Stony Brook University

OFFICIAL COPY

The official electronic file of this thesis or dissertation is maintained by the University Libraries on behalf of The Graduate School at Stony Brook University.

© All Rights Reserved by Author.

Analysis of Supercomputers and Development of a Novel Network

A Dissertation Presented

by

Reid Powell

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

December 2010

Stony Brook University

The Graduate School

Reid Powell

We, the dissertation committee for the above candidate for the Doctor of Philosophy degree, hereby recommend acceptance of this dissertation.

Dr. Yuefan Deng—Dissertation Advisor
Professor, Department of Applied Mathematics and Statistics

Dr. W. Brent Lindquist—Chairperson of Defense
Professor, Department of Applied Mathematics and Statistics

Dr. Alan Tucker—Committee Member
Distinguished Professor & Undergraduate Program Director, Department of Applied Mathematics and Statistics

Dr. Dantong Yu—Committee Member
Research Engineer & Manager, Computational Science Center, Brookhaven National Lab

This dissertation is accepted by the Graduate School.

Lawrence Martin
Dean of the Graduate School

Abstract of the Dissertation

Supercomputer Network Design and Analysis

by

Reid Powell

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

2010

As supercomputers have reached nearly 300,000 cores with 2 petaflops in Linpack performance in June 2010, energy consumption and temperature control are posing a developmental bottleneck. We retrospectively and comparatively examine all of the available data contained in the Green500 list that launched in November 2007, and the Top500 list, and propose a novel representation and analysis of the data, highlighting major evolutionary trends.

With these new insights, we introduce a new technique for generating more efficient networks by systematically interlacing bypass rings to torus networks (iBT networks). The resulting network can improve the original torus network by reducing the network diameter, node-to-node distances, and by increasing the bisection width without increasing wiring and other engineering complexity. We present and analyze the statement that a 3D iBT network proposed by our technique outperforms 4D torus networks of the same node degree. We found that interlacing rings of sizes 6 and 12 to all three dimensions of a torus network with meshes 30×30×36 generates the best network of all possible networks, including 4D torus and hypercube of approximately 32,000 nodes. This demonstrates that strategically interlacing bypass rings into a 3D torus network enhances the torus network more effectively than adding a fourth dimension, although we may generalize the claim. We also present a node-to-node distance formula for the iBT networks.

For TGL

Table of Contents

List of Figures

List of Tables

I am grateful for the years of exciting experiences, guidance, encouragement and support that my advisor, Dr. Yuefan Deng, has provided. Thank you.

I would like to extend my warmest thanks to this dissertation committee: Dr. Alan Tucker, Dr. Brent Lindquist, and Dr. Dantong Yu.

I also wish to thank the faculty and staff of the Applied Mathematics and Statistics Department at Stony Brook.

Thanks, also, to my dear friends whom I met through the Deng lab both at home and abroad.

Finally, to my family and fiancée: thank you.

This research utilized resources at the New York Center for Computational Sciences at Stony Brook University /Brookhaven National Laboratory which is supported by the U.S. Department of Energy under Contract No. DE-AC02-98CH10886 and by the State of New York.

This research used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357.

The text of this dissertation in part is a reprint of the materials as it appears in [1-2].

Chapter 1: Introduction

1.1 Latest Developments in Supercomputer Design

Although interconnection networks have been studied for decades, for example those used in telephone networks, computer telecommunications networks, and backplane buses, sources like the Top500 [3] illustrate the rapid evolution of the interconnection systems that are developed by the high performance computing community. In [4], a lack of standards and high demand for performance and reliability is cited as creating a diverse microprocessor system interconnection network landscape. Now, more than ever, is there a need for accurate and efficient methodologies for the evaluation and comparison of such networks.

1.2 Contributions

This work makes two major contributions. First, it provides analysis of supercomputer performance data over the past 18 years, highlighting major evolutionary trends.

Second, along with the insights gained by such an analysis of the high performance computing landscape, we introduce a new technique for generating more efficient networks by systematically interlacing bypass rings to torus networks (iBT networks). A 3D iBT network proposed by our technique outperforms 4D torus networks of the same node degree, as well as many other popular networks.

1.2.1 Evolutionary Performance Analysis of Supercomputers

We examine the current and historic high performance computing landscapes from four major perspectives. The first is a performance and efficiency analysis for the current Top500 systems, summarizing many features of the list in a concise way. The second considers this data over the past three years, partitioning the set of systems by major features. Next, an evolutionary analysis is performed, where the efficiencies of sets of systems over several six-month periods are summarized by moving system averages. Finally, phylogenetic analyses of major system manufacturers are performed.

1.2.2 Development of a Novel Network

Inspired by the simplicity and efficiency of the popular torus interconnection network [3, 5-10], we interlace a series of bypass rings to a base torus network to create the iBT network, achieving the low and narrowly distributed internode distances of nearly-spherical networks [11-14] and the low average internode distances of hypercube networks [15-17], while maintaining the simplicity and low node-degree of torus networks. Previous efforts have proposed similar bypass structures [18-22]; however these lacked feasibility due to prohibitively long communication links.

We justify the performance of the iBT network by comparing it with the performance of the networks listed above, as well as with other popular structures like the de Bruijn network [23], cube-connected cycles (CCC) [24], the hybrid fat tree [25], and scalable Barrel Shifter network [26]. Finally, we present an application-specific method to compare these networks

1.3 Dissertation Structure

The dissertation is organized as follows. In the following sections, we summarize the contributions of this dissertation to the field of supercomputer network design and analysis. Then, an analysis of recent architectural trends in high-performance supercomputing is presented. In Chapter 2, the problems of network design and analysis are stated and defined. The proposed networks are described in detail, followed by a detailed definition of the topological analysis tools used to compare the newly designed networks with current architectures. In Chapter 3, the results of the topological analysis are presented for all networks considered. In addition, application specific analysis data is presented. Next, in Chapter 4, the implications of the results presented are discussed and conclusions are drawn. Lastly, in Chapter 5, areas of future research are enumerated and described, from immediately feasible experiments to larger-scale future efforts.

Chapter 2: Evolutionary Performance Analysis of Supercomputers

The biannual Top500 list of the highest performing supercomputers has chronicled, and even fostered, the development of recent supercomputing platforms. Coupled with the Green500 list that launched in November 2007, the Top500 list has enabled analysis of multiple aspects of supercomputer design. In this chapter, a comparative and retrospective study, we examine all of the available data contained in these two lists and propose a novel representation and analysis of the data, highlighting several major evolutionary trends.

While there have been many efforts to develop a comprehensive tool set to evaluate and analyze supercomputing platforms, none has achieved the popularity that the Linpack benchmark and the Top500 list have. Since 1993, [3] has released a biannual list of the fastest 500 supercomputers that have run the Linpack benchmark [3]. Intended as a means to facilitate comparison between the world's most powerful supercomputers, it also consequently fostered a sense of competition between leading vendors, driving performance improvements.

As supercomputers have reached nearly 300,000 cores[1] with 2 petaflops[2] in Linpack performance in June 2010 [3, 27], energy consumption and temperature control are posing a developmental bottleneck; thus, power efficiency has garnered considerable concern in the supercomputing platform design process. It is noted in [28] that in 2001, the infrastructure and energy cost of a 1U server has exceeded its purchase cost. Since then, clusters like Green Destiny [29] have attempted to explore “power-aware” supercomputer designs. Inspired by the Top500, in an effort to bring power efficiency to the forefront of supercomputer design, the Green500 [30] has been released as listing of the world’s most powerful supercomputers, fostering competition in the realm of power reduction.

[1] JUGENE, a Blue Gene/P Solution at Forschungszentrum Juelich (FZJ) with 294912 cores.
[2] Jaguar, a Cray XT5-HE achieving 1.759 petaflops.

Together, the Top500 and Green500 provide a rich set of data for looking into the evolution of the supercomputing landscape. Previous efforts such as [31-34] have noted patterns within the first 15 years of the Top500 list. Motivation, background, and preliminary findings for the Green500 list appear in [28-30, 35-39].

This Chapter, providing a novel presentation of the Top500 and Green500 data and identifying developmental trends in supercomputer design, is organized as follows. In Section 2.1, we detail our techniques for analyzing the evolution of the top supercomputers. Correlating performance representations and analysis are presented in Section 2.2; trends are highlighted and discussed here, as well. Finally, in Section 2.3 we discuss the implications of this new perspective with respect to future developments.

2.1 Evaluation Metrics of Supercomputers

Evaluating supercomputers is a daunting task, given the many aspects of the computers that are selectively important to individual users. We employ the most popular evaluation tools such as the Top500 and Green500 while developing our own plots.

2.1.1 Top500

June 2010 marks the 35th release of the Top500 list of supercomputers. The Top500 ranks the 500 submissions with the highest R_{max} value, a measure of maximum performance a computer (in $Gflops/W$) achieved when running the HPL benchmark [3]. In addition, for each submission the following responses are measured: the sites at which computers reside, computer name, year introduced, vendor, number of cores, R_{peak} for the peak performance, and power consumption in some cases. Another metric of interest, Linpack Efficiency, is determined by the ratio $\frac{R_{max}}{R_{peak}}$. In summary, the web site has provided a rich set of data to further analyze and categorize supercomputers.

2.1.2 Green500

The Green500 list [30] creators claim to provide "rankings of the most energy-efficient supercomputers in the world. [They] raise awareness about power consumption, promote alternative total cost of ownership performance metrics, and ensure that supercomputers only simulate climate change and not create it." Between its inception in 2007 and the June release of 2009, the Green500 list was a reordering of the Top500 list in order of decreasing power efficiency measured as the maximal Mflops per Watt, a metric proposed in [40]. Power consumption of a system is measured by a digital power meter plugged into the system's power strip, and readings are sent to a profiling computer once every 20 µs (at a rate of 50 kHz). Newer versions of the Green500 list such as the Little, Open, and HPCC iterations listing different subsets of supercomputers can be found at www.green500.org [30]. For the purposes of this study, we are concerned with the power efficiency of the supercomputers appearing on the Top500 list.

2.2 Evolution Analysis of Supercomputers

In Sections 2.2.1, cross referencing the power efficiency data with the Linpack efficiency data, we examine supercomputers in a two-dimensional scatter plot: Power efficiency versus Linpack efficiency. Naturally, we can derive rich information from this representation. More desirable systems are those with both highest power efficiency and highest Linpack efficiency. The other three possibilities are also obvious: low power efficiency and high Linpack efficiency, high power efficiency and low Linpack efficiency, and low efficiencies in both dimensions. Supercomputers appear in the scatter plot grouped according to the following factors: manufacturer, parallel architecture, network interconnect, and processor family.

In Sections 2.2.2—2.2.5, with such a partitioning of the Top 500, we first present the Linpack and power efficiencies of the Top 500 supercomputers over the last six iterations of the Top500 and Green500 lists. The set of supercomputers from each list is partitioned into subsets based on the characteristics listed in Section above. Along with coloring elements from different subsets, each subset is enclosed by its convex hull to facilitate comparison between subsets. Finally, the median values for Linpack efficiency and power efficiency are denoted by dash-dot lines on the respective axes.

In each case, we not only perform "static" analysis for these categories, but we also highlight the developmental trends of these categories.

The system averages for each list release are surrounded by a corresponding ellipse, which summarizes the spread of the subset of supercomputers. The principal radii of an ellipse represent the standard deviations of a subset along their respective axes.

Figure 2.2 summarizes the latest (June 2010) state of supercomputer efficiency, highlighting the best-performing ones.

Figure 2.3—Figure 2.10 present four pairs of plots. The first plot in each pair is the efficiency scatter plot of the Top 500 supercomputers, partitioned based on particular design characteristics; the second is the evolution of the subsets' centers of mass.

2.2.1 Scatter Plot of Efficiencies

Figure 2.2 presents the supercomputers appearing on the June 2010 Top500 list. The horizontal axis is Linpack efficiency, while the vertical axis is power efficiency (as listed on [30]); the center of each disc indicates efficiencies while the radius is proportional to the Linpack performance of the corresponding supercomputer as it appears on [3]. Each label is of the form "Computer Name: (i, j)," where i and j are the Top500 and Green500 ranks of the corresponding supercomputer, respectively. Labeled supercomputers are those with ranking pairs in the set $\{(i, j) | i \leq 10 \cup j \leq 10\}$. These systems are listed in Table 1; in other words, computers listed here must be in the top 10 of the Top500 list, or in the top 10 of the Green500 list, or both. Each disc's color corresponds to one of the network types listed in the legend. The viewable area in the figure is a box with lower left and upper right corners corresponding to the minimal and maximal Linpack and power efficiencies, respectively. Rays from the bottom-left corner of this box divide the plot at $30°$ and $60°$.

Concentric arcs highlight several features of the data. Centered at the minimal efficiencies' values, these arcs indicate the distribution of the Top500 supercomputers' efficiencies by displaying selected percentiles of the order statistics $\{d_1, d_2, \dots, d_i, \dots, d_{500}\}$, where d_i represents the square of the "distance" of system i from the minimal efficiency value, i.e.

$$d_i = \left(\left(\frac{L_i - L_{,min}}{r_L}\right)^2 + \left(\frac{P_i - P_{,min}}{r_P}\right)^2\right),$$

in which r_L and r_P are the respective ranges of the Top500 supercomputers' Linpack and power efficiencies, respectively. Of the first 200 systems ordered by d_i, (below the 40th percentile), 198 are clusters from either IBM or Hewlett-Packard. The popularity of these clusters causes a significant amount of disc-overlap in Figure 2.2 with identical systems having identical efficiency values; whereas the greater diversity and therefore less disc-overlap of more efficient supercomputers makes the number of these machines appear greater in comparison.

Considering the bottom half of the Top500 with respect to d_i, all but nine of the 250 systems employ Gigabit Ethernet; this means that a single system in the top half, a Dell PowerEdge Cluster, uses Gigabit Ethernet. This feature appears clearly in Figure 2.2, as the 40% arc nearly partitions the Top500 into Gigabit Ethernet systems and non-Gigabit Ethernet systems.

Interesting things to note include China's performance share increase between 2008 and 2010, from 3% to 13%. Additionally, the United States fell from 65% to 51% over the same time period.

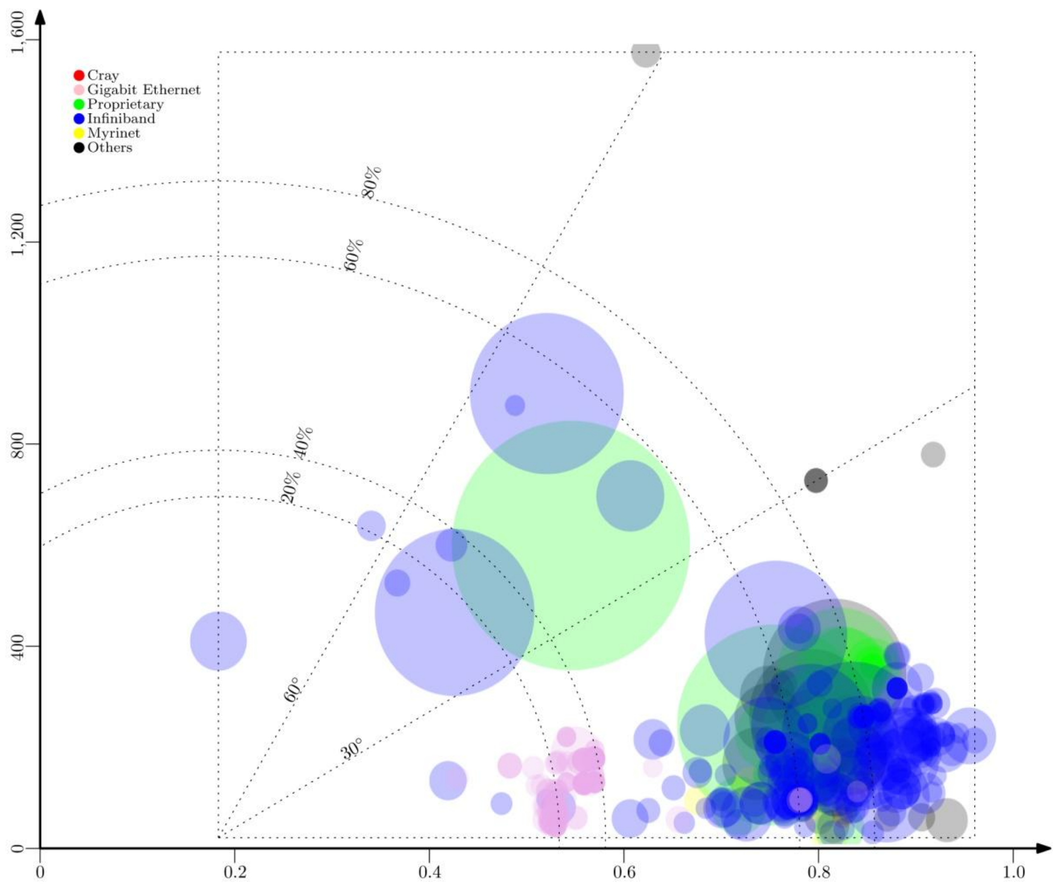


Figure 2.1. Scatter plot of the Top500 supercomputers as of November 2010.

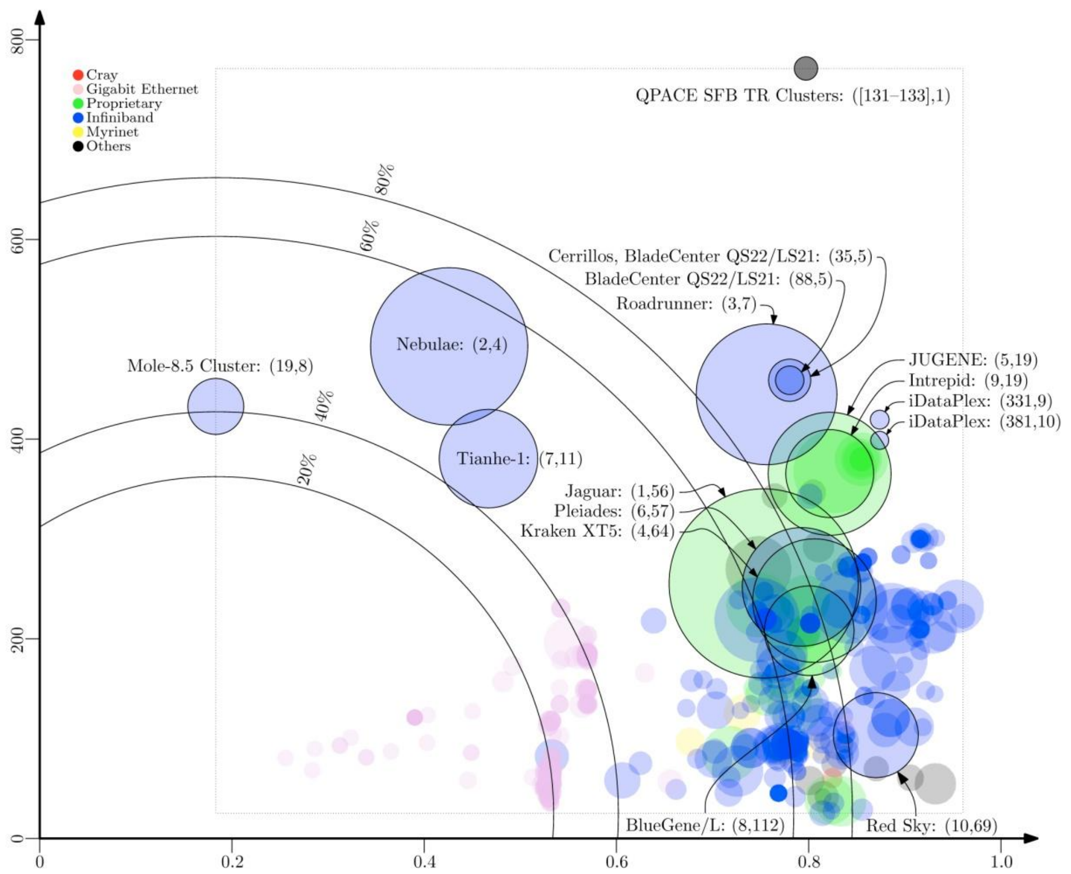


Figure 2.2. Scatter plot of the Top500 supercomputers as of June 2010.

Table 1. A summary of the top ten of Top500 and Green500 supercomputers as of June 2010.

System	Top Rank	Green Rank	Power Eff.	Linpack Eff.
Jaguar, Cray XT5-HE	1	56	0.755	253.07
Nebulae, Dawning TC3600 Blade	2	4	0.426	492.64
Roadrunner, BladeCenter QS22/LS21	3	7	0.757	444.25
Kraken XT5, Cray XT5-HE	4	64	0.808	235.77
JUGENE, Blue Gene/P Solution	5	19	0.823	363.98
Pleiades, SGI Altix ICE 8200EX/8400EX	6	57	0.794	249.58
Tianhe-1, NUDT TH-1 Cluster	7	11	0.467	379.24
BlueGene/L, eServer Blue Gene Solution	8	112	0.802	205.27
Intrepid, Blue Gene/P Solution	9	19	0.823	363.98
Red Sky, Sun Blade x6275	10	269	0.872	99.80
QPACE SFB TR Cluster (3x)	131	1	0.799	773.38
Cerrillos, BladeCenter QS22/LS21 Cluster	35	5	0.782	458.33
BladeCenter QS22/LS21 Cluster	88	5	0.782	458.33
Mole-8.5 Cluster	19	8	0.182	431.88
iDataPlex	331	9	0.876	418.47
iDataPlex	381	10	0.876	397.56

2.2.2 Network Families

Figure 2.3 and Figure 2.4 refer to list releases between November 2007 and June 2010. During this time, systems appearing with Ethernet networks used Gigabit Ethernet exclusively; therefore, when analyzing these lists, "Ethernet" refers to "Gigabit Ethernet."

In Figure 2.3, we present a scatter plot of the Top500 supercomputers grouped according to network types for six listings. The vertical axis is power efficiency ($Mflops/W$) over the range $[0,800]$; the horizontal axis is Linpack efficiency $\frac{R_{max}}{R_{peak}}$ over the range $[0,1]$. Each point represents a supercomputer; its color indicates the network type shown in the legend. The two dashed lines in each frame indicate the medians of the two efficiencies.

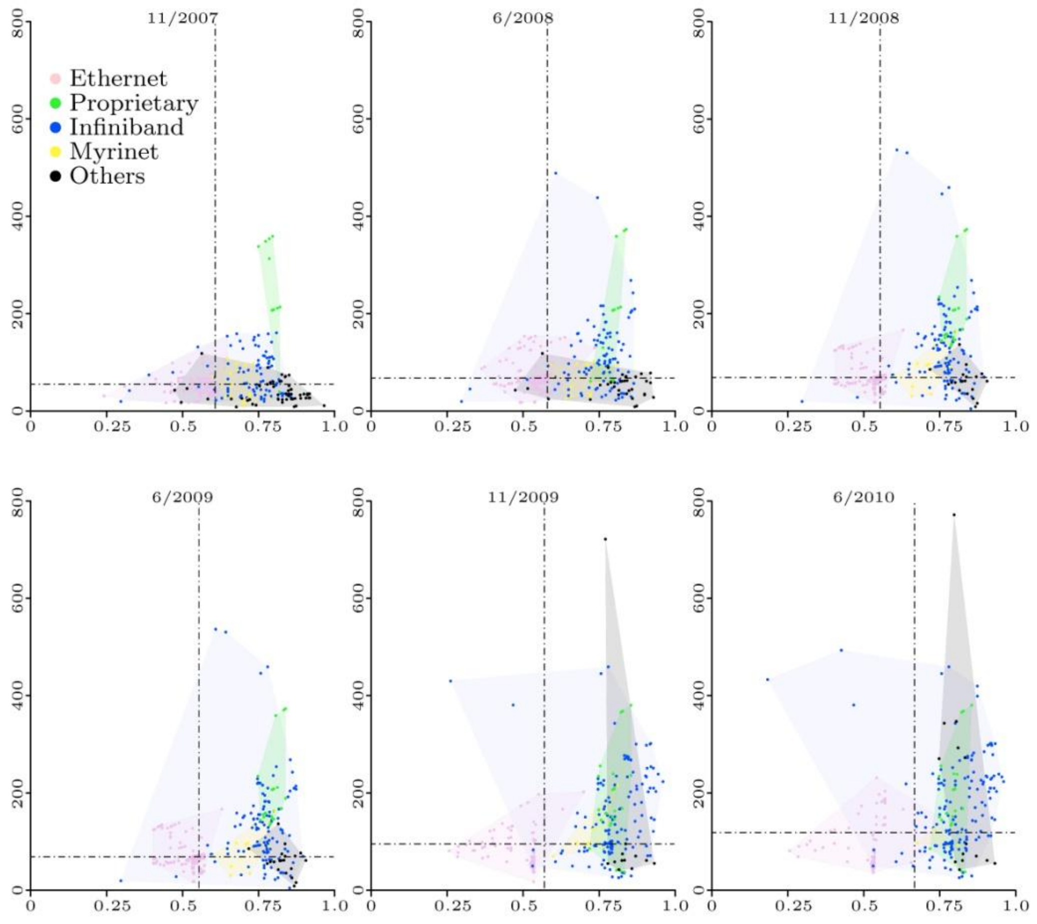


Figure 2.3. Scatter plot of the Top500 supercomputers grouped according to network type for six listings.

Platforms built with Ethernet networks reside, for the most part, in the lower left quadrant, with a maximum power efficiency of 150.45 $MFlops/W$ in November 2007 to 228.78 in June 2010. As of this date, no Ethernet supercomputers have achieved top 10% Linpack efficiency performance. The only Ethernet platforms exceeding the 90th percentile in power efficiency are two BladeCenter HS21 Clusters using Xeon quad core processors at 2.33 GHz in November of 2007. As of November 2010, there is only one system in the top 50 using an Ethernet interconnect.

In November 2007, no supercomputer was in the top 10% for both power and Linpack efficiencies. Since the release of the Green500 list, the only networks of supercomputers achieving top 10% performance in both dimensions were proprietary networks (those of BlueGene/P), Infiniband networks, and two Atipa Candor Clusters using a Myrinet 10G-MX network, the only Myrinet supercomputers to exceed the 90th percentile in any category.

As of June 2010, systems with proprietary networks such as BlueGene/P and several Cray XT systems, as well as Infiniband networks are the leaders with respect to power efficiency.

From the November 2007 release of the Top500 to the June 2010 release, supercomputers within the Myrinet network family gradually leave the list. Indeed, among the four types of networks, proprietary ones perform the best, followed by Infiniband, and then by Ethernet. The costs precisely reverse the performance order.

Figure 2.4 summarizes the efficiency evolution for various network types. The vertical axis is power efficiency ($Mflops/W$) over the range $[0,800]$; the horizontal axis is Linpack efficiency $\frac{R_{max}}{R_{peak}}$ over the range $[0,1]$. Each ellipse summarizes the efficiencies for all supercomputers with the given architecture type during a particular six month period. The center of each ellipse indicates average efficiencies while the radii of an ellipse show the standard deviations of the efficiencies.

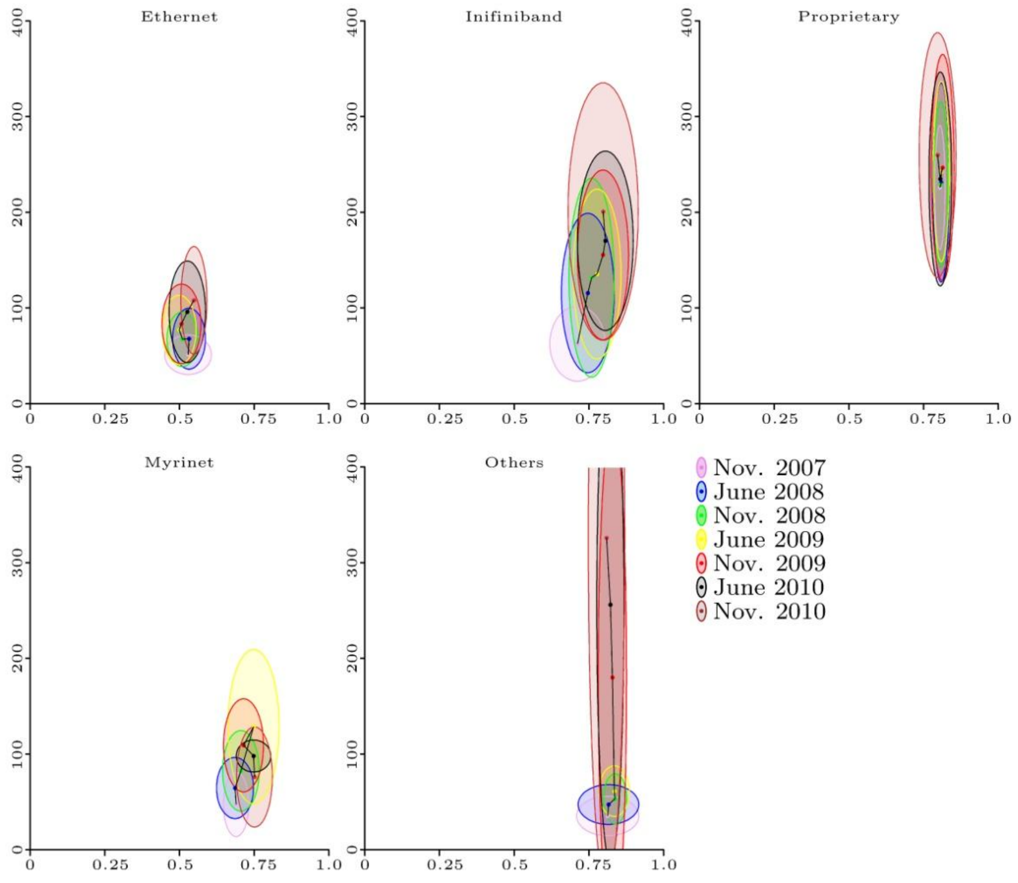


Figure 2.4. Efficiency evolution for various network types.

2.2.3 Processor Families

The major processor families represented on the Top500 list are AMD's x86 64, Intel's EM64T and IA-64, and Power; systems built with these, as well as those using other processor families, are plotted in Figure 2.5. The vertical axis is power efficiency ($Mflops/W$) over the range $[0,800]$; the horizontal axis is Linpack efficiency $\frac{R_{max}}{R_{peak}}$ over the range $[0,1]$. Each point represents a supercomputer; its color indicates the network type shown in the legend. The two dashed lines in each frame indicate the medians of the two efficiencies.

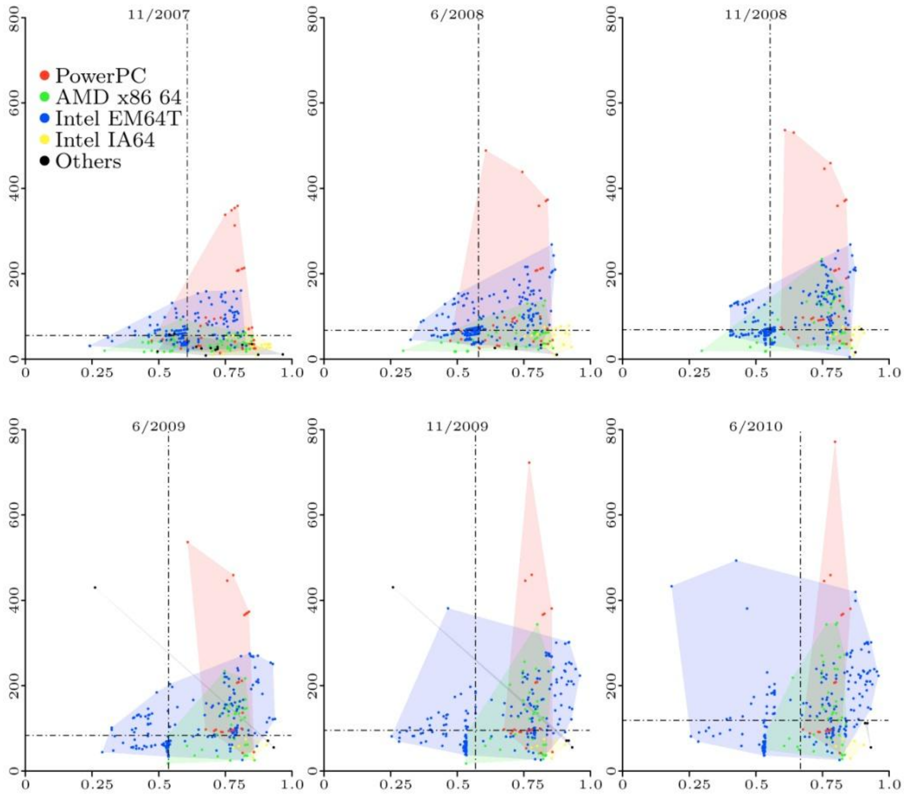


Figure 2.5. Timeline of scatter plots of the Top500 supercomputers grouped according to processor family, for six listings.

Figure 2.6 shows the evolution of the efficiencies for the various processor families. The “Others” frame shows varied distributions over the dates considered. In addition to the relatively small size of the set, elements of “Others” occupied two extremes. In June 2009, only three systems used processors not from the four major families. The GRAPE-DR Cluster also had the minimal Linpack efficiency value (0.26) for that date, yet ranked fifth overall on the Green500 List in power efficiency (428.91 $Mflops/W$). The remaining two systems, Earth Simulator—an NEC Vector system, and a Fujitsu Cluster, ranked 22 and 28 on the June 2009 Top500 list, having Linpack efficiencies over 0.91; however, they performed at 51.00 and 66.85 $Mflops/W$, respectively, on the Green500 list. In the latest (June 2010) release, the GRAPE cluster no longer appears, leaving the subset “Others” in the upper-half of the Linpack efficiency and the lower-half of the power efficiency distributions.

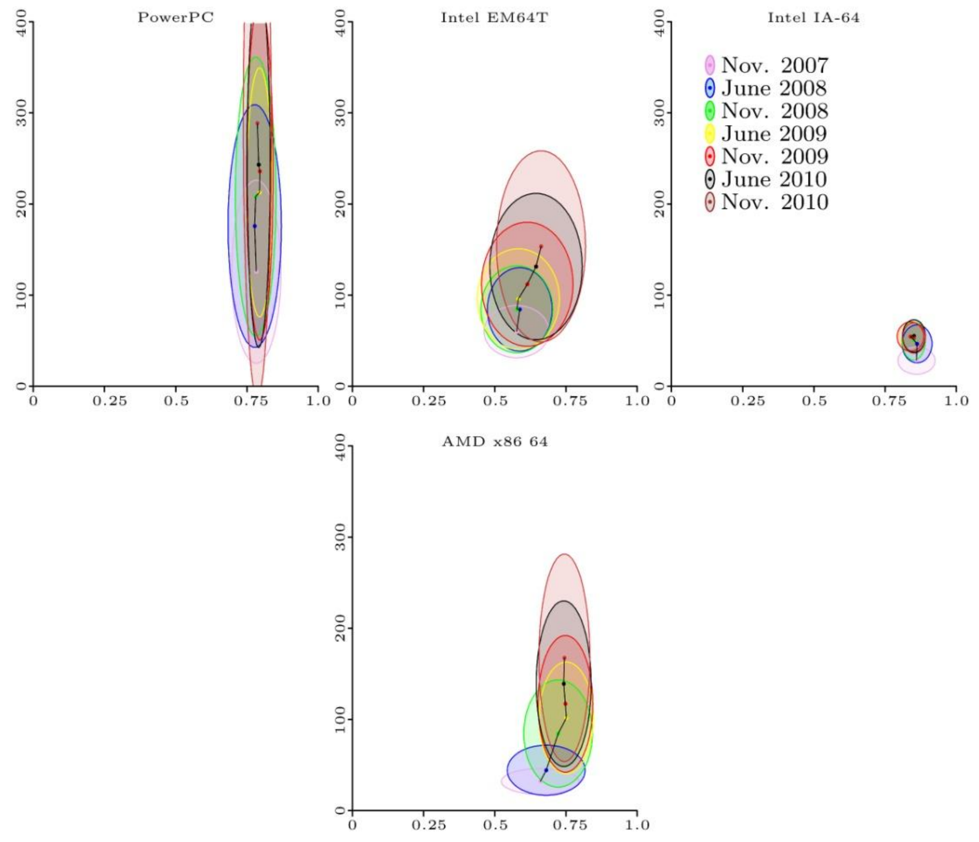


Figure 2.6. Efficiency evolution for various processor family types

Of the 42 entries using AMD x86 processors, nearly half are Cray XT MPP supercomputers, one of which is the number one Jaguar. Thirteen others are in the top 100. The 23 non-Cray supercomputers using AMD x86 chips are all clusters with a similar, though less skewed, Linpack performance distribution. The power efficiency of the AMD-based supercomputers shows consistent increase over the last five releases.

Among the largest competitors (PowerPC, Intel EM64T and IA-64, AMD x86 64), Intel's EM64T is dominant with nearly 80% system share in November 2009.

2.2.4 Architectures

In Figure 2.7, we present a scatter plot of the Top500 supercomputers grouped according to architecture type for six listings. The vertical axis is power efficiency ($Mflops/W$) over the range $[0,800]$; the horizontal axis is Linpack efficiency $\frac{R_{max}}{R_{peak}}$ over the range $[0,1]$. Each point represents a supercomputer; its color indicates the network type shown in the legend. The two dashed lines in each frame indicate the medians of the two efficiencies.

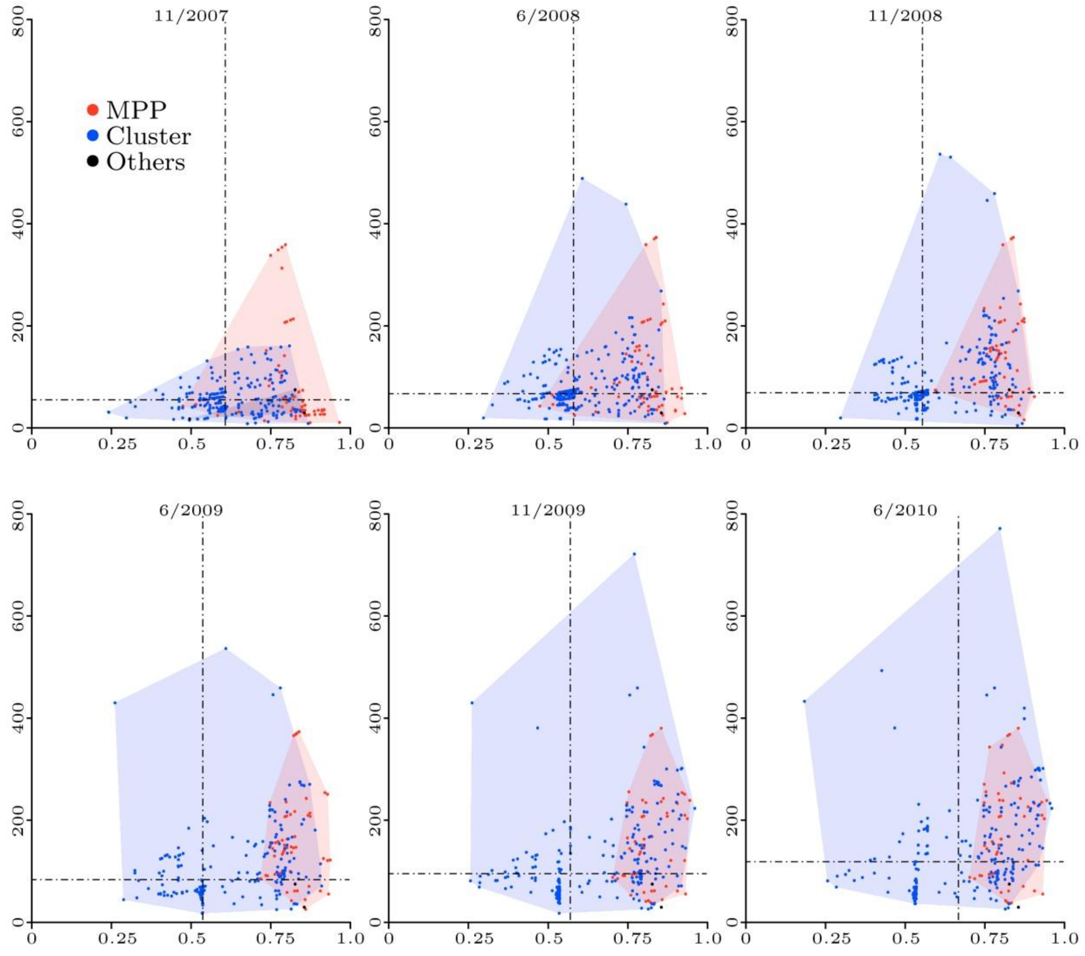


Figure 2.7. Timeline of scatter plots of Top500 partitioned by architecture.

Figure 2.7 highlights several features. The hull of the Cluster set is expanding in power efficiency. Although clusters appear to have mid-range Linpack efficiency, new cluster entrants into the Top500 list are now the most power efficient systems on the list.

On the other hand, within the set of clusters, the subset of HP 3000BL Clusters displays the following evolutionary feature. If we examine the system-center of HP Clusters, as time passes, supercomputers that are no longer powerful enough to make the Top500 disappear; however, these supercomputers are the more efficient ones. While it is reasonable to expect a certain overhead associated with supercomputer scale, the set of MPP systems does not display such characteristics. Moreover, supercomputers like BlueGene/L and P show near-constant efficiencies, regardless of scale, a clear indication of strong scaling.

Figure 2.8 summarizes the efficiency evolution for various parallel architecture types. The vertical axis is power efficiency ($Mflops/W$) over the range $[0,400]$; the horizontal axis is Linpack efficiency $\frac{R_{max}}{R_{peak}}$ over the range $[0,1]$. Each ellipse summarizes the efficiencies for all supercomputers with the given architecture type during a particular six month period. The center of each ellipse indicates average efficiencies while the radii of an ellipse show the standard deviations of the efficiencies.

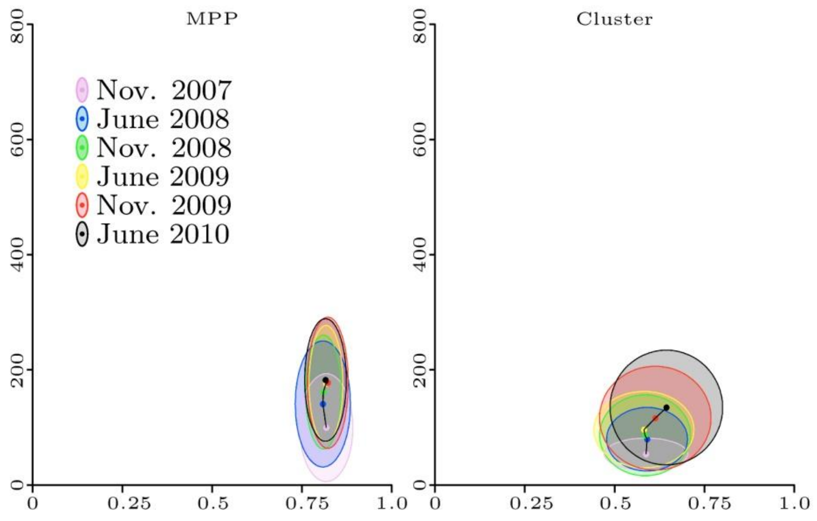


Figure 2.8. Efficiency evolution for various parallel architecture types.

2.2.5 Vendors

In Figure 2.9, we present a scatter plot of the Top500 supercomputers grouped according to vendor type for six listings. The vertical axis is power efficiency ($Mflops/W$) over the range $[0,800]$; the horizontal axis is Linpack efficiency $\frac{R_{max}}{R_{peak}}$ over the range $[0,1]$. Each point represents a supercomputer; its color indicates the network type shown in the legend. The two dashed lines in each frame indicate the medians of the two efficiencies.

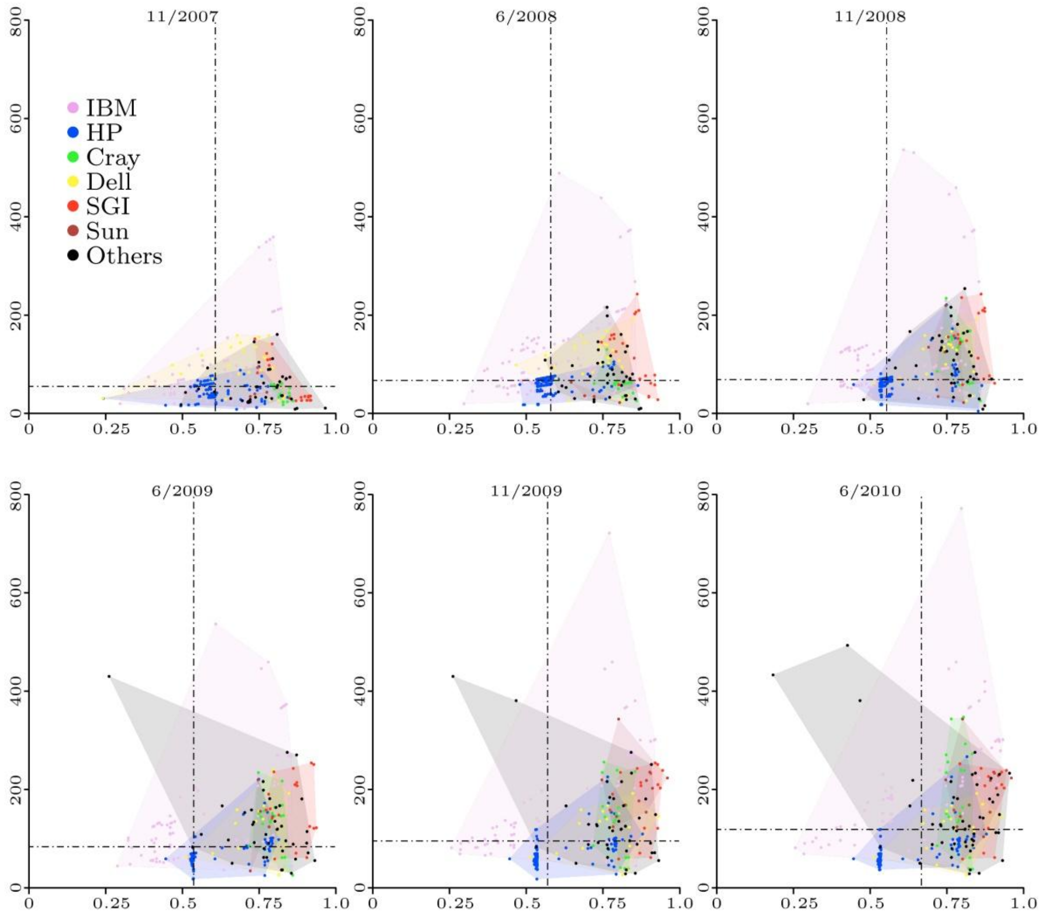


Figure 2.9. Timeline of scatter plots of the Top500 supercomputers grouped according to manufacturer, for six listings.

Figure 2.10 summarizes the efficiency evolution for various vendors. The vertical axis is power efficiency $(Mflops/W)$ over the range $[0,400]$; the horizontal axis is Linpack efficiency $\frac{R_{max}}{R_{peak}}$ over the range $[0,1]$. Each ellipse summarizes the efficiencies for all supercomputers with the given architecture type during a particular six month period. The center of each ellipse indicates average efficiencies while the radii of an ellipse show the standard deviations of the efficiencies.

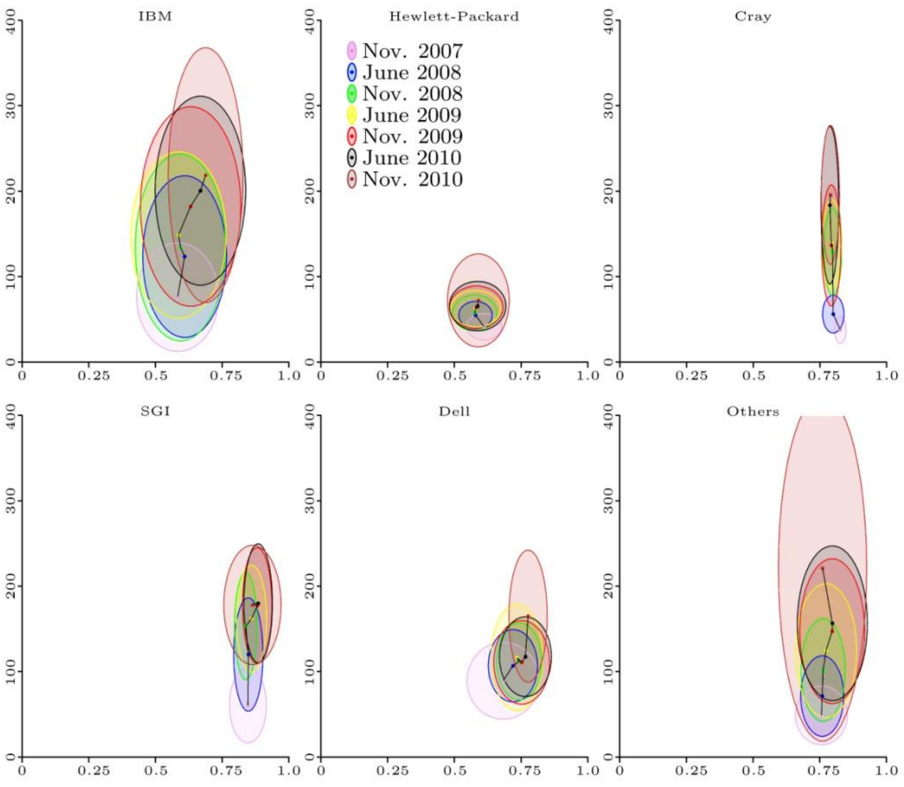


Figure 2.10. Evolution of efficiencies for Vendors.

In the June 2010 release of the Top500 list, IBM appears with 196 entries. Figure 2.10 shows consistent improvements in efficiencies for IBM systems over the last to list releases. Although Cray shows little change in system averages of power and Linpack efficiencies between November 2008 and November 2009, it shows the greatest increase of any vendor in power efficiency with the latest release. On the other hand, Cray has shown decreases in Linpack efficiency for each release since the November 2007.

Vendors that are not a part of the six most frequently appearing are summarized in the lower-right panel entitled “Others.” Companies such as Fujitsu, Hitachi, NEC, as well as self-made platforms are part of the 41 entries represented here. Of these entries, 34 use Infiniband Networks. This group shows a slight and increasing improvement in Linpack efficiency, as well as a consistent improvement in power efficiency.

2.2.6 Further Analysis for IBM Systems

Since June 2004, IBM supercomputers have consistently made up about 40% of the Top 500 systems, appearing with a variety of architectures, system models, processors, and networks. Figure 2.11 summarizes the evolution of IBM's supercomputers making the Top500 list. Each band represents a subset of systems sharing characteristics. The height of each band represents aggregated performance share, $\frac{R_{max_j}}{\sum_i R_{max_i}}$. The width of each band shows the length of time a particular type of system appeared on the Top500 list. Supercomputers are categorized in the following manner. Systems are first partitioned by architecture (SMP, MPP, Cluster), then by system type (BlueGene, BladeCenter, etc.), then, in some cases by processor type (POWER3…6, Pentium, Itanium). Finally, systems are divided by network type, as shown in the legend; here, we differentiate between systems using varieties of Ethernet networks.

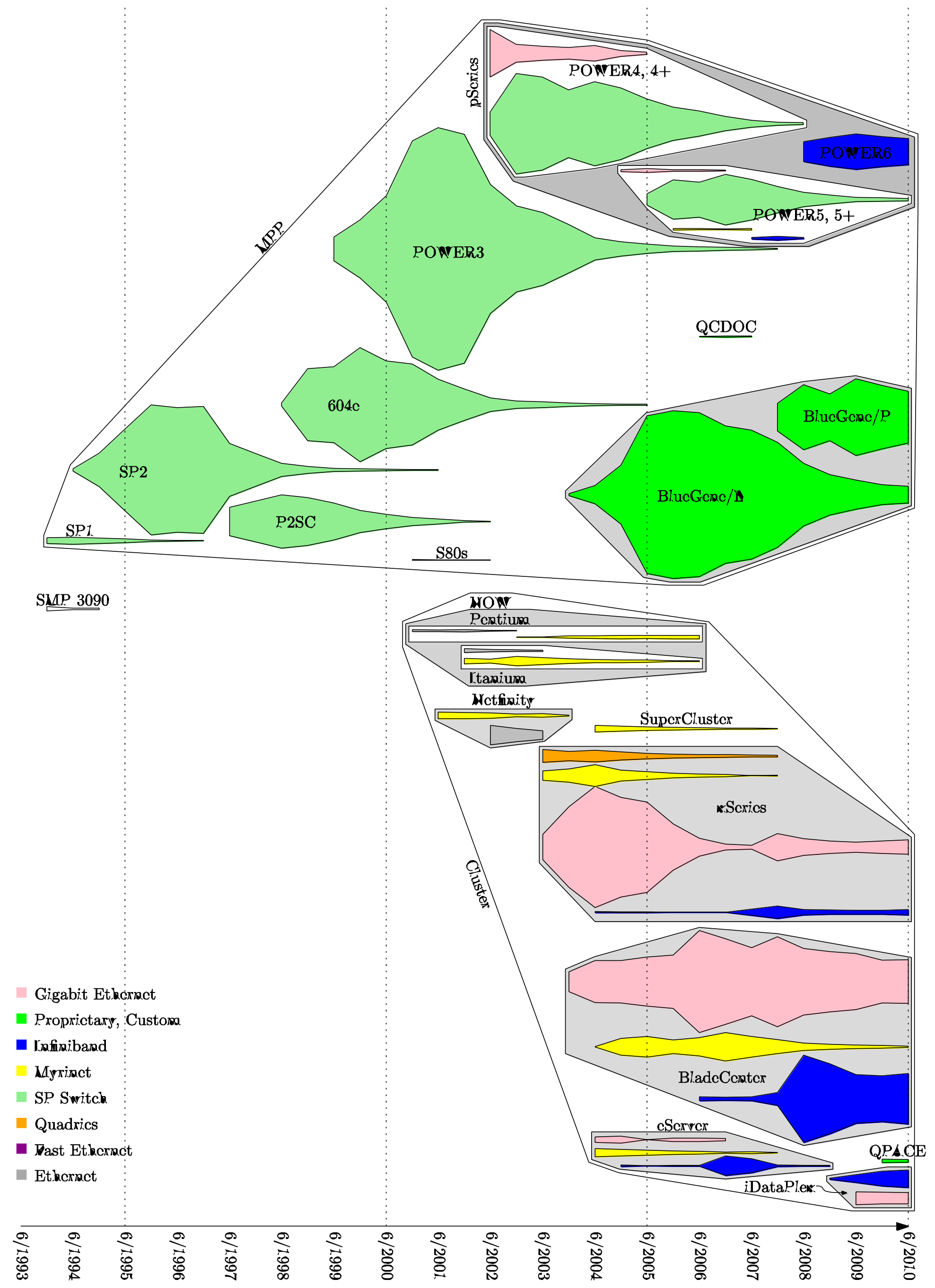


Figure 2.11. IBM Phylogeny.

In Figure 2.12 and Figure 2.13, we analyze specifically the evolution of IBM systems.

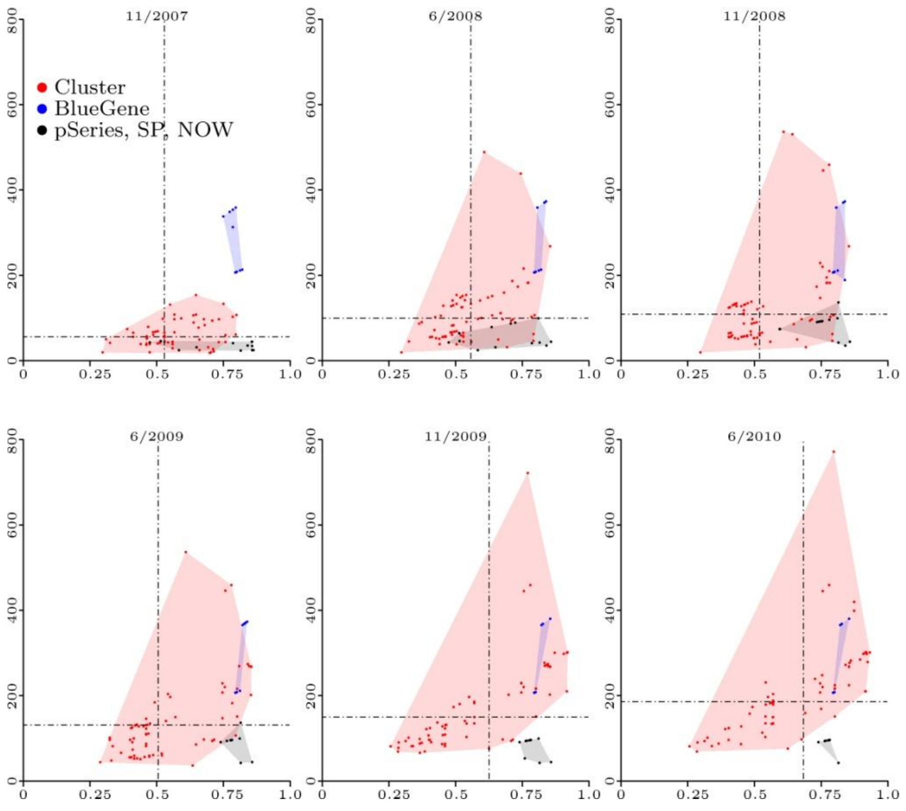


Figure 2.12. Timeline of scatter plots for IBM system types.

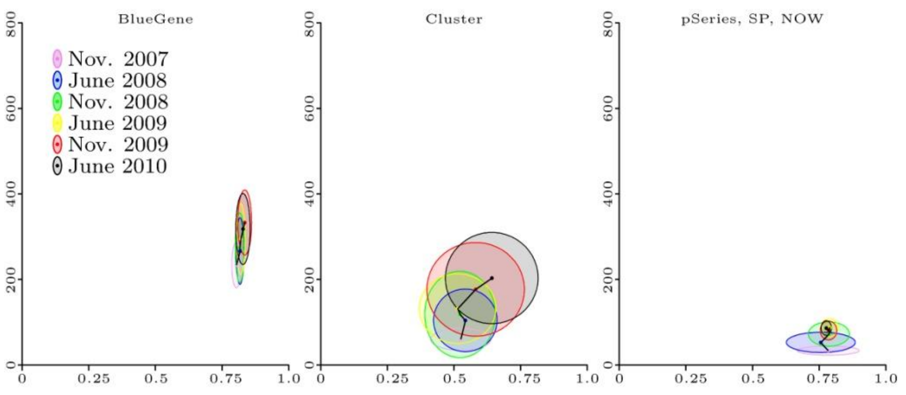


Figure 2.13. Efficiency evolution for IBM system types.

2.2.7 Further Analysis for Cray Systems

Figure 2.14 shows the introduction of the Top500 list coinciding with a decline in appearances by Cray SMP platforms and a gradual transition to MPP. Few Cray clusters appear on the list; namely, one Opteron Cluster with Myrinet interconnection and several XD1 platforms using RapidArray interconnection.

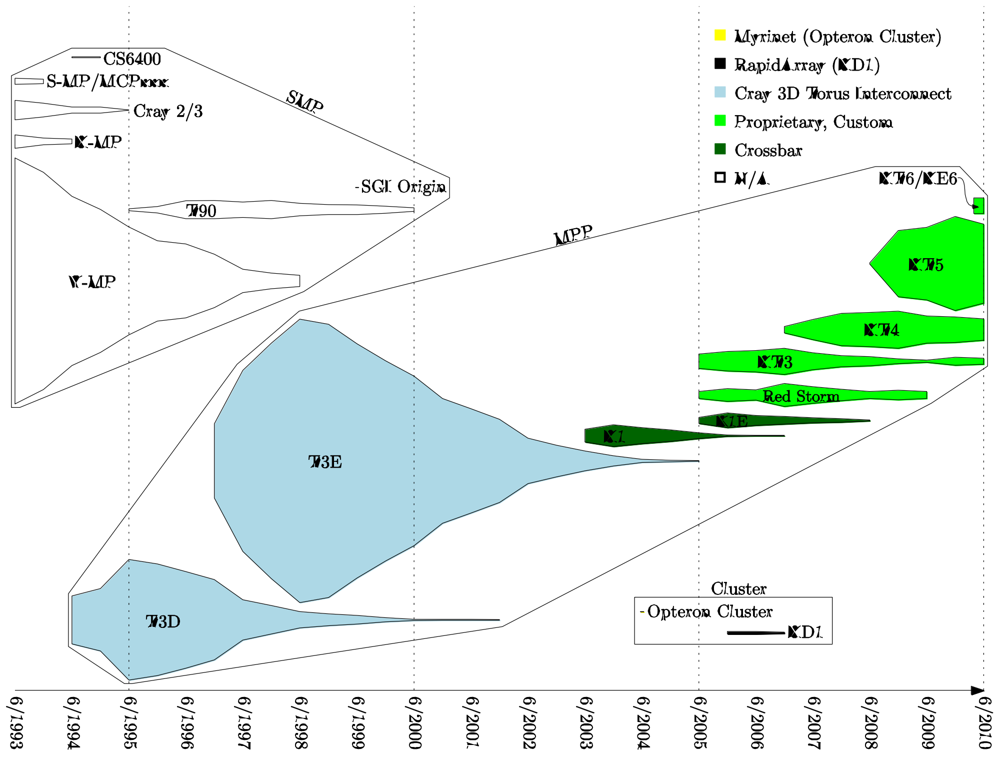


Figure 2.14. Cray Phylogeny.

2.2.8 Further Analysis for Hewlett-Packard Systems

HP Supercomputer 3000BL: a Case Study of Ethernet vs. Infiniband

When examining only Hewlett-Packard's 3000BL Cluster family, we are afforded an interesting comparison of Ethernet and Infiniband networks. In Figure 2.15, we plot the evolution of all HP Cluster Platform 3000BL supercomputers. Each of the six dated frames is a scatter plot of the platforms' efficiencies for that particular list release. Within each of these frames, the red and blue sets represent platforms built with Ethernet and Infiniband, respectively. In the bottom right panel, we summarize the evolution of these network types within the HP clusters. The left and right progressions correspond to Ethernet and Infiniband, respectively.

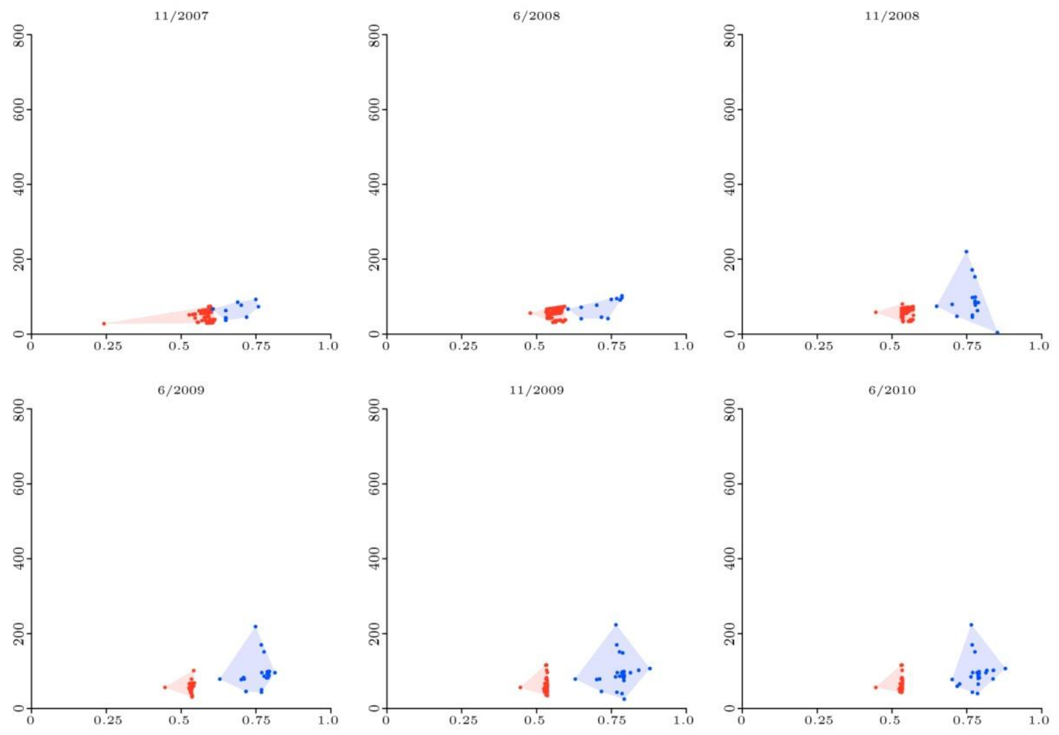


Figure 2.15. Hewlett-Packard Cluster Platform 3000BL, with two network types.

Except for one cluster appearing in November 2007, each Infiniband cluster has a higher Linpack efficiency than each Ethernet cluster.

Figure 2.16 summarizes the evolution of the HP 3000 BL cluster supercomputer-types with the Ethernet progression on the left and Infiniband on the right. Additionally, each colored point represents the system average of a particular set of the corresponding date and is surrounded by an ellipse with radii equal to the standard deviations of that set in the respective dimension.

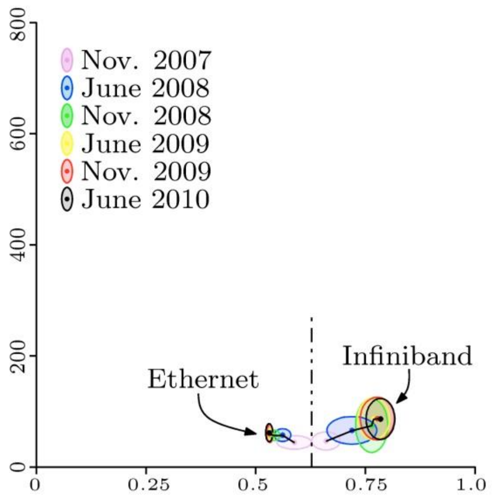


Figure 2.16. Evolution of efficiencies for Hewlett-Packard Cluster Platforms 3000BL.

2.2.9 Further Analysis for Japanese Systems

Figure 2.17 shows the taxonomic time line of Japanese vendors Fujitsu, Hitachi, and NEC. Several of these companies have collaborated with other HPC and traditionally non-HPC companies to build platforms appearing on the Top500; Figure 2.17 represents the individual efforts of these vendors.

Fujitsu systems appear on the first Top500 list with single processor computers. By list two, a transition to MPP begins. Between November 1993 and November 2001, Fujitsu's performance share is dominated by their VPP family of systems, until another transition to SMP, constellation, and cluster architectures occurs.

Hitachi platforms begin with a trajectory similar to that of Fujitsu, starting in November 1993 with single processor vector systems and making a transition to MPP systems with crossbar interconnect.

NEC follows this progression as well with the notable exception of several platforms with constellation architecture. Additionally, the then immediate impact of NEC's Earth simulator is highlighted in Figure 2.17.

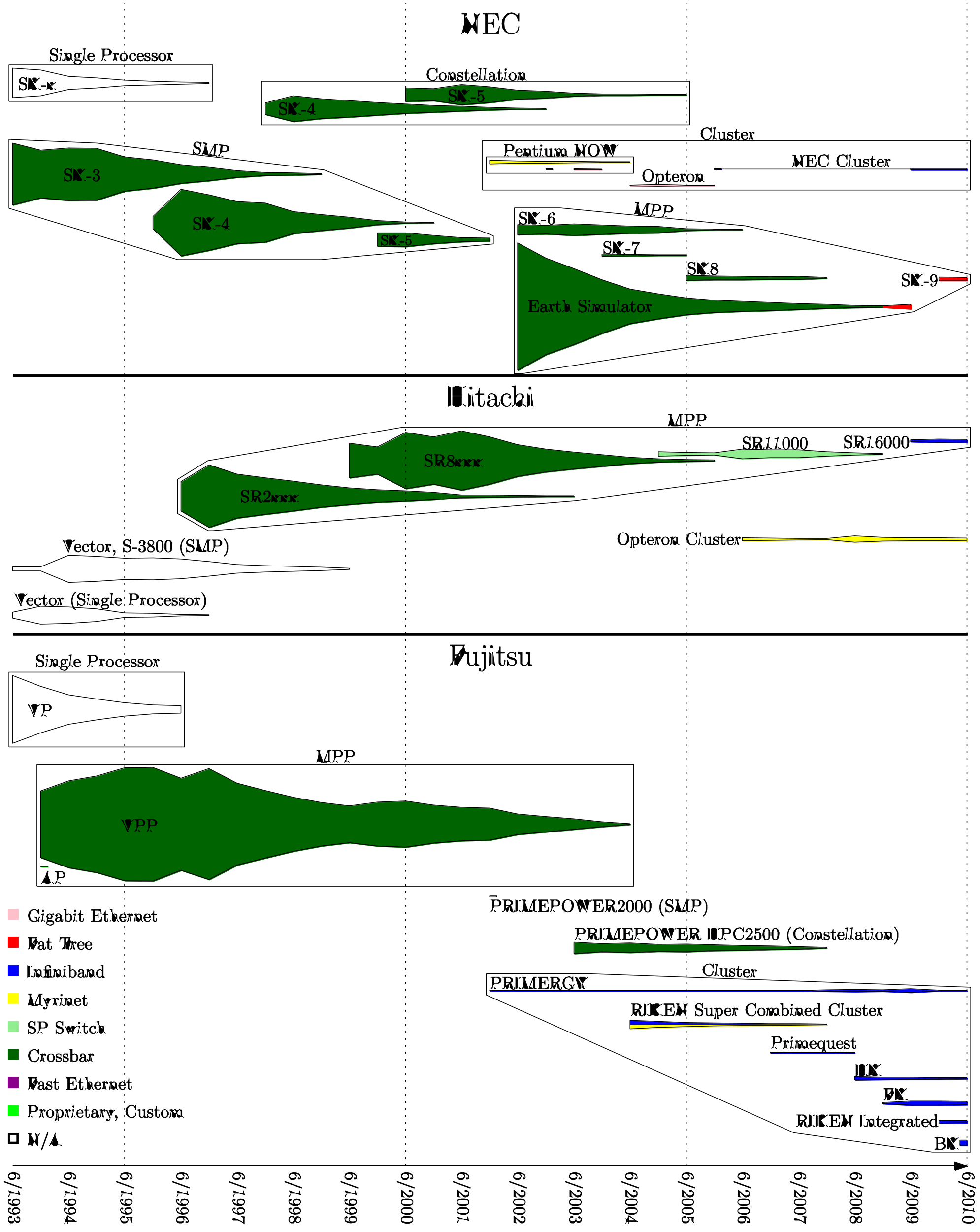


Figure 2.17. Japanese Vendor Phylogeny.

2.3 Summary Statements

We have presented and analyzed the correlations of the Linpack and power efficiencies of the past six listings of 500 supercomputers each. We group such supercomputers according to their architectures and original design manufacturers and analyze conveniently the time evolutions. Our analysis has for the first time revealed, in many cases reaffirmed, the comparative properties of such supercomputers. First, the Ethernet connected systems are low in both power (80 $Mflops/W$) and Linpack (52%) efficiencies and they have made negligible improvements over the six-listing period while the proprietary networked systems are the best in both efficiencies (220 $Mflops/W$ and 83%, respectively.) Myrinet and Infiniband are the middle-level performers. Second, Cray and IBM are leaders of power and Linpack efficiencies and both made significant progress in power efficiency over the six-listing period. HP performs the poorest with low power efficiency of below 80 $Mflops/W$ and low Linpack efficiency (70%). Third, PowerPC processors perform the best with high efficiencies in both while Intel IA-64 has the lowest power efficiency (and decent Linpack efficiency). In summary, a system with proprietary network, PowerPC, designed by IBM is the most optimal. A representative ideal system is the IBM BlueGene family.

Chapter 3: Development of a Novel Network

Advanced networking architectures [3, 5, 7-8, 10] have enabled supercomputers such as RoadRunner [41] to break the petaflop barrier. Such progress has stimulated the parallel computing community to invent more scalable interconnection networks to accommodate the ever-increasing demands on performance and functionality by incorporating millions of powerful processor cores. A scalable interconnection network, of fixed node degree, must satisfy most of the performance requirements: small diameter, large bisection width, topological simplicity, symmetry, design modularity, engineering feasibility, as well as expandability. For example, a 3D torus network (such as that in IBM's Blue Gene and Cray's T3E [5, 7-8, 10]) with up to 20,000 nodes and several smaller-scale hypercube network supercomputers [3, 15-16, 42], satisfy several of these requirements. However, network diameters grow according to $O\left(\sqrt{n}\right)$ for a torus, and $O(\log n)$ for a hypercube, where n is the network size. Growth rates for many derivatives of these networks [15-16, 24, 42-44] are similarly rapid. This defect of rapidly growing diameters greatly limits the expandability of these networks. Mesh networks of fixed dimension provide an alternative, having relatively low node degree and low engineering complexity; however they possess the disadvantages of large network diameter and small overall bandwidth. Other efforts to increase bandwidth without increasing network diameters include that of the hybrid fat-tree [25], a low-cost, low-degree network with irregular node degree. This network is susceptible to disconnection through faulty links and message contention toward roots. Other proposals have also been introduced, such as the incomplete torus and its derivatives [45] that reduce node degree at the expense of losing symmetry and topological simplicity. Honeycomb mesh and torus networks [46] received considerable early attention that faded quickly due to implementation obstacles, among other difficulties. Hexagonal networks introduced in [12] also boast a small diameter but carry a burden of a high node degree. Modifications of the traditional torus including the PEC [20], SRT [19], TESH [47], and RDT [22] networks all build upon the simplicity of mesh and torus networks, achieving improved network properties with unfavorable expandability and network cost. However, these variants demonstrated that interlacing rings of various lengths to a torus network is a profitable practice for improving network performance without adding significant engineering complexity.

Motivated by this, we propose the iBT network. The iBT network is constructed by interlacing bypass rings evenly into a torus network. We preserve the simplicity of a grid-like layout and improve the performance of the network with two bypass links per node. Our model allows for generalization of the bypass construction of the base torus to arbitrary dimensions for much larger and scalable networks, rather than being exclusively two-dimensional as in [19-20, 22]. This new network achieves a low network diameter, high bisection width, short node-to-node distances, and low engineering complexity in terms of network cost. Furthermore, the iBT network has a significantly lower node degree and network cost than those of a hypercube network with a similar number of nodes. To ensure network symmetry and modularity, we interlace rings into the torus network according to a consistent pattern. To analyze the topological properties for achieving an optimal network, we present the node-to-node hop distance distributions.

The chapter is organized as follows: We first define the iBT interconnection model and its generation scheme in Section 3.1.1. In Sections 3.1.2 and 3.1.3, we describe the evaluation criteria for networks and the performance comparisons with torus and hypercube networks, respectively. A formula for a node-to-node shortest distance is discussed in Section 3.2 and comparisons with other popular networks are given in Section 3.2.2.

3.1 iBT Interconnection Network

The iBT network, generated by interlacing bypass rings into a torus network, is an n-dimensional composite network that contains an n-dimensional torus network and additional bypass rings. Here, we provide the formal definition of the iBT network, present several topological properties of the network, enumerate the evaluation criteria used to compare networks.

Furthermore, we demonstrate the construction of a 3D iBT from a base 3D torus and search for an optimal 3D iBT configuration. As an example, we show the detailed procedure to generate an optimal iBT network with approximately 32,400 nodes and compare it to a 4D torus network of 32,768 nodes with an identical node degree of eight and to a hypercube with $2^{15} = 32{,}768$ nodes.

3.1.1 Definition

An $iBT(N_1 \times \cdots \times N_n; L = m; l = \langle l_1, \dots, l_k \rangle)$ network outgrows from an n-dimensional torus network $T(N_1 \times \cdots \times N_n)$ by interlacing l_i-hop bypass rings ($i = 1, \dots, k$) recursively into any m of the n dimensions ($m \leq n$). The m dimensions with bypass rings are referred to as the bypass dimensions and the remaining $n - m$ dimensions without bypass rings are referred to as plain dimensions. The terms $L = m; l = \langle l_1, \dots, l_k \rangle$ are referred to as a bypass scheme for generating the iBT network. This interconnection model results in a node degree of $2n + 2$, where $2n$ is from the base torus connections and the additional 2 from the bypass connections. To determine the two bypass connections for a node $p = (x_1, x_2, \dots, x_n)$, where $x_i \in \{0,1, \dots, N_i - 1\}$, $i = 1, \dots, n$, we introduce three terms: a node bypass dimension $d(p) \in \{1,2, \dots, m\}$ and a node bypass length $l(p) \in \{l_1, l_2, \dots, l_k\}$, which can be expressed as

$$d(p) = \left[\left(\sum_{i=1}^{m} x_i \right) \bmod m \right] + 1,$$

$$l(p) = l_h,$$

where

$$h = \left\lfloor \frac{(\sum_{i=1}^{m} x_i) \bmod mk}{m} \right\rfloor + 1 \in \{1, \dots, k\}.$$

Thus, a node's bypass species $s(p) = \langle d(p), l(p) \rangle$ indicates that two $l(p)$-hop bypass links have been added to the given node p in each direction along the dimension $d(p)$. For example, $iBT(32 \times 32 \times 16; L = 2; l = \langle 4,16 \rangle)$ indicates the interlacing of 4-hop and 16-hop bypass rings in the first two dimensions, i.e.,

xy-plane, of the 3D torus $T(32 \times 32 \times 16)$. A node $p = (1,1,4)$ has a bypass dimension $d(p) = 1$, a bypass length$l(p) = l_2 = 16$, and thus, its bypass species is $\langle 1,16 \rangle$, implying that p has two 16-hop bypass links in each direction along dimension x, i.e., the first dimension. Here, we summarize constraints that exist on the iBT network. Consider the network $iBT(N_1 \times N_2 \times \cdots \times N_n; L = m; l = \langle l_1, l_2, \ldots, l_k \rangle)$.

1. $N_i > 4, i \in \{1,2, \ldots, n\}$.
2. $2 \leq l_1 < l_2 < \cdots < l_k < \frac{1}{2}\min\{N_i\}$
   Constraints 1 and 2 prevent edges of different bypass rings from being redundant torus links or a single edge.
3. *Each node will exist on exactly one bypass ring.*
   These first three constraints imply that every node will have degree of $2n + 2$, where n is the number of dimensions of the network. $2n$ links appear from the n dimensions in the positive and negative directions along with two bypass links in each direction.
4. *Bypass rings are evenly interlaced*. In other words, the bypass sequence appears as defined above.
   a) This implies that *the length of the bypass sequence must be a multiple of mk*, the number of bypass dimensions and bypass lengths, respectively. Consider a node p of species $s(p) = \langle d(p), l(p) \rangle$ in a network with m bypass dimensions and k bypass lengths. To maintain even interlacing, every other species must appear exactly once in every direction before a different node with species $\langle d(p), l(p) \rangle$ appears again.
   b) Furthermore*, the length of the bypass sequence is mk*. In order to ensure nodes of differing species are not adjacent (connected by bypass links), bypass lengths must be a multiple of the bypass sequence length, mk.
   c) Equivalently, since the bypass sequence is fixed, *the nearest node in any direction with the same species will be mk hops away*.

d) Finally, *the size of the network in each bypass dimension must be a multiple of* mk.

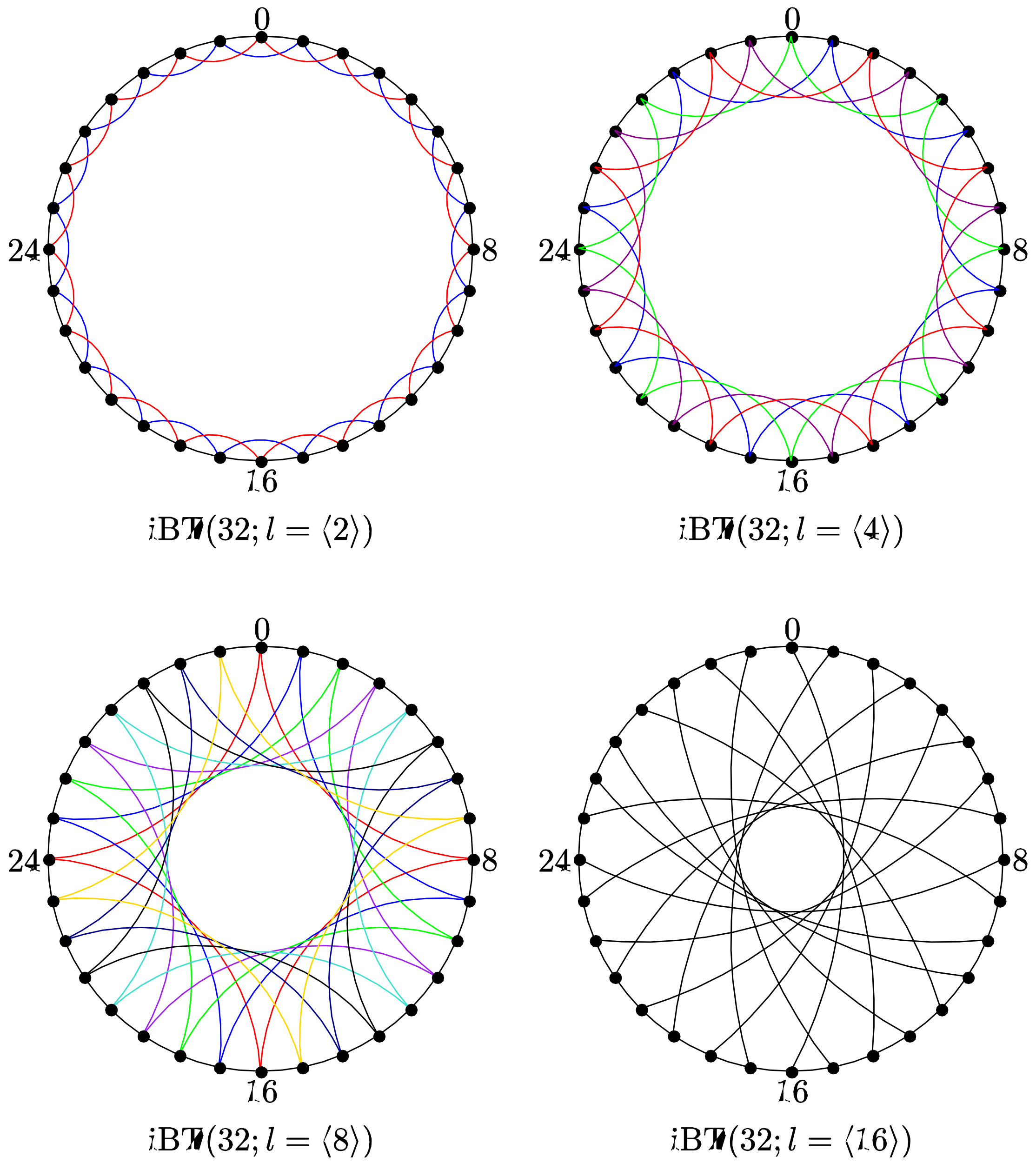

0
8
16
24
$iBT(32; l = \langle 2 \rangle)$
$iBT(32; l = \langle 4 \rangle)$
$iBT(32; l = \langle 8 \rangle)$
$iBT(32; l = \langle 16 \rangle)$

Figure 3.1. Bypass scheme for $iBT(32; L = 1; l = \langle l_1 \rangle)$.

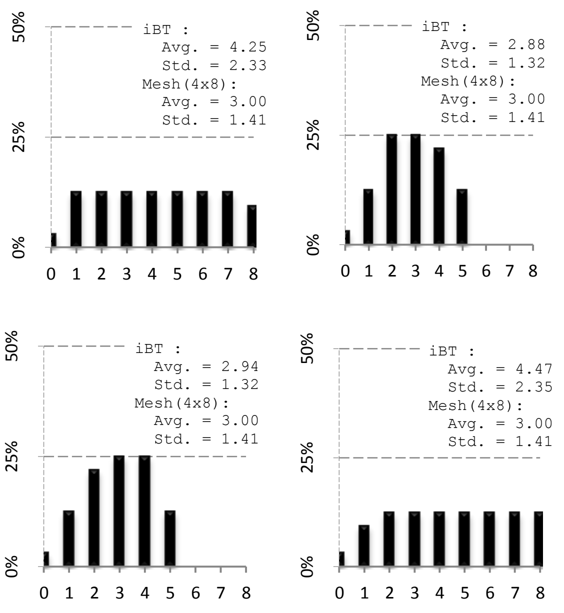
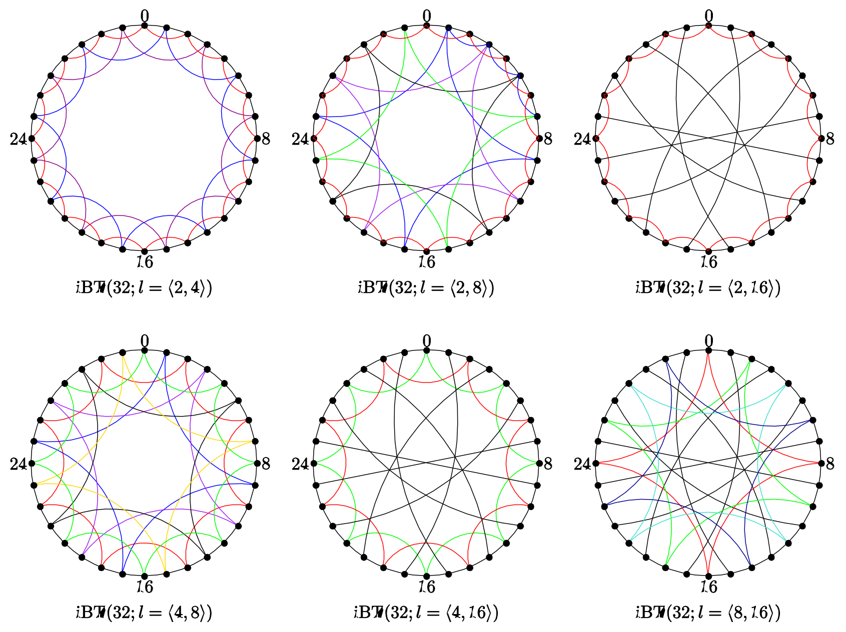


Figure 3.2. Internode distance distribution for $iBT(32; L = 1; l = \langle 2 \rangle)$, $iBT(32; L = 1; l = \langle 4 \rangle)$, $iBT(32; L = 1; l = \langle 8 \rangle)$, $iBT(32; L = 1; l = \langle 16 \rangle)$.



0
8
16
24
$iBT(32; l = \langle 2, 4 \rangle)$
$iBT(32; l = \langle 2, 8 \rangle)$
$iBT(32; l = \langle 2, 16 \rangle)$
$iBT(32; l = \langle 4, 8 \rangle)$
$iBT(32; l = \langle 4, 16 \rangle)$
$iBT(32; l = \langle 8, 16 \rangle)$

Figure 3.3. Bypass scheme for $iBT(32; L = 1; l = \langle l_1, l_2 \rangle)$.

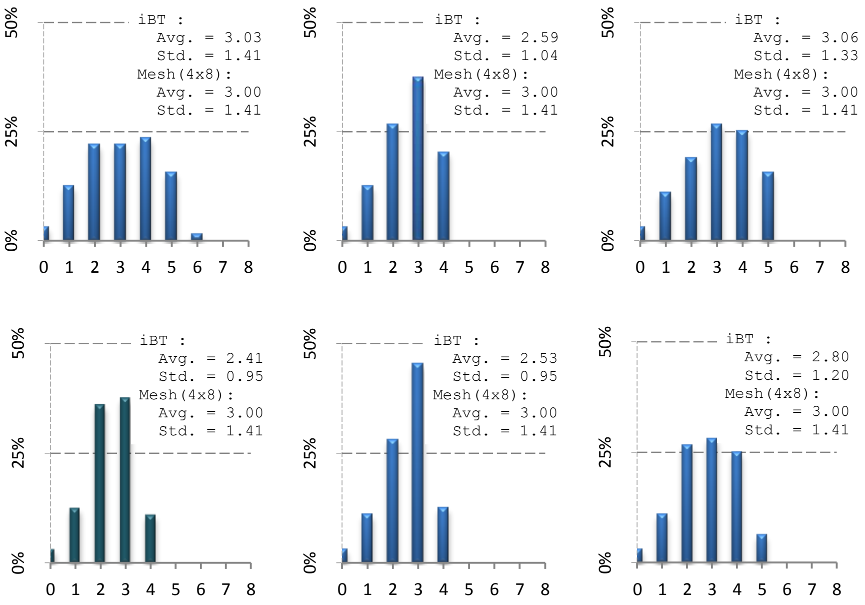


Figure 3.4. Internode distance distribution for $\boldsymbol{iBT}(\mathbf{32}; \boldsymbol{L} = \mathbf{1}; \boldsymbol{l} = \langle \boldsymbol{l_1}, \boldsymbol{l_2} \rangle)$, as labeled above.

In Figure 3.1—Figure 3.4, we show the hop distance distribution and bypass configurations for a family of 1D iBT networks written as $iBT(32; L = 1; l)$ generated from a 1D torus network with 32 nodes. A 1D torus network $T(32)$ is itself a ring. This 1D scheme is easy to follow and generalize for illustration at higher dimensions. The average node-to-node distance and standard deviation within a 32-node 2D torus network $T(4 \times 8)$ is $\mu = 3.00$ and $\sigma = 1.41$, respectively. As shown in Figure 3.1—Figure 3.4, the node-to-node distances of the resulting iBT networks are statistically smaller than those of $T(4 \times 8)$, for example, $\mu = 2.94$, $\sigma = 1.32$, for $iBT(32; L = 1; l = \langle 8 \rangle)$ and $\mu = 2.41$, $\sigma = 0.95$, for $iBT(32; L = 1; l = \langle 4,8 \rangle)$.

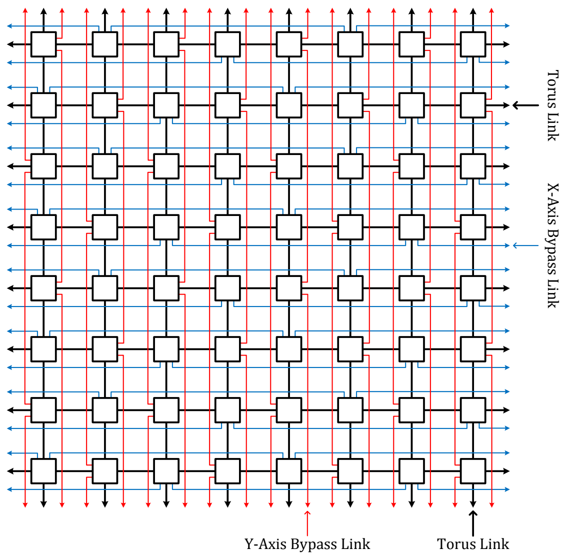


Figure 3.5. 2D $iBT(8 \times 8; L = 2; l = \langle 4 \rangle)$ with highlighted links.

In Figure 3.5, we draw all of the links for the 2D iBT network $iBT(8 \times 8, L = 2, l = \langle 4 \rangle)$. In Figure 3.6, we illustrate the bypass scheme for the 3D iBT network $iBT(30 \times 30 \times 36, L = 3; l = \langle 6,12 \rangle)$. In this figure, we only draw selected bypass links along the three easily visible faces to identify the nodes that are connected by the appropriate bypass links. Other links are omitted for the purpose of clarity.

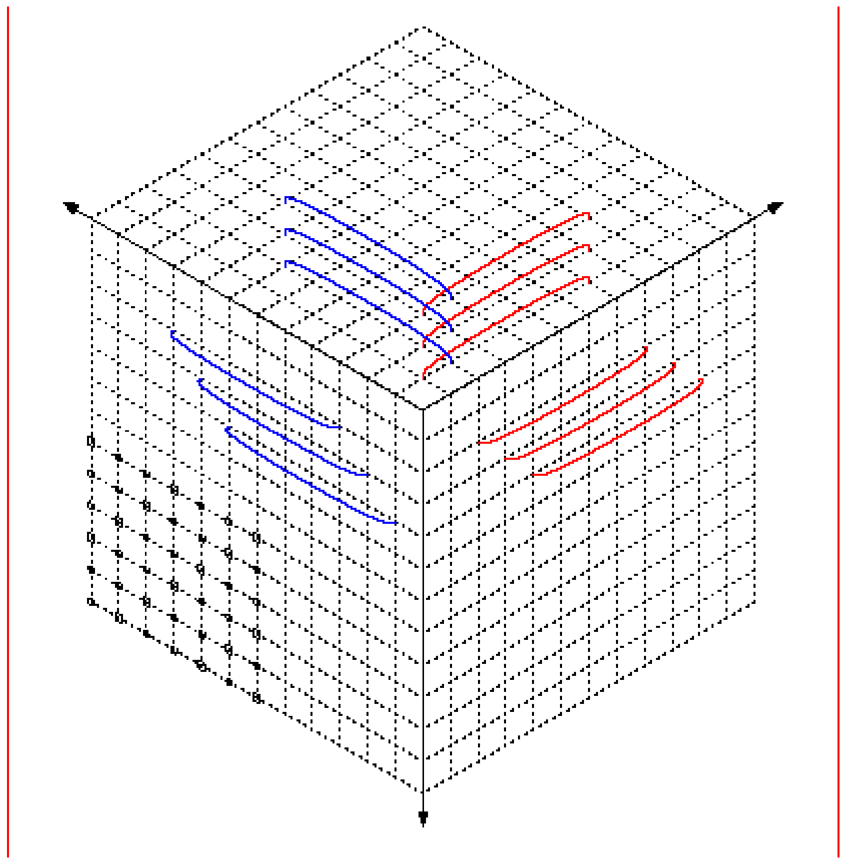


Figure 3.6. Selected links in the bypass scheme of $iBT(30 \times 30 \times 36, L = 3; l = \langle 6, 12 \rangle)$.

3.1.2 Evaluation Criteria

To evaluate a network model, we consider its diameter, average node-to-node hop distance, bisection width, and more importantly, the node-to-node hop distance distribution [48-49]. The network diameter, defined as the longest node-to-node hop distance, indicates the worst-case communication latency, while the average distance, defined as the average of the node-to-node hop distances, represents the expected communication latencies over the network. These two measures provide some information about the network while the hop distance distribution provides a richer representation of the network properties including maximum, average, and standard deviation of node-to-node distances. Additionally, we consider the bisection width to measure the aggregate network capacity and the network cost, defined as a product of diameter and node degree, for network comparison [50].

3.1.3 Search for Optimal iBT Networks

To evaluate the iBT network, it is necessary to identify which network configurations are feasible. First, we will enumerate certain constraints that exist for the iBT network.

Exhaustive Search of Feasible Networks

As an example, we begin by enumerating all networks with P processors, where $P \in [2^{15} - \Delta, 2^{15} + \Delta]$. For now, $\Delta = 0.05 \cdot 2^{15}$. With the range of network sizes fixed, the number of bypass dimensions is fixed at m, meaning the number of plain dimensions is equal to $n - m$. Commonly, for networks based on rectilinear architectures, network sizes are of the form 2^k; on the other hand, since the size of an iBT network in each bypass dimension must be a multiple of mk, its network size is not necessarily of the form 2^k. For this reason, we expand the search to a range of sizes.

For the $n = 3$ dimensional case, with node-degree $2n + 2 = 8$, even interlacing of bypass rings, constraints 1, 2, and 3, and $L = 3$, the largest number of bypass lengths is $k = 2$. Furthermore, $iBT(N_1 \times N_2 \times N_3; L = 3; l = \langle 6,12 \rangle)$ is the only class of feasible networks in this case. Further still, $30 \times 30 \times 36$ is the only network that will support these parameters and satisfy the upper bound of constraint 2.

Searching Networks up to 2^{20} nodes

To get an idea of how these iBT networks perform as network size increases, we have explored the space of networks up to size 2^{20} [51] and plot the average internode distance versus network size in Figure 3.7.

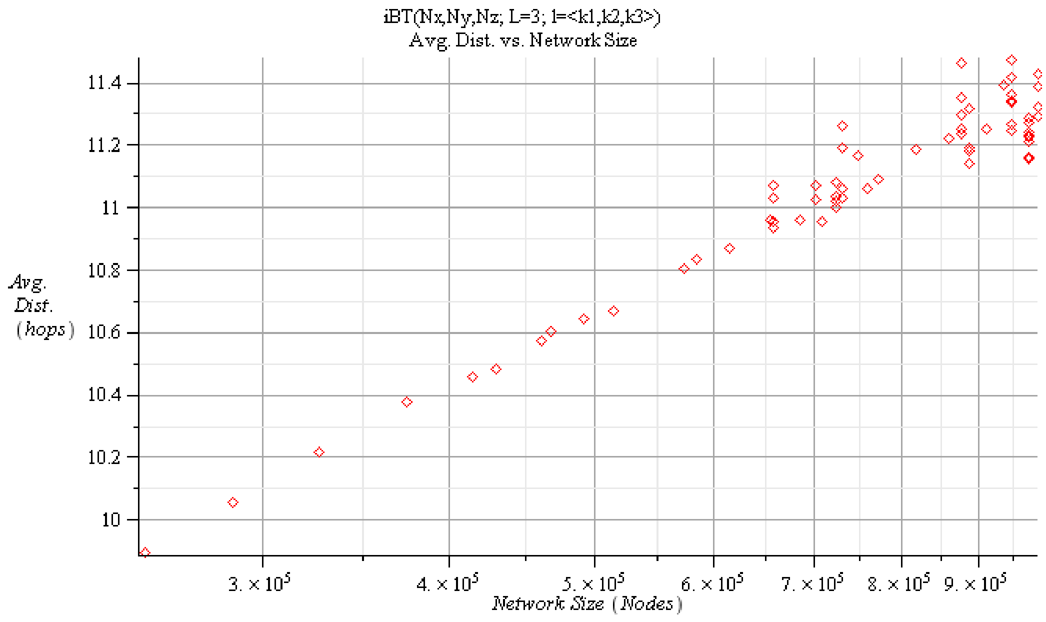


Figure 3.7. Detailed plot of iBT performance for networks up to 1,000,000 nodes [51].

3.2 Performance Analysis

In the following sections, we build the necessary methods to compare the selected networks. We choose to use popular metrics like network diameter and bisection width. Additionally, for a more detailed comparison, we also compare the internode distance distributions, highlighting the average internode distance, as well as the standard deviation of these distributions.

3.2.1 Distance Formulas, Network Diameters and Internode Distance Distributions

For all pairs of nodes in a graph, the ones with a shortest path of maximal length determine the diameter of a network. This metric proves to be useful in the comparison of network performance since it accurately characterizes the latency incurred in sending message across the network. For example, it can be used to calculate a lower bound in the time required for collective communications such as an all-to-all.

For the purposes of this study, we calculate the internode distance distributions in several ways. Where possible, we use the appropriate internode distance formula. For iBT networks, we derive such a formula. For some networks, like the RDT, we employ the Floyd-Warshall algorithm. Internode distance distribution is essential in the characterization of latency incurred during average case message passing.

For optimal routing, we always need to search for a path with the shortest internode distance [5, 8, 12]. As with many networks, there are many possible paths between a source node and destination node for iBT networks. It is much less obvious to recognize such paths for the iBT networks than for the torus network. This appears to be one of the few disadvantages of the iBT network. To overcome this, we have derived a closed-form internode distance formula for iBT networks with the bypass scheme $L \in \{2,3\}; l = \langle l_1 \rangle$. Other more complex cases can also be derived.

iBT Terminology

In $iBT(N_1 \times \cdots \times N_n; L = m; l = \langle l_1, \cdots, l_k \rangle)$ networks, the number of hops in a shortest path between a node-pair can be partitioned into two parts, $B(p_1, p_2)$ and $T(p_1, p_2)$. The part $B(p_1, p_2)$ consists of those edges along the first m bypass dimensions; $T(p_1, p_2)$ contains only edges along the remaining $n - m$ plain dimensions. Since the plain dimensions have no bypass connections, the procedure for calculating $T(p_1, p_2)$ is identical to that of a traditional torus network. Thus, we concentrate on calculating $B(p_1, p_2)$ by assuming that $m = n$. Considering this, we abbreviate the iBT networks with a uniform-length bypass connection in the first two or three dimensions as $iBT\left(N_x \times N_y; L = 2; l = \langle l_1 \rangle\right)$ and $iBT\left(N_x \times N_y \times N_z; L = 3; l = \langle l_1 \rangle\right)$, respectively.

In the $iBT\left(N_x \times N_y; L = 2; l = \langle l_1 \rangle\right)$ network, consider two points $p_1 = (x_1, y_1)^T$ and $p_2 = (x_2, y_2)^T$, where $x_i \in [0, N_x - 1]$ and $y_i \in \left[0, N_y - 1\right]$. The $sign(x)$ function is a standard sign function, while $sgn(x)$ is the signum function, defined as:

$$sign(x) = \begin{cases} -1, & x < 0, \\ 1, & x \geq 0; \end{cases}$$

and

$$sgn(x) = \begin{cases} -1, & x < 0, \\ 0, & x = 0, \\ 1, & x > 0; \end{cases}$$

respectively.

If x is a vector, the same operation applies to each of its components. For example, suppose $v = (x, y)^T$, then

$$sgn(v) = \begin{pmatrix} sgn(x) \\ sgn(y) \end{pmatrix},$$

$$|v| = \begin{pmatrix} |x| \\ |y| \end{pmatrix}.$$

Let $\delta(x, \alpha)$ be a two-point function defined as

$$\delta(x, \alpha) = \begin{cases} 1, & x = \alpha, \\ 0, & \text{otherwise}. \end{cases}$$

Now, consider an iBT derivative network in which each node is on bypass rings of each length and dimension. We will discuss a distance formula for this network and relate it to the distance formula for standard iBT networks. Let the vector $t = \left(t_x, t_y\right)^T$ be referred to as the fundamental torus distance. The magnitude of each element t_i represents the number of hops along dimension i in the $sign(t_i)i$ direction on a non-bypass shortest path from p_1 to p_2. For example, $t_x \neq 0$ indicates that the message traverses $|t_x|$ basis torus links in the positive or negative x-dimension, making the fundamental torus distance similar to the distance formula of a traditional torus. The definition of t_x is written as

$$t_x = \Delta x + \frac{N_x}{2} sgn(\Delta x)\{sign(N_x - 2\|\Delta x\|) - 1\},$$

in which $\Delta x = x_2 - x_1$. We similarly define and express t_y. In iBT networks, the set of links on a shortest path from p_1 to p_2 can be partitioned into two subsets: bypass rings and residual torus links. The set of edges forming the bypass ring subset contributes to the bypass distance $b = \left(b_x, b_y\right)^T$, a vector in which the magnitude of each component b_i is the number of bypass hops in dimension i in the $sign(b_i)i$ direction on a shortest path from p_1 to p_2. For example, $b_x \neq 0$ indicates that a message from p_1 to p_2 traverses $|b_x|$ hops of bypass rings in the positive or negative x-dimension. Thus, b_x is written as

$$b_x = round\left(\frac{t_x}{l}\right) - sgn(t_x) \cdot \delta\left(\left|frac\left(\frac{t_x}{l}\right)\right|, 0.5\right),$$

in which $round(x)$ rounds x to the nearest integer and $frac(x)$ returns the fractional part of x.

As stated previously, in addition to the bypass distance, a shortest path from p_1 to p_2 also has a residual torus link component. The residual torus distance is referred to as a vector $\hat{t} = \left(\hat{t}_x, \hat{t}_y\right)^T$ in which the magnitude of each component $\hat{t}_i$ is the number of torus hops in dimension i in the $sign(\hat{t}_i)i$ direction on a shortest path from p_1 to p_2. For example, $\hat{t}_x \neq 0$ indicates that the message routes $|\hat{t}_x|$ hops of torus links in the positive or negative dimension x. Thus,

$$\hat{t} = t - l \cdot b \Leftrightarrow \begin{pmatrix} \hat{t}_x \\ \hat{t}_y \end{pmatrix} = \begin{pmatrix} t_x - l \cdot b_x \\ t_y - l \cdot b_y \end{pmatrix}.$$

In addition to the distance vectors, we also refer to the bypass species, $s(p_1)$, of the node p_1. It is a vector defined as

$$s(p_1) = e_{[x_1 + y_1 (mod\ L)] + 1},$$

where “1” in $s(p_1)$ indicates the dimension in which the node adds bypass connections. For example, $s(p_1 = (1,2)^T) = e_2$ means p_1 adds bypass rings to the second dimension, i.e., dimension y.

The relationship among coordinates of p_1, p_2 and $\hat{t}$ is

$$x_1 + y_1 \ (mod\ L) \equiv x_2 + y_2 + \hat{t}_x + \hat{t}_y \ (mod\ L)$$

The stated definitions in $iBT\left(N_x \times N_y; L = 2; l = \langle l_1 \rangle\right)$ can all be extended to $iBT\left(N_x \times N_y \times N_z; L = 3; l = \langle l_1 \rangle\right)$.

$iBT\left(N_x \times N_y; L = 2; l = \langle l_1 \rangle\right)$

We will now relate the previous analysis to standard iBT networks. In $iBT\left(N_x \times N_y; L = 2; l = \langle l_1 \rangle\right)$ in which $l_1 = 2(k+1), k \in Z^+$, the distance between p_1 and p_2 is given by

$$D(p_1, p_2) = \|b\|_1 + \|\hat{t}\|_1 + \varphi_{xy}(p_1, p_2),$$

where

$$\varphi_{xy}(p_1, p_2) = \begin{cases} 2, & \alpha = 0, \beta = 2; \\ 2, & \alpha = 0, \gamma \in \{5,10\}; \\ 0, & \text{otherwise}, \end{cases}$$

in which $\alpha = \|\hat{t}\|_2^2$, $\beta = \|sgn(b)\|_2^2$ and $\gamma = \||sgn(b)| + s_1 + s_2\|_2^2$. The notation $\alpha = 0, \gamma \in \{5,10\}$ means that if $\alpha = 0, \gamma = 5$ or if $\alpha = 0$, $\gamma = 10$, we have $\varphi_{xy}(p_1, p_2) = 2$.

In this equation, the terms $\|b\|_1$ and $\|\hat{t}\|_1$ represent the bypass and torus hops a message needs to traverse under the assumption that a single node is on bypass rings across each of the bypass dimensions. The "penalty term," $\varphi_{xy}(p_1, p_2)$, accounts for the interlacing of bypass rings, where a given node has bypass rings in exactly one bypass dimension. For example, $\alpha = 0$ indicates no residual torus links are required, implying that $s(p_1) = s(p_2)$; meanwhile, $\beta = 2$ implies that a message has to traverse bypass rings in two dimensions. In this case, whichever bypass species a message emanates from, an additional torus hop is required to reach a node of a different species to traverse bypass hops in that bypass dimension. Then, a second torus hop is required to return to a node of the original bypass species. Thus, a valid shortest path in this case will always require a positive number of torus hops, meaning $\varphi_{xy}(p_1, p_2) = 2$. The set of α, β, γ terms together account for all source/destination bypass species cases.

***iBT*$\boldsymbol{(N_x \times N_y \times N_z; L = 3; l = \langle l_1 \rangle)}$**

In $iBT(N_x \times N_y \times N_z; L = 3; l = \langle l_1 \rangle)$ in which $l_1 = 3(k+1), k \in Z^+$, the distance between p_1 and p_2 is given by

$$D(p_1, p_2) = \|b\|_1 + \|\hat{t}\|_1 + \varphi_{xyz}(p_1, p_2),$$

where

$$\varphi_{xyz}(p_1, p_2) = \begin{cases} 4, & \alpha = 0, \gamma \in \{6,11\}; \\ 2, & \alpha \in \{0,1\}, \gamma = (5 - 2\alpha)\beta; \\ 2, & \alpha = 2, \hat{t} \cdot 1 = 0, \gamma = \beta^2 + 2; \\ 0, & \text{otherwise.} \end{cases}$$

3.2.2 Comparisons with Other Networks

In this section, we compare the topological properties of 3D iBT networks with several other networks having a number of nodes closest to that of a 4D torus network with exactly 32,768 nodes [3].

Hypercube

For the purposes of this dissertation, we will consider the hypercube in the following way. Let $H_n = (V, E)$ be a graph model of an n-dimensional hypercube consisting of a set of vertices V, and a set of edges E. When visualizing this graph, let each vertex of V have a unique permutation of the coordinates

$$\left(\begin{Bmatrix}1\\0\end{Bmatrix}, \begin{Bmatrix}1\\0\end{Bmatrix}, \begin{Bmatrix}1\\0\end{Bmatrix}, \dots \begin{Bmatrix}1\\0\end{Bmatrix}\right).$$

For convenience, we will express a permutation of the above coordinates corresponding to a vertex as an n-digit binary sequence.

In Figure 3.8 we show the cases H_0, H_1, H_2, H_3, H_4, which correspond to a single vertex (a point, or the singleton graph K_1), an edge with two vertices as endpoints (a line segment, or the path graph P_2), a circuit of length four (C_4, or a square graph), a cube, and a tesseract graph, respectively. [52-54]

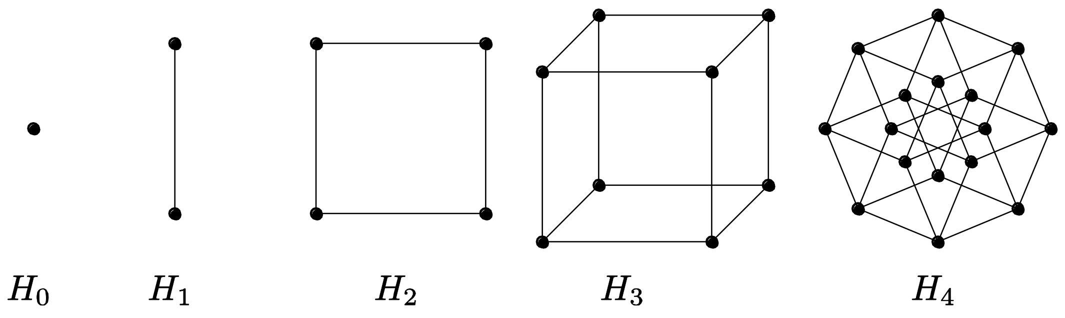


Figure 3.8. Visualization of the first 5 cases of hypercube graphs.

Hypercube Average Internode Distance

Consider the hypercube graph $H_n = (V, E)$. Since the set of vertices contains every n-digit (leading zeros allowed) binary sequence, the total number of vertices is 2^n. An edge of the graph connects two vertices if and only if the corresponding n-digit binary sequences are different by exactly one digit. For convenience, we say that two vertices are adjacent if and only if their Hamming distance is equal to 1, where the Hamming distance between two strings is the number of character places that are not equal. Since each vertex in the graph is an n-digit binary sequence, each vertex has exactly n neighbors. From [55],

$$|E| = \frac{\sum_{i=1}^{2^n} deg(i)}{2},$$

and the number of edges in the hypercube graph H_n is given by

$$|E| = \frac{n2^n}{2}.$$

To find the average internode distance for a hypercube H_n with $P = 2^n$ processors, we consider, without loss of generality, the processor $p_0 = (0,0,\dots,0)$. The average distance $\bar{d}$, where the distance between two processors represents the length of a shortest path between corresponding vertices, from p to all other vertices can be expressed with

$$\bar{d} = \frac{c_1 + 2c_2 + 3c_3 + \cdots + nc_n}{2^n - 1} = \frac{\sum_{i=1}^{n} ic_i}{2^n - 1},$$

where c_i is the number of vertices with corresponding processor address a Hamming distance i from $p_0 = (0,0,0,\ldots,0)$. In other words, c_i is the number of binary sequences with exactly i 1's, and is therefore the binomial coefficient $\binom{n}{i}$. So,

$$\bar{d} = \frac{\sum_{i=1}^{n} i \binom{n}{i}}{2^n - 1} = \frac{n2^{n-1}}{2^n - 1}.$$

The previous equation follows from the fact that

$$\sum_{i=0}^{\infty} \binom{n}{i} x^i = (1+x)^n,$$

and, when $x = 1$, that

$$\frac{d}{dx}(1+x)^n = \sum_{i=1}^{n} i \binom{n}{i} = n2^{n-1},$$

which is further discussed in [55].

Now that an expression for the average internode distance $\bar{d}$ has been derived, the additional comparative metric that we require an expression for is the variance of internode distance σ^2. Consider a set of data A, which represents distances for all pairs of processors, for which we wish to calculate the variance. Since our data set represents the entire population, we calculate the variance using the expression

$$\sigma^2 = \frac{\sum_{i=1}^{N}(A_i - \bar{A})^2}{N},$$

where N is the number of pairs (data points) and $\bar{A}$ is the mean of the set A. In this case, since A is the set of distances between each (unordered) pair of processors, we have that $N = \binom{P}{2}$.

For the term $\sum_{i=1}^{N}(A_i - \bar{A})$, we use the expression for $\bar{d}$ determined above. The set of distances A is composed of integers from the range $[1, 2, \ldots, n]$; we are left with determining the number of pairs a distance i apart.

The number of pairs 1 hop apart is equal to the number of edges $|E| = \frac{n2^n}{2}$. However, in general, we restate the problem of determining the number of pairs with a distance of i hops as the number of pairs of n-digit binary sequences whose Hamming distances are i. Since there are a total of 2^n such sequences, without loss of generality, we consider the n-digit binary sequence $B = b_1 b_2 \dots b_n$. There are $\binom{n}{i}$ sequences a Hamming distance of i away from B. Since $2^n \binom{n}{i}$ will count all pairs twice, there are $2^{n-1} \binom{n}{i}$ pairs of processors are a distance of i hops from each other. Therefore,

$$\sigma^2 = \frac{\sum_{i=1}^{n} 2^{n-1} \binom{n}{i} (i - \bar{A})^2}{\binom{P}{2}},$$

$$\sigma^2 = \frac{\sum_{i=1}^{n} 2^{n-1} \binom{n}{i} \left(i - \frac{n2^{n-1}}{2^n - 1}\right)^2}{\binom{2^n}{2}};$$

$$\sigma^2 = \frac{n}{4} \frac{(-8^n + 16^n - 28^n + 4^n + n4^n)}{(2^n - 1)^2 (4^n - 2^n)}.$$

This last statement is verified by [56].

In the following figure, the log nature of $\bar{d}$ is clear.

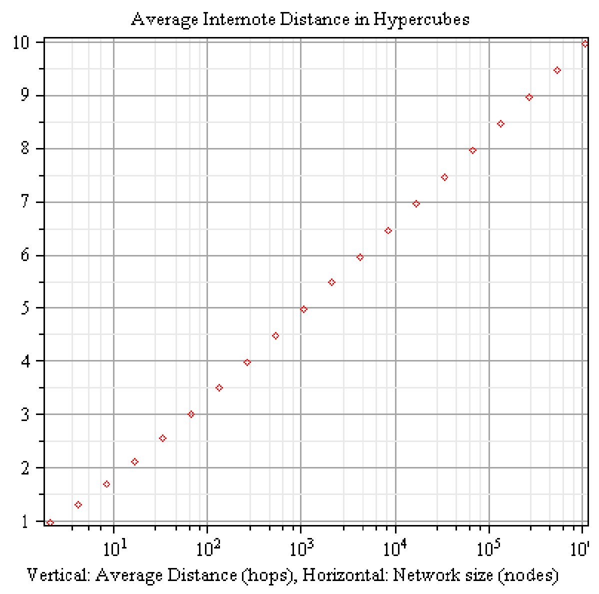


Figure 3.9. Average internode distance of Hypercubes.

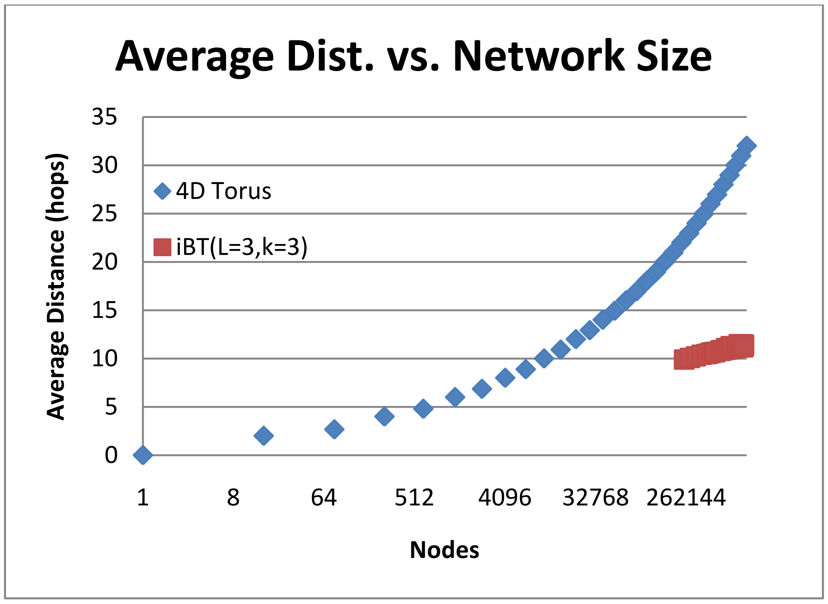


Figure 3.10. Comparison of iBT network and 4D Torus via average internode distance vs. network size. A detailed depiction of the iBT data points appears in Figure 3.7.

Cube Connected Cycles

Cube-connected cycles are inspired by the hypercube. An nth-order cube-connected cycle is generated by replacing each vertex in a d-dimensional hypercube by a cycle of length d [24, 54, 57]. One disadvantage of hypercube networks is that the degree of a node grows as d grows. Although cube-connected cycles share many properties with hypercubes, one difference is that for $d > 1$, each vertex in a cube-connected cycle has degree three.

Definition

The set of vertices is given in terms of the vertices of a hypercube in the following way.

$$V_{CCC} = V_H \times \{0,1, \dots, d-1\}$$

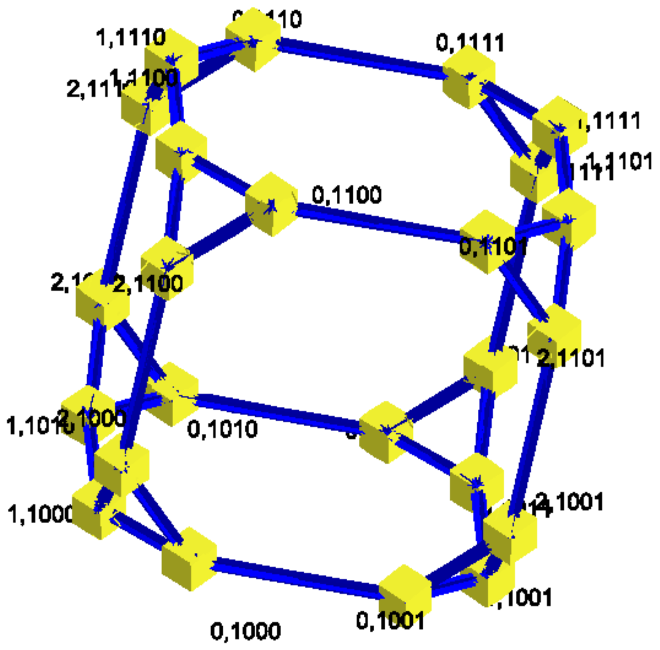


Figure 3.11. Cube-connected cycles of order 3, araged geometrically on the vertices of a truncated cube.

Consider two vertices $u = (p_u, i_u)$ and $v = (p_v, i_v)$, where p is the vertex within the base hypercube and i is the location within the cycle replacing hypercube vertex p. Vertices u and v are adjacent if any of the following are true:

1. $i_u = i_v$ and the address of p_u and p_v differ only in the ith bit,
2. $|i_u - i_v| = 1$ and $p_u = p_v$,
3. $i_u - i_v = d$ and $p_u = p_v$.

Number of Vertices and Edges

A hypercube of dimension d has 2^d vertices. Since each vertex in the hypercube is replaced by a cycle of length d, there are $|V_{CCC}| = d2^d$ vertices in the cube-connected cycle graph of dimension d. From above, each vertex has a degree of three, meaning that the total number of edges is

$$|E_{CCC}| = \frac{3d2^d}{2}.$$

Diameter

Think of each vertex $p_k = [(b_0 b_1 \ldots b_{d-2} b_{d-1}), i \in \{0,1,\ldots,d-1\}]$ as one possible reading of the binary odometer below which has a slider (with wrapping boundaries) that can be used to highlight a single bit and change its parity.

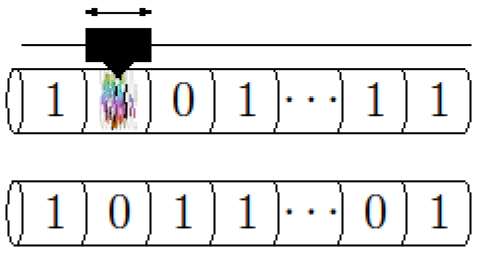


Figure 3.12. Cube-connected cycles adjacencies.

The slider has two legal "moves:"

1. Flip the currently highlighted bit (this is analogous to traversing an edge of type 1 from above)
2. Slide one digit left or right (this corresponds to traversing an edge of type 2 or 3, respectively)

So, the distance between nodes p_u and p_v in a CCC is equal to the number of slider moves from an odometer reading of p_u with the i_uth digit highlighted to a reading of p_v with the i_vth digit highlighted.

For $n \geq 4$, without loss of generality, suppose the slider starts at position 0. Further suppose that every digit needs to be changed. This contributes n bit switches and $n-1$ slides, at best. Now, from any given point on the odometer (which is really a ring because of boundary behavior), the final worst-case requirement is to force the slide as far from its current position as possible, which is about halfway around the ring, or $\left\lfloor\frac{n}{2}\right\rfloor - 1$ positions away. Therefore, the distance from any given node to a farthest node is

$$n + n - 1 + \left\lfloor\frac{n}{2}\right\rfloor - 1 = 2n + \left\lfloor\frac{n}{2}\right\rfloor - 2.$$

de Bruijn Graph

The de Bruijn graph [58-59] traditionally represents graphically the overlaps between sequences of symbols. Given an alphabet of m symbols, $s = \{s_1, \dots, s_m\}$, an n-dimensional de Bruijn graph, $G = (V, E)$, contains $|V| = m^n$ vertices, each one representing one of all possible length-n sequences of symbols from the given alphabet. Thus,

$$V = \{(s_1, \dots, s_1), (s_1, \dots, s_1, s_2), \dots, (s_m, \dots, s_m)\}.$$

Two vertices, u and v, are connected by the directed edge $\overrightarrow{(u, v)}$ in an n-dimensional de Bruijn graph if and only if the sequence corresponding to vertex v can be expressed by removing the first symbol of the sequence corresponding to vertex u and appending any symbol to the end. As a result, each vertex has indegree m and outdegree m.

Scalable Barrel Shifter

The Scalable Barrel Shifter network (SBS-net) is discussed in [26]. It is a network with $N = 2^n$ nodes. Nodes u and v are adjacent (connected by an undirected edge) if and only if $u \in \{v \pm 2^i \bmod N\}$, where $i = 0,1, \dots, n-1$. This implies that the degree of each node is $2n - 1$. The diameter is $\frac{1}{2}\log_2 N$.

Comparison results

We will analyze three iBT networks:

1. 3D iBT network with bypass rings in two dimensions $iBT(32 \times 32 \times$ *32;L=2*;
2. $iBT(64 \times 64 \times 8; L = 2)$;
3. $iBT(30 \times 30 \times 36; L = 3)$, a 3D iBT network with bypass rings in all three dimensions.

For each iBT network, we evaluate them with bypass rings of the same length or a mix of two different lengths. For comparison, we also analyze two other networks with a similar number of nodes: a 4D torus network $T(16 \times 16 \times 16 \times 8)$ and the 15D hypercube $H(2^{15})$. Since the node numbers of various network configurations are usually non-contiguous integers, it is unlikely one can find two configurations with exactly the same number of nodes. We compare two that are as close as possible: the 32,768-node $T(16 \times 16 \times 16 \times 8)$ and $H(2^{15})$ with the 32,400-node $iBT(30 \times 30 \times 36; L = 3)$.

Through exhaustive numerical search, we found the optimal iBT network of approximately 32,000 nodes to be $iBT(30 \times 30 \times 36; L = 3; l = \langle 6,12 \rangle)$. We further compare this with other networks and graphs in Table 2 and Figure 3.18. All networks listed in the table except for the 3D torus, hypercube, the CCC network [24] and the scalable Barrel Shifter [26] have a node degree of 8. As shown in Figure 3.18, these networks are grouped into three categories by their sizes. Of all 32,000-node networks of degree 8, the iBT network has the shortest average distance.

Our analysis starts from numerical experiments. Figure 3.13—Figure 3.17 illustrate the numerical results for the 3D $iBT(L = 2)$ and $iBT(L = 3)$ networks, compared with 4D torus and hypercube. Table 2 presents such network topological properties as internode distance distribution, network diameter, and bisection width. These experiments show, as expected, the dependence of the network properties on the bypass scheme; network properties behave relatively poorly at bypass extremes: too short or too long. Starting from a torus network $T(N_x \times N_y \times N_z)$, we study the following cases:

1. For $N_x \times N_y \times N_z = 32 \times 32 \times 32 = 32{,}768$, we bypass in two dimensions with uniform bypass length to generate a new network, $iBT(32 \times 32 \times 32; L = 2; l = \langle l_1 \rangle)$ with $l_1 \in \{2, 4, 6, 8, 16\}$. We found the resulting networks have relatively poor network properties for extreme bypassing lengths such as $l_1 \in \{2, 16\}$, but have better properties with mid-ranged bypass lengths such as $l_1 = 6$. As a result, $iBT(32 \times 32 \times 32; L = 2; l = \langle 6 \rangle)$ is the optimal iBT network with uniform bypass length in Figure 3.14. For $N_x \times N_y \times N_z = 32 \times 32 \times 32$ (same as above), we bypass in two dimensions with a mixture of two bypass lengths to generate a network, $iBT(32 \times 32 \times 32; L = 2; l = \langle l_1, l_2 \rangle)$, with $l_1, l_2 \in \{4, 8, 16, 32\}$. With all possible combinations, the bypassing parameter $l = \langle 4{,}16 \rangle$ generates the optimal iBT network with two bypass lengths in Figure 3.15. We also found that $iBT(32 \times 32 \times 32; L = 2; l = \langle 4{,}16 \rangle)$ excels over $iBT(32 \times 32 \times 32; L = 2; l = \langle 6 \rangle)$ in Table 2.

2. For $N_x \times N_y \times N_z = 64 \times 64 \times 8 = 32{,}768$, we bypass in two of the longest dimensions, generating a new network, $iBT(64 \times 64 \times 8; L = 2; l = \langle l_1, l_2 \rangle)$. Under the same bypass scheme $L = 2; l = \langle l_1, l_2 \rangle$, we found $iBT(64 \times 64 \times 8; L = 2)$ always outperforms $iBT(32 \times 32 \times 32; L = 2)$ in Figure 3.13 and $iBT(64 \times 64 \times 8; L = 2; l = \langle 4{,}16 \rangle)$ is the best of all possibilities, where $L = 2$ in Table 2.

3. For $N_x \times N_y \times N_z = 30 \times 30 \times 36 = 32{,}400 \approx 32{,}768$, we consider bypassing in all three dimensions to generate the network $iBT(30 \times 30 \times 36; L = 3)$. As shown in Figure 3.16—Figure 3.17, $iBT(30 \times 30 \times 36; L = 3; l = \langle l_1 \rangle)$ with $l_1 \in \{6, 9, 12\}$ demonstrated similar behavior to that of $iBT(64 \times 64 \times 8; L = 2; l = \langle l_1, l_2 \rangle)$. We also found $iBT(30 \times 30 \times 36; L = 3; l = \langle 6{,}12 \rangle)$ is the best among all of the iBT networks, better than 4D torus and similar to 15D hypercube in Figure 3.17 and Table 2.

4. As summarized in Table 2, we found that most networks have the same network diameter but different average distances and various standard deviations; a network with a larger network diameter may have a smaller average distance such as $iBT(30 \times 30 \times 36; L = 3; l = \langle l_1 \rangle)$ with $l_1 \in \{9,12\}$.

From these experiments, we make the following claims:

1. For the iBT networks with uniform bypass length, extreme bypass length achieves poorer network performance than mid-range bypass lengths.

2. An appropriate mixture of bypass lengths is favored in the interlacing arrangement for iBT networks over uniform bypass length.

3. For iBT networks with plain dimensions, a plain dimension size should be shrunk to scale to bypass dimensions for optimized performance.

4. The most efficient bypass scheme for a 3D iBT network is without plain dimensions. It is shown that, among all the possibilities of a system with approximately 32,000 nodes, $iBT(30 \times 30 \times 36; L = 3; l = \langle 6,12 \rangle)$ is the best network. It performs much better than the simple 4D torus $T(16 \times 16 \times 16 \times 8)$ with 32,768 nodes and it performs similarly to the 15D hypercube $H(15)$ with 32,768 nodes and degree 15. Its network cost of value 96 is much smaller than the 4D torus' 224 and the hypercube's 225.

5. The internode distance distribution is efficient and precise in its depiction of topological details for the comparison of networks.

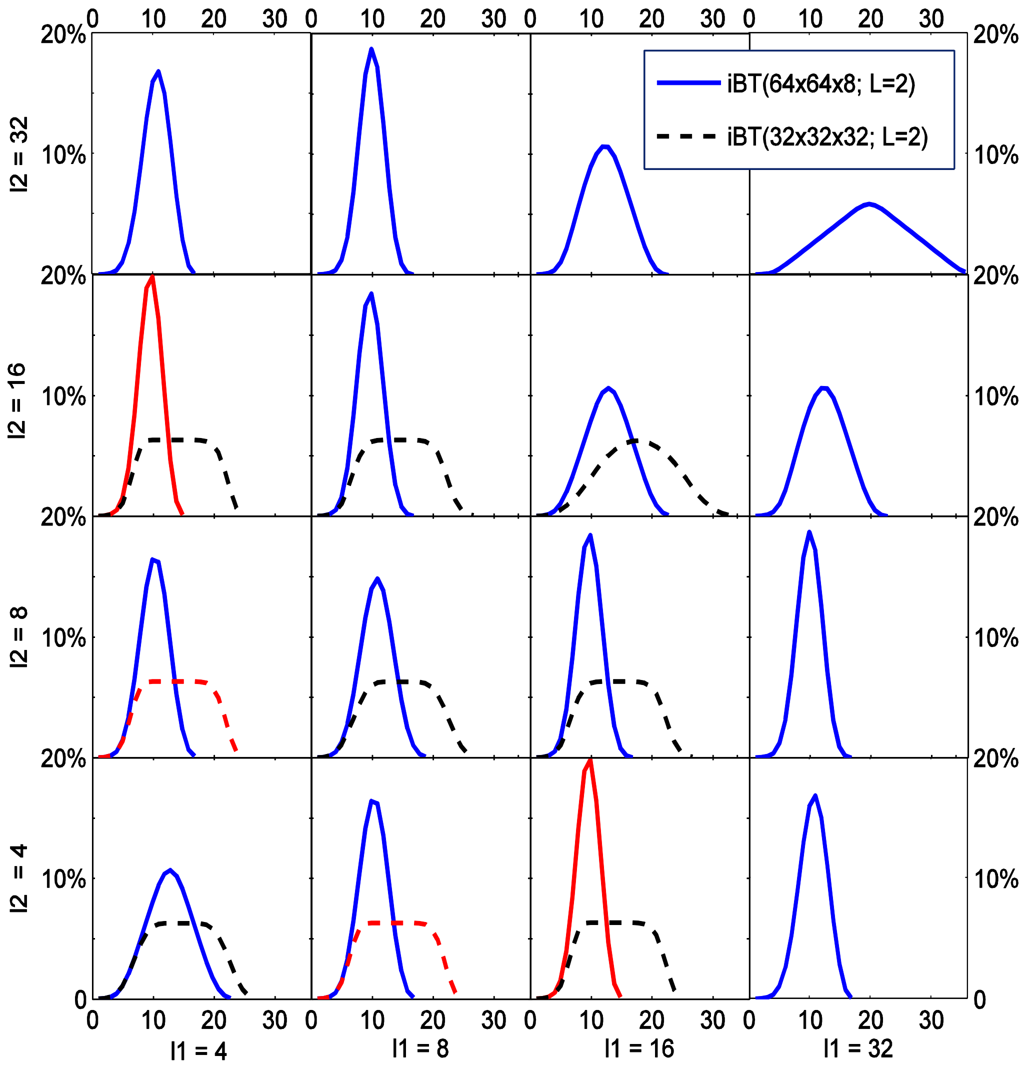


Figure 3.13. Internode distance distributions for 3D $iBT(L = 2)$.

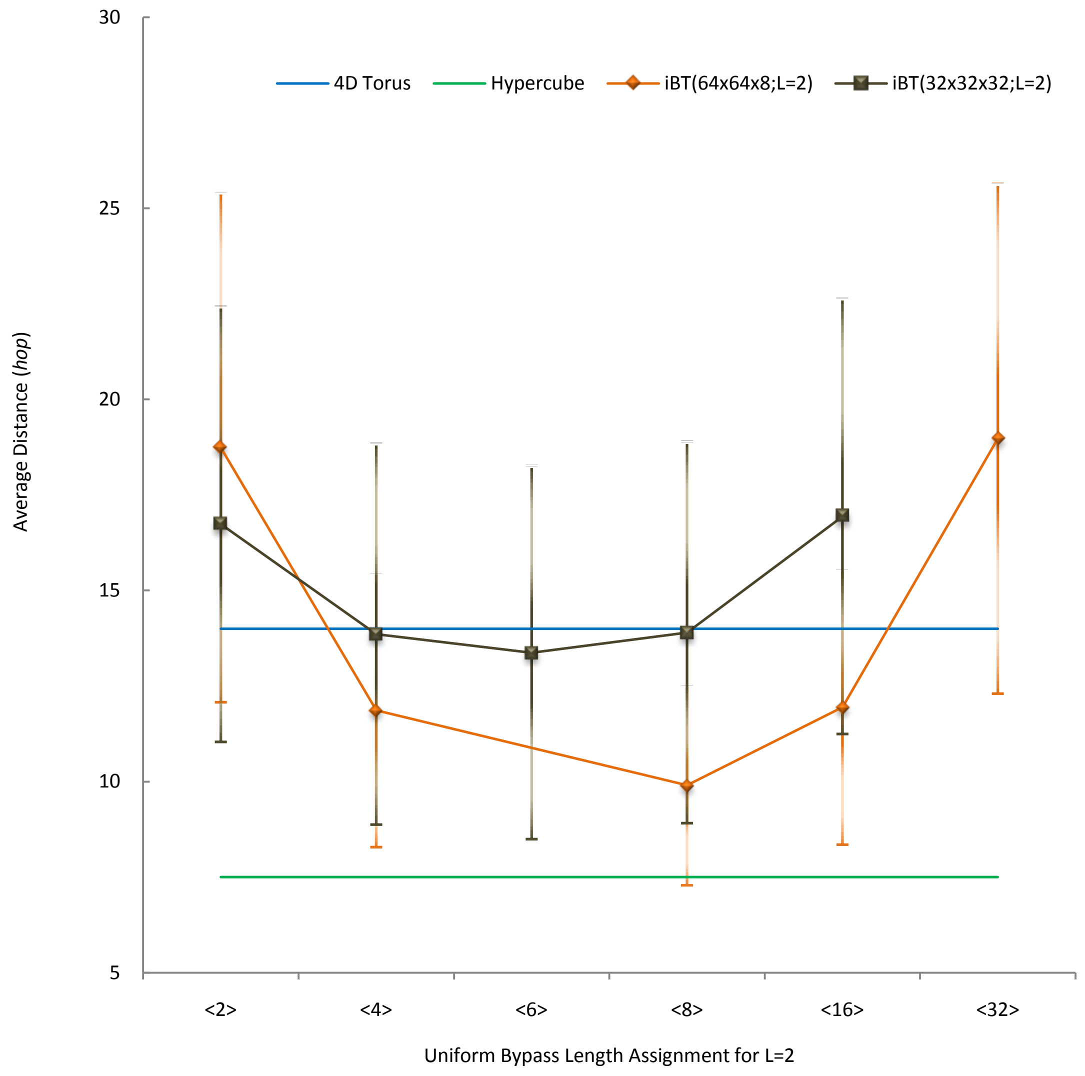


Figure 3.14. Average internode distances and standard deviations for 3D $iBT(L = 2)$ with uniform bypass length, $T(16 \times 16 \times 16 \times 8)$ and $H(2^{15})$ networks.

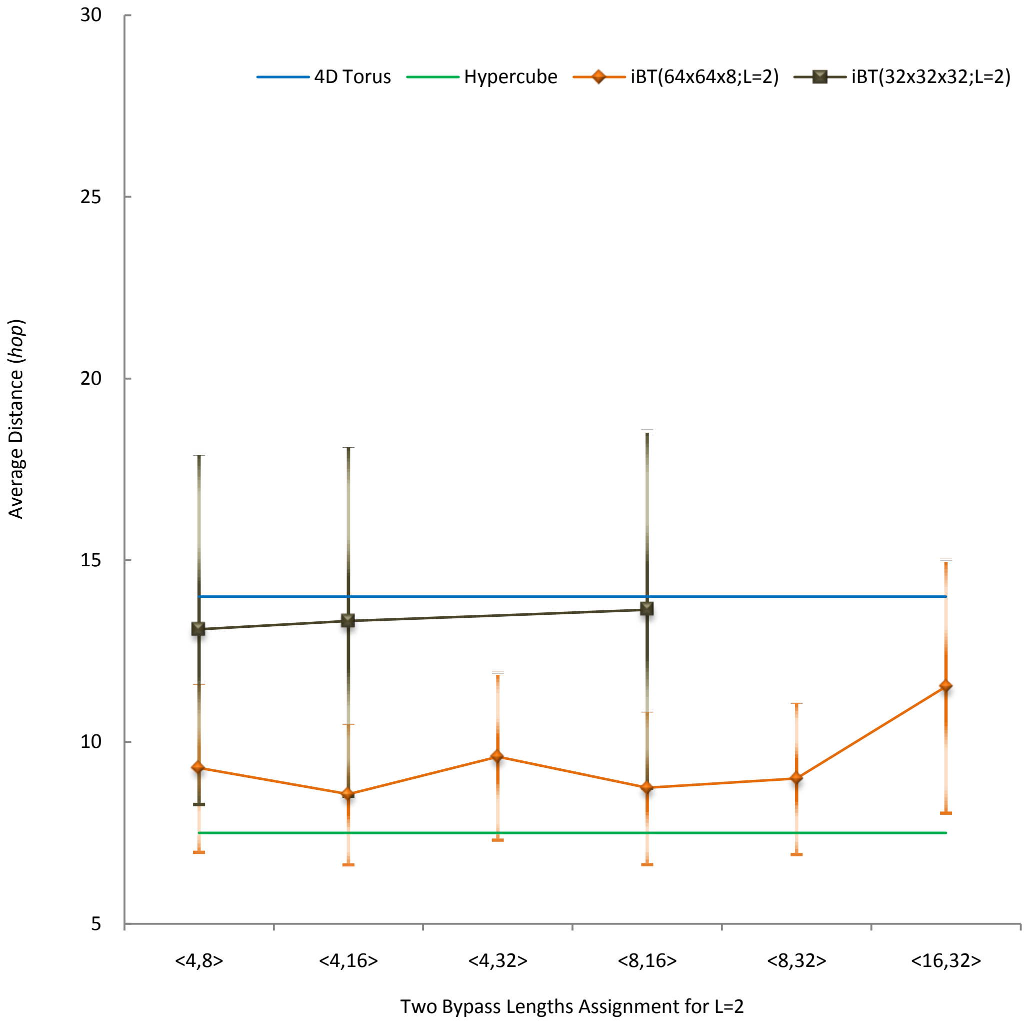


Figure 3.15. Average internode distances and standard deviations for 3D $iBT(L = 2)$ with two bypass lengths, $T(16 \times 16 \times 16 \times 8)$ and $H(2^{15})$ networks.

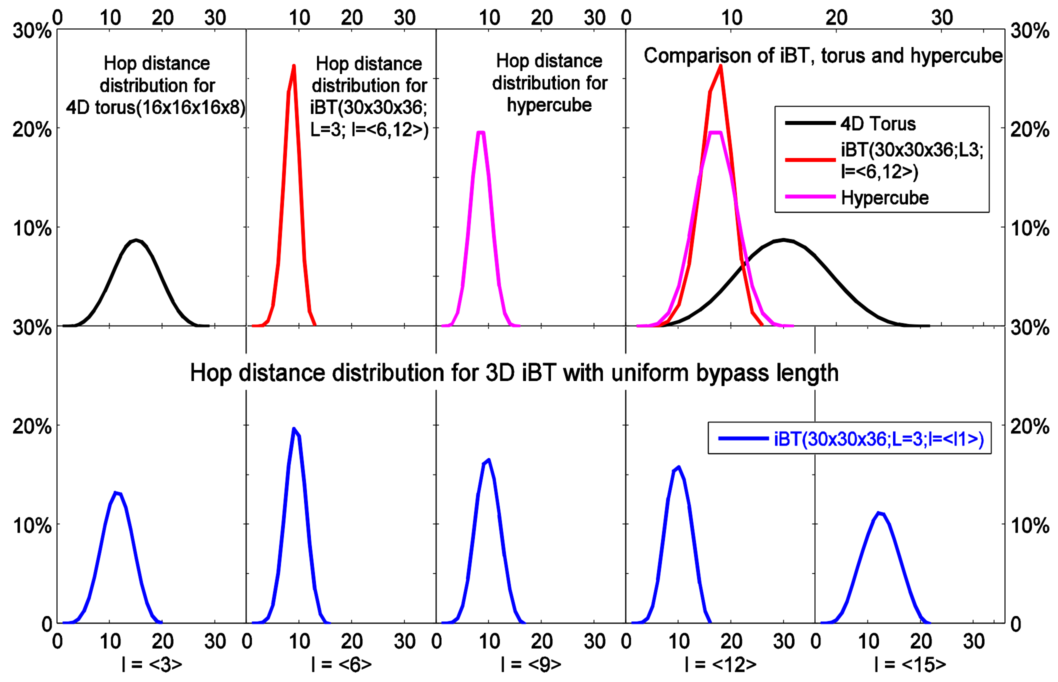


Figure 3.16. Internode distance distributions for 3D $iBT(L=3)$, $T(16\times16\times16\times8)$ and $H(2^{15})$ networks.

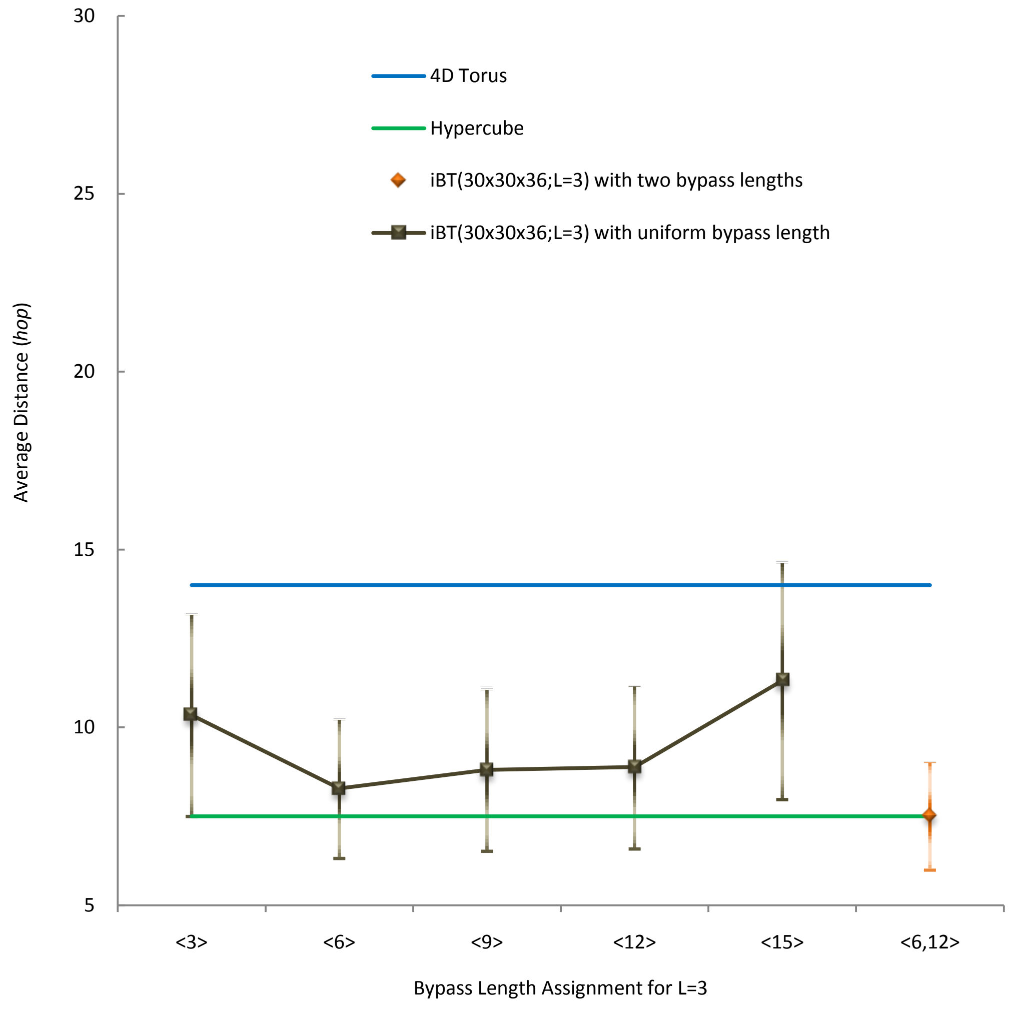


Figure 3.17. Average internode distances and standard deviations for 3D $iBT(L = 3)$, $T(16 \times 16 \times 16 \times 8)$ and $H(2^{15})$ networks.

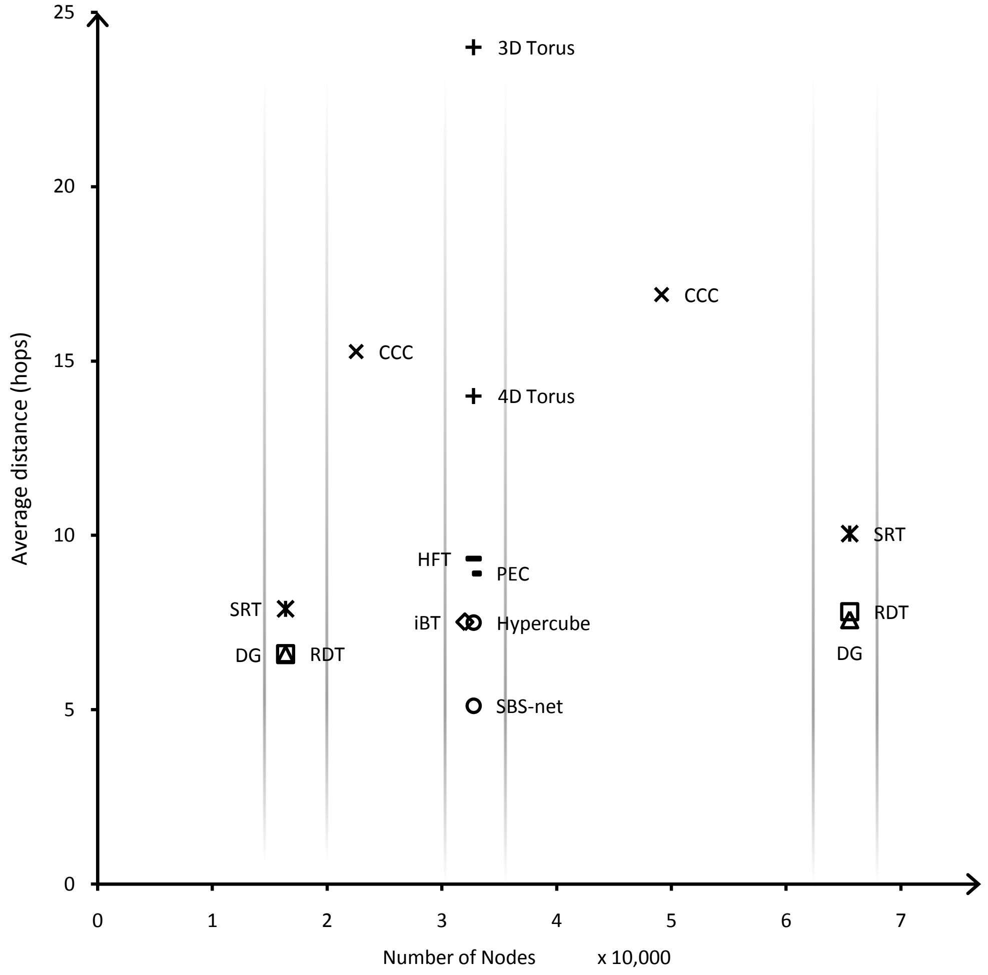


Figure 3.18. Average internode distance comparisons for several networks.

Table 2. Topological properties of iBT, Torus, and Hypercube interconnection models.

iBT models			Node Degree	Hop Dist. (*hop*)		Network Diam. (hop)	Bisection Width (link)	Network Cost
Bypass Scheme				Average	Std. Dev.			
32,32,32	2	2	8	16.7344	5.6961	33	2048	264
		4		13.8593	4.9855	26		208
		6		13.3730	4.8800	26		208
		8		13.8984	4.9872	26		208
		16		16.9414	5.6963	32		256
	2	4,8		13.1035	4.8235	24		192
		4,16		13.3257	4.8199	24		192
		8,16		13.6416	4.9104	26		208
64,64,8	2	2		18.7422	6.6664	37	2048	296
		4		11.8672	3.5835	22	3072	176
		8		9.9023	2.6172	18	6120	144
		16		11.9434	3.5945	22	8192	176
		32		18.9697	6.6687	36	8192	288
	2	4,8		9.2908	2.3277	16	4096	128
		4,16		8.5679	1.9477	14	6144	112
		4,32		9.5942	2.2946	16	6144	128
		8,16		8.7402	2.1133	16	7168	128
		8,32		8.9987	2.0954	16	7168	128
		16,32		11.5198	3.4786	22	8192	176
30,30,36	3	3		10.3464	2.8542	19	3600	152
		6		8.2800	1.9675	15	5400	120
		9		8.8034	2.2895	16	7200	128
		12		8.8827	2.3044	15	9000	120
		15		11.3114	3.3441	21	7560	168
	3	**6,12**		**7.5152**	**1.5288**	**12**	**7200**	**96**
3D Torus (32x32x32) [5, 7-8, 10]			6	24.0000	8.0312	48	2048	288
4D Torus (16x16x16x8)			8	14.0000	4.2426	28	4096	224
15D Hypercube			15	7.5000	1.9365	15	16384	225
PEC (256x128) (32,768 nodes) [20]			8	8.9068	1.8342	15	1920	120
2D SRT (128x128) (16,384 nodes) [19]			8	7.8904	1.6543	13	1664	104
2D SRT (256x256) (65,536 nodes) [19]			8	10.0529	1.9974	16	3840	128
RDT(2,4,1)/α(128x128) (16,384nodes)[22]			8	6.6113	1.2340	10	5632	80
RDT(2,4,1)/α(256x256) (65,536nodes)[22]			8	7.8076	1.4521	12	23552	96
CCC 11-11 (22,528 nodes) [24]			3	15.2685	2.8432	25	1024	75
CCC 12-12 (49,152 nodes) [24]			3	16.9020	2.9487	28	2048	84
Scalable Barrel Shifter (32,768 nodes)[26]			29	5.1111	1.1000	8	49150	232
de Bruijn Graph DG(4,7)(16,384nodes)[23]			8	6.0287	0.9025	7	32768	56
de Bruijn Graph DG(4,8)(65,536nodes)[23]			8	7.0145	0.9134	8	131072	64
Hybrid Fat Tree (32,768 nodes) [25]			[2,29]	9.3334	1.6922	15	3	60

3.3 Application-Specific Analysis of Other Networks

In the following sections, we present an application specific comparison of other networks. This is done by abstracting the network as a supply matrix and choosing an appropriate application load with which to compare networks.

3.3.1 Supply-Demand Matrices

Supercomputer systems are modeled as a graph $G = (V, E)$, where the set of vertices, V, represents the set of compute nodes. Two vertices, u and v, are connected by an edge in E if and only if the corresponding nodes are connected by communication links in the interconnection network.

In the following sections, we describe supercomputer networks and their corresponding graphs. From these graphs, we define the supply matrix to be the all-pairs distance matrix, where entry i, j represents the distance between nodes i and j in the communication network, and the length of a shortest path between corresponding vertices i and j.

n-ary k-cubes

Communication networks of this type are found in torus systems like IBM's BlueGene and QCDOC. Figure 3.19 and Table 3 summarize the structure of IBM's BlueGene while Figure 3.20 and Table 4 give examples of the truncated torus of the QCDOC network.

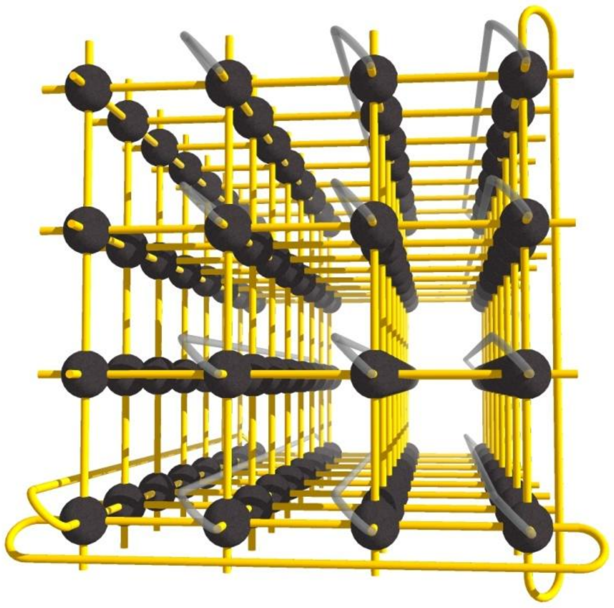


Figure 3.19. A visualization of a representative partition of BlueGene's 3D torus network.

Table 3. Representative partitions of BG/L.

Nodes	X	Y	Z
32	4	4	2
64	8	4	2
128	8	4	4
256	8	4	8
512	8	8	8
1024	8	8	16
2048	16	8	16
4096	32	8	16

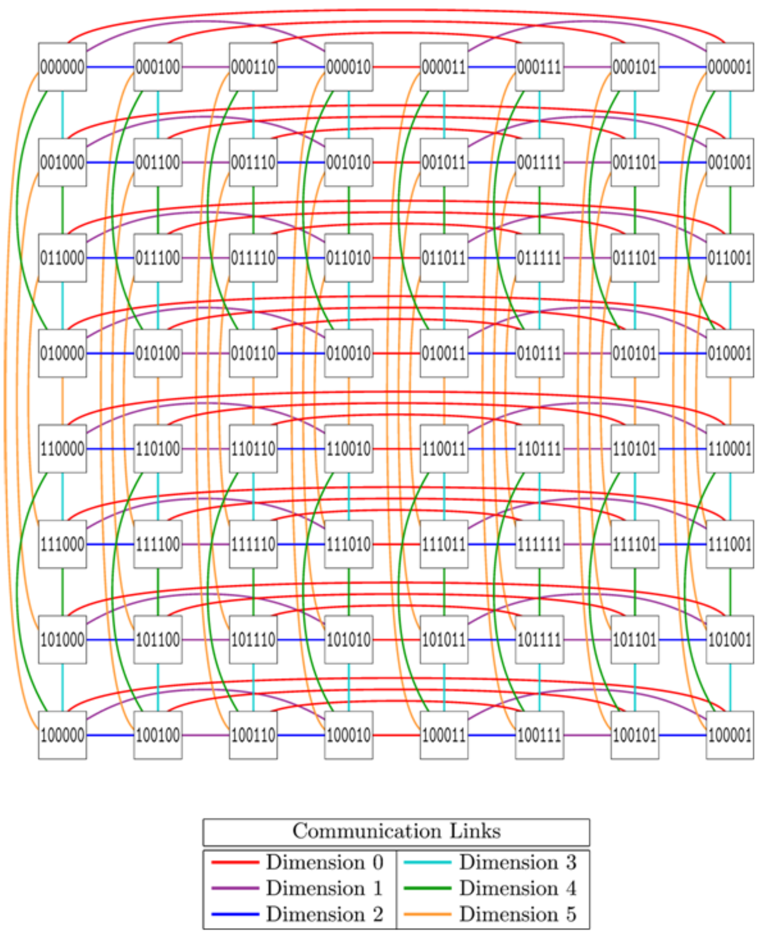


Figure 3.20. A 2D representation of a 64 node (2x2x2x2x2x2) QCDOC partition, this network is also a 2-ary 6-cube.

Table 4. Representative partitions of QCDOC.

P	X	Y	Z	K	M	N
64	2	2	2	2	2	2
512	2	2	2	4	4	4
1024	2	2	2	4	4	8
2048	2	2	2	4	4	16
4096	2	2	2	8	8	8

To calculate the distance between two vertices in this type of graph, let the address of u be $u = (u_x, u_y, \ldots)$ and the address of v be $v = (v_x, v_y, \ldots)$. Additionally, let the non-torus distance along dimension i between u and v be

$di = u_i - v_i$. The distance between u and v is then

$$\sum_{i \in \{x,y,\dots\}} \min\{|d_i|, I - |d_i|\},$$

where I is the number of nodes along dimension i. The average internode distance is then

$$H_{avg} = \binom{\prod_{j \in \{X,Y,\dots\}} j}{2}^{-1} \left(\sum_{(c,v),u<v} \sum_{i \in \{x,y,\dots\}} \min\{|d_i|, I - |d_i|\} \right).$$

3D Hexagonal Networks

The 3D hexagonal networks discussed in [12] use the following distance and average distance formulas.

$$D(u,v) = max\{|dx|, |dy|, |dz|, |dx + dy|, |dx + dz|, |dy + dz|, |dx + dy + dz|\},$$

$$H_{avg} = \binom{\frac{10}{3}t^3 - 5t^2 + \frac{11}{3}t - 1}{2}^{-1} \left(\sum_{(u,v),u<v} D(u,v) \right).$$

Lie Algebraic Networks

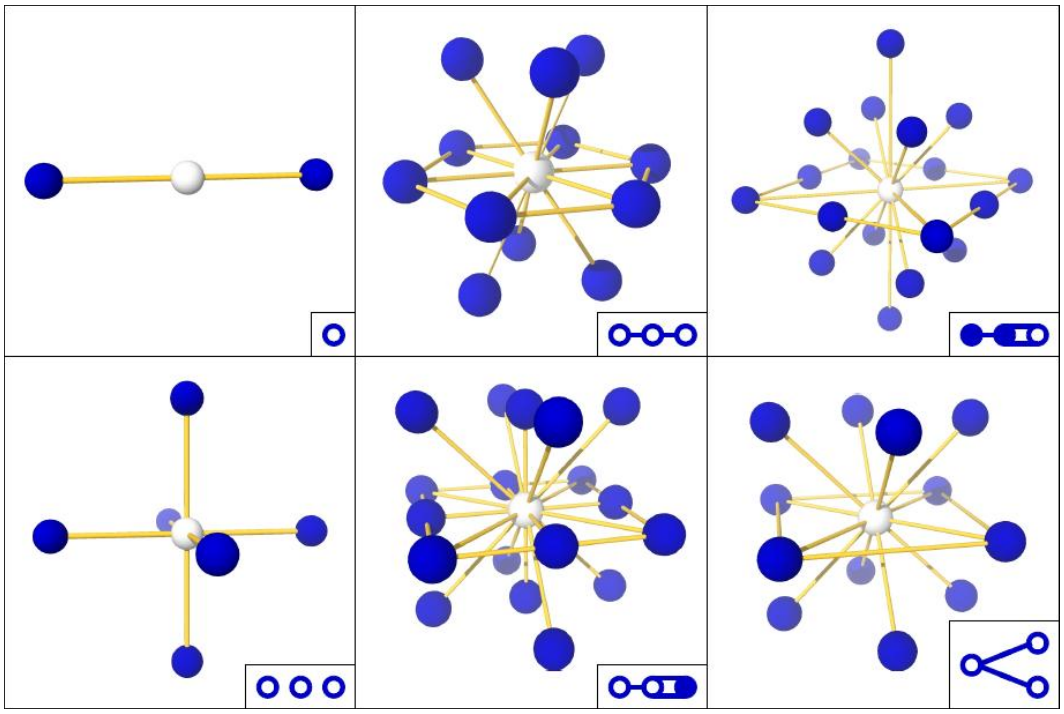


Figure 3.21. The Lie Classical Algebras. These artive partitions of BG/Lor constructing larger networks.

Generating Coordinates for Networks based on $Sp(\mathbf{6})$

The following pseudo code outlines a process which will yield 3 dimensional coordinates for nodes in networks based on the $Sp(6)$ structure.

```
GenerateVectors()
$\Delta_{13} \leftarrow 4$, $\Delta_{23} \leftarrow 2$, $\Delta_{33} \leftarrow 1$, $coords \leftarrow \{\}$, $count \leftarrow 0$
for $\Gamma_{13} \leftarrow \Delta_{23}$ to $\Delta_{13}$
    do
        for $\Gamma_{23} \leftarrow \Delta_{33}$ to $\Delta_{23}$
            do
                for $\Delta_{12} \leftarrow \Gamma_{23}$ to $\Gamma_{13}$
                    do
                        $\Xi_1 \leftarrow \Gamma_{13} + 2\Gamma_{23} - \Delta_{13} - \Delta_{23} - \Delta_{33}$
                        for $\Delta_{22} \leftarrow \Xi_1$ to $\Gamma_{23}$
                            do
                                for $\Gamma_{12} \leftarrow \Delta_{22}$ to $\Delta_{12}$
                                    do
                                        $\Xi_2 \leftarrow \Gamma_{12} - \Delta_{22}$
                                        WeightVectors
return $coords$
```

Pseudocode 1. This procedure generates 3 dimensional coordinates for nodes of networks built from the Sp(6) structure. [13]

```
WeightVectors
for $\Delta_{11} \leftarrow \Xi_2$ to $\Gamma_{12}$
    do
        $J_1 \leftarrow \Delta_{11}$
        $J_2 \leftarrow \Delta_{12} + \Delta_{22} + \Delta_{11} - 2\Gamma_{12}$
        $J_3 \leftarrow \Delta_{13} + \Delta_{23} + \Delta_{33} + \Delta_{12} + \Delta_{22} - 2\Gamma_{13} - 2\Gamma_{23}$
        for $M_1 \leftarrow -J_1$ to $J_1$ by 2
            do
                for $M_2 \leftarrow -J_2$ to $J_2$ by 2
                    do
                        for $M_3 \leftarrow -J_3$ to $J_3$ by 2
                            do
                                $coords \leftarrow coords \cup \{[M_1, M_2, M_3]\}$
                                $count \leftarrow count + 1$
```

Pseudocode 2. The WeightVectors function of the GenerateVectors() procedure [13]

Once the vectors corresponding to the node coordinates are generated, the vertices of a graph model are created in the following manner. Let

$$f(x,y,z) = 10^6 + 10^5|x| + 10^4|y| + 10^3|z| + 10^2(\operatorname{sign}(x)+1) + 10^1(\operatorname{sign}(y)+1) + \operatorname{sign}(x) + 1$$

be the function that maps the set of vectors onto integers to be used as vertex addresses. Vertices are then added by the following loop.

```
GenerateVertexSet
1  for $i \leftarrow 1$ to $|coords|$
2      do
3          $V \leftarrow V \cup f(coords_{i,x}, coords_{i,y}, coords_{i,z})$
```

Pseudocode 3. This function generates the set of vertices V of the graph model.

Finally, the set of edges is constructed by creating adjacency according to the $Sp(6)$ root structure.

```
GenerateEdgeSet
1  $neighbors \leftarrow [[2,0,0],[0,2,0],[0,0,2],[1,1,0],[1,-1,0],[1,0,1],[1,0,-1],[0,1,1],[0,1,-1]]$
2  for $u \leftarrow 1$ to $|coords|$
3      do
4          for $i \leftarrow 1$ to $|neighbors|$
5              do
6                  if $v = [coords_{u,x}, coords_{u,y}, coords_{u,z}] + neighbors_i \in V$
7                    then AddEdge$(u,v)$
```

Pseudocode 4. This function defines the set of edges E for the graph model.

In the following figure, the graph model is visualized to give an idea about its structure.

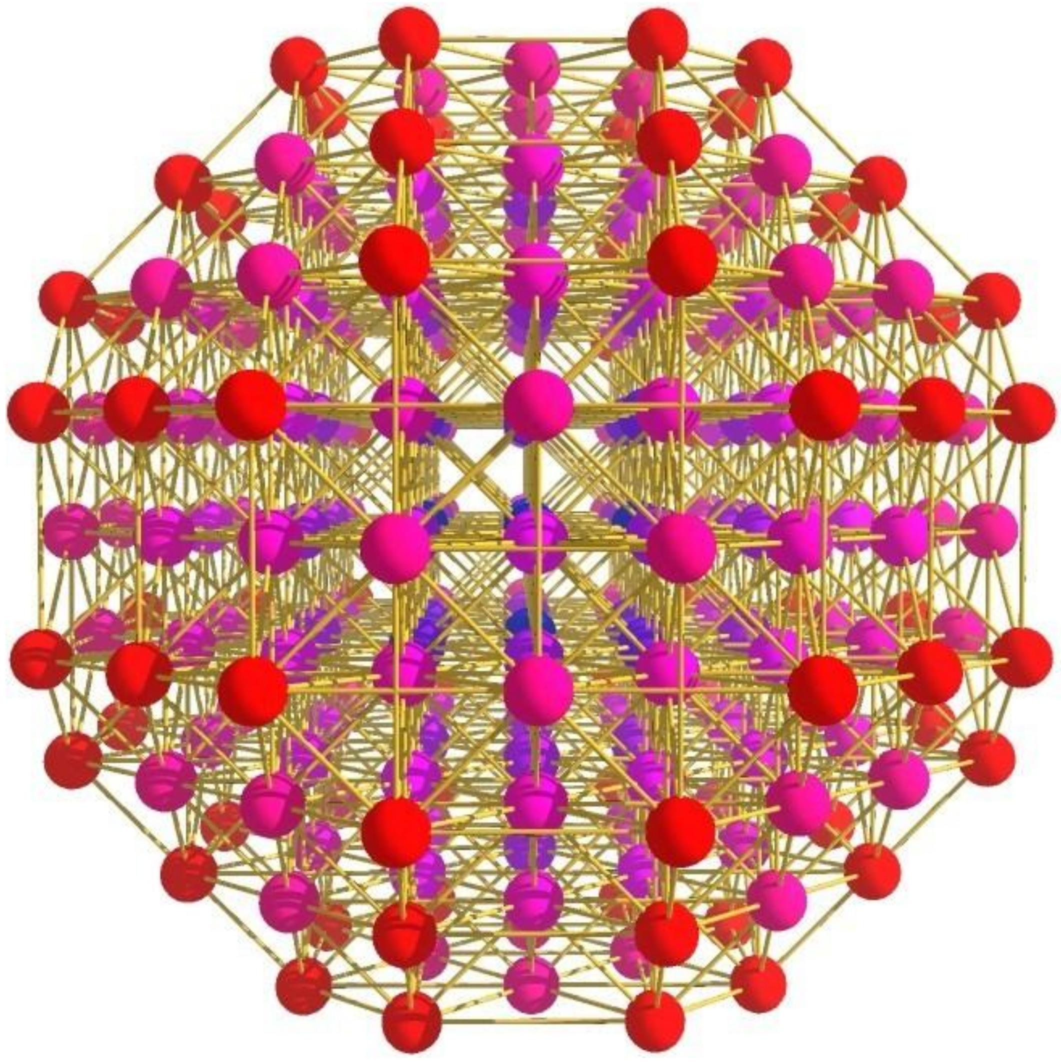


Figure 3.22. Visualization of the network built with Sp(6) structure.

The supply matrix for this structure is calculated by the Floyd-Warshall algorithm [60].

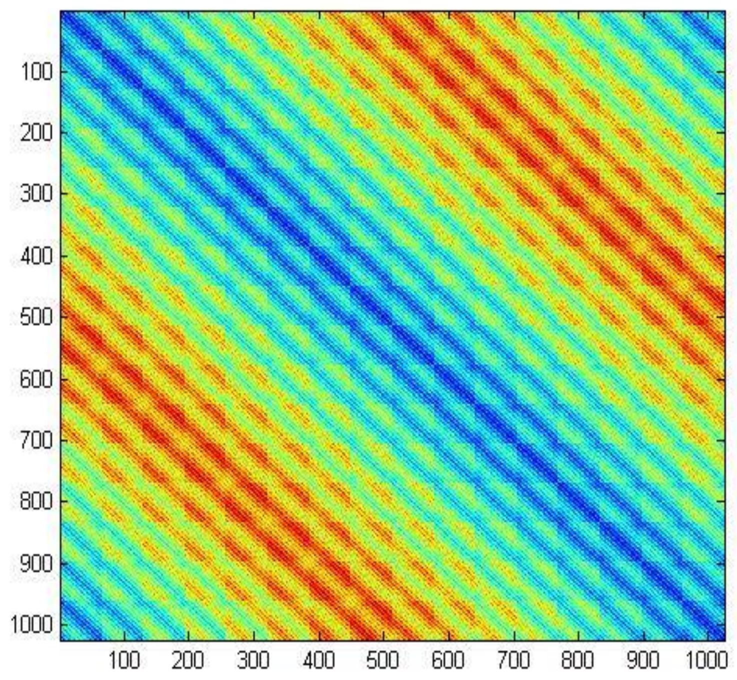


Figure 3.23. Supply Matrix for BG/L, 1024 nodes.

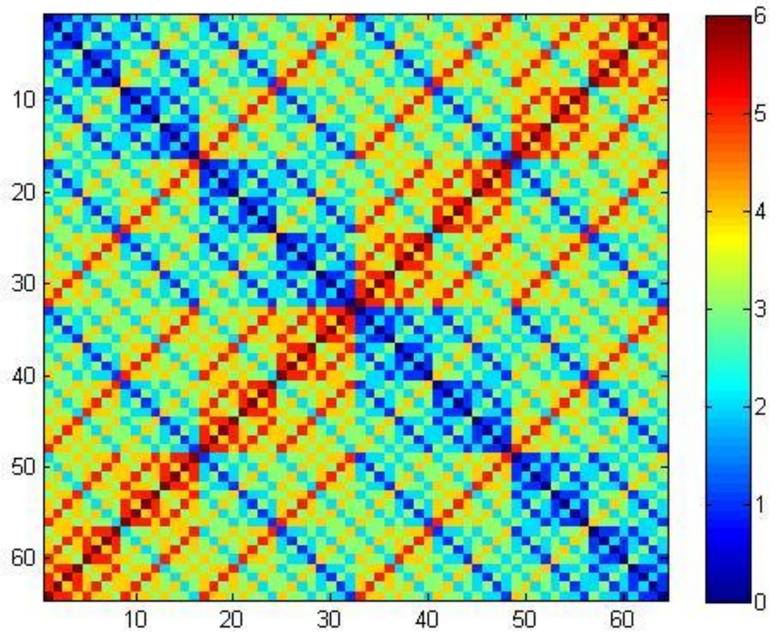


Figure 3.24. Supply matrix for QCDOC, 64 nodes.

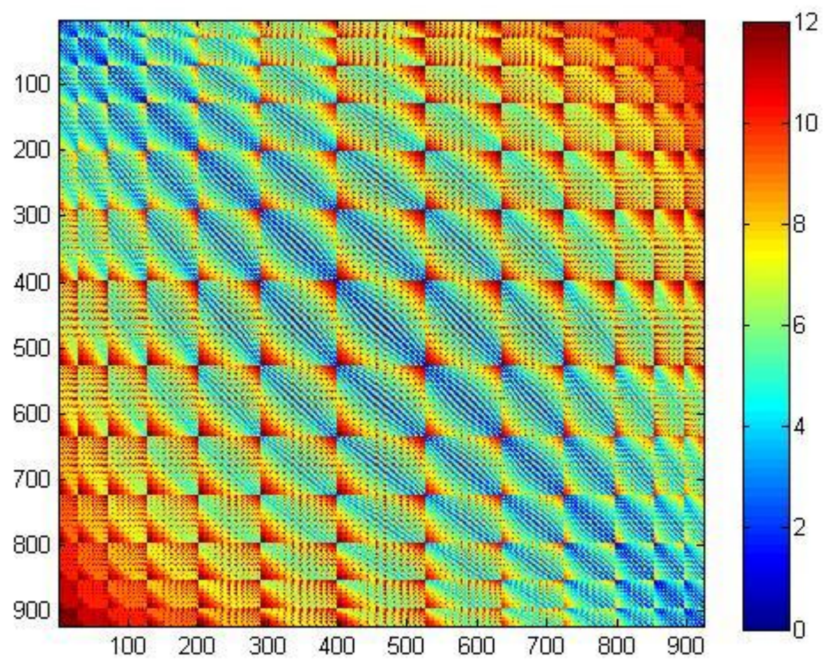


Figure 3.25. Supply matrix for 3D Hex, t=7, N=923.

Figure 3.23 through Figure 3.25 display representative supply matrices for the machines compared.

Applications are also represented as graph structures. Demand matrices are the all pairs distance matrices associated with these graphs, structures in which entry (i, j) corresponds to the total number of communication units sent from the processor corresponding to vertex i to the processor corresponding to vertex j. Here, communication units are bytes of data. Below, in the following figures, we present indicator matrices of the demand matrices corresponding to applications used to compare networks. A zero entry in the demand matrix corresponds to a white pixel in the image, whereas a non-zero entry generates a blue pixel.

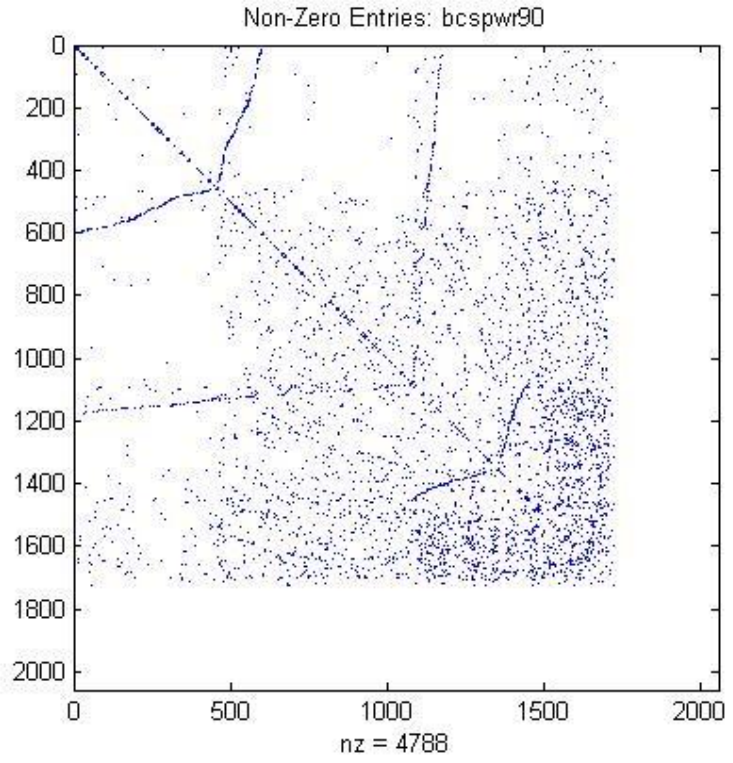


Figure 3.26. Indicator matrix of the demand matrix for BCSPWR90.

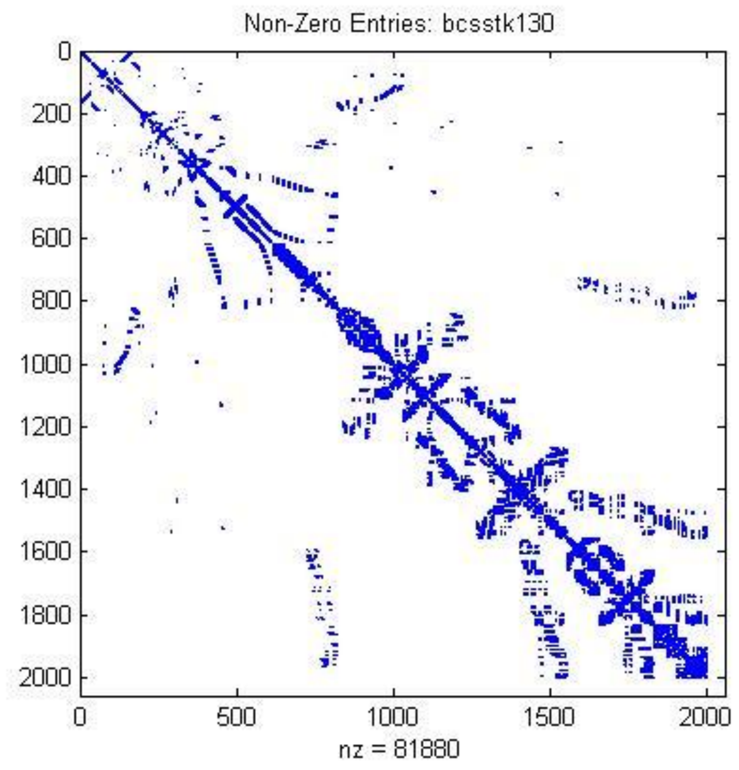


Figure 3.27. Indicator matrix for the demand matrix of BCSSTK130.

The demand matrices used in this study are from the Harwell-Boeing repository. They include:

1. YOUNG4C: Acoustic Scattering, highlighting bypass ring-like communication pattern
2. BCSPWR: Power network patterns with nearest neighbor and long distance communication
3. BSSTRUC: BCS Structural Engineering Matrices (eigenvalue matrices)
4. PSMIGR_3: Inter-county migration US inter-county migration 1965-1970; doubly stochastic; heaviest diagonals [±1, ±5]

3.3.2 Task Mapping Models

Within the task mapping model, the mapping vector, m, is a vector of length p, where entry m_i contains the label to which the task i is mapped. For example, in a rank order mapping (ROMap),

$$m = [p_{0,}p_{1,} \dots , p_{|P|-1}].$$

The objective function cost of a particular mapping m is given by

$$C = \sum_{t_i,t_j \in T} D_{i,j} \cdot S_{m_i,m_j}.$$

3.3.3 Mapping Heuristics

Here, we summarize the heuristics used to compare networks.

Simulated Annealing

Below, in Pseudocode 5, we present the simulated annealing algorithm used to calculate the objective function value of mappings.

```
SA-QAP($vars$)
1  $state \leftarrow state_0$                                ▷ Set the initial state
2  $energy \leftarrow$ SA-ComputeEnergy($state$)          ▷ Compute the energy of the initial state
3  $state_{best} \leftarrow state$                           ▷ Keep track of the current “best” state…
4  $energy_{best} \leftarrow energy$                         ▷ …and “best” energy.
5  while $k < k_{max}$ and $energy > energy_{max}$           ▷ While time remains and not good enough:
6      do
7          $state_{next} \leftarrow$ SA-Neighbor($state$)    ▷ Pick a neighbor…
8          $energy_{next} \leftarrow$ SA-ComputeEnergy($state_{next}$) ▷ …and compute its energy
9          if $energy_{next} < energy_{best}$                ▷ Is this a new best?
10           then
11               $state_{best} \leftarrow state_{next}$      ▷ If so, save the state…
12               $energy_{best} \leftarrow energy_{next}$    ▷ …and save the energy.
13         if SA-Prob$\left(energy, energy_{next}, \text{SA-Temp}\left(\frac{k}{k_{max}}\right)\right) >$ Random(0, 1)
                                                           ▷ Should we move to it?
14           then
15               $state \leftarrow state_{next}$             ▷ If so, change the state…
16               $energy \leftarrow energy_{next}$           ▷ …and change the energy.
17         $k \leftarrow k + 1$                              ▷ One more iteration is done.
18 return $state_{best}$                                     ▷ Return the “best” state found.
```

Pseudocode 5. Simulated Annealing algorithm.

Let the energy of the kth mapping be E_k. Additionally, $S^{(k)} = [M_{f_k(i),f_k(j)}]$ is the supply matrix corresponding to the kth mapping. Then,

$$\begin{aligned} E_{k+1} = E_k &- \sum_h \left(S_{p,h}^{(k)} D_{p,h} + S_{q,h}^{(k)} D_{q,h} + S_{h,p}^{(k)} D_{h,p} + S_{h,q}^{(k)} D_{h,q}\right) \\ &+ \sum_h \left(S_{p,h}^{(k+1)} D_{p,h} + S_{q,h}^{(k+1)} D_{q,h} + S_{h,p}^{(k+1)} D_{h,p} + S_{h,q}^{(k+1)} D_{h,q}\right) \\ &+ \sum_{\alpha,\beta \in \{p,q\}} S_{\alpha,\beta}^{(k)} D_{\alpha,\beta} - \sum_{\alpha,\beta \in \{p,q\}} S_{\alpha,\beta}^{(k+1)} D_{\alpha,\beta}. \end{aligned}$$

Furthermore, let $\Delta^{(k+1)} = S^{(k+1)} - S^{(k)}$. Then,

$$\begin{aligned} E_{k+1} = E_k &+ \sum_h \left(\Delta_{p,h}^{(k+1)} D_{p,h} + \Delta_{q,h}^{(k+1)} D_{q,h} + \Delta_{h,p}^{(k+1)} D_{h,p} + \Delta_{h,q}^{(k+1)} D_{h,q}\right) \\ &- \sum_{\alpha,\beta \in \{p,q\}} \Delta_{\alpha,\beta}^{(k+1)} D_{\alpha,\beta} \\ &= S^{(k+1)} : D \end{aligned}$$

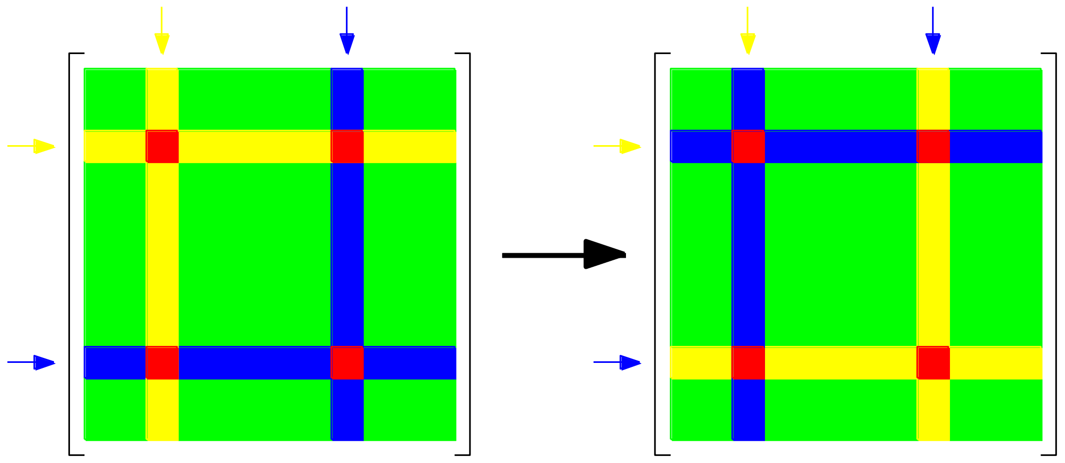


Figure 3.28. Venn diagram representing SA-Neighbor(s).

Genetic Algorithm

When mapping a set of p tasks to p processing elements, we are searching the population of $p!$ permutations of the set of tasks for a mapping permutation of minimal energy. The following pseudocode summarizes the genetic algorithm used here.

```
GA-QAP
$generation \leftarrow 0$, $currentbestfitness \leftarrow \infty$    ▷ Initialization
Generate initial (random?) sample of $\eta$ individuals, calculate their fitness
while $generation < g$ and $currentbestfitness > fitnessgoal$
    do
        Select the $\kappa$ best ranking individuals to reproduce
        Breed new generation through genetic operation, evaluate fitness
        Replace $\eta - \kappa$ worst ranking individuals with offspring
        $currentbestfitness \leftarrow$ best ranking individual's fitness
        $generation \leftarrow generation + 1$
```

Figure 3.29. Genetic algorithm pseudocode.

Crossover Operation

Given individuals (mappings) m_a and m_b, generate two new individuals m_a^* and m_b^* where

$$m_a^* = \left[m_{a_1}, m_{a_1}, \dots m_{a_{\lfloor \frac{p}{2} \rfloor}} | C(m_b) \right]$$

3.3.4 Quality of the Mapping Models

YOUNG4C

Machine	Dimension	Num Procs	ROMap	SA Best	% Impr.
so(7)	5	1051	0.8738722	0.4063812	53%
BG/L	8x8x16	1024	1	0.5722651	43%
QCDOC	2x2x2x4x4x8	1024	0.7062634	0.4993198	29%
su(4)	7	923	0.9030274	0.4631428	49%

BSSTRUC190

so(7)	5	1051	1	0.3342379	67%
BG/L	8x8x16	1024	0.769664	0.4890352	36%
QCDOC	2x2x2x4x4x8	1024	0.7839221	0.4029215	49%
su(4)	7	923	0.9734667	0.3876898	60%

662BUS

Machine	**Dimension**	**Num Procs**	**ROMap**	**SA Best**	**% Impr.**
so(7)	5	1051	0.8231054	0.2742948	67%
BG/L	8x8x16	1024	0.8289054	0.3795863	54%
QCDOC	2x2x2x4x4x8	1024	0.7865638	0.3213236	59%
su(4)	7	923	1	0.3038898	70%

BCSSTK130

so(7)	7	2703	0.8969505	0.4154848	54%
BG/L	16x8x16	2048	1	0.6110536	39%
QCDOC	2x2x2x4x4x16	2048	0.7970662	0.4960131	38%
su(4)	9	2057	0.9370016	0.4291801	54%

BCSSTH150

Machine	Dimension	Num Procs	ROMap	SA Best	% Impr.
so(7)	7	2703	0.6230841	0.3343774	46%
BG/L	32x8x16	4096	0.8999733	0.5423386	40%
BG/L	32x16x8	4096	1		
BG/L	16x8x32	4096	0.606313		
QCDOC	2x2x2x8x8x8	4096	0.5329424	0.3624994	32%

PSMIGR

so(7)	7	2703	0.6715247	0.5400175	20%
BG/L	32x8x16	4096	1	0.9269351	7%
BG/L	32x16x8	4096	0.9970267		
BG/L	16x8x32	4096	0.8043478		
QCDOC	2x2x2x8x8x8	4096	0.514379	0.4991226	3%

3.4 Summary Remarks

In all cases, ROMap had a lower energy value than the random sample. Sparse application matrices gave greater improvement over dense communication patterns. Finally, dimension priority ordering on BG/L has a drastic effect on objective function value. Figure 3.30 depicts the effect that a 15% run-time improvement will have on cost of computation for a standard queue job on Brookhaven National Labs' BlueGene machine.

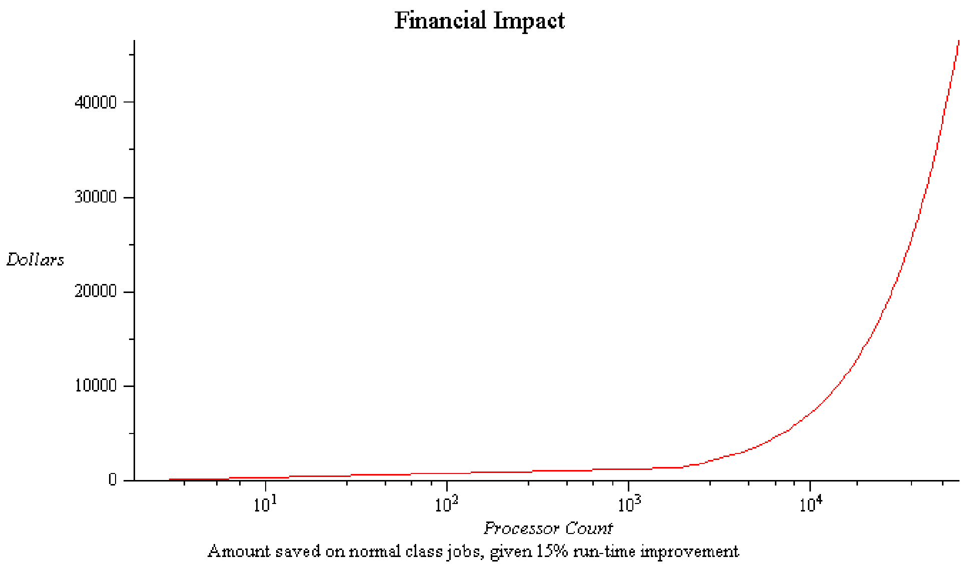


Figure 3.30. Financial Impact of task mapping.

Figure 3.30 depicts the savings given a 15% improvement in run-time versus the number of processors, given commercial use of BG/L at Brookhaven National Labs. For a 48-hour allocation of 100,000 nodes, nearly $50,000 is saved. As supercomputers move into exa-scale, efficient use of resources will save orders of magnitude more.

Chapter 4: Conclusions

Novel analysis of the vast amounts of data, such as the analyses presented here, would benefit a wide variety of groups: system integrators (e.g. Cray, HP, IBM) that assemble the system out of existing components. Their customers and the OEM vendors can influence them in deciding what components to use. This work, in turn, may influence their customers as they see a comparative analysis.

System manufacturers, like IBM, that build system from proprietary components would also benefit greatly from this work. The systems they create are targeted for specific markets and power efficiency is one of many constraints. This analysis highlights, among other features, that PowerPC with optimal networks can be energy-efficient. Other Off-the-Shelf “super” computer makers produce rather energy-draining platforms due to the convenience of off-the-shelf components.

Finally, users of supercomputing platforms will benefit greatly from this research. It quickly and concisely distills the most important features of the current supercomputing landscape.

Chapter 5: Future

The work presented here has many promising avenues of future research. With the recent release of several new versions of the Green500 list, it is clear that the Top500 list and the Green500 list are growing in popularity, and fostering a large community of individuals from diverse computing backgrounds. Development of a website that will display these analyses, allowing users to customize views, is underway using the prefuse flare visualization toolkit for Java (http://flare.prefuse.org) with the aim of integration with the Top500 website.

Currently, the analysis is relevant to many from the parallel computing community. However, additional analysis, such as partitioning of the Top500 supercomputers according to network media, like copper and fiber optic cable, would offer an alternate and more specific perspective on the performance and efficiency of supercomputer networks.

With respect to the iBT network, generalization of the iBT distance formula to higher dimensional networks will facilitate collective communication algorithms implementation of the iBT architecture. Ultimately, this work would result in the construction of an iBT system.

Finally, in the area of network comparison and evaluation, a higher degree of accuracy and efficiency will be pursued by improving solution methods. Namely, by increasing neighborhood radius and parallelizing the simulated annealing algorithm. Additionally, network models that account for features such as path diversity and application/network models that consider network contention are obvious but vast next steps.

List of References

1. Zhang, P., R. Powell, and Y. Deng, *Interlacing Bypass Rings to Torus Networks for More Efficient Networks.* IEEE Transactions on Parallel and Distributed Systems, 2010. **99**.

2. Powell, R. and Y. Deng, *Analysis of Power and Linpack Efficiencies of the World's Top 500 Supercomputers.* Parallel Computing, 2010(Submitted).

3. *Top 500 Supercomputer Sites*. 2010 [cited 2010; www.top500.org]. Available from: http://www.top500.org.

4. Duato, J., S. Yalamanchili, and N. Lionel, *Interconnection Networks: An Engineering Approach*. 2002: Morgan Kaufmann Publishers Inc. 650.

5. Adiga, N.R. *An Overview of the BlueGene/L Supercomputer*. 2002.

6. Adiga, N.R., et al., *Blue Gene/L torus interconnection network.* IBM J. Res. Dev., 2005. **49**(2): p. 265-276.

7. Anderson, E., et al., *Performance of the CRAY T3E multiprocessor*, in *Proceedings of the 1997 ACM/IEEE conference on Supercomputing (CDROM)*. 1997, ACM: San Jose, CA. p. 1-17.

8. Blumrich, M., et al., *Design and Analysis of the BlueGene/L Torus Interconnection Network*. 2003: Yorktown Heights.

9. Heidelberger, P., *Overview of the BlueGene/L Torus Network.*

10. Scott, S.L. and G.M. Thorson, *The Cray T3E Network: Adaptive Routing in a High Performance 3D Torus.*

11. Carle, J. and J.F. Myoupo. *Topological properties and optimal routing algorithms for three dimensional hexagonal networks*. in *High Performance Computing in the Asia-Pacific Region, 2000. Proceedings. The Fourth International Conference/Exhibition on*. 2000.

12. Decayeux, C. and D. Seme, *3D hexagonal network: modeling, topological properties, addressing scheme, and optimal routing algorithm.* Parallel and Distributed Systems, IEEE Transactions on, 2005. **16**(9): p. 875-884.

13. Deng, Y. and J.E.M. Hornos, *Group Theory Representation Insights for Supercomputer Design*. 2009: Stony Brook, NY.

14. Zhen Zhang, W.X.a.M.H., *Optimal Routing Algorithm and Diameter in Hexagonal Torus Networks*, in *Advanced Parallel Processing Technologies*, S.B. Heidelberg, Editor. 2007, Springer Berlin / Heidelberg: Berlin. p. 241-250.

15. Bhuyan, L.N. and D.P. Agrawal, *Generalized Hypercube and Hyperbus Structures for a Computer Network.* IEEE Trans. Comput., 1984. **33**(4): p. 323-333.

16. Efe, K., *A Variation on the Hypercube with Lower Diameter.* IEEE Transactions on Computers, 1991. **40**(11).

17. Gaughan, P.T. and S. Yalamanchili, *Adaptive routing protocols for hypercube interconnection networks.* Computer, 1993. **26**(5): p. 12-23.

18. Dally, W.J., *Express Cubes: Improving the Performance of k-ary n-cube Interconnection Networks.* IEEE Trans. Comput., 1991. **40**(9): p. 1016-1023.

19. Inoguchi, Y. and S. Horiguchi, *Shifted Recursive Torus Interconnection for High Performance Computing*, in *Proceedings of the High-Performance Computing on the Information Superhighway, HPC-Asia '97*. 1997, IEEE Computer Society.

20. Kirkman, W.W. and D. Quammen. *Packed exponential connections-a hierarchy of 2-D meshes*. in *Parallel Processing Symposium, 1991. Proceedings., Fifth International*. 1991.

21. Tao, L. *Practical Routing and Torus Assignment for RDT*. 2004.

22. Yang, Y., et al., *Recursive Diagonal Torus: An Interconnection Network for Massively Parallel Computers.* IEEE Trans. Parallel Distrib. Syst., 2001. **12**(7): p. 701-715.

23. Samatham, M.R. and D.K. Pradhan, *The de Bruijn Multiprocessor Network: A Versatile Parallel Processing and Sorting Network for VLSI.* IEEE Trans. Comput., 1989. **38**(4): p. 567-581.

24. Preparata, F.P. and J. Vuillemin, *The cube-connected cycles: a versatile network for parallel computation.* Commun. ACM, 1981. **24**(5): p. 300-309.

25. Harwood, A., Hong Shen, *A Low Cost Hybrid Fat-tree Interconnection Network*. 2003, unknown.

26. Chaki, N., et al., *A New Logical Topology Based on Barrel Shifter Network over an All Optical Network.*

27. Bland, A.S., et al. *Jaguar: The World's Most Powerful Computer*. in *CUG*. 2009.

28. Feng, W.-c. and T. Scogland, *The Green500 List: Year One*, in *23rd IEEE International Parallel & Distributed Processing Symposium (IPDPS)*. 2009.

29. Feng, W.-c., *Global Climate Warming? Yes ... in the Machine Room*, in *Clusters and Computational Grids for Scientific Computing*. 2006.

30. *Green 500*. 2010 [cited 2010 1/6/2010]; Available from: www.green500.org.

31. Meuer, H.W., *The TOP500 Project: Looking Back Over 15 Years of Supercomputing Experience.* Informatik-Spektrum, 2008. **31**(3): p. 203-222.

32. Feitelson, D.G., *On the Interpretation of Top500 Data.* Int. J. High Perform. Comput. Appl., 1999. **13**(2): p. 146-153.

33. Feitelson, D.G., *The Supercomputer Industry in Light of the Top500 Data.* Computing in Science and Engg., 2005. **7**(1): p. 42-47.

34. Dongarra, J., et al., *Biannual Top-500 Computer Lists Track Changing Environments For Scientific Computing.* SIAM News, 2000. **34**(9).

35. Feng, W.-C., *The Importance of Being Low Power in High-Performance Computing.* Cyberinfrastructure Technology Watch Quarterly, 2005. **1**(3).

36. Sharma, S., H. Chung-Hsing, and F. Wu-chun. *Making a case for a Green500 list*. in *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*. 2006.

37. Feng, W.-c., *Green Computing for a Clean Tomorrow*, in *Dean's Forum on Energy Security and Sustainability*. 2006.

38. Feng, W.-c. and K.W. Cameron, *The Green500 List: Encouraging Sustainable Supercomputing.* Computer, 2007. **40**(12): p. 50-55.

39. Feng, W.-c., F. Xizhou, and C. Rong, *Green Supercomputing Comes of Age.* IT Professional, 2008. **10**(1): p. 17-23.

40. Chung-Hsing, H. *Towards Efficient Supercomputing: A Quest for the Right Metric*. 2005.

41. Barker, K.J., et al., *Entering the petaflop era: the architecture and performance of Roadrunner*, in *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*. 2008, IEEE Press: Austin, Texas.

42. Harary, F., J.P. Hayes, and H.-J. Wu, *A survey of the theory of hypercube graphs.* Computers & Mathematics with Applications, 1988. **15**(4): p. 277-289.

43. Heun, V. and E.W. Mayr, *Efficient Embeddings into Hypercube-like Topologies.* The Computer Journal, 2003. **46**(6): p. 632-644.

44. Kaushal, R.P. and J.S. Bedi, *Comparison of hypercube, hypernet, and symmetric hypernet architectures.* SIGARCH Comput. Archit. News, 1992. **20**(5): p. 13-25.

45. Parhami, B. and D.-M. Kwai, *Incomplete k-ary n-cube and its derivatives.* J. Parallel Distrib. Comput., 2004. **64**(2): p. 183-190.

46. Stojmenovic, I., *Honeycomb Networks: Topological Properties and Communication Algorithms.* IEEE Transactions on Parallel and Distributed Systems, 1997. **8**: p. 1036-1042.

47. Jain, V.K. and S. Horiguchi, *VLSI considerations for TESH: a new hierarchical interconnection network for 3-D integration.* Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, 1998. **6**(3): p. 346-353.

48. Dally, W. and B. Towles, *Principles and Practices of Interconnection Networks*. 2003: Morgan Kaufmann Publishers Inc.

49. Dally, W.J., *Performance Analysis of k-ary n-cube Interconnection Networks.* IEEE Trans. Comput., 1990. **39**(6): p. 775-785.

50. Parhami, B., *Swapped interconnection networks: Topological, performance, and robustness attributes.* J. Parallel Distrib. Comput., 2005. **65**(11): p. 1443-1452.

51. Paiva, F. and Y. Deng, *iBT Network Analysis.* 2009 (In Progress).

52. Weisstein, E.W., *Hypercube Graph*, in *MathWorld--A Wolfram Web Resource*. 2009.

53. Wikipedia, *Hypercube --- Wikipedia, The Free Encyclopedia*. 2009.

54. Pemmaraju, S. and S. Skiena, *Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica* 2003, Cambridge: Cambridge University Press.

55. Tucker, A., *Applied Combinatorics*. 5th ed. 2007, Hoboken, NJ: John Wiley & Sons.

56. Maplesoft, *Maple 14*. 2010.

57. Weisstein, E.W. *Cube-Connected Cycle*. MathWorld--A Wolfram Web Resource [cited 2010.

58. de Bruijn, N.G., *A Combinatorial Problem.* Koninklijke Nederlandse Akademie v. Wetenschappen, 1946. **49**: p. 758-764.

59. Weisstein, E.W. *de Bruijn Sequence*. MathWorld--A Wolfram Web Resource [cited 2010.

60. Ahuja, R., T. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*. 1993: Prentice Hall.