Visualizing the Five-dimensional Torus Network of the IBM Blue Gene/Q

Collin M. McCarthy†, Katherine E. Isaacs†, Abhinav Bhatele*, Peer-Timo Bremer*, Bernd Hamann†

†Department of Computer Science, University of California, Davis, California 95616 USA
*Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, California 94551 USA
Email: †{cmccarthy, keisaacs, bhamann}@ucdavis.edu, *{bhatele, ptbremer}@llnl.gov

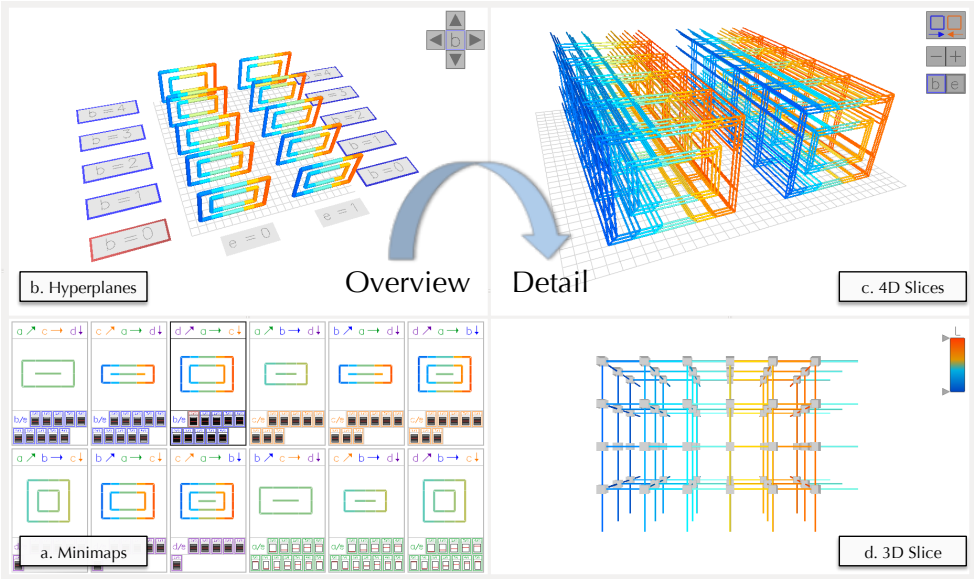

Fig. 1: Visualization of IBM Blue Gene/Q Five-dimensional torus interconnection network using four linked views.

***Abstract*—Understanding the interactions between a parallel application and the interconnection network over which it exchanges data is critical to optimizing performance in modern supercomputers. However, recent supercomputing architectures use networks that do not have natural low-dimensional representations, making them difficult to comprehend or visualize. In particular, high-dimensional torus networks are common and are used in four of the top ten supercomputers and eight of the top ten on the Graph500 list. We present a new visualization of five-dimensional torus networks. We use four connected views depicting the network at different levels of detail, allowing analysts to observe general large-scale traffic patterns while simultaneously viewing individual links or outliers in any specific section of the network. We demonstrate this approach by analyzing network traffic for a pF3D simulation running on the IBM Blue Gene/Q architecture, and show how it is both intuitive and effective for understanding and optimizing parallel application behavior.**

I. Introduction

Massively parallel applications require many processes running in a carefully orchestrated and efficient way to achieve maximum performance. In particular, processes running on different nodes in the network typically exchange large amounts of data which often causes performance bottlenecks. This is explained by a number of factors such as the algorithm that dictates the frequency and overall need for communication, the mapping of processes onto physical nodes, the MPI calls and their implementation, and the underlying routing algorithms. The combined effects are difficult to predict making optimization challenging. Analysts can record the number of packets sent over each link during execution under different conditions, and then analyze the traffic to gain more insight into the observed performance. Visualization tools can aid in understanding this data by providing topological context to the variability of network usage. This context can reveal spatial patterns, such as correlated usage among directions of the torus, or local behavior arising from imbalanced communication needs, which are rarely available in purely statistics-based approaches. Existing visualizations have depicted interconnection networks with natural two-dimensional (2D) or three-dimensional (3D) embeddings such as trees and low-dimensional meshes and tori. However, newer network topologies such as higher-dimensional tori do not have this property, so visualizations have not been available thus far. To help bridge this gap, we present a visualization of the five-dimensional (5D) torus network of the IBM Blue Gene/Q.

Each node in the BG/Q has 16 cores capable of up to four hyperthreads, for a maximum of 64 processes per node, and

each node has ten pairs of incoming and outgoing links, two in each torus direction A, B, C, D and E. The E-direction is constrained to be no more than two nodes wide, meaning each node has double the bandwidth with its neighbor in the E-direction. While our visualization tool is general in the sense that it can be applied to any 5D torus or mesh, we utilize the fact that the E-direction is a maximum of two nodes wide to simplify and reduce the number of views when looking at slices of the network traffic data. We have implemented our visualization as a module in Boxfish [1] that provides filtering capabilities on our metrics and enables linking with other Boxfish modules.

II. Visualizaton Approach

Our visualization (Fig. 1) is composed of four views ranging in level of detail from a highly aggregated overview to a focused slice showing all elements. The *minimaps view* provides an overview of possible torus projections. The *hyperplanes view* shows the full set of overviews for a chosen projection. The *4D slices view* displays individual link data in up to five dimensions. The *3D slice view* shows all nodes and links in a single 3D subtorus. Views can be resized or collapsed, allowing users to focus on a view of interest while maintaining context from the other views. Metrics of interest, e.g., number of packets, are encoded using color.

A. 2D Projection of a 3D Torus

Three of our four views use the 2D projection of the 3D torus (Fig. 2) and minimap overview of Landge et al. [2].

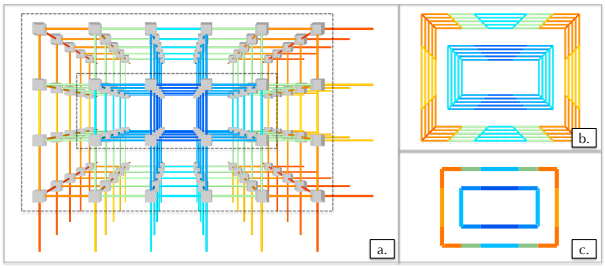

Fig. 2: Correspondence between (a) the 3D visualization of a 3D torus, (b) the 2D projection, and (c) the minimap projection.

The 2D projection regards the mesh-representation of the 3D torus as a series of nested cylinders, as shown by the dashed lines (Fig. 2a). Each cylinder is projected into two dimensions as a series of closely nested rectangles by representing links going into the torus as diagonal links in the projection (Fig. 2b). These maintain nesting with the other cylinders. This representation does not suffer from occlusion but removes approximately half of the links from the view and shortens the diagonal links. The minimaps remove the diagonal links of the 2D projection and aggregate the nested rectangles of a cylinder into a single rectangle (Fig. 2c). Using this projection we are able to take advantage of our users' familiarity with an existing visualization, and provide a simple but effective way of aggregating a multi-dimensional subspace.

B. Slicing and Projecting the 5D Torus

We obtain a 3D slice of the 5D torus by taking all nodes with the same coordinate in two of the dimensions, one of which we always choose to be E as it is only two nodes wide. Performing this operation for all combinations of the two dimensions partitions the 5D torus into 3D subtori, to which we can apply the 2D projection discussed in Section II-A. For example, Fig. 3b shows minimaps for each 3D torus obtained from slicing in the B- and E-directions. The two highlighted minimaps are shown in full in the corresponding 4D slice view (Fig. 3c). All the minimaps in the hyperplanes view are aggregated into a single minimap in Fig. 3a.

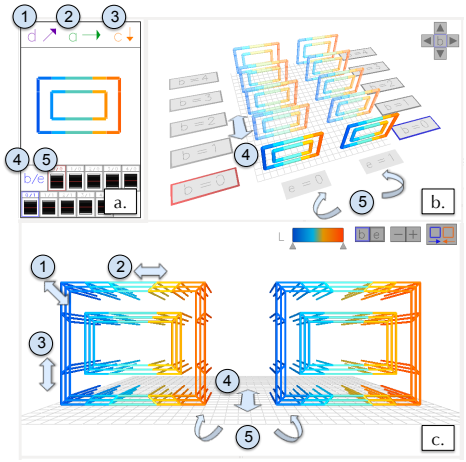

Fig. 3: The selected minimap assigns three of the torus directions A-D to the diagonal (1), horizontal (2), and vertical (3) directions of the 2D projection. The remaining direction is shown as multiples (4) in each view and drawn explicitly in the 4D slice (c). Each view has a consistent, specific handling of the E-direction (5).

The minimap view contains twelve summary minimaps of different projections (Fig. 1a). By selecting one of these, the user chooses which projection is used in all other views – in other words, which three of the A-, B-, C-, and D-directions are mapped to the diagonal, horizontal and vertical directions of the 2D projection. Selecting minimap {D,A,C} in Fig. 3a sets D to the diagonal, A to the horizontal, and C to the vertical links in the hyperplanes and 4D slices views (Fig. 3bc). The remaining direction, B, extends into the third dimension in both views. Each view takes advantage of the two-node long E-direction, allowing us to view the 5D torus as two 4D torus halves (Fig. 1c).

We provide two ways of representing the E direction. 4D slices from both E=0 and E=1 can be viewed side-by-side, as in Fig. 3c, or with an inset view where the two 4D torus halves are nested slightly offset from one another at a +/- 65 degree angle (Fig. 4). This approach makes horizontal, vertical, and diagonal links appear as double-wide links, while allowing us to show E links explicitly.

To further reduce occlusion in the 4D slice view, the user can select which 3D tori are represented via the labels in the hyperplanes view (Fig. 3b) or the glyphs in the minimaps view (Section II-C). Selection of which subtorus is shown in the 3D slice view is done similarly.

Each of the four main views serve a unique purpose. The minimaps view aggregates all 3D tori of the specified three dimensions, giving the user a comprehensive overview. The hyperplanes view shows all of the 3D tori as separate minimaps, arranged to evoke the extra dimensions and visually

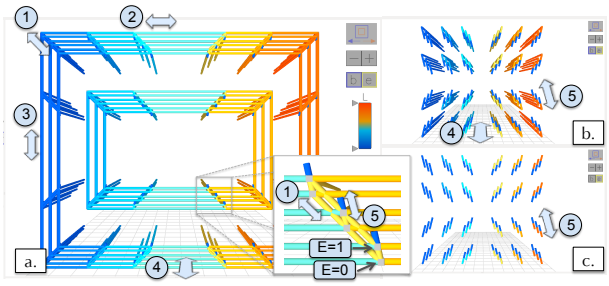

Fig. 4: Link drawing options of the 4D slice inset view: (a) All directions, (b) fourth and fifth dimension, (c) fifth dimension only.

differentiate them from the minimaps view. This view allows the user to easily select which 3D tori to view in more detail in the 4D slices view, and provides a better context for what is being shown in the remaining views. The 4D slices view shows individual links, and allows the user to analyze links that connect the adjacent 3D tori of the dimension set. Finally, the 3D slice view, while showing the smallest subset of the total network, allows for exploration of all links within the selected 3D torus. This last view is the most widely understood amongst our user base and also shows the nodes between the links which can be colored with a (possibly different) metric.

C. Representing Variance in Aggregated Views

Each projection in the minimaps view is aggregated in two dimensions. This aggregation can mask potentially interesting distributions, as shown in Fig. 5 where the selected minimap appears to have constant traffic but displays variations in the hyperplanes and 4D slice views. As the minimaps are used to gain the initial overview and navigate the 5D torus, it is essential to be able to identify variance within them quickly.

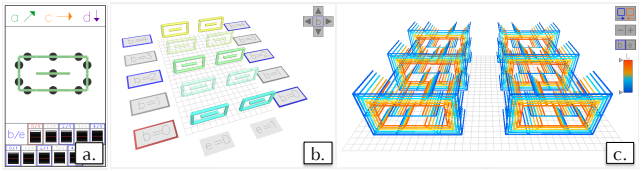

Fig. 5: The selected minimap looks constant (green) but the corresponding hyperplanes show variance in the B-direction and the 4D slice view shows variance in the A-direction.

Under each of the minimaps we draw a glyph for each of the 3D subtori aggregated by the minimap (Fig. 6b). These glyphs are similar to a box-and-whisker plot, showing the mean, standard deviation, and total spread of the mapped metric values. By clicking on the glyph the user can turn on or off that hyperplane in the 4D slice view, which also updates which links are aggregated in the minimap so only selected planes are used in the minimap construction. To provide even more detail, we optionally depict the variance in each segment of the minimap using circles (Fig 6a).

III. Case Study

One of the greatest benefits of a topology-specific visualization tool is the ability to clearly map performance data onto physical links, and to make visible which dimensions or specific links are being underutilized or overburdened. Here, we use our visualization to analyze performance data

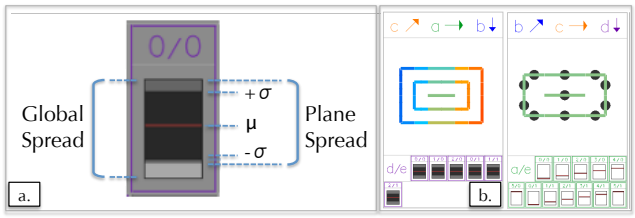

Fig. 6: (a) Glyph showing the distribution of metric values for one of the hyperplanes aggregated by a minimap. (b) Low (left) and high (right) inter-plane variance depicted by hyperplane glyphs.

gathered from a recent study on task mapping on the IBM Blue Gene/Q [3]. We look at mappings of pF3D [4], a laser-plasma interaction code developed at LLNL.

In topology-aware task mapping [5], processes are placed on hardware nodes based on the specific network topology to reduce the overall communication time. The aim is to minimize network congestion, a difficult task considering that messages shared between processes must often traverse multiple hops to reach their destination. For larger-diameter torus networks this is even more difficult, as communication between distant nodes places an additional burden on the shared links in-between. Nevertheless, an intelligent task mapping can provide significant speedups in communication time.

A significant portion of pF3D's communication time is spent performing 'Alltoall' operations which take place in the X- and Y-directions of the 3D Cartesian domain of the physical simulation. These are performed by sub-communicators, subsets of the processes that communicate together, over which MPI collective calls such as `MPI_Alltoall` take place. In the X direction, the Alltoall is carried out by a sub-communicator of 32 processes with fixed Y and Z domain coordinates. Similarly, the Y sub-communicator is 16 processes with fixed X and Z domain coordinates. The task mapping aims to optimize these orthogonal sub-communicators simultaneously.

The first mapping we examine is the default for BG/Q, ABCDET, where T stands for thread, with processes are filled along T, then E, and so on. We use two threads per core and all 16 cores per node. This method assigns the maximum amount of processes to a single node before moving on to the next one, resulting in X sub-communicators being completely contained on a node. The second mapping is a tiling generated using Rubik [6], whereby each Z-plane of pF3D is mapped to a torus tile of size ABCDET = (4, 4, 4, 4, 2, 1), meaning each process in the plane is on a different node. Though this mapping does not take advantage of shared memory for the X Alltoalls as the Default does, it may make better use of bandwidth across the network, which was shown to be beneficial on previous architectures [6].

Fig. 7 shows time spent in MPI calls (left) and packets in the network (right) for both mappings in a 4,096 node (131,072 process) run. This data demonstrates that the Tile mapping far surpasses the Default mapping, reducing the total MPI time by 64% and the maximum number of packets by 66%. To explore why we observe this behavior, we employ visualization.

First we begin with an overview using the minimaps to compare the two mappings. Fig. 8 shows that traffic is evenly

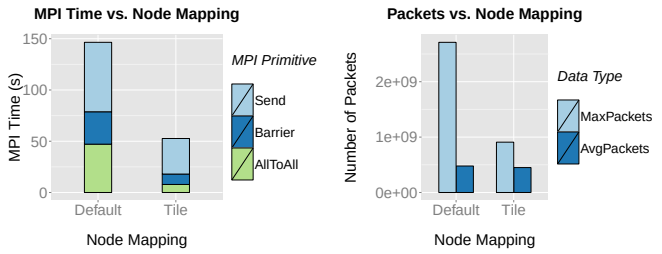

Fig. 7: Performance data for pF3D simulation.

distributed and generally moderate for the Tile mapping, as indicated by the even blue color. The Default mapping shows high utilization (orange) in the D direction and very low utilization in all other directions. We suspect this overuse of the D direction leads to congestion and lower performance.

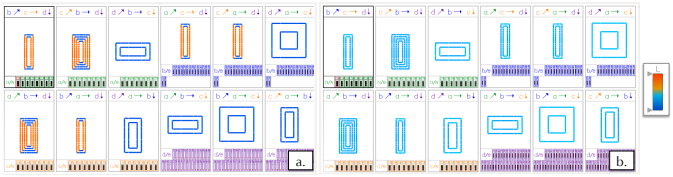

Fig. 8: pF3D link usage, (a) Default vs. (b) Tile mapping.

The reason for this imbalanced use of links in the Default mapping becomes clear when we use the highlighting and filtering capabilities of our visualization to focus on a single sub-communicator. The X sub-communicators are completely on-node, resulting in no communication, so we examine a Y sub-communicator. Fig. 9 shows the nodes and links used by a single sub-communicator performing a Y Alltoall. The links utilized by this sub-communicator are solely in the D and E directions. The A, B, and C directions are not used.

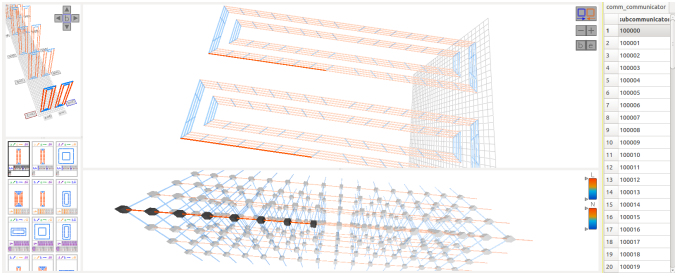

Fig. 9: Single Y sub-communicator under the Default mapping. The 4D slices show that this sub-communicator occupies adjacent D links for both E=0 and E=1. The 3D slice shows detail in the E=0 subtorus.

Fig. 10 and 11 show X and Y sub-communicators for the Tile mapping. As the Tile mapping has no more than one process per sub-communicator per node, we expect 32 nodes in the X sub-communicator and 16 in the Y. The 4D slice and 3D slice views show the X sub-communicator uses the C, D, and E directions while the Y sub-communicator uses the A and B directions. The egalitarian use of the torus directions results in the even use of links seen in Fig. 8.

IV. Conclusion

We have presented an intuitive multi-view visualization tool for exploring performance on 5D torus networks which

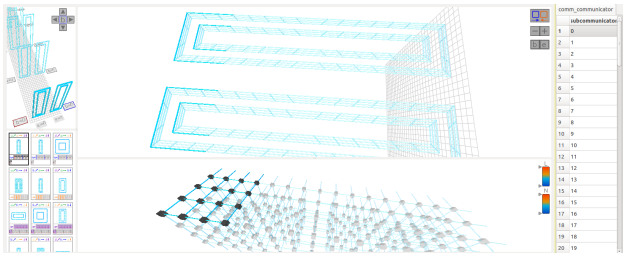

Fig. 10: Single X sub-communicator under the Tile mapping. The 4D slices show this sub-communicator occupies adjacent C and D links for both E=0 and E=1. The 3D slice shows detail in the E=0 subtorus.

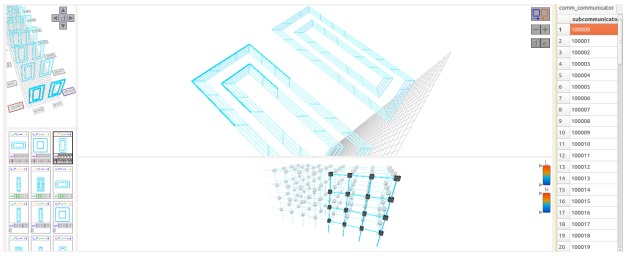

Fig. 11: Single Y sub-communicator under the Tile mapping. The 4D slices show this sub-communicator occupies A and B links for E=0 only. The 3D slice shows all links of this sub-communicator.

captures the topological structure of the network despite the high dimensionality. Through a case study on task mapping of a highly scalable production simulation, we have demonstrated the effectiveness of this design for identifying network traffic patterns and understanding complex task layouts.

Acknowledgment

The authors would like to thank Nikhil Jain for providing guidance regarding BG/Q link counter data and Todd Gamblin for his helpful feedback. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-CONF-661000.

References

[1] K. E. Isaacs, A. G. Landge, T. Gamblin, P.-T. Bremer, V. Pascucci, and B. Hamann, "Abstract: Exploring performance data with Boxfish," in *Proc. of the 2012 SC Companion*, ser. SCC '12, 2012, pp. 1380–1381.

[2] A. G. Landge, J. A. Levine, K. E. Isaacs, A. Bhatele, T. Gamblin, M. Schulz, S. H. Langer, P.-T. Bremer, and V. Pascucci, "Visualizing network traffic to understand the performance of massively parallel simulations," *IEEE Trans. on Vis. and Comp. Graphics (Proc. InfoVis '12)*, vol. 18, no. 12, pp. 2467–2476, 2012.

[3] A. Bhatele, N. Jain, K. E. Isaacs, R. Buch, T. Gamblin, S. H. Langer, and L. V. Kale, "Improving application performance via task mapping on IBM Blue Gene/Q," in *Proc. of IEEE Intl. Conf. on High Performance Computing (to appear)*, ser. HiPC '14, Dec. 2014.

[4] C. H. Still, R. L. Berger, A. B. Langdon, D. E. Hinkel, L. J. Suter, and E. A. Williams, "Filamentation and forward brillouin scatter of entire smoothed and aberrated laser beams," *Physics of Plasmas*, vol. 7, no. 5, pp. 2023–2032, 2000.

[5] A. Bhatele, "Topology Aware Task Mapping," in *Encyclopedia of Parallel Computing*, D. Padua, Ed. Springer Verlag, 2011.

[6] A. Bhatele, T. Gamblin, S. H. Langer, P.-T. Bremer, E. W. Draeger, B. Hamann, K. E. Isaacs, A. G. Landge, J. A. Levine, V. Pascucci, M. Schulz, and C. H. Still, "Mapping applications with collectives over sub-communicators on torus networks," in *Proc. of the ACM/IEEE Intl. Conf. on Supercomputing (SC12)*, ser. SC '12, Nov. 2012.