Interconnection Networks in Petascale Computer Systems: A Survey

ROMAN TROBEC, Jožef Stefan Institute
RADIVOJE VASILJEVIĆ and MILO TOMAŠEVIĆ, University of Belgrade
VELJKO MILUTINOVIĆ, Maxeler Technologies and MISANU Belgrade
RAMON BEIVIDE, University of Cantabria
MATEO VALERO, Barcelona Supercomputing Center

This article provides background information about interconnection networks, an analysis of previous developments, and an overview of the state of the art. The main contribution of this article is to highlight the importance of the interpolation and extrapolation of technological changes and physical constraints in order to predict the optimum future interconnection network. The technological changes are related to three of the most important attributes of interconnection networks: topology, routing, and flow-control algorithms. On the other hand, the physical constraints, that is, port counts, number of communication nodes, and communication speed, determine the realistic properties of the network. We present the state-of-the-art technology for the most commonly used interconnection networks and some background related to often-used network topologies. The interconnection networks of the best-performing petascale parallel computers from past and present Top500 lists are analyzed. The lessons learned from this analysis indicate that computer networks need better performance in future exascale computers. Such an approach leads to the conclusion that a high-radix topology with optical connections for longer links is set to become the optimum interconnect for a number of relevant application domains.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Network topology, network communications*

General Terms: Design, Performance

Additional Key Words and Phrases: Interconnection networks, high performance parallel computers, exascale computers, Top500 list

ACM Reference Format:
Roman Trobec, Radivoje Vasiljević, Milo Tomašević, Veljko Milutinović, Ramon Beivide, and Mateo Valero. 2016. Interconnection networks in petascale computer systems: A survey. ACM Comput. Surv. 49, 3, Article 44 (September 2016), 24 pages.
DOI: http://dx.doi.org/10.1145/2983387

1. INTRODUCTION

Interconnection networks (ICNs) have always had important roles in cooperating computer systems. From the very first multiprocessors to the latest high-performance

This work is supported in part by the Slovenian Research Agency Grant No. P2-0095.
Authors' addresses: R. Trobec, Jožef Stefan Institute, Department of Communication Systems, Ljubljana, Slovenia; email: roman.trobec@ijs.si; R. Vasiljević, School of Electrical Engineering, University of Belgrade, Serbia; email: beholder@sezampro.rs; M. Tomašević, School of Electrical Engineering, University of Belgrade, Serbia; email: mvt@etf.rs; V. Milutinović, Maxeler Technologies London, United Kingdom and MISANU (Mathematical Institute of the Serbian Academy of Sciences and Arts) Belgrade, Serbia; email: vm@etf.rs; R. Beivide, University of Cantabria, Spain; email: ramon.beivide@unican.es; M. Valero, Barcelona Supercomputing Center and Technical University of Catalonia, Spain; email: mateo@ac.upc.edu.

DOI: http://dx.doi.org/10.1145/2983387

parallel computers (HPCs), the ICNs, along with the type of central processing unit (CPU), are the major hallmark of a system. In computing systems with an enormous number of cooperating computers, the ICN's performance is becoming more important than the performance of the CPU, because the execution time depends more and more on the communication time rather than on the calculation time [Grama et al. 2003]. The ICN very much determines the efficiency and scalability of a HPC on most real-world parallel applications. It can shorten the overall execution time and increase the number of processors that can be efficiently exploited, both of which lead to a higher ultimate speedup.

The performance of an ICN depends on many factors, technological and topological, but among the most important are the network topology, the routing, and the flow-control algorithms, which represent the focus of this survey. While the routing and flow-control algorithms have advanced to a state where efficient techniques are known and used, we have not reached an equal level of sophistication in the development of topologies [Dally and Towles 2004; Duato et al. 2002]. From the viewpoint of this survey, a lack of sophistication means a lack of constant monitoring of the changes in technological trends, followed by proper adjusting of the ICN topologies to the changed values of the three exposed focal attributes.

In fact, many topologies already present at the dawn of parallel computing are still being widely applied. With modern standards like Infiniband, vendors and end users do not have the capability to alter the routing and flow-control. However, the end users do have the freedom to define a number of different topologies, based on an anticipated usage. A further step in performance increase is made possible by an improved ICN topology or by innovative technological approaches in optical networking [Benner et al. 2005], which could solve the current ICN bottlenecks, that is, message latency and non-efficient collective communications. It is believed that ICNs will be able to adapt dynamically to the current application in some optimal way in the near future.

This survey of existing interconnection networks for the highest-performance supercomputers does expose some of the weaknesses in existing networks that may create scalability problems, thus paving the way for improvements and innovations in future exascale systems. The results of this work are also applicable to future highest-performance, data-flow systems [Flynn et al. 2013]. The survey attempts to analyze the topologies that have been employed most often in the most powerful systems from the Top500 list [Top500 2015], which ranks systems according to their performance results obtained from the High Performance Linpack (HPL) benchmark [Dongarra et al. 2003]. Regardless of the large number of ICN topologies proposed in the literature, based mostly on their graph properties, this survey is focused on the ICN topologies that are currently used in petaflop (PFLOP) HPCs. First, it examines the trends in the ICNs of the most powerful systems from the Top500 lists over a time period that spans the past two decades. Then, it concentrates on the ICNs from the present top-level systems. Finally, based on experiences from past and present, and the technology trends of the present and future, it tries to establish some proposals for future ICNs that are expected to better fit the needs of exascale HPC systems [Coteus et al. 2011].

2. BACKGROUND AND STATE OF THE ART

ICNs can be classified in many ways and characterized based on various parameters. Graph theory provides an elegant mathematical framework for a number of ICN parameters. However, for further analyses and extrapolation to future ICN topologies, some additional factors, like physical constraints, should also be considered. Examples were intentionally made, self-contained, and simple, to include only the issues of relevance for the extrapolation process introduced in this article. The most important topological ICN properties are node degree, diameter, regularity, symmetry, and path diversity. The main performance properties are channel bandwidth, bisection

bandwidth, and latency. For expert technical readers, ICNs are discussed in great detail in many reference books, for example, Dally and Towles [2004] and Duato et al. [2002].

2.1. Parameters of ICNs

An ICN can be modeled as a graph $G(N, C)$, where N is a set of communication nodes and C is a set of channels between the nodes [Dally and Towles 2004]. The node degree d (or radix) is the number of channels through which it is connected to other nodes. Usually, only the ports for the network communication are counted, because their number is both essential and topology dependent. Moreover, a communication node needs additional channel(s) for the connection to the processing element(s), for example, service or maintenance channels, that are not shown in the figures and not explored in detail in this survey. Many real implementations of ICNs are based on symmetric regular graphs because of their fruitful topological properties that lead to a simple routing and fair load balancing under the uniform traffic.

Bandwidth is the amount of data that can be transferred in a certain amount of time. When an ICN is cut into equal-sized parts, the bandwidth across the worst-case cut is called the bisection bandwidth (BBW); when normalized with nodes, it is the bisection bandwidth per node (BBWN). Both depend on the topology and the corresponding channel bandwidth. Moving a packet from one node to another requires going through some elements (routers or switches), and this is a path, while the number of communication nodes traversed is the hop count. In the best case, two nodes communicate through the minimum path that has the minimum hop count (l). However, because the hop count varies with the source and destination nodes, it can be averaged over all possible combinations (l_{avg}). An important characteristic of any topology is the diameter, defined as the maximum of all the minimum hop counts (l_{max}).

Latency is the time required for a packet to travel from the source node to the destination node. Many applications (especially with short messages) are latency sensitive, such as climate modeling [Kerbyson and Jones 2005], finite-element analysis [Shainer et al. 2009], and so on. While the bandwidth of an individual channel can be increased, theoretically without limits, the latency is ultimately bounded by the velocity of light and the physical distances between the nodes. From the application point of view, the software overhead also adds a considerable amount of latency [Buh et al. 2014].

The data-transfer units that are closely related to the bandwidth and to the latency are the packet, the FLow control digIT (FLIT), and the PHysical digIT (PHIT). The packet is the smallest amount of data that can be transferred by hardware. The FLIT is the amount of data used to allocate the buffer space in some flow-control techniques. The packet consists of one or more FLITs. The PHIT is the amount of data that can be transferred during a single cycle, for example, it is the width (in bits) of a link.

Any given topology, even in higher-dimensional space, has to be mapped to the three-dimensional (3D) world, starting from chips, through printed-circuit boards (PCBs), and, finally, placed into cabinets. Chips and PCBs have some limits on the number of Input/Output (I/O) pins. The length of the cables must be kept to a minimum as the data rate over copper wires decreases with the square of the length. Nodes that are close in higher-dimensional space may be relatively distant in the 3D world. The ICN chips are often not in the center of power–aware considerations, but their power dissipation is frequently comparable to that of processors [Mellanox 2013]. Increasing the ICN bandwidth can bring the same benefit as increasing the CPU clock; however, both approaches are limited by a dramatic increase in the power consumption.

2.2. Classification of ICNs

ICNs can be classified into direct and indirect networks [Dally and Towles 2004]. Hybrid approaches are also found in some cases [Peñaranda et al. 2016]. A network is direct

when a node is directly connected to its neighbors. For example, a fully connected network has direct links between any two nodes. Since the fully connected direct network has the complexity of $O(N^2)$, where N is the number of nodes, it cannot be used for building large systems. Usually, a node is only connected directly to a subset of other nodes, while the communication to the remaining nodes is achieved by routing messages through intermediate nodes (e.g., mesh, hypercube).

An indirect network connects the nodes through one or more levels of switches. An ideal switch is the fully connected crossbar. The crossbar enables a connection from any input to any output port if the output port is not already in use. Large crossbars are not a feasible solution, because of a number of related problems, mainly because of its $O(N^2)$ complexity. However, smaller crossbars are used inside routers and switches as basic building blocks. The switches are usually organized into stages using a regular connection pattern between the stages (multistage ICNs).

Since a number of smaller switches are used, the resulting indirect network can be blocking, blocking rearrangeable, or non-blocking [Dally and Towles 2004]. Because of conflicts, a blocking network cannot provide a connection between a source and an arbitrary destination node, even if the destination is free. In blocking rearrangeable networks, existing connections can be rearranged in such a way that a new connection can be established. Non-blocking networks can connect any source to any destination, as long as the destination is not busy, due to there being multiple paths between them [Dally and Towles 2004].

The distinction between direct and indirect networks is less clear nowadays. Every direct network can be represented as an indirect network since every node in the direct network can be represented as a router with its own processing element connected to other routers. However, for both direct and indirect ICNs, the full crossbar, as an ideal switch, is the heart of the communications.

2.3. ICNs Most Often Used in Top500 Supercomputers

Nowadays, the most widely used network topologies in larger systems are direct k-ary n-cubes and indirect fat-trees. We consider only the basic type for each topology, while its variations are commented on for a particular system.

2.3.1. k-Ary n-Cube. The multidimensional k-ary n-cube topologies can be treated as generalized regular meshes. The parameters k and n represent the number of nodes along each dimension and the spatial dimension, respectively. A k-ary n-cube can be constructed from k-times k-ary (n-1)-cubes by connecting the nodes with identical positions into rings [Grama et al. 2003]. N nodes can be connected in a ring (N-ary 1-cube), toroidal 4-mesh ($\sqrt[2]{N}$-ary 2-cube), 3D torus ($\sqrt[3]{N}$-ary 3-cube), or hypercube (2-ary d-cube).

The number of nodes connected in a k-ary n-cube is $N = k^n$ if k is the same in all dimensions, while the diameter is $l_{\max} = n(k/2)$ and the average distance $l_{avg} = l_{\max}/2$ for even k and $l_{avg} = n(k/4-1/(4k))$ for odd k [Dally and Towles 2004]. The number of bidirectional links in a k-ary n-cube is $L = dN = nk^n$, where d is the degree. The k-ary n-cubes have a simple recursive structure but a poor expansion scalability, in particular if k is the same in all dimensions.

(a) Torus and mesh. The torus and mesh are n–dimensional grids with k nodes in each dimension and a total number of nodes $N = k^n$. If the border links in the grid wrap around, then they form a torus. The bisection bandwidth of a 3D torus with N nodes is $O(N^{2/3})$. The folding of the torus makes the link lengths among all the nodes equal, but this uniform length is twice as long as the original short link length [Dally and Towles 2004]. The use of optical links would, however, overcome this problem. For different practical reasons, 2D and 3D tori with different numbers of nodes per dimension have

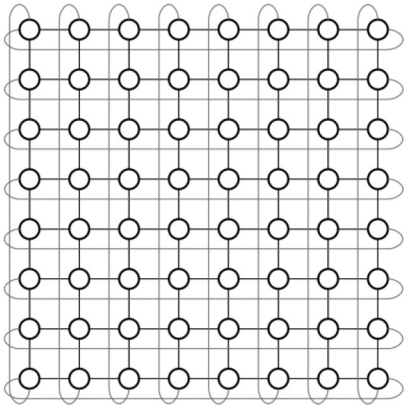

Fig. 1. 2D torus (8-ary 2-cube) topology. Only adjacent nodes have links in between, and all the nodes have the same fixed degree. A network of arbitrary size can be built using the same node design. Circles, communication nodes; curves or lines, communication links.

been used, which degrades the performance. As shown [Camara et al. 2010], twisted, mixed-radix tori can eliminate the network bottlenecks and recover the edge symmetry that can balance the links' utilization.

The popularity of the 3D torus has several reasons. It has excellent packaging properties with uniformly short links, for both the logical and the physical distances. Another advantage of the 3D torus is that many scientific problems are three dimensional in nature and a large portion of the communication is between adjacent nodes. Finally, the node degree is independent of the number of nodes. Figure 1 presents the structure of a 2D torus built from $N = 8 \times 8$ communication nodes with $d = 4$. Higher dimensions of a torus (Hyper-D torus) are hard to visualize, but each higher dimension contributes a new pair of connections to each node, resulting, for example, in $d = 10$ for a 5D torus.

(b) *d*-mesh. The performances of meshes can be improved by an increased node degree. An interesting example is the d-mesh ICN [Trobec 2000] that fits most of the requirements of the present-day technology. The basic idea is to use a classic mesh-based topology and add additional diagonal links in order to shorten the network diameter but still preserve the existing advantages of the mesh-based ICNs. Let the topology of a d-mesh consist of $N = X \times Y$ nodes. The nodes are placed in Y rows, each of length X, and they can thus be uniquely represented by coordinate pairs (x, y), $1 \leqslant x \leqslant X, 1 \leqslant y \leqslant Y$ with the node (x, y) located in the xth column and the yth row. For simplicity, we restrict ourselves to the toroidal d-meshes with symmetrical links, that is, if a link to a neighboring node with coordinates $(x + a, y + b)$ is chosen, then the symmetric link to the node with coordinates $(x - a, y - b)$ must also be used.

The properties of d-meshes depend on the applied link-adding technique. For example, to shorten the diameter, a few longer links can be used [Arden and Lee 1982]. The optimum d-meshes can be defined as those having the minimum sum of their diameter $l_{\max}$ and the average distance l_{avg} [Trobec 2009]. An illustrative example for an optimal toroidal 6-mesh in a system with $N = 64$ nodes is given in the left-hand part of Figure 2 with $l_{\max} = 4$ and $l_{\text{avg}} = 2.69$. The 1-neighborhood of a node (x, y) in this 6-mesh is $(x \pm 1, y)$, $(x, y \pm 1)$, $(x \pm 3, y \pm 2)$, for example, for node (5,5), it is denoted by dark-gray filled nodes connected with bold links. In the right-hand part of Figure 2 just the toroidal links are shown. Note that d-meshes with a higher radix can be easily constructed, which can significantly decrease the ICN diameter, for example, the 16-mesh has $l_{\max} = 2$ and $l_{\text{avg}} = 1.72$.

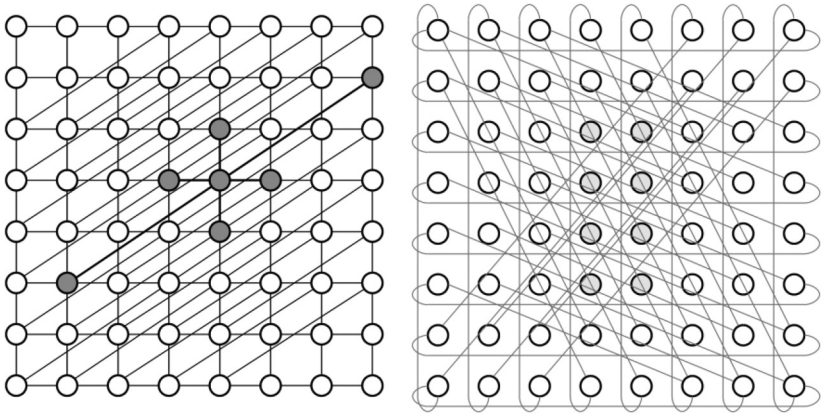

Fig. 2. Optimum toroidal 6-mesh for $N = 64$ with a basic 4-mesh and two additional links per node (left). Dark-gray nodes are directly connected 1-neighborhood of node (5,5). Toroidal links are shown separately (right). Circles, communication nodes; curves or lines, communication links.

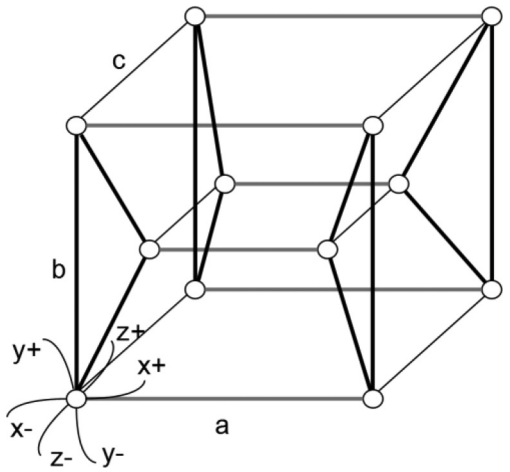

Fig. 3. Building block of a K computer: local 3D mesh/torus with 12 communication nodes. Circles, communication nodes; lines, communication links; a, b, c, dimensions inside the local block; x±, y±, z± – links for systemwide 3D torus.

(c) Tofu. The Tofu ICN was introduced by the Fujitsu K computer as a customized version of a six-dimensional mesh/torus. The basic building block of the ICN is composed of 12 nodes that are connected in a small 3D mesh/torus (see Figure 3) made from two 1D meshes that are just pairs of connected nodes, marked with black and bold gray lines for the dimensions c and a, respectively. 1D meshes are connected in the third dimension b with four 3-node rings marked with bold black lines. Such a choice balances the number of links (and hence the link width) with a fault-tolerance ability. The local 3D meshes/tori are connected in a systemwide 3D torus. The routers have 10 ports: four ports for the local 3D mesh/torus and six ports for the systemwide 3D torus (x±, y±, z±, curved lines, drawn on a single node only for clarity).

(d) Hypercube. A hypercube is another gridlike topology. Instead of changing the number of nodes in a dimension, the number is kept fixed ($k = 2$) and the number of dimensions varies. A 1D hypercube is just a pair of two connected nodes, a 2D hypercube is a square with four nodes, a 3D hypercube is a cube, and so on (Figure 4). Higher-dimensional hypercubes can be constructed recursively by connecting two lower-order

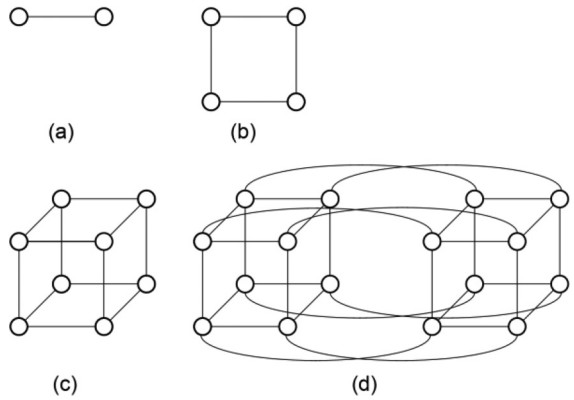

Fig. 4. Hypercubes in up to four dimensions (a) 2-ary 1-cube, (b) 2-ary 2-cube, (c) 2-ary 3-cube, and (d) 2-ary 4-cube. The BBWN is constant. Each added dimension doubles the number of nodes, as well as the number of links. The maximum distance grows as the logarithm of N. Circles, communication nodes; curves or lines, communication links.

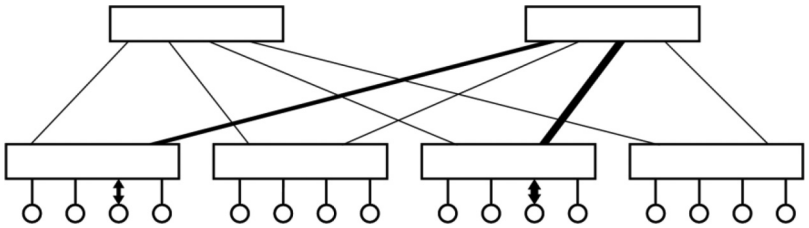

Fig. 5. Two-level fat-tree topology built with crossbars having 6 and 4 ports in the lower and upper levels, respectively. With an appropriate number of links between the switches, any bisection bandwidth can be achieved. Circles, nodes; rectangles, communication switches; lines and arrows, bidirectional communication links.

hypercubes. When adding another dimension, the number of nodes is doubled, and the number of links is doubled; consequently, the degree of all nodes has to be increased. An n–dimensional hypercube consists of $N = 2^n$ nodes, each connected to d other nodes with $l_{\max} = \log_2 n = d$ and $l_{\text{avg}} = d/2$. Since the number of nodes and the bisection bandwidth are $O(2^n)$, the BBWN is constant. Before the wormhole flow-control was invented [Dally and Seitz 1987], a great advantage of the hypercube over the 3D torus was its lower hop count and, consequently, a lower latency, $O(\log n)$ vs. $O(N^{1/3})$. With wormhole flow-control the torus has a comparable latency, assuming no other load in the network. However, the hypercube still has the advantage of a perfect bandwidth scaling.

2.3.2. Fat-Tree/Clos. The fat-tree/Clos is an indirect topology with two names and can be drawn in two very distinct, but isomorphic, ways. Consider a tree of switches with the processing nodes as leaves, all connected with links of the same speed. As the communication progresses from the leaves to the root, it is clear that the links closer to the root will become a bottleneck. To remedy this problem, the links closer to the root should be faster (higher clock, multiple links, wider or "fatter") [Leiserson 1985]. A drawback of such a technique is that a very fast switch is needed for the root node. Consider the case with multiple links between two levels of switches. Instead of multiple links from the lower- to the higher-level switch, the links can be distributed along higher-level switches (Figure 5). It might, however, look like there are no gains, as the number of higher-level switches is the same as the number of links to the lower-level switches. Nevertheless, the resulting tree has multiple roots, so there is no need for an

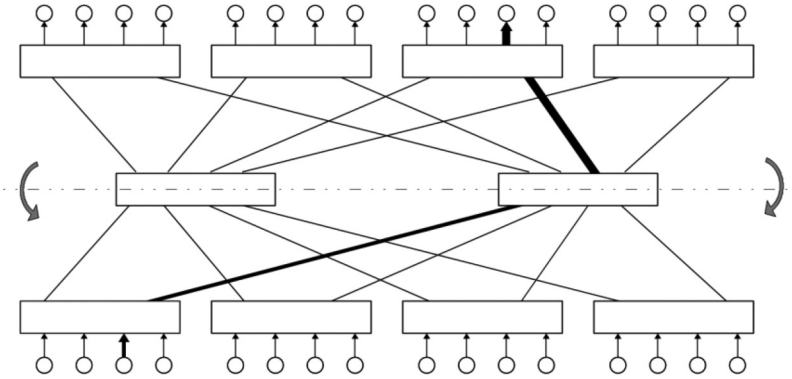

Fig. 6. Clos network $(m, n, r) = (2, 4, 4)$ with $N = 16$ nodes. Bottom nodes are message sources and upper nodes are message destinations. Circles, nodes; rectangles, communication switches; lines and arrows, unidirectional communication links.

even faster root switch. The level of the tree is the choice of the user or the designer and depends on a number of parameters. In practice, the fat-tree level is usually from 2 to 4.

Now consider a more abstract network named Clos [Clos 1953], characterized by a triplet (m, n, r). It is composed of a number of switches, arranged into three stages: input, middle, and output (Figure 6). The input stage consists of r switches, each being an $n \times m$ crossbar. The middle stage consists of m switches, each being an $r \times r$ crossbar. The output stage consists again of r switches, each one being an $m \times n$ crossbar. A network with any number of odd stages is composed recursively, that is, the middle stage is replaced with a three-level Clos network. One of the most important properties of this topology is that it remains non-blocking with a sub-quadratic number of switches if $m \geq 2n$-1 and rearrangeable non-blocking for $m = n$ (see Section 2.2 for a description of blocking and non-blocking networks).

Practical folded Clos/fat-tree networks are usually built to be rearrangeable because the strictly non–blocking property doubles the number of links. Also, non–blocking networks block in practice, because finding a free path in the network requires information about the complete network. This information is not available as each switch has to make decisions based on local information only [Hoefler et al. 2008]. In most cases, when a network or switch is called non-blocking, it is actually rearrangeable blocking.

Although Figures 5 and 6 look very different, in fact this is the same network with $N = 16$ nodes. The flow of a packet in the fat-tree goes up to the first switch that is shared by the source and destination nodes (bolder lines) and down, until it reaches the destination node (the boldest lines). The links between the nodes and switches are bidirectional. In the Clos network, the packet goes from the bottom switches at the source nodes, through the middle switches, and then to the upper switches at the destination nodes. If we consider that the links are unidirectional with nodes and switches composed of a source and a destination part, and "fold" the Clos network across the middle switches, then it is intuitively clear that the path from the input to the middle stage in the Clos network is the same as going up to the root in the fat-tree, while the path from the middle to the output stage is the same as from the root to the leaves of the fat-tree.

The fat-tree with k levels is equivalent to a Clos network with $2k$-1 stages. However, it is not possible to establish a straightforward correspondence between the parameters of the Clos and fat-tree networks. The fat-tree can have any number of levels, but with a fixed number of ports towards higher and lower levels, while the basic Clos networks have only three stages. On the other hand, a Clos network with more than three stages can have arbitrary parameters for the inner stages. So, an equivalent fat-tree would

Table I. l_{max}, l_{avg}, BBW and BBWN for k-Ary n-Cubes, Fat-Tree, and Dragonfly with Specified ICN Configuration and Radix for $N = 4096$ Nodes

ICN	ICN configuration	radix	l_{avg}	l_{max}	BBW	BBWN
2-D torus	64×64	4	32	64	128	32
32-mesh	64×64	32	2.99	4	1024	16
3-D torus	$16 \times 16 \times 16$	6	12	24	512	85.3
BGQ (5-D)	$4 \times 8 \times 8 \times 8 \times 2$	10	7.5	15	1024	102.4
Tofu (6-D)	$7 \times 7 \times 7 \times 2 \times 3 \times 2$	10	6.33	12	147	14.7
Hypercube (12-D)	$2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$	12	6	12	2048	170.7
Fat-tree	3 level	32	4.87	5	4096	128
Dragonfly	$4 \times 96 \times 11$	48	3.69	5	289	20

not have the same bandwidth ratio between the levels, and even blocking/non-blocking characteristics could differ from level to level. The main advantage of the fat-tree is that any BBW can be achieved with an appropriate arrangement of the switches.

2.3.3. Dragonfly. To achieve a high interconnection scalability with the minimum number of expensive global links, the Dragonfly topology was proposed in Kim et al. [2008], which effectively uses high-radix routers. In order to achieve this goal, not just at a graph level but also in cost-effective practical implementations, the Dragonfly departs from established topologies.

The Dragonfly is built as a hierarchical network of routers at the group and system levels. Each router is connected to p computing nodes, a-1 routers from the same group, and h global links between groups, resulting in a router radix $k = p + a + h$-1. A group is formed from a routers connected via an intra-group ICN using local links only, resulting in ap group nodes and ah global links. From the perspective of global links, a group acts as a virtual router with a radix $k' = a(p + h)$. To secure the full utilization of the expensive global links, the above parameters should satisfy $a \geq 2h$, $2p \geq 2h$.

Intra-group and inter-group topologies can be selected individually, but for practical reasons, the intra-group topology should be a high radix, putting a premium on exploiting the dense packaging, while inter-group topologies should be fully connected. A few inter-group optical multilane links provide a fast data transfer between groups, resulting in a network of almost 10^5 computing nodes with the maximum distance being just five hops—four within the source and destination groups and one between the groups.

The Dragonfly has good path diversity, but to effectively exploit it, advanced routing algorithms are needed. For example, if a source router does not have a direct global link to the destination router, then the communication traffic could become unbalanced [Fuentes et al. 2015]. To balance it, some adaptation is needed, possibly obtained by indirect adaptive routing. This fact makes the Infiniband ineffective in the Dragonfly because the current Infiniband employs just static routing approaches.

To compare the described topologies, the results for l_{max} and l_{avg}, in the number of hops, and BBW and BBWN, in the link bandwidth, are listed in Table I for k-ary n-cubes, fat-tree, Dragonfly, and d-mesh. The configuration of each topology is given in a separate column, that is, the number of nodes in each dimension for k-ary n-cubes; the number of levels and the type of switches for fat-tree; and the number of nodes, routers, and groups for Dragonfly.

Table II provides the average distances for the different numbers of nodes. The results were obtained by using the closest possible set of parameters, for each topology and for each system size, that maintain the important metrics, for example, distances and BBWs near optimal. Since all the considered topologies are isomorphic, the distances were found by simply counting the hops from a selected node towards all the rest of the nodes in the ICN (for fat-tree and Dragonfly), by using given formulas (for

Table II. Average Distances for Different Numbers of Nodes and Near-Optimum ICN Parameters

Nodes	32-mesh	3D torus	BGQ (5D)	Tofu (6D)	Hypercube	Fat-tree(R32)	Fat-tree(R64)	Dragonfly
1,024	2.47	7.50	6.00	6.53	5.00	4.46 3L	2.94 2L	3.50
2,048	2.74	9.46	7.00	7.48	5.50	4.83 3L	2.97 2L	3.62
4,096	2.99	12.00	8.00	9.21	6.00	4.87 3L	4.48 3L	3.69
8,192	3.37	15.00	9.00	11.21	6.50	4.93 3L	4.74 3L	3.72
16,384	3.66	19.00	11.00	10.43	7.00	6.26 4L	4.87 3L	3.73
32,768	3.97	24.00	13.00	16.21	7.50	6.73 4L	4.94 3L	3.74
65,536	4.31	30.23	15.00	20.54	8.00	6.86 4L	4.97 3L	3.75
131,072	4.65	37.99	17.00	24.37	8.50	6.97 4L	6.48 4L	3.75
262,144	5.03	48.00	21.00	30.24	9.00	8.47 5L	6.74 4L	3.75

Note: R stands for radix and L stands for fat-tree level.

mesh/torus topologies), or by using a custom optimization program for d-meshes. For modified topologies of Blue Gene/Q and K-computer, the physical constraints of the original systems are preserved so the systems are not expanded in the "smallest" local dimensions. For fat-trees, radix 32 is chosen for two reasons: It is close to the contemporary Infiniband radix 36 and it simplifies the construction of the power of two size systems, while radix 64 is used to show the benefits of an increased radix. Furthermore, in fat-trees all the switches are rearrangeable non-blocking and the complete systems have full BBWs, that is, no bandwidth tapering toward the last level of the switches. The Dragonfly radix was like in fat-trees and was selected for an ease of comparison with other systems.

It is clear that the high-radix ICNs achieve better results. The Dragonfly and fat-tree are winners, however, on account of the additional and expensive communication links or switches. d-meshes, on the other hand, remains fully regular with isomorphic communication nodes and communication links.

3. PAST: LESSONS LEARNED

Looking at the Top500 list can give us an impression of how the trends in employing ICNs in the most powerful HPCs were changing over the past two decades. In the early days of HPCs, when vector-based systems with a small number of powerful CPUs dominated the Top500 list, there was a clear distinction—the HPCs employed scalable networks and the vector systems were crossbar based. While the best performance could be achieved with a small-to-modest number of fast CPUs, the conceptually simple and very efficient crossbar was an appropriate choice. Over time the speed of the CPU increased as well as its total number, resulting in a tremendous rise in the computational speed of the entire system. The ICNs followed a similar trend in bandwidth, while the latency settled at around 1μs. Crossbars are not acceptable with a large number of ports and such systems slowly disappeared from the Top500 list.

From an analysis of previous Top500 lists of supercomputers it follows that the average growth in HPC system performance has started to flatten out since 2013. This trend was detected even earlier, in 2008, for the top-of-the-line supercomputers that operate with edge technologies. To some extent, this was compensated for by accelerator/co-processor technologies and by advanced ICNs that are in the topic of this article. The most common communication technologies in the Top500 systems are Infiniband (46%) and 10G Ethernet (36%). As a consequence, the systems with a fat-tree topology are much more numerous in the Top500 list, mainly due to systems based on Infiniband.

The Top10 supercomputers of the past two decades are followed in the rest of this section because they are changing relatively quickly compared to the Top500 lists, demonstrating the highest diversity of custom interconnect technologies and original

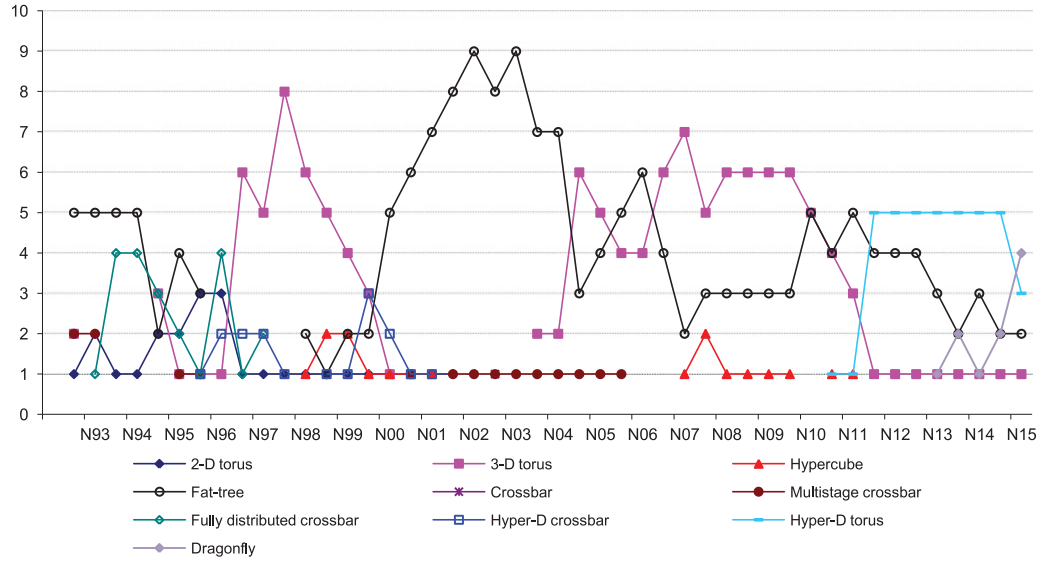

Fig. 7. Number of ICNs of a particular type in the Top10 of the Top500 lists. Legend: y-axis, number of ICNs, x-axis, June 1993 to November 2015 (one point per half a year).

ICN topologies that could become candidates for the exascale era [Coteus et al. 2011]. In addition, they represent the huge investments in advancing solutions that will serve in the extrapolation to future developments and approaches. Figure 7 presents the number of ICNs of a particular type in the Top10 of Top500 supercomputers from 1993 (when this list first appeared) to November 2015.

In the early 1990s, the Top10 was shared between early massive parallel systems from Thinking Machines and Meiko and parallel vector systems from Cray, NEC, Fujitsu, and Hitachi, with a prevailing use of the fat-tree topology. The systems built by Cray and Japanese vendors with various crossbar architectures dominated in the middle 1990s. At the same time, the Cray T3D was the only HPC in the Top10 with the 3D torus topology. In the late 1990s, the 3D torus filled a number of places in the Top10, mainly because of the Cray T3E systems.

At the start of the new millennium, a noticeable change in the Top10 occurred. With the Advanced Simulation and Computing (ASC) initiative [NNSA 2013], a swing back to the fat-tree topology is evident, led by the IBM Scalable POWERparallel based systems and followed by Compaq. The hypercube could only be found in the SGI systems, while the Intel systems used the 2D torus. This can be partially explained by the new trend of using workstation-based nodes and the newly emerged Message Passing Interface (MPI) standard. The fat-tree offered a clear advantage: Each node was connected to one switch, with switches further interconnected with each other in the desired way, thus greatly decoupling the architectures of the nodes and the switches from each other. At that time, the cluster technology matured, especially due to the introduction of the Myrinet and Quadrics cards and switches, which were much faster and had a significantly lower latency than Ethernet. They also used the fat-tree topology, mainly with two or three levels. The first systems that used Quadrics appeared in the Top10 in November 2001, while the first system that used Myrinet appeared a year later. Meanwhile, the Infiniband specification emerged and vendors started to market the appropriate cards and switches. Although Infiniband can support different topologies, the fat-tree is almost exclusively used in practice. The first system with Infiniband appeared in the Top10 in November 2003.

From the middle of the past decade, the Top10 changed significantly with the introduction of the IBM BlueGene/L and the Cray RedStorm. The 3D torus-based systems started to gain ground. Although not well suited for connecting workstation-based nodes, the 3D torus was ideal for a tightly integrated system, which is best illustrated by Blue Gene/L. The 3D torus and fat-tree remained in approximately equal numbers in the Top10 until November 2007, after which the 3D torus gained a clear lead, mainly because of the Cray XT systems (several of them being constantly upgraded) and the IBM Blue Gene/L and Blue Gene/P systems. Until the June 2011 list update, the 3D torus-based systems prevailed in the Top10, but now fat-trees seem to be becoming more popular again. Stimulated by an increased number of CPUs today, new interconnection approaches appear, for example, hyper-D torus and Dragonfly, which aim to optimize the price/performance ratio.

Besides the topology, the flow-control and routing techniques implemented in ICNs proved to be very important for providing maximum bandwidth and minimum latency. With the increased number of nodes in the petascale era, the reliability has become more important, and all topologies are required to tolerate the faults of individual nodes without any significant performance degradation.

4. PRESENT

4.1. Detailed Overview of ICNs in the Top10 PFLOP Computers

In the November 2008 Top500 list update, the IBM BladeCenter, nicknamed Roadrunner, was the first system to achieve more than 1 PFLOP on the HPL, not just by increasing the number of CPUs but also by introducing a number of unique features [Barker et al. 2008]. The nodes consist of two dual-core Opteron CPUs and four PowerXCell 8i, resulting in 40 cores per node. Most of the floating-point performance was provided by the PowerXCells, announcing the future manycore trends. The large number of cores per node gave a relatively modest number (3,060) of high-performance nodes, thus enabling a four-level fat-tree topology. Soon after the PFLOP era, the race began towards exaflops (EFLOPs), first solely by an increased number of nodes and computing accelerators but later, because of the enormous power consumption, also by improved ICNs. In the remaining part of this section we focus in more detail on the state-of-the-art ICN technology and implementation.

Tianhe-2. For more than 3 years Tianhe-2 has occupied first place in the Top500 list from November 2015 with 33.9 PFLOPs in the HPL benchmark [Dongarra and Heroux 2013]. Most of its power comes from the enormous number of cores (3,120,000), approximately 5.5 times more than the Cray XK7 Titan, which is ranked in second place. Its hybrid design utilizes Intel Xeon CPUs and Intel Xeon Phi accelerators. The system consists of 16,000 nodes, with each node having two Xeons (with 12 cores and 422.4 GFLOP peak performance) and 3 Xeon Phis (with 57 cores and 3009.6 GFLOP peak performance).

The ICN called TH Express-2 is custom built, based on the fat-tree topology, with 13 high-radix switches of 576 ports, on the top level, with either electrical or optical links. Proprietary high-radix routers and network-interface Application-Specific Integrated Circuits (ASICs) implement a proprietary communication protocol. Tianhe-2 achieves about 62% of its peak performance.

Cray X-series. The history of the Cray X-series follows that of the classic, massively parallel, HPC systems based on the 3D torus topology. The company's developers constantly introduced new ICN approaches. Originally, Cray designed and built the RedStorm system, as described in Brightwell et al. [2005a], for Sandia National Laboratories, with the XT3 system [Jeffrey et al. 2006] as its commercial spin-off, followed by the XT4 [Alam et al. 2007] and the XT5 [Worley et al. 2009]. All the

Table III. Summarized Characteristics of Cray XT/XE/XK Systems

System	Cores per node	Router	Link bandwidth Peak/Sustained (GB/s)	Router Bandwidth (GB/s)
XT3	1 or 2	SeaStar	7.6/4	45.6
XT4	2 or 4	SeaStar2	7.6/6	45.6
XT5	4 or 6	SeaStar2+	9.6/6	57.6
X2	4	YARC	2.3/?	300
XE6	8 or 12	Gemini	9.4/5.8	168
XK7	8 or 12	Gemini	9.4/5.8	168

XTs use AMD Opterons, but the direct connection between the CPU and the SeaStar router chip via the HyperTransport link was a breakthrough innovation compared to the classic network interfaces that reside on the I/O bus [Brightwell et al. 2005b]. The SeaStar performs DMA tasks and also handles the MPI functionality. To keep up with the increasing speed of the CPU, the original SeaStar router was upgraded to the SeaStar2 and SeaStar2+ by increasing the bandwidth and keeping the latency low. Supporting nodes (including I/O nodes) are integrated into the 3D torus and there is no separate I/O network.

XT5 also supports vector CPUs (X2, codename Black Widow), described in Abts et al. [2007], and multithreaded CPUs. The vector CPUs are connected by a modified fat-tree topology, while the routing is performed with a YARC router (Yet Another Router Chip) having notable features: a high radix and bandwidth, a low latency appropriate for distributed shared memory systems, and narrow links. The bridge chip connects this network to the rest of the XT5 structure. The details of the micro-architecture and the rationales for the design choices are given in Kim et al. [2005b] and Scott et al. [2006]. The last CPU upgrade of the Jaguar from four to six cores enabled it to take over the Top500's first place from Roadrunner (Blade Center) in the November 2009 list, with significantly higher performance, without accelerators. It stayed ahead of the Chinese systems, with a higher peak performance, until the November 2010 list update.

The Cray XE6 is a newer generation of the Cray systems. Architecturally, it is very similar to the XT series, but there are some significant differences, for example, a new Gemini routing chip. As in the XT series, the CPU is an AMD Opteron but with 8 or 12 cores. The Gemini router has as up to 48 ports, which is an unusually large number for a 3D torus router. Eight ports are used for the connection to 2 HyperTransport links, and 40 others are used to form the 3D torus. Each torus connection consists of 4 links, giving 10 available groups. In the x and z dimensions, 16 links in 4 groups are used, while in the y direction only 8 links in 2 groups are used. This feature provides additional safety against single–link failures. Gemini supports Remote Memory Access (RMA), but without the cache coherence associated with Opteron caches, Fast Memory Access (FMA) for the low-latency transfer of a small amount of data, and a Block Transfer Engine (BTE) for the asynchronous transfer of a large amount of data. The BTE was already present in the T3D, the first Cray HPC system. The overall design of Gemini has much more in common with the YARC from X2 than with the SeaStar, because of the tiled design that supports a large number of ports, the same PHIT size, and two flow-control techniques (cut through externally and wormhole internally). The complete design of the Gemini router is described in Alverson et al. [2010]. Some benchmark data are available in Vaughan et al. [2011].

Finally, the XK7 is essentially the XE6 system (see Table III) with additional Graphics Processing Unit (GPU) accelerator cards. The largest installed XK7 is Titan, in second place on the Top500 list from November 2015, with 18,688 nodes, each having 16-core Opterons and NVidia GPGPU cards. The Blue Waters system, based on

Table IV. Summarized Characteristics of Blue Gene/L/P/Q Systems

	Blue Gene/L	Blue Gene/P	Blue Gene/Q
Node	Dual PowerPC 440 700MHz	Quad PowerPC 450 850MHz	17 core Power A2 1.6GHz
Cache per node	32KB I-cache, 32KB D-cache	32KB I-cache, 32KB D-cache	16KB I-cache, 16KB D-cache
Torus network	2.1GB/s	5.1GB/s	40GB/s
Collective network	2.1GB/s	5.1GB/s	N/A

the XE6 with 22,640 nodes (16 core Opteron) and with 4,224 nodes (16-core Opteron and general-purpose GPU (GPGPU)), is not benchmarked with HPL, but, due to its larger number of CPUs, it should achieve a higher percentage of its peak and better performance on applications without GPGPU support.

Blue Gene series. Third place on the Top500 list from November 2015 is reserved for Sequoia, based on the Blue Gene (BG) approach with a very large number of nodes that have a modest performance and, consequently, low power consumption. Note that places 5, 9, and 10 are also occupied by BGs with a smaller number of cores, which demonstrates the scalability of the BG architecture. Hence, it is not a surprise that BG has been successfully evolving for almost 15 years. In the initial Blue Gene/L system (BGL) [TheBlueGene/LTeam 2002], a complete node, with two PowerPC 440s, three levels of caches, and ICNs, was on a single chip. There were five ICNs: the 3D torus for point-to-point communication, the tree network for collective communication, the global barrier network, and two Ethernet networks, one for I/O and the other, with test access port defined by Joint Test Action Group (JTAG) IEEE 1149.1 standard, for control and test purposes. The torus network achieved a total link bandwidth of 175MB/s. The role of the communication coprocessor was assigned to the second core.

In the second-generation Blue Gene/P (BGP) [TheBlueGene/PTeam 2008], the node architecture, the ICN architectures, and the ratio among the CPU's speed, memory bandwidth, and ICN bandwidth remain the same as in the BGL. However, each node has four PowerPC 450 processors running at 850MHz, and the links are 2 bits wide, giving a link speed of 425MB/s. The BGP has a dedicated DMA engine that off-loads a portion of the MPI functionality, thus enabling a greater overlap between the computation and the communication.

The Blue Gene/Q (BGQ), the third generation of BG systems, brings further innovations in terms of both system architecture and ICN design. Each BGQ node has 17-core processors that run at 1.6GHz with a support for transactional memory. The ICN is a 5D torus with 4GB/s per port. Interestingly, the last dimension of the ICN connects only two nodes, which demonstrates the compromise of preserving higher ICN dimensions for an increased bisection bandwidth and for a low hop count, while retaining most of the 3D torus's packaging advantages. Another change in BGP is the absence of a separate barrier and collective networks and their integration into the main torus network. Table IV summarizes the characteristics of the BG systems. The bandwidths of the networks represent the total router bandwidth per node.

Fujitsu K computer. The Fujitsu K computer is ranked in fourth place in the Top500 list from November 2015 as a result of a number of unique architectural innovations that provide enormous peak performance and excellent efficiency. It is built with 88,128 Fujitsu-developed SPARC64 VIIIfx processors having eight cores. Many of the K computer advantages come from the Tofu ICN, which better balances conflicting requirements, for example, an optimal allocation of resources versus a maximization of system utilization. In addition, the Tofu ICN enables efficient fault-tolerance approaches by core and link faults. A higher dimensionality and a variety of connections

provide more opportunities to best fit the various applications. For example, a local 3D torus can be embedded in many different ways into the Tofu ICN. To improve the performance in cases when the overlapping of computation and communication is possible, four communication engines per router are available that share all the links to other routers. There is also support for collective communication in the form of collective operation (reduce) and synchronization (barrier) operation without involving the processor. The K computer is extensively described in Ajima et al. [2009].

Cray XC30/40. Fifth place belongs to the already-described BGQ, while the sixth to ninth places in the Top500 list from November 2015 are again occupied by Cray XC30/40 systems, which feature a number of modifications regarding the XT/XK series. On the node side, it brings two Intel Xeons with up to 16 cores, instead of AMD Opterons, and on the ICN side it brings the Dragonfly topology and the Aries router. The Aries has many similarities with YARC, Gemini, and even SeaStar, but there are new features, mostly to efficiently support the Dragonfly topology and the scalability.

The basic building block is a blade with 4 nodes and one Aries router chip. The Aries router has 48 ports (like Gemini), with 8 ports being used for the bidirectional connections of 4 nodes on the blade and the remaining 40 ports being used for the ICN. There is strong emphasis on the hierarchical package and locality. Sixteen blades are connected in a chassis, three chassis form a cabinet, and two cabinets form a Dragonfly group with 384 nodes and 96 Aries routers. Inside a chassis, all 16 blades are fully connected, occupying an additional 15 router ports. Inside a group, each of the 16 blades are connected through 3 ports to the corresponding blades in 5 remaining chassis, occupying an additional 15 router ports. Hence, a minimal path between any two routers in a group is two hops at most. Each Aries provides 10 remaining ports for global links, resulting in 960 ports in total for the inter-group communications. Because four-lane optical links are used, the number of groups is limited to 241, since each group can be connected to a maximum of 240 other groups. The Dragonfly parameters for such a system are $p = 8$, $\alpha = 30$, and $h = 10$.

Besides support to the Dragonfly topology and to the adaptive routing, the Aries router provides additional features, which can be classified as performance oriented and reliability oriented. The performance-oriented features include the following: support for address translation and support for RMA, BTE, and the Collective Communication Engine. Reliability features are provided in the form of Error Correcting Codes (ECC) and Cyclic Redundancy Check (CRC) on all the critical parts of the system as well as in the form of quick and accurate failure reporting.

Stampede. Tenth place on the Top500 list from November 2015 belongs to a state-of-the-art classic cluster built by Dell. The main processor is an Intel Xeon with 8 cores, while the role of acceleration is delegated to the Intel Phi. The Stampede system has a straightforward ICN topology. It is built as a two-level fat-tree. The first level consists of 320 36-port leaf switches, each of them connected to 20 computing nodes and 8,648-port core switches with two links. The core switches are built as two two-level fat-trees effectively resulting in a three-level fat-tree.

4.2. Lessons Learned from the Present

A summary of the HPC technologies, architectures, and ICNs of the PFLOP computers on the Top10 list from November 2015 are given in Table V.

One of the most important lessons to be learned from the present Top10 systems involves the scalability, on which the ICN has an important impact. The scalability is important at every level. The basic building block must be easily connected to other blocks in a uniform way. Moreover, the same building block must be used to build ICNs of different sizes, with only a small performance degradation for the maximum-size

Table V. Top 10 PFLOP Computers from the November 2015 List with Their Main Technological and Architectural Attributes

Rank	Computer name	No. of cores	R_max [TFlOP]	Power [kW]	R_max/Power [TFLOP/kW]	System platform	Provider	Processor	Accelerator	Interconnect	ICN topology
1	Tianhe-2	3,120,000	33.862	17.880	1.89	TH-IVB-FEP	NUDT	Intel Xeon E5-2692 12C	Intel Phi 31S1P	Custom: TH Express-2	Fat-tree
2	Titan	560,640	17590	8.209	2.14	XK7	Cray Inc	Opteron 6274 16C	Nvidia K20x	Custom: Gemini	3D torus
3	Sequoia	157,286	17173	7890	2.18	Blue Gene/Q	IBM	Power BQC 16C	None	Custom	5D torus
4	K computer	707,024	10510	12660	0.83	Sparc 64	Fujitsu	SPARC64 VIIIfx 2.0 GHz	None	Custom: Tofu	6D mesh/ torus
5	Mira	786,432	8586	3945	2.18	Blue Gene/Q	IBM	Power BQC 16C	None	Custom	5D torus
6	Trinity	301,056	8100	11079	0.73	XC40	Cray Inc	Intel Xeon E5-2698v3	Nvidia K20x	Custom: Aries	Dragon-fly
7	Piz Daint	115,984	6271	2325	2.70	XC30	Cray Inc	Intel Xeon E5-2670 8C	Nvidia K20x	Custom: Aries	Dragon-fly
8	Hazel Hen	185,088	5640	7403	0.76	XC40	Cray Inc	Intel Xeon E5-2680v3	Nvidia K20x	Custom: Aries	Dragon-fly
9	Shaheen II	196,608	5537	7235	0.77	XC40	Cray Inc	Intel Xeon E5-2698v3	Nvidia K20x	Custom: Aries	Dragon-fly
10	Stampede	462,462	5168	8520	0.61	Power Edge C8220	Dell	Intel Xeon E5-2680 8C	Intel Phi SE10P	Infini-band FDR	Fat-tree

system. Both the fat-tree and the torus satisfy these requirements, although in very different ways. The fat-tree typically adds one or two levels of switches for increasing system sizes by one order of magnitude and uses relatively large crossbars (8×8 at least) or the Infiniband (36×36). Each switch increases the latency, but the BBWN can be held constant or only slightly decreased. The torus uses an opposite strategy: Each router only slightly increases the latency, but for an order-of-magnitude-larger system, a much larger number of routers has to be added (approximately double in each dimension). Even without the perfect scalability of the BBW, the 3D nature of many applications could mitigate this problem.

However, scaling up a system beyond a certain size with fat-tree and keeping the BBW constant can be very expensive. The best demonstration is the Stampede system, which uses a pruned tree in order to reduce the excess number of cables and core switches. From this perspective, the disappearance of hypercube from contemporary parallel architectures is understandable. With a fixed number of links, the scalability is limited. However, the hypercube's scalability problems are not solely due to the number of ports needed in switches but also because of the need for long cables with intricate connectivities between cabinets. In other words, even when provided with switches that have dozens of ports, hypercube connectivity may not be the best use of such switches. This trend might change, because with Infiniband switches it is possible to build very large hypercubes with off-the-shelf components only, for example, Infiniband switches with 24 or 36 ports. Thus, in the exascale range, the hypercube may become interesting again due to its basic properties of a constant BBW and a low hop count and possibly a simpler cabling than in fat-trees for the same level of performance.

The focus of this survey is directed to the network topologies and the number of ports in the switches. Concerns about the other relevant parameter, the length of the cabling, are resolved through the introduction of optics, which is a promising approach to overcoming the scalability problems [Arimilli et al. 2010]. Since long wires dictate the cost, the packaging can resolve this by making a balance between copper for short links and optics for longer links [Kim et al. 2008]. The next technological step towards more efficient ICNs could be the use of optics, both between boards and between chips on the same board [Taubenblatt 2012]. Finally, all the more-or-less successful ICNs have one common feature: They are relatively easy to understand for most engineers. Any new topology entering the arena of ICNs must also have these qualities.

Even though they appear to be very diverse on first inspection, the Top 10 systems share several common approaches that could evolve in the winning exascale system. The top-of-the-line systems have, on average, more than 500,000 computing cores with a diverse power efficiency from 0.61 to 2.70 [TFLOP/kW]. By using similar processors, in the past few years, the list confirms that semiconductor technology has reached the end of its historic scaling, which is hard to sustain on account of the larger number of cores, attached accelerators, and the improved interconnection technologies. A closer look at ICNs reveals the almost-exclusive use of custom technologies and high-radix switches that enable the construction of high-dimensional topologies. In order to keep the expenses within manageable limits, the number of long optical links should be minimized on account of the smaller flexibility. Such an approach could retain the historic scaling over the next decade.

5. DISCUSSION

5.1. Lessons from the Past and Present

Past developments indicate that two issues are of great importance for the future developments of ICNs. A larger number of interconnection wires can be tolerated as far as the technology is concerned, which brings more performance as far as the application is concerned. This implies that the optimal ICN in future systems will be characterized

by a larger wire count. However, using the same topologies with wider links (larger PHIT) may not be the best solution.

On the other hand, the past teaches us that moderate-degree implementations, not dependent on the system size, might be preferred in the future. This is due to the fact that system implementations in higher dimensions could imply non–trivial technological overheads, for example, in a dimension-dependent degree of nodes that becomes too large for practical implementations with an increased number of nodes, or in an inability to manage the balance between the number of links per node and its I/O throughput.

Regarding the ICN scalability, it seems that regular topologies with a node architecture that is independent of the system's dimensions will dominate. Such an approach implies a system node composed of a computing unit and a communications switch with a balanced system throughput and nodal I/O ability.

One of the serious anticipated problems with a 3D spatial implementation is the limited possibility for power dissipation because of the problems with higher temperatures in the central part of the system. An additional potential problem is the more difficult accessibility to the inner nodes. Planar topologies are important for the networks on a chip and convenient for off-chip, higher-radix routers.

Current and future trends show that the number of nodes in HPCs will constantly grow. However, such huge systems can only be built and managed if they are composed of simple and equal building blocks. Consequently, an ICN with nodes of a constant degree are preferred. Despite this, the diameter of a future ICN should also be small. These somehow contradicting objectives and the problems mentioned at the beginning of this section have been partially addressed by the Tofu ICN, which contributed to the Top500 dominance and the efficiency of the K computer.

Another trend of great importance is the emergence of the Partitioned Global Address Space (PGAS) model for parallel programming in forms of new languages or extensions to existing languages [Charles et al. 2005]. As this model resembles the shared-memory model but with a strong emphasis on locality, we can expect more applications to use it in the future, especially considering the fact that the PGAS is much simpler and more scalable than the distributed shared memory since there is no need for the caching of remote memory.

There is an increasing trend to build faster and larger HPC systems. The communication hardware is tightly coupled to the processors, and it is optimized for the MPI standard, with microkernel-based operating systems or stripped-down Linux. The fat-trees and hyper-D tori are used almost exclusively in the area of HPC. Many other topologies have simply disappeared, while only Tofu and Dragonfly can be assumed as new.

Implementations include combinations of off-the-shelf versus custom interconnects and processors, often with data-flow-based accelerators. The choice of ICN topology is largely influenced by the performance of the node and the interconnection technology. A smaller number of nodes makes fat-trees a good solution, while with a larger number of nodes the fat-tree would have more levels and thus an increased latency with a higher hop count.

An important current trend of an increasing number of nodes is also followed by an increasing number of cores in each node, which is mostly dictated by the ever-increasing number of cores per processor. Thus, the positioning of a single router into the center of a small star interconnect is applied. Finally, some hybrid approaches could become favorable; therefore, they deserve constant attention.

5.2. Proposals Based on the Lessons

Computational requirements determine the efficiency of any parallel algorithm. Numerous approaches for more efficient and scalable ICNs [Kim et al. 2005a], for accelerated communications [Nüssle et al. 2013], and for several other improvements in ICNs

have been published in the past; however, the communication patterns and the chosen interconnection topology can still importantly contribute to the improved efficiency. In general, the quantity of communications increases with the number of processors, which limits the speedup of parallel programs. In order to mitigate the problem, the calculations and communications must run in parallel whenever possible [Marjanović et al. 2010; Lawry et al. 2002]. Furthermore, communications should be able to exploit the parallel communication paths in the ICN as much as possible [Subotic et al. 2010]. To support scalability, we require an isomorphic ICN composed of network nodes with the same architecture, even if the number of nodes is increased [Dally and Towles 2004].

We feel that an important step forward towards the exascale era is possible with further improvements to ICNs. One obvious approach to increasing the ICN performance is to increase the radix of well-known topologies, for example, the hyper-D torus. The main advantage of such an approach is in the fact that many important algorithms are already optimally implemented for lower-radix ICNs, which simplifies the upgrading for higher radices [García et al. 2012]. However, the potential disadvantages are in the loss of some of the desirable properties, for example, the physical compactness of the 3D torus, or the fact that algorithms being developed for lower-radix routers do not scale well for high-radix routers.

The increase in radix can be achieved using two main strategies, the first one being a moderate increase of the radix and the second one being a significant increase of the radix. There are no exact rules where the low radix ends and high radix starts, but radices less than 12 can be safely classified as low, while radices higher than 40 can be classified as high, while for radices between these values the classification is somewhat unclear. The first strategy is exemplified in the Fujitsu K and IBM BlueGene/Q with the resurrected concept of some dimensions being much smaller than others in order to improve the packaging properties, while still enjoying a significant improvement over the classic 3D torus. The second strategy is exemplified in the Cray YARC, which was built for distributed shared memory systems that favor shorter packets and higher radices, or in the Cray Aries, which provides the efficient, fine-grained collective communication.

5.3. Motivation for the Future

The ICNs based on high-radix routers, a smart combination of copper and optical links, and Networks-on-Chip (NoC) for the communications on the computing core level could become a winning combination in the future developments of HPC. Compared with k-ary n-cubes, the performance of d-meshes in a node-to-node communication is competitive with respect to the maximum and average distances between the nodes, to the scalability in the number of nodes and their degree, and to the structural regularity of nodes and links [Trobec et al. 2009]. A direct consequence is the possibility to construct a wide variety of d-meshes, from low radix (in the range of 4–12) and midrange radix (12–40) to high radix (40 and more) in the same framework, thus better adapting to the users' needs and technological factors.

The reasonableness of the above option is indirectly confirmed by Stafford et al. [2010], which follows a similar strategy. The authors propose a king mesh and a king torus, which are topologically richer than their classic counterparts, since they also include diagonal links in addition to the usual orthogonal links. Their experimental analysis shows that a performance improvement can be achieved in this way. There are other promising ICNs tailored for the multiple processor cores on NoC published recently, for example, dense Gaussian networks [Martínez et al. 2006], based on the family of dense degree four circulant graphs with a richer connectivity and a smaller diameter than classical tori.

We expect that in the near future there will be proposals for several alternative approaches that can improve the effectiveness and flexibility of future ICNs. This will

offer a lot of options for advancing the parallel performances of HPCs, in particular, in combination with optical transmission and switching technologies. The ICNs will become adaptive to communication flows and media, inter-/intra-node physical distance, network faults, and so on, thus taking over an important part of the responsibility for the computation speed from the processors.

5.4. Benchmarking

There is a significant difference in the HPL efficiency between leading systems. This is best exemplified by a Tianhe-2 and Titan comparison. Tianhe-2 features an enormous number of cores, 5.5 times that of Titan, yet it scores less than 2 times better. On the other hand, Sequoia has an almost 2 times fewer number of cores, while the score and power consumption are more than 2 times lower, with a substantially lower peak performance. The K computer together with BlueGene/Q demonstrate that excellent performance and power efficiency are attainable without accelerators. If the maximum efficiency is considered, then Tianhe-2 and Titan are comparable with 61.7% versus 64.9% of their respective peaks, while the Sequoia and K computer have significant advantages with 85.3% and 93.1%.

However, to effectively analyze the efficiency of an ICN (including the related software stack), the peak HPL per node performance should be used as a point of comparison as opposed to the theoretical peak. This is a significant issue as the efficiency of a single core CPU or GPU considerably varies, thus potentially covering up the differences between ICNs. Another significant benchmarking factor is the ratio of the accelerator to CPU performance. As this ratio increases, dependency on libraries, application tuning, and dependence on general "accelerator friendliness" increases, too. For example, the Cray XC30 Piz Daint is in many respects comparable to other Cray systems, which enables a direct comparison of the 3D torus and Dragonfly topologies. Based on HPL data only, Dragonfly does perform significantly better than the torus compared to a Titan using the same accelerators and comparable CPUs.

The results from the benchmarking must be looked at in an appropriate way. The HPL benchmark provides an upper limit to the performance for scientific applications. In many cases, the sustained performance on real applications is quite modest compared to the HPL scores. The deficiency of having a single-type benchmark was recognized. To better stress the CPU/memory system, as well as the ICN's impact, the HPC Challenge Benchmark suite was created, where the HPL is only one of the benchmarks [Luszczek et al. 2005]. While such a comprehensive benchmark suite provides much of the valuable data, its main advantage is its weakness, too, since it lacks the simplicity of a single number HPL. To improve on that, the High Performance Conjugated Gradient benchmark was created [Dongarra et al. 2003], with the main purpose being to provide a single number performance scale like HPL but with a significantly higher correlation with application performances. Still, the HPL is very valuable because of its large base of results spanning over a period of more than two decades. Although obvious, it is worth remembering that the main purpose of those systems is not running the HPL or any other benchmark but running the application codes [Flynn et al. 2013]. This calls for very careful statements about "the best" or "the fastest" system in the world. The Top10 list has been relatively stable for some time and this is perhaps a sign that both supercomputer users and vendors are looking for significantly faster and more scalable systems based on innovative computation and communication approaches.

6. CONCLUSIONS

The principal aim of this survey is to shed more light on the types of ICNs used in past and present supercomputers to establish the basis for their evaluation and to provide major requirements for future interconnects. We have shown that with a constantly

increasing number of processing elements, various modifications to the existing ICN topologies can be expected to fulfill the scaling requirements. We suggest the following plan of actions for the selection of future ICNs: a technology-relevant design parameter is interpolated through the past and extrapolated into the future. Its design value is determined to meet the projected technology-determined value at the time when the system will be applied in production. The same is repeated for all technology-relevant design parameters. If different pieces of analysis on the single parameter level lead to contradictory conclusions, then a multi-variant optimization has to be applied.

The question of an optimal node radix is of a crucial importance. In Kim et al. [2005b] a methodology is presented for the latency estimation and its minimization with an optimal radix. A single packet size (actually, the averaged sizes of packets) was assumed for the point-to-point communications only. Such an assumption is reasonable for distributed, shared-memory systems where most communications are with the nearest neighbors through equal packet sizes. The considered data movements are typically a single word and a single cache line with cache-coherency information. For general-purpose computing clusters, the situation can substantially differ, with message sizes ranging from a single word to kilobytes and with a packetization overhead on the message layer, which becomes significant in short messages. Many data movements are based on collective communication with eventual barriers and varied communication statistics. All these factors suggest that the optimization of a nodal radix should be based on much broader criteria, possibly in an adaptive way.

The communication nodes of ICNs will take over more and more decisions. They should be able to adapt the routing strategies to the current communication load in a cognitive way. Even more than before, the available technology will be very influential on the design decisions, including the choice of topology. A combination of ICN topologies in a single system was used in the past with promising results [Laudon and Lenoski 1997], and it is one of the ideas behind Dragonfly. The same idea can be seen in Tofu, which has some similarity with Dragonfly, although the starting points and motivations for these two topologies vastly differ. Comparing Tofu and Dragonfly, it is possible to draw parallels between small tori and groups and between systemwide and intergroup connections. The main difference is the scale: Tofu has a larger number of small groups, while Dragonfly has a smaller number of large groups and, as a consequence, radically different intergroup connections: 3D torus versus fully connected network.

It is evident that today the EFLOP on the HPL is well beyond the current technology with imposed space and power constraints. A promising technological approach is packaging a complete node in a single chip, where processors and routers are on one die and the memory on another, each manufactured with the process optimized for their respective functions. Such an integration could bring performance improvements and reduced power consumptions due to a mitigated pin number and chip-to-chip interface limitations, decreased latencies, and many available large memory banks. The Intelligent Random Access Memory (IRAM) project [Kozyrakis et al. 1997], which proposed to integrate a high-performance processor and the Dynamic Random Access Memory (DRAM) on the same die, demonstrated the above listed improvements of Systems-on-Chips (SoC). To fully exploit this promising technology, a few factors still must be resolved, mainly the cooling of the SoC and the loss of freedom regarding the choice of CPU and the amount of memory.

It can be stated that there is a constant search for faster and larger systems and their ICNs from the early beginnings of parallel computing in the 1950s. However, as the technology is changing constantly, the responses to this challenge also change [Friedman 2008; Kim et al. 2005a]. A naive answer remains simple—a network topology that enables an efficient usage of the available computational resources is the right

solution. The lessons learned by the extrapolation of the actual approaches in the past and the current top-of-the-line parallel computers led us to the conclusion that novel solutions in ICNs are needed for the exascale era. They will be based on (i) high-radix communication nodes for improved bandwidth, (ii) optical links for low communication latency, (iii) scalable interconnection topologies for efficient connection of millions of nodes, and (iv) adaptivity of these features to specific application domains. The limitations remain in a sustainable power consumption and in acceptable costs. However, the ultimate methodology that will enable an entry into the exascale era still remains an open research question.

REFERENCES

D. Abts, A. Bataineh, S. Scott, G. Faanes, J. Schwarzmeier, E. Lundberg, T. Johnson, M. Bye, and G. Schwoerer. 2007. The cray blackwidow: A highly scalable vector multiprocessor. In *Proceedings of ACM/IEEE Conference on Supercomputing*. 1–12.

Y. Ajima, S. Sumimoto, and T. Shimizu. 2009. Tofu: A 6D mesh/torus interconnect for exascale computers. *Computer* 42, 11, 36–40.

S. R. Alam, J. A. Kuehn, R. F. Barrett, J. M. Larkin, M. R. Fahey, R. Sankaran, and P. H. Worley. 2007. Cray XT4: An early evaluation for petascale scientific simulation. In *Proceedings of ACM/IEEE Conference on Supercomputing*, 1–12.

R. Alverson, D. Roweth, and L. Kaplan. 2010. The gemini system interconnect. In *Proceedings of the IEEE 18th Annual Symposium on High Performance Interconnects*. 83–87.

B. W. Arden and H. Lee. 1982. A regular network for multicomputer systems. *IEEE Trans. Comput.* C-31, 1, 60–69.

B. Arimilli, R. Arimilli, V. Chung, S. Clark, W. Denzel, B. Drerup, T. Hoefler, J. Joyner, J. Lewis, J. Li, N. Ni, and R. Rajamony. 2010. The PERCS high-performance interconnect. In *Proceedings of the 18th IEEE Symposium on High Performance Interconnects*. 75–82.

K. J. Barker, K. Davis, A. Hoisie, D. J. Kerbyson, M. Lang, S. Pakin, and J. C. Sancho. 2008. Entering the petaflop era: The architecture and performance of roadrunner. *International Conference for High Performance Computing, Networking, Storage and Analysis*, 1–11.

A. F. Benner, M. Ignatowski, J. A. Kash, D. M. Kuchta, and M. B. Ritter. 2005. Exploitation of optical interconnects in future server architectures. *IBM J. Res. Dev.* 49, 4–5, 755–775.

R. Brightwell, W. Camp, B. Cole, E. DeBenedictis, R. Leland, J. Tomkins, and A. B. MacCabe. 2005a. Architectural specification for massively parallel computers: An experience and measurement-based approach: Research articles. *Concurr. Comput. Pract. Exper.* 17, 10, 1271–1316.

R. Brightwell, K. Pedretti, and K. D. Underwood. 2005b. Initial performance evaluation of the Cray SeaStar interconnect. *Proceedings 13th Symposium on High Performance Interconnects*, 51–57.

T. Buh, R. Trobec, and A. Ciglič. 2014. Adaptive network-traffic balancing on multi-core software networking devices. *Comput. Netw.* 69, 19–34.

J. M. Camara, M. Moreto, E. Vallejo, R. Beivide, J. Miguel-Alonso, C. Martinez, and J. Navaridas. 2010. Twisted torus topologies for enhanced interconnection networks. *IEEE Trans. Parallel Distrib. Syst.* 21, 12, 1765–1778.

P. Charles, C. Grothoff, V. Saraswat, C. Donawa, A. Kielstra, K. Ebcioglu, C. von Praun, and V. Sarkar. 2005. X10: An object-oriented approach to non-uniform cluster computing. *SIGPLAN Not.*, 40, 10, 519–538.

C. Clos. 1953. A study of non-blocking switching networks. *Bell Syst. Technol. J.* 32, 406–424.

P. W. Coteus, J. U. Knickerbocker, C. H. Lam, and Y. A. Vlasov. 2011. Technologies for exascale systems. *IBM J. Res. Dev.* 55, 5, 581–592.

W. J. Dally and C. L. Seitz. 1987. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. Comput.* C-36, 5, 547–553.

W. J. Dally and B. Towles. 2004. *Principles And Practices of Interconnection Networks.* Morgan Kaufmann.

J. J. Dongarra and M. A. Heroux. 2013. *Toward a New Metric for Ranking High Performance Computing Systems.* Sandia National Laboratories.

J. J. Dongarra, P. Luszczek, and A. Petitet. 2003. The LINPACK benchmark: Past, present and future. *Concurr. Comput. Pract. Exper.* 15, 9, 803–820.

J. Duato, S. Yalamanchili, and L. Ni. 2002. *Interconnection Networks.* Morgan Kaufmann.

M. J. Flynn, O. Mencer, V. Milutinovic, G. Rakocevic, P. Stenstrom, R. Trobec, and M. Valero. 2013. Moving from petaflops to petadata. *Commun. ACM*, 56, 5, 39–42.

J. Friedman. 2008. New views of the structure of the universe. *The IPSI BgD Transactions Advanced Research*, 4, 5–6.

P. Fuentes, E. Vallejo, C. Camarero, R. Beivide, and M. Valero. 2015. Throughput unfairness in dragonfly networks under realistic traffic patterns. In *Proceedings of the IEEE International Conference on Cluster Computing*. 801–808.

M. García, E. Vallejo, R. Beivide, M. Odriozola, C. Camarero, M. Valero, G. Rodríguez, J. Labarta, and C. Minkenberg. 2012. On-the-fly adaptive routing in high-radix hierarchical networks. In *Proceedings of the International Conference on Parallel Processing*. 279–288.

A. Grama, A. Gupta, V. Karypis, and V. Kumar. 2003. *Introduction to Parallel Computing*, 2nd ed. Pearson Education Limited, Essex, England.

T. Hoefler, T. Schneider, and A. Lumsdaine. 2008. Multistage switches are not crossbars: Effects of static routing in high-performance networks. In *Proceedings of the IEEE International Conference on Cluster Computing*, 116–125.

S. V. Jeffrey, R. A. Sadaf, H. D. Thomas, Jr., R. F. Mark, C. R. Philip, and H. W. Patrick. 2006. Early evaluation of the cray XT3. In *Proceedings of the 20th IEEE International Parallel & Distributed Processing Symposium*. 1–10.

D. J. Kerbyson and P. W. Jones. 2005. A performance model of the parallel ocean program. *International J. High Perform. Comput. Appl.* 19, 3, 261–276.

E. J. Kim, G. M. Link, K. H. Yum, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, and C. R. Das. 2005a. A holistic approach to designing energy-efficient cluster interconnects. *IEEE Trans. Comput.* 54, 660–671.

J. Kim, W. J. Dally, S. Scott, and D. Abts. 2008. Technology-driven, highly-scalable dragonfly topology. *35th International Symposium on Computer Architecture*, 77–88.

J. Kim, W. J. Dally, B. Towles, and A. K. Gupta. 2005b. Microarchitecture of a high radix router. In *Proceedings 32nd International Symposium on Computer Architecture*, 420–431.

C. E. Kozyrakis, S. Perissakis, D. Patterson, T. Anderson, K. Asanovic, N. Cardwell, R. Fromm, J. Golbus, B. Gribstad, K. Keeton, R. Thomas, N. Treuhaft, and K. Yelick. 1997. Scalable processors in the billion-transistor era: IRAM. *Computer*, 30, 9, 75–78.

J. Laudon and D. Lenoski. 1997. The SGI Origin: A ccNUMA highly scalable server. *SIGARCH Comput. Archit. News*, 25, 2, 241–251.

W. Lawry, C. Wilson, A. B. Maccabe, and R. Brightwell. 2002. COMB: A portable benchmark suite for assessing MPI overlap. In *Proceedings of the IEEE International Conference on Cluster Computing (ICCC'02).*472–475.

C. E. Leiserson. 1985. Fat-trees - universal networks for hardware-efficient supercomputing. *IEEE Trans. Comput.* 34, 10, 892–901.

P. Luszczek, J. J. Dongarra, D. Koester, R. Rabenseifner, B. Lucas, J. Kepner, J. McCalpin, D. Bailey, and D. Takahashi. 2005. *Introduction to the HPC Challenge Benchmark Suite.* Electronic Book.

V. Marjanović, J. Labarta, E. Ayguadé, and M. Valero. 2010. Overlapping communication and computation by using a hybrid MPI/SMPSs approach. In *Proceedings of the 24th ACM International Conference on Supercomputing*. 5–16.

C. Martínez, E. Vallejo, R. Beivide, C. Izu, and M. Moretó. 2006. Dense gaussian networks: suitable topologies for on-chip multiprocessors. *Int. J. Parallel Program.* 34, 3, 193–211.

Mellanox. 2013. *Mellanox company site*. Sunnyvale, California. Retrieved from http://www.mellanox.com.

NNSA. 2013. *Advanced Simulation & Computing*. National Nuclear Security Administration, USA. Retrieved from http://www.nnsa.energy.gov/asc.

M. Nüssle, H. Fröning, S. Kapferer, and U. Brüning. 2013. Accelerate communication, not computation! In *High-Performance Computing Using FPGAs*, 507–542.

R. Peñaranda, C. Gómez, M. E. Gómez, P. López, and J. Duato. 2016. The k-ary n-direct s-indirect family of topologies for large-scale interconnection networks. *J. Supercomput*, 72, 1035–1062.

S. Scott, D. Abts, J. Kim, and W. J. Dally. 2006. The blackwidow high-radix clos network. In *Proceedings of the 33rd International Symposium on Computer Architecture*, 16–28.

G. Shainer, T. Liu, J. Liberman, J. Layton, O. Celebioglu, S. A. Schultz, J. Mora, D. Cownie, and V. Holst. 2009. LS-DYNA productivity and power-aware simulations in cluster environments. In *Proceedings of the 7th European LS-DYNA Conference*.

E. Stafford, J. L. Bosque, C. Martinez, F. Vallejo, R. Beivide, and C. Camarero. 2010. A first approach to king topologies for on-chip networks. In *Proceedings of the 16th International Euro-Par Conference on Parallel Processing: Part II*, 428–439.

V. Subotic, J. C. Sancho, J. Labarta, and M. Valero. 2010. A simulation framework to automatically analyze the communication-computation overlap in scientific applications. In *Proceedings of the IEEE International Conference on Cluster Computing*. 275–283.

M. A. Taubenblatt. 2012. Optical interconnects for high-performance computing. *J. Lightwave Technol.* 30, 448–458.

TheBlueGene/LTeam. 2002. An overview of the bluegene/l supercomputer. In *Proceedings of the ACM/IEEE 2002 Conference on Supercomputing*. 1–22.

TheBlueGene/PTeam. 2008. Overview of the IBM blue gene/p project. *IBM J. Res. Dev.* 52, 1.2, 199–220.

Top500. 2015. *Top500 supercomputers site*. Retrieved from http://www.top500.org.

R. Trobec. 2000. Two-dimensional regular d-meshes. *Parallel Comput.* 26, 13–14, 1945–1953.

R. Trobec. 2009. Evaluation of d-mesh interconnect for SoC. Parallel Processing Workshops, 2009. ICPPW'09. International Conference on. 507–512.

R. Trobec, U. Borštnik, and D. Janežič. 2009. Communication performance of d-meshes in molecular dynamics simulation. *J. Math. Chem.* 45, 2, 503–512.

C. Vaughan, M. Rajan, R. Barrett, D. Doerfler, and K. Pedretti. 2011. Investigating the impact of the cielo cray XE6 architecture on scientific application codes. *IEEE International Symposium on Parallel and Distributed Processing*, 1831–1837.

P. H. Worley, R. F. Barrett, and J. A. Kuehn. 2009. Early evaluation of the cray XT5. *Cray User Group Conference.* New York, NY.

Received November 2015; revised July 2016; accepted July 2016