2013 42nd International Conference on Parallel Processing

Distributed Shortcut Networks: Layout-aware Low-degree Topologies Exploiting Small-world Effect

Van K. Nguyen, Nhat T. X. Le
Ha Noi University of Science and Technology,
1 Dai Co Viet Road, Ha Noi, Viet Nam
vannk@soict.hut.edu.vn, thongnhat313@gmail.com

Ikki Fujiwara, Michihiro Koibuchi
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
{ikki, koibuchi}@nii.ac.jp

***Abstract*—Low communication latency becomes a main concern in highly parallel computers and supercomputers. Random network topologies are best to achieve low average shortest path length and low diameter in hop counts between nodes and thus low communication latency. However, random topologies lead to a problem of increased aggregate cable length on a machine room floor. In this context we propose low-degree non-random topologies that exploit the small-world effect, which has been typically well modeled by some random network models. Our main idea is to carefully design a set of various-length shortcuts that keep the diameter small while maintain an economical cable length. Our experimental graph analysis showed that our proposed topology has low diameter and low average shortest path length, which is considerably better than those of a counterpart 2-D torus and is near to those of a counterpart random topology with the same average degree. Meanwhile, the proposed topology has average cable length drastically shorter than that of the counterpart random topology. Our cycle-accurate network simulation results show that the proposed topology has lower latency by 15% and almost the same throughput when compared to torus with the same degree.**

***Index Terms*—Network topologies, small-world networks, interconnection networks, high-performance computing.**

I. Introduction

As the scale of many-core high-performance computer systems increases, their scientific parallel applications usually become latency-sensitive [1]–[3]. Thus designing low-latency interconnection networks is essential for these systems. Switch delays (e.g., about 100 nanoseconds in InfiniBand QDR) are relatively large when compared to the link delays (i.e. 5 ns/m). To achieve low end-to-end communication latency, a topology of switches should thus have low diameter and low average shortest path length, both measured in numbers of switch hops [3].

Topology design has been excitingly discussed for low-radix vs. high-radix networks, especially for exascale computing systems. IBM has been attempted a 3D torus for BlueGene/L and a 5D torus for BlueGene/Q supercomputers for low-radix cases, and Dragonfly [4] for Power 775 "datacenter-in-a-rack" systems for a high-radix case. However, we should not forget that low-degree topologies have been historically used in a mainstream of supercomputers, because of (1) their simple management mechanisms for faults [5], [6], (2) easy integration of network router/network interface and computation cores to a single chip or to a board (it can be regarded as "switch-less" network), (3) straightforward layout of switches with relatively short cables in a machine room [7], and (4) easiness in debugging custom communication protocol. Indeed, tori are employed by six of the top 10 supercomputers in TOP500 ranking in June 2012 [8]. In "Tofu" 6D torus network in the K-computer, three dimensions are fixed at $2 \times 3 \times 2$, but the remaining three dimensions are scalable and must be chosen carefully [6]. We can thus regard it as a 3D torus network when attempting to scale the topology. In this work, we target at low-degree topologies, i.e., up to six inter-switch links per switch.

Random topologies are generated either as fully random graphs [9] or by adding random shortcuts to classical topologies [3], [10]. These topologies achieve low diameter, low average shortest path length, and are thus a good candidate for the low-degree topology target [3]. However, a practical concern for random topologies is *their long cable length* for a physical deployment [3], [9], [11]. Even in the case of non-random topologies, for example, the first generation Earth Simulator required over two thousand kilometers of cabling [12], while the K-computer requires one thousand kilometers [6]. The use of random links further increases aggregate cable length [11]. A layout-conscious random topology using permutation has recently been proposed. It has low diameter and average shortest path length in high-radix networks, but less reduction is reported in low-radix networks [11].

Other practical concerns with random topologies are on routing implementation and traffic balancing. With respect to the former, random topologies cannot support custom routing implementation while several non-random topologies can exploit topological regularity to make routing logic simple and small (e.g., dimension-order routing). Thus, a topology-agnostic deadlock-free routing, such as up*/down* routing is assumed [13], which however can introduce traffic imbalance.

In this context we propose *Distributed Shortcut Networks (DSN)*, namely low-degree non-random topologies that "learn" some exciting design features from random topologies such

0190-3918/13 $26.00 © 2013 IEEE
DOI 10.1109/ICPP.2013.71

as small-world properties while still feature a simple routing logic. Our new interconnect architecture has *a much shorter cable length* than counterpart random shortcut topologies while keeping a similar (small) constant degree and a logarithmic diameter. Theoretically, the improvement can be a $\log n$ factor, asymptotically, compared to the random shortcut topology DLN-2-2 [3] of n nodes arranged in a 1-D fashion.

The contribution of our work is as follows.

- Although DSNs do not use the random effect of shortcut links to shorten path hops, they have similar low diameter and similar average shortest path hops to those of the counterpart random topologies in all network sizes, i.e., 64 – 2048 switches in our graph analysis. Besides the basic DSN topology, we also consider a variety of extensions that have different properties of diameter (and average shortest path hops) vs. degree and that also cope with other important issues such as to avoid deadlocks in our custom routing implementation.
- In the practical perspective, our DSNs surprisingly have similar average cable length to the same-degree torus in conventional floor layout of supercomputers in a machine room. When compared to a random topology with the same degree, they reduce the average cable length by up to 38%.
- Cycle-accurate network simulation results show that DSNs reduce end-to-end network latency by up to 15% compared to the same-degree torus topology.

The rest of this paper is organized as follows. The principles of our DSN design and related work are discussed in Sections II and III. In Section IV, we define and analyze our basic DSN topology in theoretical aspects. We then briefly discuss some initial work on the extended topologies in Section V. In Section VI, using experimental graph analysis, we compare DSN to counterpart non-random and random topologies, based on the computation of average cable length when laid out in a machine room. In Section VII, we use discrete-event simulation to compare them. Section VIII concludes the paper and shows our future work.

II. Distributed Shortcut Networks (DSN): Exploiting Small-World Properties in Designing Shortcuts

As a subject of extensive studies, small-world networks have been introduced to model network structures of popular real-world large-scale networks [14]. The small-world effect can be spoken of some special low-degree small-diameter networks that support distributed search, i.e., routing using local information only. In his seminal work for analyzing the algorithmic perspective of the small-world effect [15], Kleinberg showed that a selective addition of a few random long-range links can drastically reduce the diameter and can make the distributed search possible. Motivated by this important observation, we consider designing efficient low-degree topologies which also exploit the pattern of cleverly-created long-range links as we observed in these mentioned small-world models.

Let us briefly review some existing topologies, which are (inspired by) small-world networks and have helped to shape our proposed topology. DLN-x, a Distributed Loop Network of degree x [3], consists of n vertices arranged in a ring and additional shortcuts between vertices i and j such that $j = i + \lfloor n/2^k \rfloor \mod n$ for $k = 1, \ldots, x-2$. This graph has a logarithmic diameter for $x = \log n$. Koibuchi et al. [3] also introduced a topology called DLN-x-y, where y random links are added to each node of the DLN-x. These graphs have constant degrees and logarithmic diameters with constant x, y. Kleinberg's small-world network [15], on the other hand, consists of a grid and additional random shortcuts that prefer nearby nodes. This graph features the small-world effect in full: short routes between any two nodes are abundantly provided and also are easy to find using just local information.

In scrutiny, however, all these above topologies have some significant shortcomings. DLN-$\log n$ has a logarithmic diameter but has degree $\log n$, which is not low as desired. DLN-x-y with constant x, y has a logarithmic diameter and a constant degree but routing needs a global knowledge of the topology and, most crucially, it has a very large total cable length. Although having shorter cable length, a constant degree and a logarithmic diameter, Kleinberg's small-world network uses greedy routing, which can find paths with length only quadratic to the optimum [16].

In our approach, the basic idea is also to use the same type of shortcut links as in DLN-$\log n$ but utilize them in a smarter, more economical manner (also similarly reflected in Kleinberg's topology). In DLN-$\log n$, each node u has its own super shortcut jumping a distance of $n/2$. This is redundant and too expensive in terms of cable length. It will be cheaper if each node can "borrow" the super shortcut of the next-door neighbor. Therefore we consider adding only one shortcut of each type to a group of adjacent nodes rather than to each node. This may increase the diameter a bit, but can drastically reduce the degree and the cable length. A close observation shows that the distribution of shortcuts and their lengths in our proposal is quite similar to those in Kleinberg's random topology. However, our proposed topology has fixed (non-random) shortcut links, which are intentionally designed to save cable length. Moreover, our way to create shortcuts enables to use an efficient deadlock-free routing algorithm.

III. Related Work

Traditionally, low-diameter topologies were often designed using a hierarchical structure or node permutation. For example, shuffle-based topologies such as De Bruijn graphs [17] or (n, k)-star graphs [18] have desirable diameter-and-degree properties: De Bruijn has 12-and-4 for 3,072 vertices while Kautz has 11-and-4 and Pradhan has 12-and-5. Hierarchical structures are also well studied for targeting low degrees. Other typical low-degree topologies include Hypernet that consists of subnets mutually connected by a complete graph [19] and Cube Connected Cycles (CCC) that replaces each node in a hypercube with a ring of nodes. The latter has a constant degree three for any size of network. Hypernet has 19-and-5 for 4,608 vertices and CCC has 23-and-3. Although we have to carefully consider their layouts in a machine room when applying them to supercomputers, they still have interesting diameter-and-degree properties.

The low diameter property of random graphs has been extensively discussed in literature. Having the effect of greatly reducing the graph diameter, random links have been exploited for modeling real-world complex networks, such as social networks and the Internet topologies. The *small-world* property of these networks has received a fair amount of attention in literature, initiated by the landmark small-world paper by Watts and Strogatz [20]. Small-world and random graphs have recently been proposed for designing data center networks with increased expandability, fault tolerance and throughput [9], [10]. They can be applied in low-radix switch era. They are competitive to our DSN topologies and we compare them to DSN in terms of path hops and end-to-end communication latency in cycle-accurate simulation in Sections VI and VII.

To save the aggregate cable length, two methods are recently proposed in [11], which considers generating quasi-random topologies that has low path hops—almost comparable with its counterpart fully random topologies—and studies their physical layout on a floorplan. One of their methods randomizes the links after optimizing the physical layout, while the other optimizes the layout after randomizing the links. However, they implicitly assume relatively high-radix networks compared to its network size to obtain a small diameter and a small average shortest path length.

IV. Basic Topology and Custom Routing

A. Basic approach

Let us introduce our basic approach and give a more detailed review of the existing topologies that have motivated our approach.

Consider topology DLN-x [3], a Distributed Loop Network of degree x, where n vertices are arranged in a ring and a shortcut is added between vertices i and j such that $j = i + \lfloor n/2^k \rfloor \mod n$ for $k = 1, \ldots, x-2$. For DLN-$\log n$, obviously, for any given destination node t, any other node u has at least one shortcut link that halves the distance to t. Thus it is easy to see that this graph has a logarithmic diameter. These long-range shortcuts have made possible, what we call, the distance halving technique, which is widely used in analyzing graphs as well as designing topologies with small diameter. Generally, in these graphs when a node u searches for a path to another node v, it will find a long link from itself or a nearby node that goes closer to v by at least half of the distance between u and v for a given predefined distance metrics. As mentioned, DLN-x with $x = \log n$ has a logarithmic diameter and also has a natural simple routing logic but has a degree $\log n$ which is not low as desired. Besides, DLN-x-y with constant x, y [3] has a logarithmic diameter and a constant degree but its routing needs a global topology knowledge. Unfortunately, a uniformly-distributed random shortcuts can make the total cable length very long.

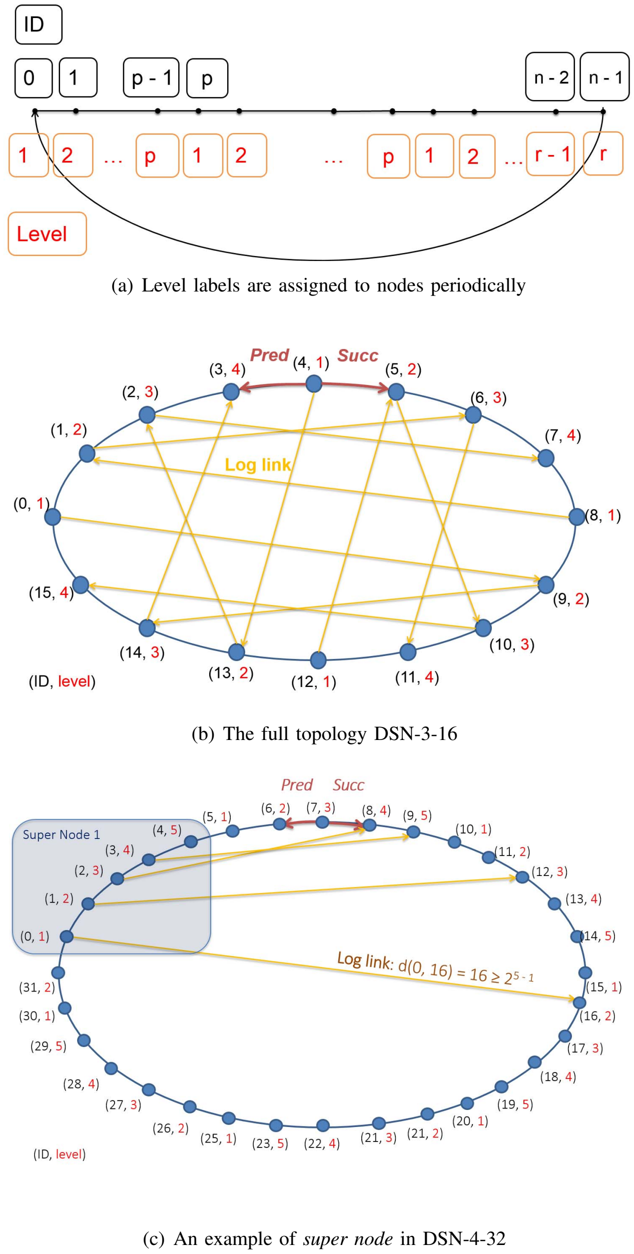
Now look closer at Kleinberg's small-world network [15]. A base grid graph of $n \times n$, where each node has at most 4 links to its neighbor nodes, is augmented by random shortcuts. A shortcut from node u goes to any other node v with a probability inversely proportional to the square of the lattice distance between u and v. Kleiberg's small-world network has a short cable length, a constant degree and a logarithmic diameter. This graph fully features the small-world effect and provides an abundant choice of short routes between any two nodes. Moreover, any node u can find a path of length $O(\log^2 n)$ to any other node v by using local information only. This is done by using greedy routing which works naturally based on the distance halving technique. However, the greedy routing can only find paths of length $\theta(\log^2 n)$ [16], asymptotically quadratic of the minimum.

Our proposed topology inherits those existing topologies mentioned above to achieve a low degree (small constant), a logarithmic diameter and *a custom simple routing to find almost optimum paths*; but we especially focus on *reducing cable length*. Our basic idea is to utilize the kind and variety of shortcut links as well as in DLN-$\log n$ but in a smarter, more economical manner (which is also somehow reflected in Kleinberg's topology). We give a full set of $\log n$ shortcuts, spanning distances $n/2^k$ for $k = 1 \ldots \log n$, to each group of $\log n$ adjacent nodes, which we call super nodes. Compared to DLN-2-2, obviously, this selection and deployment of shortcuts helps to reduce the total cable length to a factor of $\theta(\log n)$. Still, this lessened density of shortcuts is enough to make the distance-having technique work well and produce routes of logarithmic length.

B. Topology Description

Below we show the detailed construction of our basic topology DSN-x-n with n nodes. We also call it DSN-x or just DSN for short when the context is clear. The integer parameter x, conditioned to be $1 \leq x \leq p-1$ with $p = \lceil \log n \rceil$, represents the size of the set of different-length shortcuts.

- n vertices are arranged in a ring and each node has an ID number from 0 to $n-1$. Each node i shares two local undirected links with adjacent neighbors $(i-1) \mod n$ and $(i+1) \mod n$, which are called *predecessor* (*pred* for short) and *successor* (*succ* for short) links, respectively.
- Each node also has a numeric label from 1 to p, which is called the *level* of the node. The levels are assigned to nodes periodically: level $i = \overline{1 \ldots p}$ is assigned to nodes $k \times p + i - 1$ where $k = 0, 1, 2, \ldots, \lfloor n/p \rfloor$.
- Each node that has a level $l \leq x$ has one shortcut link going to node j that has level $l+1$ and has the minimum clockwise distance to i but at least $\lfloor n/2^l \rfloor$. We call this type of shortcut as the level-l shortcut, which has length at least $n/2^l$.
- For a node with level l we say that it has a height of $p + 1 - l$. Thus the higher a node is the farther its shortcut goes. The nodes with height p have shortcuts going farthest that is half of the ring.



(a) Level labels are assigned to nodes periodically

(b) The full topology DSN-3-16

(c) An example of *super node* in DSN-4-32

Figure 1. The basic DSN topology.

Figure 1 illustrates in detail about our topology construction. Fig. 1(a) illustrates on level assignment, while 1(b) presents a full network for the case of $n = 16$ and $x = 3$. To have a simple view of our DSN topology, imagine each group of p adjacent nodes to be collapsed into one big *super node*. You then obtain exactly a DLN-x topology of these super nodes as illustrated in figure 1(c). Later in certain clear context we also use a "super node" to indicate a group of p adjacent nodes.

Our DSN topology has a natural simple routing, which is induced from a simple routing logic of its super graph, i.e., the DLN-x of super nodes as mentioned above. Reviewing the simple routing logic of this super-graph, DLN-X topology, each super node U will find a proper shortcut (amongst its x shortcuts) to go half a distance to a destination super node V. This step can be repeated at most p times before the destination V is hit. Thus, the routing for DSN can follow the above as the blueprint but adding a few small steps for moving within a super node for finding the proper shortcuts.

Consider the details in performing a routing task from a source node s to a destination node t. Without loss of generality, we assume that $0 \leq s < t < n$ and $d = t - s$ is the distance between the two. Let $l = \lfloor \log \frac{n}{d} \rfloor + 1$ where $n/2^l < d \leq n/2^{l-1}$. We need to find a path from s to a node u at height $p + 1 - l$ (i.e., of level l) in order to use a proper shortcut that would go at least halfway through towards destination t. More generally, the main idea is that, at any current node u, we need to find a node v nearby (within the same super node) that is high enough to look over to t unless u is high enough.

Figure 2 presents the detailed algorithm. The algorithm has three phases: (i) PRE-WORK is for the source node s to find such a nearby node v with proper height, (ii) MAIN-PROCESS is to successively use distance-halving shortcuts to come into the locality of t, and (iii) FINISH is to walk to t using the local links. In the first PRE-WORK phase, the starting node s usually need to find a nearby node u at a proper height. If the level of s, $l_s = s \mod p + 1$, is greater than l, then we need to move up in height from s to find u; otherwise we need to move down from s.

Note that the MAIN-PROCESS is a loop. It goes downhill from the current node u to find a node v with the required height and takes the distance-halving shortcut, and repeats again with the new current node u. MAIN-PROCESS stops just when the shortcuts can no longer be used, where the current node u is close enough to t. Then we can simply walk to t on the local links. More specifically, this loop stops if either of the following three conditions occurs:

- $l_u = x + 1$, i.e., u is at the level that no longer has a shortcut.
- u is in the clockwise side of t, i.e., the shortcut just taken overshoots t.
- $t - u \leq p$, i.e., u is close enough to t and further taking a shortcut will overshoot t.

We name this union the LOOP-STOP condition.

C. Properties and Analysis

Note that we only define our DSN-x for $x = \overline{1 \ldots p-1}$. We denote $r = n \mod p$ in the remainder of this paper.

Fact 1 (On the degrees): The possible vertex degrees in DSN are 2 (only if $x < p - 1$), 3, 4 and 5. The average degree is ≤ 4 and there are at most p vertices with degree 5.

Proof: By definition, each node at a level l between 1 and x is assigned a level-l shortcut. We say this node has an *outgoing* shortcut (figuratively speaking, because links are indirect). A level-l node may also often have one or two level-$l-1$ shortcuts coming into it ($l > 1$). Nodes at level 1 or $l > x$ have no incoming shortcut. Therefore a typical node in

DSN-Routing Algorithm Pseudo-code

```
procedure DSN-ROUTING($s, t$)
    $u \leftarrow s$
    $l \leftarrow \lfloor \log \frac{n}{d_{ut}} \rfloor + 1$          ▷ i.e. $\frac{n}{2^l} \leq d_{ut} \leq \frac{n}{2^{l-1}}$
  ▷ PRE-WORK Phase ————————
    while $l_u > l$ do                 ▷ $l_u$– level of $u$
        $u \leftarrow u.pred$
        $l \leftarrow \lfloor \log \frac{n}{d_{ut}} \rfloor + 1$
    end while
  ▷ MAIN-PROCESS Phase ————————
    repeat
        if $l_u = l$ then
            $u \leftarrow u.shortcut$
        else
            $u \leftarrow u.succ$
        end if
        $l \leftarrow \lfloor \log \frac{n}{d_{ut}} \rfloor + 1$
    until $l_u = x + 1$ or $d \leq p$ or overshooting $t$
  ▷ FINISH Phase ————————
    repeat
        $u \leftarrow u.succ$ or $u \leftarrow u.pred$
    until $u = t$
end procedure
```

Figure 2. Our custom routing algorithm for DSN-x.

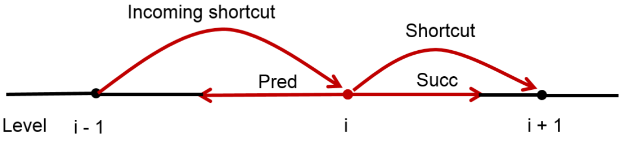
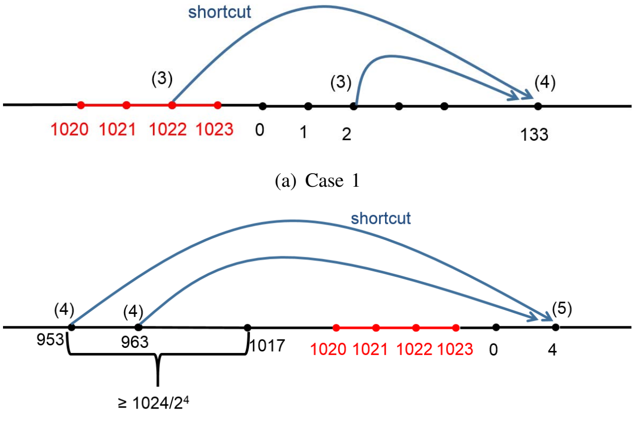


Figure 3. A typical node with degree 4.



(a) Case 1

(b) Case 2

Figure 4. Two cases of degree 5 for $n = 1024, p = 10, r = 4$. (i) denotes the level of node. Node in red are the last r nodes of the ring.

DSN has degree 4, namely a Pred, a Succ, an outgoing and an incoming shortcuts (see figure 3).

For $x = p-1$, the minimum degree is 3 since each node has at least one shortcut, outgoing or incoming, and the nodes at level $l > x+1$ only have degree 2. The number of nodes with degree 3 is at most $2n/p$, possibly counting nodes at level 1 (no incoming shortcuts) and nodes at level $x+1$ (no outgoing shortcuts). It is quite obvious that 5 is the maximum degree in DSN-x because a node can have no more than 2 incoming shortcuts.

Now we prove the upper bound on the number of nodes with degree 5. Suppose that a node t has degree 5, i.e., t has two incoming shortcuts, namely from nodes t_i and t'_i with level i. It is also not hard to point out that there are only two cases of having degree 5 as follows.

- The first case of having degree 5: When $n - r \leq t_i \leq n - 1$, $t'_i = (t_i + r) \mod n$ and the condition below holds:
$$i < [(t_i + n/2^i) \mod n] \mod p \leq p - r.$$
Clearly, there are at most r nodes as such t_i (see Figure 4(a)).
- The second case of having degree 5: When $t'_i = t_i + p$, $i > r$ and the condition below holds:
$$n + i - p - r \leq t_i + n/2^i < t_i + n/2^i + p \leq n + i$$
Clearly, there are at most $p - r$ nodes as such t_i (see Figure 4(b)).

In summary, there are at most p nodes with degree 5.

■

Observation: It is also not hard to show that the expected number of nodes with degree 5 is $\leq p/2$. Its proof is omitted due to space limitation since this fact is not that important and its proof is long and tedious.

Fact 2 (On the routing diameter): In our routing algorithms on DSN-x with $x > p - \log p$, the maximum path length is $\leq 3p + r$.

Proof: Let's consider a path from source s to destination t in our routing algorithm. First, in the PRE-WORK we successively take Pred links to go up, i.e. to smaller level, to find a proper node with level l. Clearly, it takes at most p hops in the PRE-WORK. In the MAIN-PROCESS, however, we always move up (in the hight of node), i.e. going down with levels. Obviously, the MAIN-PROCESS also takes at most p hops in the path length. Now we just need to consider the FINISH. It is a simple local walk, of which we will show an upper bound on the length.

Notice an invariant throughout the MAIN-PROCESS: d_u, i.e. the distance between the current node u and destination t, is always at most $n/2^{l_u-1}$. This is clear, because at this height and level (l_u), by definition, the shortcut from u will jump over at least half of this distance to t (a typical distance halving argument). Now, let us consider the situation when the MAIN-PROCESS just finishes. If LOOP-STOP is met by $l_u = x+1$, we have $d_u = t - u \leq n/2^{l_u-1} = n/2^x \leq n/2^{p-\log p} = p$. If LOOP-STOP is met by overshooting t, then the overshoot

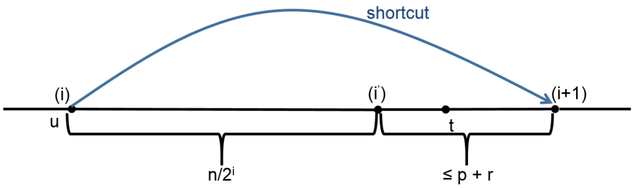


Figure 5. Case of overshoot. (i) denotes the level of node.

distance is at most $p + r$ (see Figure 5). Generally, from any node we go at most $p + r$ hops to find another node at any given level. Thus, regardless of how MAIN-PROCESS stops, the distance to t that it leaves for the FINISH to cover is at most $p + r$.

Overall, using our routing algorithm the path from any s to any t takes at most $3p + r$ hops. ■

Fact 3 (On the diameter): For $x > p - \log p$, DSN-x has diameter $\leq 2.5p + r$.

Proof: It is possible to find a path shorter than ones found by our routing algorithm as follows. We almost imitate this algorithm but in the PRE-WORK we take moves to the closest node at the required height (or level), either using successive Pred or Succ links, which reduces the max hop number in the PRE-WORK to at most $p/2$ (from p as in our routing algorithm — fact 2). Thus, such a path has length at most $2.5p + r$. ■

We conclude the above facts by the below theorem, which only considers the worst case.

Theorem 1: On the properties of the DSN:

a. The degree of vertices is mostly 4 but the maximum is 5 at at most p vertices.
b. For $x > p - \log p$, the diameter is at most $2.5p + r$
c. For $x > p - \log p$, the routing diameter is at most $3p + r$

Theorem 1 establishes upper bounds on some important factors of our proposed topology. The next theorem considers the average values instead.

Theorem 2: On the properties of the DSN:

a. For $x > p - \log p$, for s and t uniformly selected from the n nodes, the expected length of the s–t path found by our routing algorithm is $\leq 2p$ while that of the shortest s–t path is $\leq 1.5p$.
b. If the nodes are arranged evenly in a line of length n (distance between two adjacent node is 1), the average length of the shortcuts is $\leq n/p$ and the total cable length of the network is $\leq n^2/p + 2n$. Using the same setting, the topology DLN-2-2 [3] has average length of shortcut $n/3$. Thus the cable length in DSN is shorter than in DLN-2-2 about a factor of $p/3$.

Proof(Sketch): a. Because the expected level of s is $p/2$, the PRE-WORK will take expectedly no more than $p/2$ hops to any desired height. A similar argument will show that the FINISH phase takes expectedly $p/2$ hops. The MAIN-PROCESS as before can take up to p hops. Therefore, the expected length of an s–t path found by our routing algorithm is $\leq 2p$.

Now we aim to create a shorter s–t path by simulating our routing algorithm but making better choices in PRE-WORK and FINISH. From s we can choose to go either clockwise or counterclockwise to node u or v, respectively, that are at a necessary height required for the MAIN-PROCESS. Of course, u and v are distance p apart. Let X be the random variable to count the number of hops between u and s. We simulate PRE-WORK but choose between u and v the nearest one to s. Therefore, the expected number of hops in this simulated PRE-WORK is $E[\min\{X, p-X\}]$ which is $p/4$ (a basic probability problem). A similar argument (a bit more complicated) will also show that a simulated FINISH takes expectedly $p/4$ hops. Thus, the expected length of the shortest s–t path is $\leq 1.5p$.

b. We omit the proof of this part which only requires some basic integral work, although a bit tedious. ■

Our theorems above show that our proposed topology can strongly outperform DLN-2-2 in terms of cable length, while being quite similar in terms of degree and path length. However, our custom routing algorithm does not often find the minimum path and hence, we still leave space for improvement in future work. Our basic topology has been defined in a general format. We can specifically choose n as a multiple of p, and hence $r = (n \mod p) = 0$ so that we can avoid having the last super node with only $r < p$ nodes, which does not have a full set of levels and shortcuts (Figure 4 shows an incomplete, final node in red color). Such an incomplete super node makes it harder to manage our routing, e.g., it can enlarge the overshoot. The overshoot is normally at most p but can be enlarged to $p+r$. In the next section, we also consider to have n as an arbitrary integer but still avoid having an incomplete super node.

V. Possible Extensions

Our basic topology can be extended or augmented in several ways. In this section we consider extending our basic topology and our DSN-routing algorithm for solving different issues or improving on certain performance factors.

A. On Deadlock-free Routing

We can extend our basic topology and our custom routing algorithm to absolutely avoid having deadlocks in using wormhole or cut-through routing modes. Let us discuss our main idea first. If the DSN-routing algorithm had composed of only two parts, PRE-WORK and MAIN-PROCESS, it were deadlock-free because it had a natural Up-Down walking on the height of nodes. Remind that, the PRE-WORK is to go uphill to a proper height (so can look over to destination t) while the MAIN-PROCESS is to go downhill gradually, one step down each time taking a shortcut or a Succ link towards the destination. However, in reality we also have to deal with the FINISH, where we can possibly go uphill (using Pred links to get back over the overshoot distance). Note that we use the same type of links, the Pred, in both PRE-WORK and FINISH. Hence, two PRE-WORK and FINISH link sequences that are resulted in two separate routing tasks can still mess up together, creating a deadlock. This issue can be solved by introducing a separate virtual channel for moving uphill to the preceding node within the same super node (while decreasing the level by one). Alternatively, we can instead add a new local link per node for this purpose. We call them *Up* links.

Another cause of deadlock is that multiple routing threads all are in the FINISH phase can create a deadlock loop over the whole network ring. To solve this we can also further create $2p$ virtual channel, or alternatively, add $2p$ *Extra* links, that are $(i, i-1)$ for $i = 1 \dots 2p$. In both cases, we fix $x = p-1$ so that all super nodes will have a full set of shortcuts. We use DSN-E to call our extended topology with the new additional links (*Up* and *Extra* links), while we use DSN-V to call our basic topology with the additional virtual channels.

Theorem 3: In DSN-E networks, if we extend DSN-routing by using the Up links in the PRE-WORK and Extra links when available in the FINISH, then this extended routing algorithm is deadlock-free and the routing diameter is still $\leq 3p + r$. Similar results apply for DSN-V networks when we use new, separate virtual channels in place of the Up and Extra links accordingly.

Proof (Sketch): We analyze deadlock possibility in our routing algorithm by using a special *Channel Dependency Graph* (CDG), in which each vertex is not a single channel or link but a combination (or group) of links (see Figure 6). By nature of our routing algorithm, the three groups of links — *Up, Succ and Shortcut, Pred and Extra* — are used completely separately in the three phases of our routing algorithm, i.e., *PRE-WORK*, *MAIN-PROCESS* and *FINISH*, respectively. It is easy to observe that there are only two such dependencies between two distinct groups and no deadlock situation can occur due to them. Hence, we only need to consider deadlock possibility in each individual groups. However, the ways we use *Up* links (in PRE-WORK) or *Succ and Shortcut* (in MAIN-PROCESS) are such that the changings of height (or level) are always monotone, i.e., gradually moving uphill (in height) in PRE-WORK or gradually going downhill in MAIN-PROCESS. This property assures no deadlock possibility in each of these two groups. The only remaining deadlock possibility is in the *Pred and Extra* group of the FINISH phase, i.e., the existence of possibility of a cycle consisting of multiple FINISH sequences (of links used by each FINISH phase). Here we introduce the use of Extra link to break such a cycle: when destination t is in the range of $0 \dots 2p-1$, FINISH will only use Extra links for this part of the route inside this range. In summary, our routing algorithm is deadlock-free. The upper bound on routing diameter can be shown similarly as in fact 2.

■

B. Improving routing diameter

In DSN-$(p-1)$ the last few shortcuts from a super node, specifically the $\log p$ shortest shortcuts at the $\log p$ lowest heights and at level $l > p - \log p$, are not really helpful. It is easy to see that they are simply $(i, i+p+1)$ for any $i \mod p = l$. If used, these links will overshoot destination t. Therefore it is better not to have them but instead to add in some short links, a few per each super node, that can actually help to reduce the FINISH. Below is such a topology construction that uses DSN-x, with $x = p - \lfloor \log p \rfloor$, as a base and add a few such links to each group within a super node.

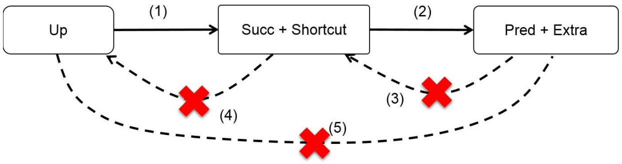


Figure 6. Channel Dependency Graph of three link groups. (1) and (2) are dependencies. There is no cycle (1-4), (2-3), (1-2-5).

Construction DSN-D-x: Let $q = \lfloor p/x \rfloor$. We add links $(iq, (i+1)q)$ for $i = 1, 2, \dots, w = \lfloor n/q \rfloor - 1$ and $((w+1)q, 0)$. Clearly, this helps to reduce the long local walks in the steps PRE-WORK and FINISH up to $(1 - \frac{1}{x})p$ per each step. For DSN-D-2, we add just two more short links per super node that help to reduce the diameter to only $\frac{7}{4}p$ (from $2.5p + r$). Our routing algorithm can also be updated a little bit to reduce routing diameter to $2p$ (from $3p + r$).

C. Improving flexibility

Here we aim to loosen the strict condition in constructing our topology. We allow each super node to have a flexible size, that is p and possibly plus/minus a few. For example, for $n = 1024$ we can arrange the nodes in 6 super nodes of size 10 and 4 of size 11. All the super nodes will still have the same set of shortcuts, i.e., the four 11th nodes don't have any shortcut. This arrangement will help to fix the issue of incomplete super node as well as to tolerate with node addition or failure. For convenience in managing and routing, we can extend the ID system as follows. We start with a convenient n (e.g. 1020), form the basic topology with integer IDs and whenever we need to add a new node between two initial node i and $i+1$ we give it a fractional ID such as $i + \frac{1}{2}$. The integer nodes will be considered the major nodes that come with shortcuts, the fractional are minor without shortcut. For example, a size-1024 topology can look like the basic DSN-10-1020 with 4 more added minor nodes, which can have ID such as $10\frac{1}{2}$, $20\frac{1}{2}$, $30\frac{1}{2}$ and $40\frac{1}{2}$. The routing will still be basically the same with the additional rule that is to route to a minor node we need to firstly route to the major node just before it, and then use Succ links to reach it.

D. Avoiding the overshoot

For the basic DSN, any shortcut must go from level $k-1$ to level k, which causes the overshoot issue. It could be a bit ugly to have an overshoot distance almost $p + r$ ($r = n \mod p$): normally, the overshoot distance is at most $p - 1$ but the existence of the final, incomplete super node I (that may

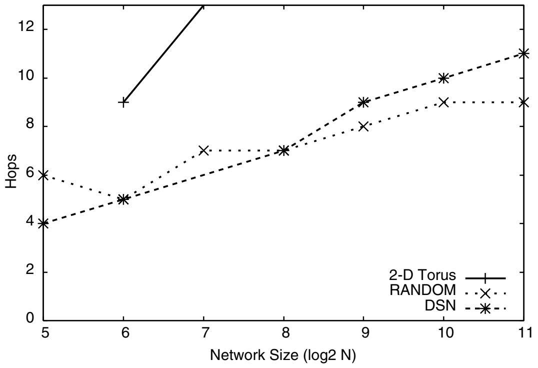


Figure 7. Diameter vs. network size for DSN, torus and RANDOM topologies.

not have a required level k) can delay the walk for finding the next k-level node further by the extra r local links within I. To avoid this overshoot problem totally, we can twist the routing algorithm a bit: at a node i if we find that the originally selected shortcut turns out to be an overshoot, we shoot first take a local link to node $i+1$ and then take the shortcut from it, which is much shorter and definitely lands before the destination t. Note that this new routing algorithm will help to reduce a lot in the FINISH, but may prolong the MAIN-PROCESS. We intend to make it concretely done in our future work.

VI. Topology and Layout Analysis

In this section we firstly compare our newly proposed topology (DSN) with a typical non-random topology (torus) and a random topology (DLN-2-2 [3]) in terms of diameter and average shortest path length by means of graph analyses. Next we compute the average cable length considering their floorplan in a machine room using parameters of recent interconnect technology.

We focus only on our basic topology in this paper to evaluate it from a practical perspective. Evaluation of our topology extensions will be made in our future work.

A. Diameter and Average Shortest Path Length versus Network Size

Our basic DSN topology has an average degree 4 (or a bit less). Thus we compare it with same-degree counterparts, a 2-D torus and a DLN-2-2, using the same network size (the number of nodes). Below we denote these topologies as DSN, torus and RANDOM, respectively. Figures 7 and 8 show the diameter and the average shortest path length of each topology, respectively. Lower values in both graphs are considered better.

In all the network sizes, RANDOM topology achieves the lowest diameter and the lowest average shortest path length. When compared to torus, DSN improves the diameter and the average shortest path length by up to 67% and 55%, respectively. These values are not far from those of RANDOM, which has an exact degree 4. Therefore our DSN is expected to have a similar performance to that of RANDOM topology with the same average degree.

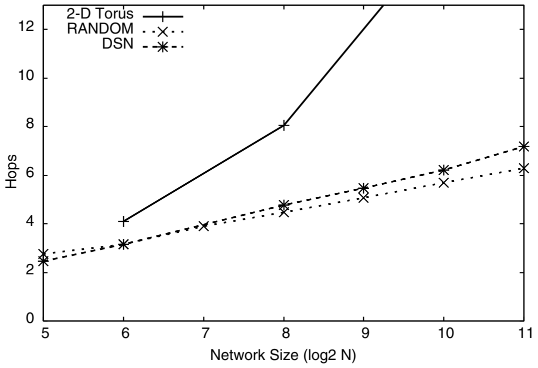


Figure 8. Average shortest path length vs. network size for DSN, torus and RANDOM topologies.

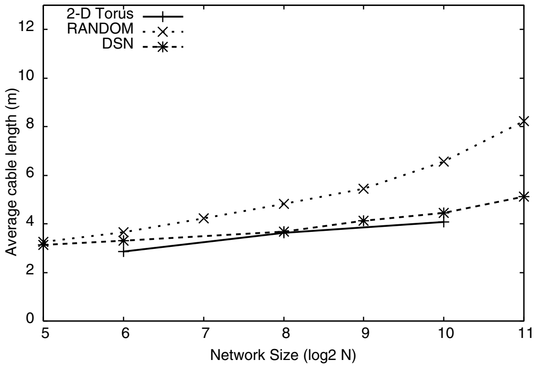


Figure 9. Average cable length vs. network size for DSN, torus and RANDOM topologies.

As the network size becomes large, both RANDOM and DSN achieve considerably lower path length than that of torus. Therefore we can say that RANDOM and DSN topologies have a good scalability.

B. Average Cable Length and Layout

We estimate the cable length required to deploy the topologies onto a physical layout of cabinets. We assume a physical floorplan that is sufficiently large to align all cabinets on a 2-D grid. Formally, assuming m cabinets, the number of cabinet rows is $q = \lceil \sqrt{m}\ \rceil$ and the number of cabinets per row is $p = \lceil m/q \rceil$. We assume that each cabinet is 0.6m wide and 2.1m deep including space for the aisle, following the recommendations in [21]. The distance between the cabinets is computed using the Manhattan distance. We estimate average cable length based on [22]: 2m intra-cabinet cables and a 2m wiring overhead added to the length of inter-cabinet cables at each cabinet. We ignore cables between compute nodes and switches, since their lengths are constant regardless of the layout. We assume that each cabinet has 16 switches.

The average cable length of each topology is computed and shown in Figure 9. Lower values of the average cable length are considered better. Notice that the layout of 2-D torus is well studied, such as folded method for uniform link length. However, the aggregate cable length of folded torus is the same as that of the corresponding original torus in which only wraparound links become long. Thus we fairly compare 2-D torus and our basic DSN in terms of cable length.

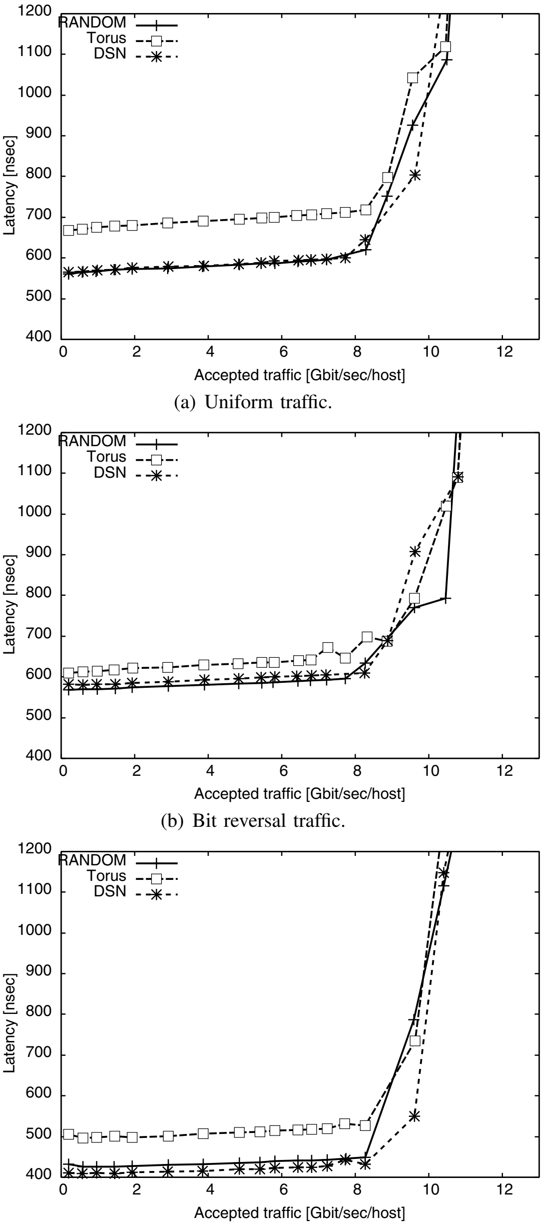
For RANDOM topology, the average cable length increases significantly as the network size becomes large. We can say that RANDOM topology has a serious issue of trading a lot of cable length for shorter hop count. By contrast, our DSN topology features (i) an average cable length similar to that of the 2-D torus and much shorter than that of RANDOM topology, and (ii) an average path length comparable to that of RANDOM topology. In another analysis, our DSN with degree 6 surprisingly has shorter average cable length than 3-D torus in conventional floor layout of supercomputers in a machine room. The total cost of interconnects (the price of switches and cables plus installation cost) increases in proportion to the cable length assuming high-bandwidth optical cables over 10Gbps [4], [23]. We thus expect that our DSN topology has a good economy. From this topological analysis, we recommend using our DSN topology for low-radix network era in supercomputers.

VII. Simulation Results

A. Parameters

We use a cycle-accurate network simulator written in C++ [3]. Every simulated switch is configured to use virtual cut-through switching. A header flit transfer requires over 100ns that includes the routing, virtual-channel allocation, switch allocation, and the flit transfer from an input channel to an output channel through a crossbar. The flit injection delay and the link delay together are set to 20ns. We use the topology-agnostic adaptive routing scheme described in [24], with up*/down* routing for the escape paths. This topology evaluation uses four virtual channels. The network size is set to 64 switches. Each switch has four compute nodes.

We simulate two synthetic traffic patterns that determine each source-and-destination pair: *random uniform*, and *bit-reversal*. These traffic patterns are commonly used for measuring the performance of large-scale interconnection networks [25]. In addition to these typical traffic patterns, we evaluate “neighboring” traffic in which 90% of packets are sent to neighboring nodes in 2-D array layout, whereas the remainder packets are sent to destination nodes randomly selected. The neighboring traffic considered in the evaluation is used for the performance evaluation under heavy local accesses. The hosts inject packets into the network independently of each other. In each synthetic traffic the packet size is set to 33 flits (one of which is for the header). Each flit is set to 256 bits. Effective link bandwidth is set at 96 Gbps. We pick relatively small packet sizes since we wish to study the performance of latency-sensitive traffic that consists of small messages [1]. Our results quantify two metrics: *latency* and *throughput*. The latency is the elapsed time (in nsec) between the generation of a packet at a source host and its delivery at a destination host.



(a) Uniform traffic.

(b) Bit reversal traffic.

(c) Neighboring traffic.

Figure 10. Latency vs. accepted traffic for DSN, torus and RANDOM topologies (degree 4).

The throughput is the largest amount of traffic (in Gbit/sec) accepted by the network before the network is not saturated.

B. Evaluation

Figure 10 plots the results of DSN, torus and RANDOM topologies. All the topologies have similar throughput, and the difference appears in their latency under low-traffic load. The average shortest path lengths in 64-switch network are 3.2, 3.2 and 4.1 hops for DSN, RANDOM and torus topologies, respectively (see Figure 8). As expected from these values, DSN and RANDOM topologies have almost the same curves.

Traffic patterns also affect the performance gain. DSN improves the latency by 4.3% with bit reversal traffic and by 15% with uniform traffic when compared to torus. We thus observe that DSN always outperforms torus in terms of latency. Our cycle accurate network simulation confirms that the latency strongly depends on the average shortest path length of topology. We thus expect that our DSNs maintain lower latency near to RANDOM topology as the network size become large, e.g., 2048 switches as shown in our graph analysis.

We have also created simulations using our custom routing. Our initial work has also obtained exciting results that our custom routing makes traffic significantly more balanced than using up*/down* routing. Hence, our custom routing can lead to better throughput for heavier traffic. We do not discuss these results in detail due to the scope and space limitation of this paper.

VIII. Conclusions

We proposed distributed shortcut networks (DSNs), which achieve low-latency communication using low-degree switches, mainly targeting on HPC off-chip interconnects. Our DSN topology features a carefully designed set of various-length shortcuts, which reflects small-world networks in achieving small diameter while ensuring a very economical cable length. We also consider extending our basic DSN topology to a variety of topology designs that have different diameter-vs-degree properties and that also cope with other important issues. They include topologies (with custom routing algorithm) for avoiding deadlocks or avoiding overshoots in routing, or constructions that possess the flexibility in supporting network sizes and switch degrees.

Our graph analyses showed that the proposed basic topology has low diameter and low average shortest path length, which is considerably better than those of a counterpart 2-D torus and near to those of a random topology (DLN-2-2 [3]) with the same average degree. Moreover, the average cable length of the proposed topology is drastically shorter than that of DLN-2-2. It is near to that of the same-degree torus. Our cycle-accurate network simulation showed that the proposed topology reduces latency by 15% and has almost the same throughput when compared to the torus with the same degree.

As with a typical non-random topology, it is possible to exploit the structure of our DSN topologies to create a custom routing algorithm with a natural routing logic. We also proposed a deadlock-free routing algorithm, by which the routing logic at each switch is expected to be simple and small. Our future work will attempt to analyze our custom routing with respect to the strength in creating traffic balance. We will also aim to design a deadlock-free minimal custom routing on DSNs as well as to analyze further on our extended topologies.

Acknowledgments

This work was partially supported by KAKENHI #25280018 and #25730068.

References

[1] K. Scott Hemmert et al, "Report on Institute for Advanced Architectures and Algorithms, Interconnection Networks Workshop 2008," http://ft.ornl.gov/pubs-archive/iaa-ic-2008-workshop-report-final.pdf.

[2] J. Tomkins, "Interconnects: A Buyers Point of View," ACS Workshop, 2007.

[3] M. Koibuchi, H. Matsutani, H. Amano, D. F. Hsu, and H. Casanova, "A Case for Random Shortcut Topologies for HPC Interconnects," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, 2012, pp. 177–188.

[4] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-Driven, Highly-Scalable Dragonfly Topology," in *Proc. of the International Symposium on Computer Architecture (ISCA)*, 2008, pp. 77–88.

[5] P. Coteus and et. al., "Packaging the Blue Gene/L supercomputer," *IBM Journal of Research and Development*, vol. 49, no. 2/3, pp. 213–248, Mar/May 2005.

[6] Y. Ajima, S. Sumimoto, and T. Shimizu, "Tofu: A 6D Mesh/Torus Interconnect for Exascale Computers," *IEEE Computer*, vol. 42, pp. 36–40, 2009.

[7] I. Fujiwara, M. Koibuchi, and H. Casanova, "Cabinet Layout Optimization of Supercomputer Topologies for Shor ter Cable Length," in *Proc. of International Conference on Parallel and Distributed Com puting, Applications and Technologies*, Dec 2012.

[8] Top 500 Supercomputer Sites, http://www.top500.org/.

[9] A. Singla, C.-Y. Hong, L. Popa, and P. B. Godfrey, "Jellyfish: Networking Data Centers Randomly," in *Proc. of USENIX Symposium on Network Design and Implementation (NSDI)*, 2012.

[10] J. Y. Shin, B. Wong, and E. G. Sirer, "Small-World Data Centers," in *Proc. of the Symposium on Cloud Computing*, Oct. 2011.

[11] M. Koibuchi, I. Fujiwara, H. Matsutani, and H. Casanova, "Layout-conscious random topologies for hpc off-chip interconnects," in *19th International Conference on High-Performance Computer Architecture (HPCA)*, Feb. 2013, p. XX.

[12] "Earth simulator project," http://www.jamstec.go.jp/es/en/index.html.

[13] A. Mejia, M. Palesi, J. Flich, S. Kumar, P. López, R. Holsmark, and J. Duato, "Region-based routing: A mechanism to support efficient routing algorithms in nocs," *IEEE Transactions on VLSI Systems*, vol. 17, no. 3, pp. 356–369, 2009.

[14] D. Watts and S. Strogatz, "Collective dynamics of small-world networks," *Nature*, vol. 393, pp. 440–32, 1998.

[15] J. Kleinberg, "The small-world phenomenon: An algorithmic perspective," in *STOC*, 2000.

[16] C. Martel and V. Nguyen, "Analyzing Kleinberg's (and other) smallworld models," in *PODC*, 2004.

[17] M. R. Samatham and D. K. Pradhan, "The De Bruijn Multiprocessor Network: A Versatile Parallel Processing and Sorting Network for VLSI," *IEEE Trans. on Computers*, vol. 38, no. 4, pp. 567–581, 1989.

[18] S. B. Akers, B. Krishnamurthy, and D. Harel, "The Star Graph: An Attractive Alternative to the n-Cube," in *Proc. of the International Conference on Parallel Processing (ICPP)*, 1987, pp. 393–400.

[19] K. Hwang and J. Ghosh, "Hypernet: A communication-efficient architecture for constructing massively parallel computers," *IEEE Trans. on Computers*, vol. 36, no. 12, pp. 1450–1466, 1987.

[20] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[21] HP, "Optimizing facility operation in high density data center environments , technoloogy brief," 2007. [Online]. Available: http://h18004.www1.hp.com/products/servers/technology/whitepapers/datacenter.html

[22] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: a cost-efficient topology for high-radix networks," in *Proc, of the International Symposium on Computer Architecture (ISCA)*, 2007, pp. 126–137.

[23] J. Mudigonda, P. Yalagandula, and J. C. Mogul, "Taming the flying cable monster: a topology design and optimization framework for data-center networks," in *Proc. of the USENIX conference on USENIX annual technical conference*, 2011.

[24] F. Silla and J. Duato, "High-Performance Routing in Networks of Workstations with Irregular Topology," *IEEE Trans. on Parallel Distrib. Syst.*, vol. 11, no. 7, pp. 699–719, 2000.

[25] W. D. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2003.