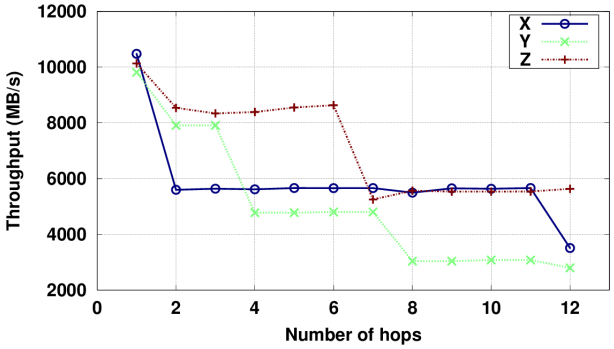
To evaluate the effects of network anisotropy on communications involving more advanced and widely used MPI communication patterns over a larger number of nodes, we ran some of the collective OSU microbenchmarks on communicators of different shapes. Given the arrangement of network links in Figure 1c, messaging along the Y direction may be expected to suffer from reduced transfer rate, a hypothesis corroborated by individual point-to-point experiments such as in Figure 3. However, an optimal node ordering and matching between MPI ranks and network topology may lead collectives saturating the links to greatly improve their performance in the Y direction. To illustrate this, we show in Figure 4 the throughput attained when a pair of processes transfers data to another pair of processes (one-to-one), placed in parallel network paths. As can be seen in the figure, the aggregate transfer rate increases for those transfers in the Y direction, because they feature separate links. In contrast, the double X and Z links become shared and do not provide a throughput increase with respect to the single-transfer case of Figure 3a. The placement of the dual nodes per ASIC along the Y direction brings another performance improvement for transfers in this coordinate, as nodes are closer with regard to the costly network hops.

This behavior led by the Y-wise node placing and topology matching is well suited to be exploited, e.g., by rowwise and—to a minor extent—planewise collective communications. To illustrate the performance of these, we conducted experiments involving splitting a contiguous cube of allocated nodes into rows and planes and subsequently invoking collective operations on each individual split, i.e., *rowwise*, where collectives are performed among the ranks placed in the same row in a given direction (X, Y, or Z), and *planewise*, where the collectives are performed among the ranks placed along the same plane (XY, XZ, and YZ). We modified the OSU `MPI_Alltoall` and `MPI_Allgather` microbenchmarks to implement these conditions. Note that the same collective will be happening concurrently in all splits.

Our rowwise experiments (Figures 5a and 5b) show re-




(a) Unidirectional.




(b) Bidirectional.

Figure 3: Internode MPI transfer rate microbenchmarks for 1 MB messages.

ductions in latency of up to 74% (`alltoall`) and up to 54% (`allgather`) when using Y rows with respect to the X direction. Additionally, our planewise benchmarks (Figures 5c and 5d) show up to 59% and 53% reductions in latency, respectively, when comparing those including the Y direction (XY and YZ) with those not including Y links (XZ).

Since the number of Y links is half the number in the X and Z directions, we identified a nonstraightforward performance behavior that can be exploited by production libraries and applications. The unexpected benefit in the Y direction is explained by the fact that the internal connection between nodes sharing a Gemini ASIC offers roughly the same throughput as the interrouter X and Z links. However, if leveraging a Y-wise arrangement of the double nodes of each network Cartesian point, Y single links are used by only two nodes, whereas the double X and Z links are used to communicate up to four nodes. In a saturated regime like ours, this means that adjacent nodes in the Y direction will experience around twice as much throughput as in X and Z.

## 4.3 Stencil Communications

The `MPI_Cart` family of calls provide rankwise Cartesian topology information. These functions allow users to define a Cartesian virtual topology and allow MPI tasks to obtain information about the topology they are part of. Although the underlying MPI implementation is not required to take into account the actual network topology when distributing the ranks across the user-specified topology, a good matching between this logical rank distribution and the actual network topology is highly desired in order to improve communication performance. This becomes especially relevant in those applications communicating mostly with their neighbors, such as those performing halo exchanges.

To assess the implications of our anisotropic network on stencil communications, we developed a *halo* test as a rough indication of the communication pattern of a range of production applications. The following are its features:

**Communications** Via nonblocking MPI primitives, with one process per node performing the halo exchange in a box disposal: 4 neighbors in the case of 2D and 8 neighbors for the 3D case, forming a periodic mesh.

**Memory** All dimensions feature the same size: 65,536 *double* elements per dimension for 2D tests (a total of 4 MB bidirectional transfers per rank performing a halo exchange with its neighbors), while 1,536 in the case of 3D (216 MB of data transfers per halo exchange). Data are actually stored in memory following a proper layout, so elements across Y and Z dimensions are not contiguous. Data are packed and unpacked in temporary buffers for network exchange. This process is interleaved with the actual exchange process and is considered in our timings. Single rows for 2D and planes in the 3D case are exchanged with neighbors.

**Timing** Average time spent in communications involving all nodes. To compute the throughput, we consider the entire amount of sent and received data with all the neighbors. Experiments are repeated 50 times, and the average is represented, obtaining a maximum relative standard deviation of 2.5%.

The key for an efficient halo exchange is to arrange the ranks taking into account the network topology, in order to minimize the number of links the data travel along. However, the Cray MPI implementation ignores the network topology; that is, it effectively ignores the `reorder` parameter in the `MPI_Cart_create` function, hence relying on the node placement order (see Section 3.3). This ordering is the fruit of a heuristic algorithm designed to be beneficial for a wide range of communication patterns. However, applications specifying an MPI Cartesian topology would highly benefit from a matching to the actual network topology. Our evaluations considered three different distributions:

**Plain** Based on their original ordering, ranks are distributed along the X, next Y, and then Z dimensions.

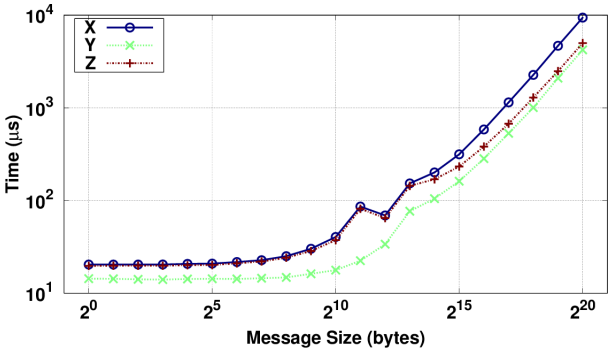
**Cart_create** The MPI library distributes the ranks. This effectively results in a Y-major ordering for 2D topologies and Z-Y-X in the case of 3D.

**Custom** MPI topology matches the network topology, accomplished by employing the network topology information provided by the PMI interface in MPICH.

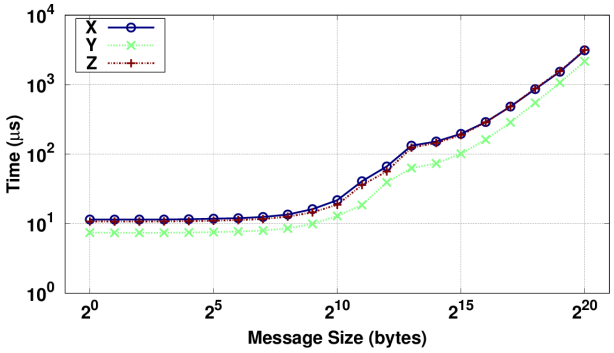
In order to avoid performance artifacts introduced by network hops, and hence clearly reveal the potential of MPI and network topology matching, our tests were run in a block of contiguous nodes within the network topology.

The results of our 2D halo benchmarks are shown in Figure 6a[1], covering a range from 2 to 12 nodes in each dimen-

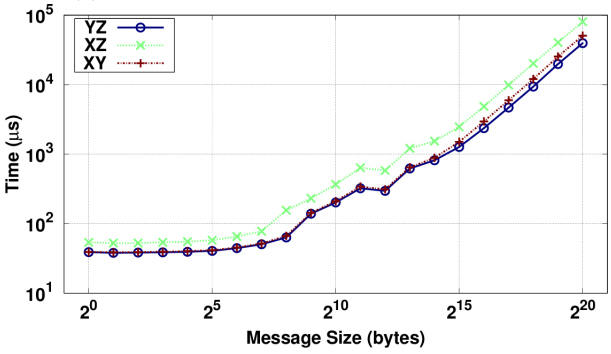
[1]Our halo tests spend a large portion of time in the data pack and unpack stages, and hence these timings should not be directly compared with our micro-benchmark results.



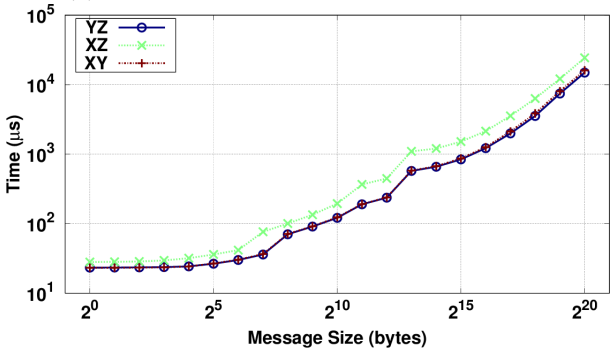

(a) Rowwise `MPI_Alltoall` on a $7 \times 7 \times 7$ cube.




(b) Rowwise `MPI_Allgather` on a $7 \times 7 \times 7$ cube.




(c) Planewise `MPI_Alltoall` on a $7 \times 7 \times 7$ cube.




(d) Planewise `MPI_Allgather` on a $7 \times 7 \times 7$ cube.

Figure 5: Topology-dependent collectives.

sion. The network topology-aware placement (`Custom`) outperforms the two other placement possibilities for a dimension of 6 or more nodes. Up to 4% improvement is observed for $12 \times 12$ nodes with respect to the MPI-driven distribution (`Cart_create`). On the other hand, the naïve distribution (`Plain`) shows a slightly better performance of up to 1.4% in the case of a $5 \times 5$ node topology with respect to the MPI-assisted ordering, revealing a slightly better matching to the default node ordering algorithm. This difference lies in the fact that those data along other than the X dimension are not contiguous in memory, and hence the packing/unpacking stages are more costly. Because of the network anisotropy, if those have to be transferred along a dimension offering a lesser throughput, transfer times to different neighbors are further imbalanced, causing an overall overhead. Moreover, the topology matching distribution favors scalability, since data have to traverse fewer links among logical neighbors.

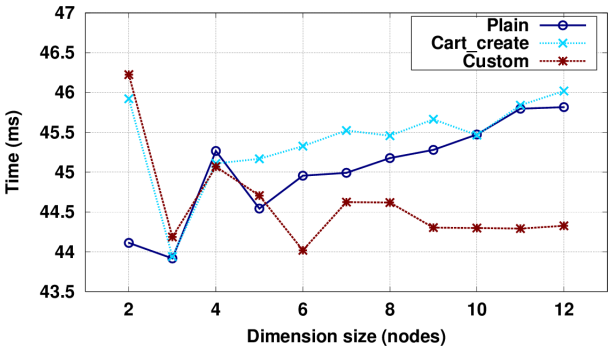
Figure 6b depicts our halo benchmarking for a 3D topology, ranging from 3 to 8 nodes per dimension. In this case, the Z-Y-X distribution algorithm (`Cart_create`) outperforms the naïve X-Y-Z sorting (`Plain`) up to 4% in the case of a 6-node dimension cube. Again, this difference is because of the network anisotropic properties. On the other hand, the `Custom` ordering outperforms the `Plain` distribution in 8.5%, and the `Cart_create` in 5%, ensuring a minimum network link usage, and depicting the relevance of the introduction of topology-aware ordering in MPI libraries.

Note that minimizing network hops on data transfers is expected to be highly beneficial in production environments, where noncontiguous nodes are likely to be assigned by the global job scheduler. However, an optimal matching between the Cartesian topology defined for an application and the actual network topology is nontrivial. Employing existing mapping libraries such as LibTopoMap [7] for this purpose is left for future work and as an interesting topic to be considered by MPI library developers.

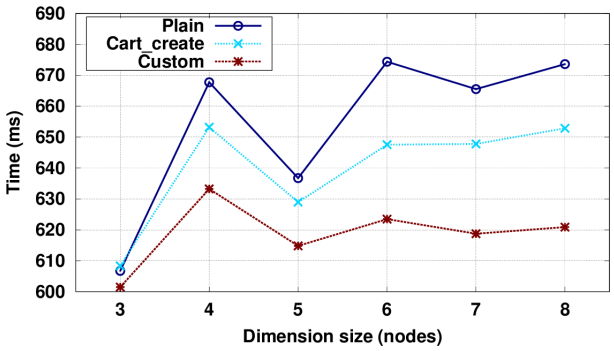
## 5. CONCLUSIONS

In this paper we studied the anisotropic implications of Cray Gemini networks on MPI communications. After a characterization of this interconnect by means of point-to-point benchmarks, we evaluated the behavior of MPI collectives along the different dimensions and planes of the network topology of our target system. We proved that when the nodes sharing a network coordinate are considered to be placed along the dimension featuring lesser network links, communication performance along that direction is higher than expected, hence maximizing the use of the network resources. Additionally, we demonstrated by means of a halo benchmark that including awareness of the network topology in the MPI library can outperform the heuristic-based rank ordering of this system.

The results of our study demonstrate the benefits of an MPI-network topology matching for common application scenarios. Since the current MPI implementation on Cray systems ignores the `reorder` parameter meant for this purpose, currently topology-sensitive applications need to be concerned about network placement, aided (one hopes) by external libraries. This situation poses an unnecessary overhead for application developers, who should be able to rely on the capabilities of the MPI library for this purpose. As suggested in this paper, MPI libraries would benefit from the incorporation of existing topology matching work. Although




(a) 2D.




(b) 3D.

Figure 6: Halo benchmarks.

this paper focuses on the Cray Gemini interconnect, other systems that are still missing the topology reorder capability may benefit from its implementation as well.

## Acknowledgments

We thank Torsten Hoefler, from ETH Zürich, for his valuable comments and suggestions to improve the quality of this paper. This research is part of the Blue Waters sustained petascale computing project, supported by NSF (award no. OCI 07-25070) and the state of IL. This work is also part of the "System Software for Scalable Applications" PRAC allocation support by the National Science Foundation (award number OCI-1036216). This work was supported in part by a NEIS-P2 subaward from NCSA/UIUC and in part by the U.S. Dept. of Energy under contract DE-AC02-06CH11307.

## 6. REFERENCES

[1] Carl Albing, Norm Troullier, Stephen Whalen, Ryan Olson, Joe Glenski, Howard Pritchard, and Hugo Mills. Scalable node allocation for improved performance in regular and anisotropic 3D torus supercomputers. In *Recent Advances in the Message Passing Interface*, volume 6960 of *LNCS*. 2011.

[2] Pavan Balaji, Anthony Chan, William Gropp, Rajeev Thakur, and Ewing Lusk. Non-data-communication overheads in MPI: Analysis on Blue Gene/P. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pages 13–22. Springer, 2008.

[3] Pavan Balaji, Anthony Chan, Rajeev Thakur, William Gropp, and Ewing Lusk. Toward message passing for a million processes: Characterizing MPI on a massive scale Blue Gene/P. *Computer Science-Research and Development*, 24(1-2):11–19, 2009.

[4] Pavan Balaji, Harish Naik, and Narayan Desai. Understanding network saturation behavior on large-scale Blue Gene/P systems. In *15th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 586–593. IEEE, 2009.

[5] Arthur S. Bland, Jack C. Wells, Bronson Messer, Oscar R. Hernandez, and James H. Rogers. Titan: Early experience with the Cray XK6 at Oak Ridge National Laboratory. In *CUG 2012*, May 2012.

[6] Cray Inc. *The Gemini Network Rev 1.1*, August 2010.

[7] Torsten Hoefler and Marc Snir. Generic topology mapping strategies for large-scale parallel architectures. In *Proceedings of the 2011 International Conference on Supercomputing (ICS'11)*, June 2011.

[8] Antonio J. Peña and Sadaf R. Alam. Evaluation of inter- and intra-node data transfer efficiencies between GPU devices and their impact on scalable applications. In *The 13th International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 2013.

[9] Howard Pritchard, Igor Gorodetsky, and Darius Buntinas. A uGNI-based MPICH2 nemesis network module for the Cray XE. In *18th European MPI Users' Group conference on Recent advances in the Message Passing Interface (EuroMPI'11)*, 2011.

[10] The MPI Forum. http://www.mpi-forum.org, 2013.

[11] The Ohio State University. OSU Micro-Benchmarks. http://mvapich.cse.ohio-state.edu/benchmarks, 2013.

[12] Abhinav Vishnu, Monika Bruggencate, and Ryan Olson. Evaluating the potential of Cray Gemini interconnect for PGAS communication runtime systems. In *19th Annual Symposium on High Performance Interconnects (HOTI)*, 2011.

[13] Alan J. Wallcraft. Benchmarking an ocean model. *NAVO MSRC Navigator*, (Fall), 1999.