This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

On Data Center Network Architectures for Interconnecting Dual-Port Servers

Dawei Li and Jie Wu, *Fellow, IEEE*

Abstract—During the past decade, various novel Data Center Network (DCN) architectures have been proposed to meet various requirements of large scale data centers. In existing works that consider server-centric DCN architectures, the lengths of a server-to-server-direct hop and a server-to-server-via-a-switch hop are assumed to be equal. We propose the concept of Normalized Switch Delay (NSD) to distinguish a server-to-server-direct hop and a server-to-server-via-a-switch hop, to unify the design and analysis of server-centric DCN architectures for interconnecting servers with two network interface cards. In [1], the authors claim that BCN is the largest known architecture to interconnect dual-port servers, with diameter 7, given a switch port number. We notice that the existing DPillar [2] architecture accommodates more servers than BCN does under the same configurations. Motivated by this fact, we consider a fundamental problem: maximizing the number of dual-port servers, given network diameter and switch port number; and give an upper bound on this maximum number. Then, we propose three novel architectures that try to achieve this upper bound: SWCube, SWKautz, and SWdBruijn, based on the generalized hypercube, Kautz graph, and de Bruijn graph, respectively. The number of servers that SWCube can accommodate is comparable to that of DPillar. The number of servers that SWKautz and SWdBruijn can accommodate is generally greater than that of DPillar. Compared with three existing architectures, the three proposed architectures, SWCube, SWKautz and SWdBruijn demonstrate various advantages. Analysis and simulations on the newly proposed architectures also show that they have nice properties for DCNs, such as low diameter, high bisection width, good fault-tolerance, and the capability of efficiently handling network congestion.

Index Terms—Data center networks (DCNs), dual-port servers, generalized hypercubes, Kautz graphs, de Bruijn graphs.

1 Introduction

Data centers provide various internet-based/online services, such as search engines, email, online video, social networking, online gaming, and large-scale computations, etc.. Data centers also provide the infrastructure services such as GFS, Bigtable, MapReduce and Dryad [3]–[6]. As the increasing of the service demand will never end, the number of servers in modern and future data centers is required to be very large, for example, hundreds of thousands or millions.

A great challenge in data centers is how to design network architectures to interconnect large numbers of servers. Traditional tree-based architectures have been shown to be difficult in meeting the requirements of Data Center Networks (DCNs). During the past decade, various novel DCN architectures have been proposed. Considering whether the interconnection intelligence is put on the switches or on the servers, these architectures fall into two categories, namely, switch-centric designs and server-centric designs [7]. In switch-centric designs, such as Fat-Tree [8], VL2 [9], HyperX [10], and Flattened Butterfly [11], switch functionality is extended to meet the interconnection need, while servers do not need to be modified for interconnection purposes. Thus, high-end switches may be needed, which significantly increases the interconnection cost. In server-centric designs, such as DCell [12], BCube [13], FiConn [14], HCN & BCN [1], DPillar [2] and MCube [15], switches are only used as cross-bars while servers act as both computing nodes and packet relay nodes. Server-centric designs have the advantage of using only low-end layer-2 switches, thus reducing the cost; also, by putting the interconnection intelligence on servers, they provide a higher degree of programmability. However, an important disadvantage of server-centric designs is that servers usually have much larger processing delays than switches; also, by using servers for forwarding packets, the overhead and/or workload on servers may be increased. While switch-centric architectures are still the main stream; server-centric architectures are also becoming available commercially. Thus, our work on sever-centric architectures will definitely contribute to modern and future DCN designs.

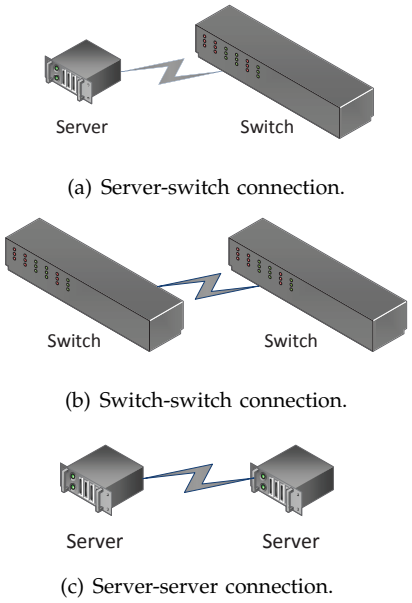
Three basic connections may exist in a data center: server-switch (or switch-server) connections, switch-switch connections and server-server connections, as shown in Fig. 1. In switch-centric-designs, servers usually only connect to a local switch; then, switches connects to other switches to enable remote communications. In other words, only sever-switch and switch-switch connections can exist in a switch-centric design, and server-server connections are usually not allowed. In server-centric designs, servers may have multiple NIC ports; there may exist server-server connections and server-switch connections; and switch-switch connections are usually not allowed.

Considering the number of NIC ports used on servers,

• *Dawei Li and Jie Wu are with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, E-mail: {dawei.li, jiewu}@temple.edu*

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers



(a) Server-switch connection.

(b) Switch-switch connection.

(c) Server-server connection.

Fig. 1. Basic connections in data center networks.

architectures that belong to server-centric designs, can be further classified into two sub-categories. In the first sub-category, servers can have more than 2 ports, such as DCell and BCube; in the second sub-category, servers have no more than 2 ports, such as FiConn, HCN & BCN, DPillar, and MCube. Since Commodity-off-the-Shelf (COTS) servers in data centers often only have 2 NIC ports [1], [2], [14], restricting the server degree in a DCN to be no more than 2, the time and human power needed to upgrade hundreds of thousands of servers can be avoided; also, the packet relay overhead on the servers can be reduced, compared to the case in which more than 2 ports are used on servers.

In this paper, we consider server-centric DCN architecture designs, where servers only have 2 NIC ports, and where only low-end layer-2 switches are used. Obviously, to design DCN architectures, various aspects should be considered, such as the number of servers that an architecture can accommodate, network diameter, bisection width, interconnection cost, and fault-tolerance, etc.. Our goal of designing DCNs is to scale-out the network (increase the number of servers) and maintain or improve the network performance. Our main contributions are as follows:

- First, we notice that in existing works [1], [2], [12]–[14], the lengths of a server-to-server-direct hop and a server-to-server-via-a-switch hop are assumed to be equal. We call this assumption the HOmogeneous Hop (HOH) assumption. However, as servers' packet forwarding capabilities will increase significantly in future DCNs, it may not be suitable to neglect the processing delay at switches [16]. Thus, design and comparison based on this assumption may be inappropriate. For example, in FiConn and BCN, there exist server-to-server-direct hops, while in DPillar, any two servers are not directly connected. In this paper, we propose the concept of Normalized Switch Delay (NSD), which is defined as the switch's packet forwarding delay divided by the server's forwarding delay (when they have no other load), to distinguish these two kinds of sever-to-server hops to unify the design and analysis of server-centric DCN architectures for interconnecting dual-port servers. Specifically, we assume that the length of a server-to-server-via-a-switch hop is counted as $1 + c$, where c is the NSD, $(0 \leq c \leq 1)$, and a server-to-server-direct hop is still counted as 1; thus, we have the HEterogeneous Hop (HEH) assumption. Correspondingly, we can calculate the diameter of the network architecture under this new assumption.
- Second, we ask the following fundamental question: what is the maximum number of dual-port servers that any architecture can accommodate at most, given network diameter d, and switch port number n? Motivated by the *Moore Bound* [17], which provides the upper bound on the number of nodes in a graph given the maximum graph diameter and node degree, we give an upper bound on the maximum number of dual-port servers in a DCN, given network diameter d and switch port number n. In [1], the authors claimed that BCN is the largest known architecture to interconnect dual-port servers, with diameter 7, given a switch port number. We notice that the existing DPillar architecture accommodates more servers than does BCN under the same configurations. Besides, BCN and DPillar still show big gaps between the numbers of dual-port servers that they can accommodate, and the upper bound number.
- Third, we propose three novel DCN architectures which try to approximate the upper bound. The first one is called SWCube, which is based on the generalized hypercube [18]. SWCube accommodates a comparable number of servers to that of DPillar. Specifically, SWCube accommodates less servers than DPillar when the switch port number is small and the network diameter is large, and accommodates more servers than DPillar when the switch port number is large and the network diameter is small. The other two are SWKautz and SWdBruijn, which are based on the Kautz graph [19], [20] and the de Bruijn graph [21], respectively. They always accommodate more servers than DPillar. Then, we compare various DCN architectures on several aspects, namely, the design flexibility, the number of servers given a network diameter, the bisection width, the hardware interconnection cost per server, and the influences of c (NSD) on different architectures under the HEH assumption. Analysis and simulations on the newly proposed architectures reveal that they also have nice properties for DCNs such as high degree of regularity, high bisection

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

width, good fault-tolerance, and efficient handling of network congestion.

The remaining of the paper is organized as follows. Some basic definitions are given in Section 2. In Section 3, we provide the upper bound of the maximum number of dual-port servers that any architecture can accommodate at most, given network diameter d, and switch port number n. We then design three novel architectures that try to approximate this upper bound, which are defined in Sections 4, 5, and 6, respectively. Three existing architectures are reviewed in Section 7. In Section 8, we compare our proposed architectures with the three existing ones in various aspects. Section 9 evaluates our proposed architectures in detail. Conclusions and future work are sketched in Section 10.

2 Preliminaries

In our considerations, all switches are low-end COTS ones. We assume that all servers only have two NIC ports. To construct large DCNs, we assume that the switch port number, n is at least 4. We focus on server-centric designs; thus, switch-to-switch-direct connections are not considered in our design and analysis. Since there are two kinds of nodes, namely, servers and switches in a DCN, some concepts should be made clear as compared to those in a traditional graph. Before further discussion, we give some definitions.

We define that a *hop* is a path, from one node to another node of the same kind, which consists of no other nodes of the same kind. Thus, we have *switch-to-switch* hops and *server-to-server* hops. According to our assumption, there does not exist *switch-to-switch-direct* hops. Server-to-server hops consist of *server-to-server-direct* hops and *server-to-server-via-a-switch* hops. The *length* of a path between two servers is the number of server-to-server-direct hop(s), plus $1 + c$ times the number of server-to-server-via-a-switch hop(s) in the path. Again, c can be regarded as the *Normalized Switch Delay (NSD)*, which is the switch's packet forwarding delay divided by the server's forwarding delay. We explicitly consider c because switch's forwarding delay may not be neglected in modern and future DCNs. Besides, c is assumed to be less than or equal to 1, because we predict that servers' packet forwarding ability will not increase to the extent of outperforming switches. The *distance* of two servers is the length of the shortest path between the two servers. The *diameter* of a DCN architecture is the maximum distance among all pairs of servers. The distance between servers and network diameter are critical factors on the communication delay/latency in DCN, which is an important metric for DCN design. The *bisection width* of a DCN architecture is the minimum number of links that must be removed to partition the network into two equal halves. The bisection width of a DCN provides key information on the potential throughput that the network can have, and thus, has great influences on the performances of the network itself and the applications that are hosted in the data center.

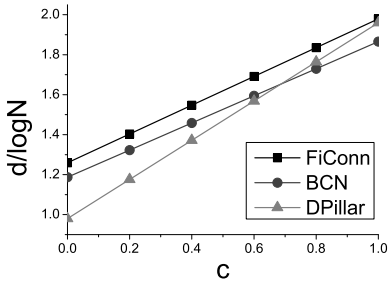


Fig. 2. Scaled diameter of FiConn, BCN, and DPillar for different c values.

To give a preview of the influence of NSD, we compare the diameter of three existing architectures, namely, FiConn, BCN, and DPillar, when different values are chosen for c, given the same switch port number $n = 48$, server degree 2, and approximately equal numbers of servers. Readers may refer to Section 7 for details on FiConn, BCN, and DPillar. We use approximately equal numbers of servers because it is almost impossible for these architectures to have exactly the same number of servers under any configurations. We choose: FiConn$(48, 2)$ with 361,200 servers and diameter 7, BCN$(32, 16, 1, 1)$ with 787,968 servers and diameter 7, and DPillar$(48, 4)$ with 1,327,104 servers and diameter 6. The initial diameter values are calculated assuming $c = 0$. As we can see, the numbers of servers that the three architectures have still differ from each other significantly. Thus, we calculate the *scaled diameter*, which is the diameter divided by the logarithm of the number of servers of an architecture, when c chooses different values. As shown in Fig. 2, under the HOH assumption, DPillar has the lowest scaled diameter; as c increases, the scaled diameter of BCN tends to be comparable with, or even lower than, that of DPillar. This tendency is intuitively correct because, in DPillar, all server-to-server hops are server-to-server-via-a-switch hops, while in BCN, lots of server-to-server-direct hops exist. As c increases, server-to-server-via-a-switch hops will contribute more to the diameter.

3 Maximizing the Number of Servers Given Network Diameter and Switch Port Number

A basic idea of designing DCN architectures is to scale-out the network (increase the number of servers) and maintain or improve the network performance at the same time. Although the latter (the network performance) itself includes many aspects and is not easy to express explicitly, traditional graph theory can be applied to address the former (increase the number of servers). This aspect motivates us to ask the following fundamental question: what is the maximum number of dual-port servers that an architecture can accommodate,

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

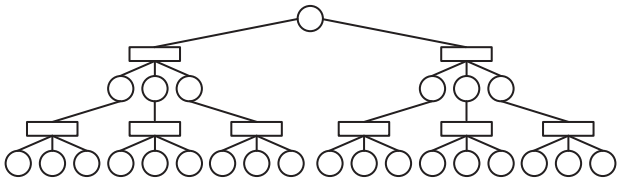


Fig. 3. The architecture greedily constructed to maximize the number of servers has a diameter $d = 2(1+c)l, l = 2$.

given network diameter d, and switch port number n? A similar problem in traditional graph theory is the degree/diameter problem: find graphs with a maximal number of nodes with given constraints of maximum degree δ and diameter d. Compared to the traditional graph model (where only one kind of node exists), in a DCN, two kinds of nodes exist. Given the server degree 2, some related work in the traditional graph theory can be applied for analyzing DCNs. The Moore Bound gives an upper bound for the degree/diameter problem.

Moore Bound: The maximum number of nodes in a graph, given diameter constraint d and node degree δ is

$$\begin{aligned} N &\leq 1+\delta+\delta(\delta-1)+\cdots+\delta(\delta-1)^{d-1} \\ &= 1+\delta\sum_{i=0}^{d-1}(\delta-1)^i. \text{ [17]} \end{aligned} \quad (1)$$

Illustration: Any node can reach at most δ other nodes within distance 1. Each of the δ nodes can reach another $\delta-1$ nodes within distance 2, because one degree has already been used for connecting to the original node. Extending to distance d, the upper bound on the maximum number can be calculated.

Reverting to our DCN architecture scenario, we start with the situation when $c = 0$, which is the HOH assumption that all existing works adopted.

Theorem 1: For $c = 0$, given switch port number n, $(n \geq 4)$, the maximum number of dual-port servers that any DCN architecture, with diameter less than or equal to d (d is a positive integer), can accommodate is

$$N_v \leq N_v^{ub} = \frac{2(n-1)^{d+1}-n}{n-2}. \quad (2)$$

For the HEH assumption, where $0 \leq c \leq 1$, we have:

Theorem 2: Given switch port number n, $(n \geq 4)$, the maximum number of dual-port servers that any DCN architecture, with diameter less than or equal to d (d is an arbitrary positive number), can accommodate is

$$N_v \leq N_v^{ub} = \frac{2(n-1)^{\lceil \frac{d}{1+c} \rceil+1}-n}{n-2}. \quad (3)$$

The proofs for Theorems 1 and 2 can be found in the Supplemental Material.

However, like the Moore Bound, the upper bound may not be achievable. Consider a graph greedily constructed to maximize the network order within l server-to-server-via-a-switch hops, as shown in Fig. 3. Notice that we consistently use rectangles to represent switches, and circles to represent servers in all DCN architectures. The network in Fig. 3 accommodates at most $N_v = (2(n-1)^{l+1}-n)/(n-2)$ servers. However, this network actually has a diameter $d = 2(1+c)l$. In terms of d and n, $N_v \leq (2(n-1)^{\lceil d/(2(1+c)) \rceil+1}-n)/(n-2)$, which is much less than the upper bound.

As we can notice, when $c = 0$, the upper bound is approximately $2n^d$. The numbers of servers that three existing architectures for interconnecting dual-port servers, namely, FiConn, BCN, and DPillar can accommodate show big gaps between the upper bound. In traditional graphs, a d-dimensional r-ary generalized hypercube has diameter d and network order (the number of nodes in a network) r^d; a Kautz graph with $r+1$ symbols and diameter d has network order r^d+r^{d-1}; a d-dimensional r-ary de Bruijn graph has diameter d and network order r^d. These facts motivate us to design larger order DCN architectures, based on the generalized hypercube, the Kautz graph, and the de Bruijn graph. The following three sections present our three novel DCN architectures: SWCube, SWKautz, and SWdBruijn. When calculating the diameter of SWCube, SWKautz, SWdBruijn, we assume $c = 0$. Since server-to-server hops are all server-to-server-via-a-switch hops in SWCube, SWKautz, and SWdBruijn, the actual diameter is $1+c$ times the diameter calculated assuming $c = 0$.

4 SWCUBE

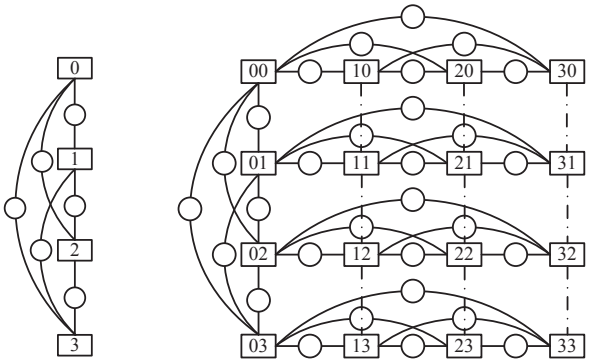
4.1 The Generalized Hypercube and SWCube Construction

We denote a k-dimensional generalized hypercube [18] by $H^k_{r_1\times r_2\times\cdots\times r_k}$. The ith dimension is with radix r_i, $\forall i = 1, 2, \cdots, k$. A node W is represented by a k-tuple: $W = w_1w_2\cdots w_k$, where $0 \leq w_i \leq r_i-1$, $\forall i = 1, 2, \cdots, k$. Two nodes are connected directly by an edge if and only if their addresses differ at one bit. We consider that a set of nodes are along the same dimension i, if all of their addresses differ only at the ith bit. We can see that nodes along the same dimension form a complete graph.

We design a novel DCN architecture for interconnecting dual-port servers based on the generalized hypercube. The new architecture can be constructed logically as follows: 1.) replace the nodes in the original generalized hypercube with switches; 2.) insert one server into each edge that connects two switches. We denote the resulting DCN architecture by SWCube($r_1 \times r_2 \times \cdots \times r_k, k$) because the SWitches form a generalized hyperCube, $H^k_{r_1\times r_2\times\cdots\times r_k}$. Fig. 4(a) and Fig. 4(b) show a 1-dimensional and a 2-dimensional SWCube, respectively, where $r_1(= r_2) = 4$. Note that in Fig. 4(b), the interconnections of switches and servers along the 2nd, 3rd, and 4th columns are represented by dotted lines.

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers



(a) 1D SWCube. (b) 2D SWCube.

Fig. 4. SWCube.

Since we replace nodes in the original hypercube with switches, switches in SWCube can adopt the same addressing scheme for nodes in the original generalized hypercube. In SWCube, each server is uniquely identified by the two switches that it directly connects to. Thus, we represent a server by $V = (V^1, V^2)$, where $V^1 = v_1^1 v_2^1 \cdots v_k^1$ and $V^2 = v_1^2 v_2^2 \cdots v_k^2$ represent the two switches that the server directly connects to. Since, in the original generalized hypercube formed by the switches, two switches are adjacent if and only if they differ at one bit, for each server, there exists only one $i \in \{1, 2, \cdots, k\}$ such that $v_i^1 \neq v_i^2$.

4.2 Properties of SWCube

The number of switches in an SWCube($r_1 \times r_2 \times \cdots \times r_k, k$) is $N_w = \prod_{i=1}^{k} r_i$. Since switches along the same dimension form a complete graph, each switch connects to the other $r_i - 1$ switches via a server along the ith dimension. Thus, the number of ports that are used in each switch is: $n = \sum_{i=1}^{k}(r_i - 1)$. The number of servers in an SWCube is actually the number of edges in the original generalized hypercube, which can be calculated as the number of switches N_w, times the switch port number n, divided by 2: $N_v = (\prod_{i=1}^{k} r_i)(\sum_{i=1}^{k}(r_i - 1)/2)$.

Theorem 3: The bisection width of SWCube($r_1 \times r_2 \times \cdots \times r_k, k$) is $\min_{m=1}^{k}(1/4) r_m \prod_{i=1}^{k} r_i$.

Proof: We borrow the idea from [10] and prove the theorem as follows. The bisection width of SWCube($r_1 \times r_2 \times \cdots \times r_k, k$) is realized by cutting one of its dimensions in half. If dimension m is cut, we consider the number of switches of the SWCube without extending in dimension m, which is $\prod_{i=1, i\neq m}^{k} r_i$. Cutting a complete graph with r_m nodes requires removing $(r_m/2)(r_m/2)$ edges. Cutting SWCube($r_1 \times r_2 \times \cdots \times r_k, k$) in dimension m requires cutting $\prod_{i=1, i\neq m}^{k} r_i$ complete graphs with r_m nodes. Thus, the bisection of such a cut is $(r_m/2)(r_m/2)\prod_{i=1, i\neq m}^{k} r_i = (1/4) r_m \prod_{i=1}^{k} r_i$. If $(1/4) r_m \prod_{i=1}^{k} r_i$ is the smallest among all the dimensions of SWCube($r_1 \times r_2 \times \cdots \times r_k, k$), then, $(1/4) r_m \prod_{i=1}^{k} r_i$ determines the bisection width of SWCube($r_1 \times r_2 \times \cdots \times r_k, k$). Thus, this theorem holds. Notice that, we do not consider the case when r_m is indivisible by 2, since this only incurs some rounding(s) on $r_m/2$. □

For symmetry and regularity, we can choose $r_1 = r_2 = \cdots = r_k = r$. We denote the constructed architecture by SWCube(r, k). According to Theorem 3, the bisection width of SWCube(r, k) is $r^{k+1}/4$. When the number of ports used in a switch is $n = 8$, we can choose $r = 5, k = 2$. SWCube($5, 2$) can accommodate a total of 100 servers. When the number of ports used is $n = 16$, we can choose $r = 5, k = 4$. SWCube($5, 4$) can accommodate a total of 5000 servers. Fig. 4(a) and Fig. 4(b) represent an SWCube($4, 1$) and an SWCube($4, 2$), respectively; the numbers of ports used in each switch are 3 and 6, respectively.

We say that two servers $S = (S^1, S^2)$ and $D = (D^1, D^2)$ are along the same dimension if and only if the four switches, which the two servers connect to, S^1, S^2, D^1 and D^2, differ at most one bit.

Lemma 1: The distance of two servers that are along the same dimension is at most 2.

Proof: As the SWCube is constructed, if two servers $S = (S^1, S^2)$ and $D = (D^1, D^2)$ are along the same dimension, the four switches they connect to, S^1, S^2, D^1 and D^2 are also along the same dimension. Since all switches along the same dimension form a complete graph (disregarding the servers), S^1 can reach D^1 via one intermediate server; or S^1 and D^1 are the same switch. Thus, the shortest path between S and D is no greater than $S \rightarrow S^1 \rightarrow (S^1, D^1) \rightarrow D^1 \rightarrow D$, where S^1 and D^1 represent switches, and (S^1, D^1) represents the intermediate server, if it exists. Thus the shortest path contains at most 3 servers, and the distance between S and D is at most 2. □

Lemma 2: The distance of two servers that are not along the same dimension is at most $k + 1$.

Proof: Consider servers $S = (S^1, S^2)$ and $D = (D^1, D^2)$ again. Since all switches form a generalized hypercube, each switch, say S^1, can reach another, say D^1, at most k hops by correcting one bit via each switch-to-switch-via-server hop. The total number of servers in such a path from S^1 to D^1 is at most k. Including S and D themselves in a path from S to D, there are at most $k + 2$ servers in the path from S to D. Thus, the distance between two servers is at most $k + 1$. □

Theorem 4: The diameter of an SWCube(r, k) is $d = k + 1$.

Proof: The theorem directly follows from Lemma 1 and Lemma 2. □

Theorem 5: In terms of network diameter and switch port number, the number of servers in an SWCube(r, k) is

$$N_v = \frac{k r^k (r-1)}{2} = \frac{n(\frac{n}{d-1} + 1)^{d-1}}{2}. \quad (4)$$

Proof: The number of switches in an SWCube(r, k) is $N_w = r^k$. The number of ports that are used on each switch is $n = k(r - 1)$. Since $d = k + 1$, $r = n/(d - 1) + 1$.

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

The number of servers in SWCube(r,k) is $N_v = kr^k(r-1)/2 = n(n/(d-1)+1)^{d-1}/2$. □

Given a switch port number n, for a regular SWCube, $n = k(r-1)$, where k and $r-1$ are positive integers. For $n = 16$, k can be 1, 2, 4, 8 and 16. Table 1 shows the choices of k and corresponding other values, given switch port number $n = 16$. Notice that a huge gap on the total number of servers exists between the $k = 8$ column and the $k = 16$ column. This gap results from the assumption that all r_i's are equal; in practical designs, this assumption is not necessary, and r_i's and k can choose more flexible values to accommodate the desired number of servers and to meet other requirements.

TABLE 1
Choices of k given $n = 16$

k	1	2	4	8	16
r	17	9	5	3	2
d	2	3	5	9	17
N_w	17	81	625	6561	65536
N_v	136	648	5000	52488	524288

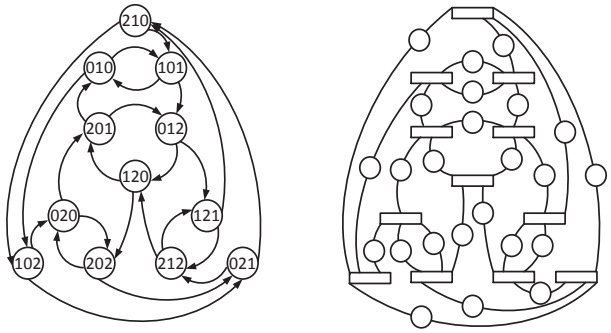
5 SWKautz

5.1 The Kautz Graph and SWKautz Construction

A k-dimensional Kautz directed graph [19], [20] with $r+1$ symbols is denoted by KA(r,k). The node set of KA(r,k) is given by all possible strings of length k where each symbol of the string is from the set $Z = \{0,1,2,\cdots,r\}$ with the restriction that two consecutive symbols of the string are always different. In other words, a string $w_1w_2\cdots w_k$ can represent a node in a KA(r,k) if $w_i \in Z, \forall 1 \le i \le k$ and $w_i \neq w_{i+1}$ $\forall 1 \le i \le k-1$. There exists a directed edge from node $W^1 = w_1^1w_2^1\cdots w_k^1$ to node $W^2 = w_1^2w_2^2\cdots w_k^2$ if and only if W^2 is a left-shifted version of W^1, i.e., $w_2^1w_3^1\cdots w_k^1 = w_1^2w_2^2\cdots w_{k-1}^2$, and $w_k^2 \neq w_{k-1}^2$. A 3-dimensional Kautz graph with 3 symbols is illustrated in Fig. 5(a).

The total number of nodes in a KA(r,k) graph is $(r+1)r^{k-1} = r^k + r^{k-1}$ because the first symbol of a string representing a node has $r+1$ choices, while the other $(k-1)$symbols have r choices in order to make sure two consecutive symbols of every string are not equal. The network diameter of KA(r,k) is k. Each node is with an indegree r and an outdegree that is also r.

We construct a DCN architecture for interconnecting dual-port servers based on a k-dimensional Kautz graph with $n/2+1$ symbols; in other words, $r = n/2$, where n is the switch port number. The new architecture can be constructed logically as follows: 1.) replace each node in the original KA$(n/2,k)$ graph with an n-port switch; 2.) remove the direction of all the edges and insert a server into each edge. The architecture constructed as such is named SWKautz$(n/2,k)$ because the SWitches form a Kautz graph, KA$(n/2,k)$. Fig. 5(b) shows a SWKautz$(2,3)$, whose base Kautz graph is KA$(2,3)$ shown in Fig. 5(a). In SWKautz$(2,3)$, the switch port number is 4.



(a) KA$(2,3)$. (b) SWKautz$(2,3)$.

Fig. 5. The Kautz graph and SWKautz.

Since the SWKautz architecture is constructed by replacing each node in the Kautz graph with an n-port switch, the addressing or labeling scheme of the switches can be identical to that of the nodes in the original Kautz graph. That is to say, each switch is represented by a k-digit string $w_1w_2\cdots w_k$, where, $w_i \in \{0,1,\cdots,n/2\}, \forall 1 \le i \le k$ and $w_i \neq w_{i+1}, \forall 1 \le i \le k-1$. Note that there may be two servers connecting two same switches W^1 and W^2, since in the original Kautz graph, there may exist a directed edge from W^1 to W^2 and a directed edge from W^2 to W^1 at the same time. Thus, we use an ordered pair (W^1, W^2) to represent a server, W; i.e., $W = (W^1, W^2)$. W^1 and W^2 are the server W's left switch and right switch, respectively. We also say that the left switch, W^1, of W is W's *home switch*. Note that (W^2, W^1) represents a different server than does (W^1, W^2).

5.2 Properties of SWKautz

The bisection width of a k-dimensional Kautz graph with $r+1$ symbols is $\Theta(r^{k+1}/k)$ [22]. The following theorem directly follows from this fact.

Theorem 6: The bisection width of an SWKautz$(n/2,k)$ is $\Theta((n/2)^{k+1}/k)$.

Theorem 7: The diameter of an SWKautz$(n/2,k)$ is $d = k+1$.

Proof: Consider two servers $S = (S^1, S^2)$ and $D = (D^1, D^2)$, where S^1 and S^2 are the switches that server S connects to, and D^1 and D^2 are the switches that server D connects to. Since all switches form a k-dimensional Kautz graph, while a k-dimensional Kautz graph has diameter k, the longest path from any switch, say S^1, to another, say D^1, is at most k switch-to-switch-via-server hops. Thus the longest path from S to D includes at most $k+2$ servers. The network diameter of SWKautz$(n/2,k)$ is $d = k+1$. □

Theorem 8: In terms of network diameter and switch port number, the number of servers in an SWKautz$(n/2,k)$ is

$$N_v = (\frac{n}{2})^d + (\frac{n}{2})^{d-1}. \quad (5)$$

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

Proof: The number of switches in an SWKautz$(n/2, k)$ is equal to the number of nodes in the original Kautz graph, and can be calculated as: $N_w = (n/2)^k + (n/2)^{k-1}$. The number of servers in an SWKautz$(n/2, k)$ is equivalent to the number of edges in the original Kautz graph, which can be calculated as: $N_v = n((n/2)^k + (n/2)^{k-1})/2 = (n/2)^{k+1} + (n/2)^k = (n/2)^d + (n/2)^{d-1}$. □

6 SWdBruijn

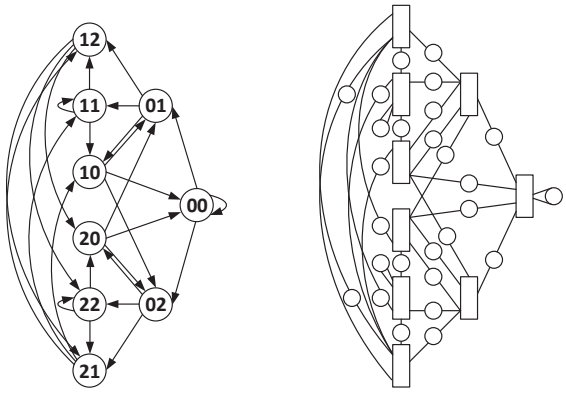
6.1 de Bruijn Graph and SWdBruijn Construction

An k-dimensional r-ary de Bruijn graph [21] consists of r^k nodes. Each node is represented by a k-digit string, $w_1w_2\cdots w_k$, where $0 \le w_i \le r-1$, $\forall i = 1, 2, \cdots, k$. There exists a directed edge from node $W^1 = w_1^1 w_2^1 \cdots w_k^1$ to node $W^2 = w_1^2 w_2^2 \cdots w_k^2$ if and only if W^2 is a left-shifted version of W^1, i.e., $w_2^1 \cdots w_k^1 = w_1^2 w_2^2 \cdots w_{k-1}^2$; under this condition, if W^2 is identical to W^1, we say this node (W^2 or W^1) has a directed self circle. A 2-dimensional 3-ary de Bruijn graph is illustrated in Fig. 6(a). We can notice that node 00 has a directed self circle. Actually, for all of the nodes $w_1w_2\cdots w_r$, where $w_1 = w_2 = \cdots = w_r$, there exists such a directed self circle.

The k-dimensional r-ary de Bruijn graph has diameter $d = k$. Each node is with indegree r and outdegree r, as well. The number of edges is $(2r)r^k/2 = r^{k+1}$.

We construct a novel DCN architecture based on a k-dimensional $n/2$-ary de Bruijn graph; in other words, $r = n/2$, where n is the switch port number. The new architecture can be constructed logically as follows: 1.) replace each node in the original k-dimensional $n/2$-ary de Bruijn graph with an n-port switch; 2.) remove the direction of all the edges and insert a server into each edge. By doing these, the nodes with a self circle in the original de Bruijn graph will connect to $n-1$ servers, with 2 ports connecting to the same server and the other $n-2$ ports connecting to $n-2$ other servers. Of course, we can remove one edge that the switch uses to connect the same server and connect another server using the left-over port. By doing this, the network can support an additional n servers; compared to $(n/2)^k$, this number is negligible. Thus, we do not choose to add these n servers in order to keep the regularity of the addressing scheme. The structure constructed as such is denoted by SWdBruijn$(n/2, k)$ because the SWitches form a k-dimensional $n/2$-ary de Bruijn graph. Fig. 6(b) shows an SWdBruijn$(3, 2)$.

Since the SWdBruijn architecture is constructed by replacing each node in the de Bruijn graph with an n-port switch, the addressing or labeling scheme of the switches can be identical to that of the nodes in the original de Bruijn graph. Thus, each switch is represented by a k-digit string $w_1w_2\cdots w_k$, where $0 \le w_i \le n/2-1$, $\forall 1 \le i \le k$. Note that there may be two servers connecting the two same switches W^1 and W^2, since in the original de Bruijn graph, there may exist a directed edge from W^1 to W^2 and a directed edge from W^2 to W^1 at the same time. Thus, we use an ordered pair (W^1, W^2) to represent a server, W; i.e., $W = (W^1, W^2)$. W^1 and W^2 are server W's left switch and right switch, respectively. Note that (W^2, W^1) represents a different server than does (W^1, W^2).



(a) de Bruijn ($r = 3, k = 2$). (b) SWdBruijn$(3, 2)$.

Fig. 6. The de Bruijn graph and SWdBruijn.

6.2 Properties of SWdBruijn

As readers may have already noticed, the definitions for Kautz graph and de Bruijn graph are similar. The constructions of SWKautz and SWdBruijn are also similar. Thus, SWKautz and SWdBruijn share many similar characteristics and properties. In this subsection, we just list some important characteristics of SWdBruijn, omitting the explanations and proofs.

- The bisection width of an SWdBruijn$(n/2, k)$ is $\Theta((n/2)^{k+1}/k)$.
- The diameter of an SWdBruijn$(n/2, k)$ is $d = k+1$.
- In terms of network diameter, d, and switch port number, n, the number of servers in an SWdBruijn$(n/2, k)$ is $N_v = (n/2)^d$.

7 Related Existing Works

In this section, we review three main DCN architectures that also consider interconnecting dual-port servers. Notice that MCube [15] only uses 6-port switches; thus, its application is very limited. Besides, the number of servers that an MCube can accommodate is much less than those of FiConn, BCN and DPillar; thus, we do not consider MCube in our paper. As we have mentioned, all existing works adopt the HOH assumption, in other words, $c = 0$.

7.1 FiConn

FiConn [14] is a recursively defined architecture. FiConn$(n, 0)$ is the basic construction unit, which consists of n servers and an n-port switch connecting them. If there are a total of b servers with one port remaining in a FiConn$(n, k-1)$ $(k > 0)$, the number of

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

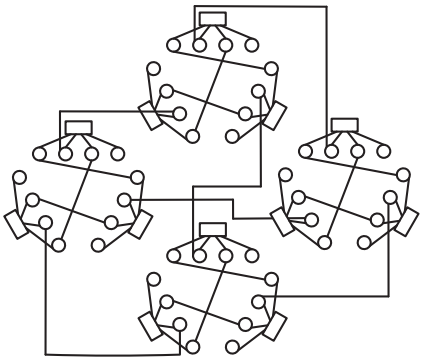
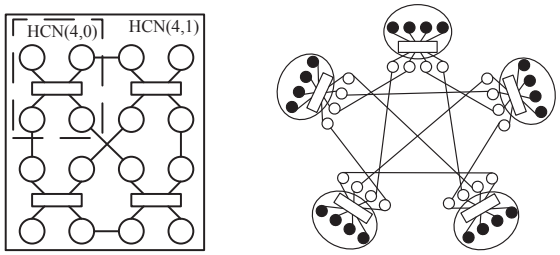


Fig. 7. FiConn$(4,2)$.



(a) HCN. (b) BCN.

Fig. 8. HCN and BCN.

FiConn$(n,k-1)$'s in a FiConn(n,k) is equal to $b/2+1$. In each FiConn$(n,k-1)$, $b/2$ servers out of the b servers with one port remaining are selected to connect the other $b/2$ FiConn$(n,k-1)$'s using their second ports, each for one FiConn$(n,k-1)$. Fig. 7 shows a FiConn$(4,2)$, which consists of four FiConn$(4,1)$'s.

The diameter of a FiConn(n,k) is $d=2^{k+1}-1$. The number of servers in a FiConn(n,k) is $N_v \geq 2^{k+2}(n/4)^{2^k}$, which, in terms of d and n, can be represented as follows:

$$N_v \geq 2^{\log_2(d+1)+1}(n/4)^{(d+1)/2} = 2(d+1)(n/4)^{(d+1)/2}.$$

The number of switches in a FiConn(n,k) is $N_w = N_v/n$. The average server degree in FiConn(n,k) is $2-1/2^k$. Table 2 provides some feasible k and d values for FiConn.

TABLE 2
k, d values for FiConn

k	0	1	2	3	4
d	1	3	7	15	31

7.2 HCN & BCN

HCN [1] is also a recursively defined architecture. A high-level HCN(n,h) employs a low level HCN$(n,h-1)$ as a unit cluster, and connects many such clusters by means of a complete graph. HCN$(n,0)$ is the smallest module, which consists of n dual-port servers and an n-port switch. For each server, its first port is used to connect to the switch, and its second port is used to interconnect with another server in different smallest modules for constituting larger networks. An HCN(n,i) $(i>0)$ is formed by n HCN$(n,i-1)$'s and has n servers that still have one remaining port, each in an HCN$(n,i-1)$ for further expansion. Fig. 8(a) shows an HCN$(4,1)$ which connects four HCN$(4,0)$'s.

A BCN architecture can be represented by BCN(α,β,h,γ), where h denotes the level of BCN in the first dimension, and γ denotes the level of a BCN, which is selected as the unit cluster in the second dimension. When $0 \leq h < \gamma$, the BCN(α,β,h,γ) is simply BCN(α,β,h), which is the same as an HCN(α,h), except that in each of the smallest unit BCN$(\alpha,\beta,0)$'s, there are β slave servers that can be used to expand the network in the second dimension. $\alpha=n-\beta$ is the number of master servers that can be used to expand the network in the first dimension in a BCN$(\alpha,\beta,0)$. The shortest path length among all of the server pairs in BCN(α,β,h) is at most $d=2^{h+1}-1$. Actually, in order to maximize the number of dual-port servers, the case when $h<\gamma$ needs not to be considered, since an HCN(n,h) will have more servers than a BCN(α,β,h).

When $h \geq \gamma \geq 0$, the shortest path length among all of the server pairs in BCN(α,β,h,γ) is at most $d=2^{h+1}+2^{\gamma+1}-1$. The number of servers is $N_v = \alpha^{h-\gamma}(\alpha^\gamma(\alpha+\beta)(\alpha^\gamma\beta+1))$. Apparently, in this case, $N_v \leq n^{h-\gamma}n^\gamma n(n^{\gamma+1}+1) \leq n^{h+\gamma+3}$. Since $d=2^{h+1}+2^{\gamma+1}-1 \geq 2\sqrt{2^{h+1}\cdot 2^{\gamma+1}}-1$, we have $h+\gamma+2 \leq 2\log_2((d+1)/2)$. Thus, $N_v \leq n^{2\log_2((d+1)/2)+1}$. The number of switches is $N_w = N_v/n$. All slave servers have degree 2, $\alpha(\alpha^\gamma\beta+1)$ master servers have degree 1, while all other master servers have degree 2. Thus, the average server degree is $2-1/(\alpha^{h-1}n)$. Since HCN is just a special case of BCN, and has a known small network order, we focus on BCN only. More details on HCN & BCN can be found in [1]. Table 3 provides some feasible h, γ and d values for BCN. Fig. 8(b) shows a BCN$(4,4,0,0)$.

TABLE 3
h, γ, d values for BCN

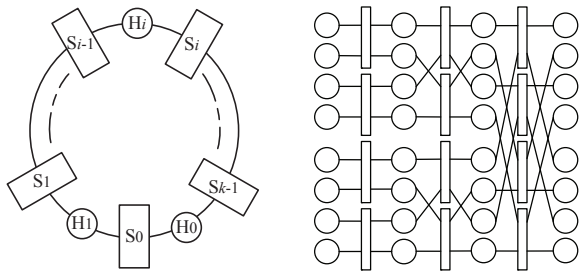
d	$h=0$	$h=1$	$h=2$	$h=3$
$\gamma=0$	3	5	9	17
$\gamma=1$	1	7	11	19
$\gamma=2$	1	3	15	23
$\gamma=3$	1	3	7	31

7.3 DPillar

The DPillar architecture, which is based on the butterfly network, can be represented by DPillar(n,k), which consists of k server columns and k switch columns; H_i and S_i $(0 \leq i \leq k-1)$ represent server and switch columns, respectively. The server and switch columns are alternately placed along a cycle, as shown in Fig. 9(a). A server in each server column is connected to the two switches in its two neighboring switch columns. For a switch in column S_i, half of its n ports are connected

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers



(a) Vertical view. (b) Horizontal view.

Fig. 9. DPillar.

to $n/2$ servers in H_i, and the other half are connected to $n/2$ servers in $H_{i+1 \mod k}$. Each server column has $(n/2)^k$ servers; each switch column has $(n/2)^{k-1}$ switches. Fig. 9(b) shows a DPillar$(4,3)$.

The diameter, the number of switches in a DPillar(n,k) are $d = k + \lfloor k/2 \rfloor$, $N_w = k(n/2)^{k-1}$, respectively. The number of servers in a DPillar(n,k) is $N_v = (2d/3)(n/2)^{2d/3}$, when k is even and $N_v = ((2d+1)/3)(n/2)^{2d/3}$, when k is odd. Every server's degree is 2, because both of its two ports are used. Table 4 provides some feasible k and d values for DPillar.

TABLE 4
k, d values for DPillar

k	1	2	3	4	5	6	7	8	9	10	11	12	13
d	1	3	4	6	7	9	10	12	13	15	16	18	19

8 On the Comparison of Various Architectures

We compare various architectures in several aspects. The first one is the design flexibility of different architectures; our discussion focuses on the choices of network diameters assuming $c = 0$. The second is the fundamental one: the number of servers that an architecture can accommodate, given the same configurations. The third aspect is the bisection width. The fourth aspect is the interconnection cost per server in an architecture. Last, we investigate the influence of c values on the architectures.

8.1 Design Flexibility

As a recall, the network diameters of FiConn and BCN are $d = 2^k - 1 (k \geq 0)$ and $d = 2^{h+1} + 2^{\gamma+1} - 1 (h \geq \gamma \geq 0)$, respectively. Thus, FiConn and BCN allow very limited network diameter values, due to the integer constraints of k for FiConn, and h and γ for BCN. In a regular SWCube where $r_1 = r_2 = \cdots = r_k$, $n = k(r-1) = (d-1)(r-1)$, it is only required that $d-1$ is a divisor of n. As we have mentioned, in practice, r_is can differ from each other. Thus, SWCube allows flexible choices of diameter values. DPillar also allows flexible choices of diameters, in that d only needs to be in the form of $k + \lfloor k/2 \rfloor$. SWKautz and SWdBruijn allow the most flexible choice of network diameters because they can choose arbitrary positive integer diameters independent of switch port number, and their architectures themselves do not incur additional constraints.

8.2 The Number of Servers Given d and n

As we have shown, FiConn and BCN allow very few options for d. Besides, the number of servers that FiConn accommodates is approximately $2(d+1)(n/4)^{d/2}$, which is strictly less than that of DPillar, SWCube, SWKautz, and SWdBruijn. The number of servers BCN accommodates is less than $n^{2\log_2((d+1)/2)+1}$, which is also less than that of DPillar, SWCube, SWKautz, and SWdBruijn for most n and d values. We do not include FiConn and BCN for comparison in Fig. 10.

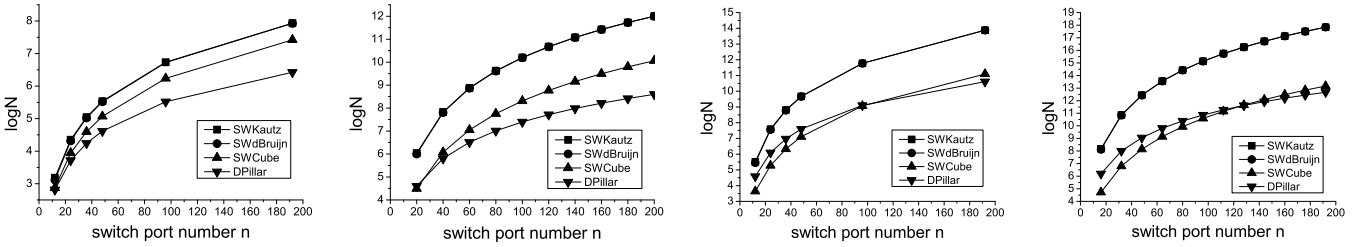
We choose four typical diameter values for comparing DPillar, SWCube, SWKautz, and SWdBruijn, $d = 4, 6, 7, 9$. When $d = 4$ and $d = 7$, we vary $n = 12, 24, 36, 48, 96, 192$. When $d = 6$, we vary $n = 20, 40, \cdots, 200$. When $d = 9$, we vary $n = 16, 32, 48, \cdots, 192$. Results for different d values are shown in Fig. 10(a), Fig. 10(b), Fig. 10(c), and Fig. 10(d), respectively. As we can see, for small d values, SWCube can accommodate more servers than DPillar. For large d values, DPillar will exceed SWCube when n is small; when n is sufficiently large, SWCube still accommodates more servers than DPillar. Under various configurations, SWKautz and SWdBruijn almost always outperform DPillar and SWCube in terms of maximizing the number of servers. The numbers of servers that SWKautz and SWdBruijn accommodate are the most close to the upper bound, even under the HEH assumption when $c \neq 0$. Also notice that, the numbers of servers that SWKautz and SWdBruijn can accommodate are almost overlapped.

8.3 Bisection Width

In this subsection, we compare the bisection width of various architectures given a network diameter d, and the switch port number n. The bisection width of FiConn is at most $\Theta(N_v/d)$ [14]. The bisection width of BCN$(\alpha, \beta, h, \gamma)$ for $h \geq \gamma$ is $\alpha^h(\alpha^\gamma\beta + 2)\beta/4$ [1], which is $\alpha^{h+\gamma}\beta^2(1 + 2/(\alpha^\gamma\beta))/4 \leq n^{h+\gamma+2} \leq n^{2\log_2(d+1)/2}$. The bisection width of DPillar(n,k) is $(n/2)^k \approx N_v/(2d/3)$ [2]. Based on Theorems 3, 4, 6, and 8, we can see that the bisection widths of SWCube is approximately $N_v/(2(d-1))$, and the bisection width of SWKautz is $\Theta(N_v/(d-1))$. Again, SWdBruijn has a similar bisection width as that of SWKautz's, i.e., $\Theta(N_v/(d-1))$. Notice that, in the above expressions, N_v is the number of servers that the corresponding architecture can accommodate. Since DPillar, SWCube, SWKautz, and SWdBruijn can accommodate more servers that those of FiConn, the bisection width of DPillar, SWCube, and SWKautz are also higher. The bisection width of BCN is in the order of $n^{2\log_2(d+1)/2}$, while the bisection widths

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

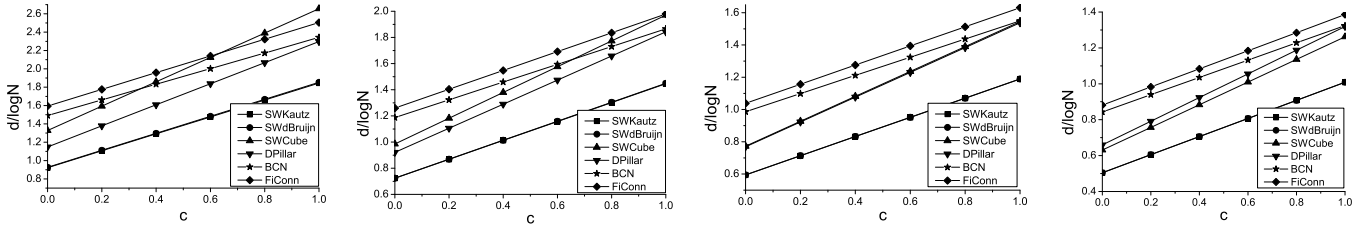


(a) $d = 4$. (b) $d = 6$. (c) $d = 7$. (d) $d = 9$.

Fig. 10. Number of servers given network diameter and various switch port numbers.

TABLE 5
Hardware Interconnection Cost Comparison

	FiConn(n,k)	BCN(α,β,h,γ)	DPillar(n,k)	SWCube(r,k)	SWKautz$(n/2,k)$	SWdBruijn$(n/2,k)$
N_w/N_v	1/n	1/n	2/n	2/n	2/n	2/n
average server degree	$2-1/2^k$	$2-1/(\alpha^{h-1}n)$	2	2	2	2
cost per server	$P_w/n+P_l(2-1/2^k)$	$P_w/n+P_l(2-1/(\alpha^{h-1}n))$	$2P_w/n+2P_l$	$2P_w/n+2P_l$	$2P_w/n+2P_l$	$2P_w/n+2P_l$



(a) $n = 24$. (b) $n = 48$. (c) $n = 96$. (d) $n = 192$.

Fig. 11. Scaled diameter of FiConn, BCN, DPillar, SWCube, SWKautz for different c values.

of DPillar, SWCube, SWKautz, and SWdBruijn are generally in the order of n^d/d. These rough comparisons show that SWCube, SWKautz, and SWdBruijn can provide high bisection widths.

8.4 Hardware Interconnection Cost per Server

We compare the hardware interconnection cost when all architectures use n-port switches. Assume that the price of an n port switch is P_w, and that the price of a cable/link is P_l. Based on architectures' switch-number to server-number ratio and server degree, different architectures' hardware interconnection costs per server can be calculated as in Table 5. As we can see, the cost on links of different architectures do not differ much; in fact, it is very close to $2P_l$ for all architectures. The cost per server on switches of DPillar, SWCube, SWKautz, and SWdBruijn is twice that of FiConn and BCN; which is the main reason why DPillar, SWCube, SWKautz, and SWdBruijn can accommodate more servers than FiConn and BCN. At only twice the cost, DPillar, SWCube, SWKautz, and SWdBruijn provide a large order of increase on the number of servers.

8.5 Influence of c on Various Architectures

We investigate the influence of c (NSD) on various architectures. For this purpose, we choose a fixed $d = 7$ when $c = 0$. Under the HEH assumption, When c chooses different values, the scaled diameter of different architectures can be calculated. Fig. 11(a), Fig. 11(b), Fig. 11(c) and Fig. 11(d) show the comparisons when $n = 24, 48, 96$ and 192, respectively. Since, in FiConn and BCN, server-to-server-direct hops exist, when c increases, their scaled diameter will not increase as sharply as that of DPillar. Though FiConn and BCN have a large scaled diameter when $c = 0$, as c increases, their scaled diameters tends to be comparable with that of DPillar. SWCube has a larger scaled diameter than DPillar when switch port number is small; when the switch port number becomes large, SWCube tends to have a comparable, or even lower scaled diameter than that of DPillar. SWKautz always has the lowest scaled diameter because its network order is much greater than that of all the others. The scaled diameter of SWdBruijn is almost overlapped with that of SWKautz, due to their various similarities.

9 Evaluation of SWCube and SWKautz

In this section, we revert to the situation when $c = 0$, which is the assumption that all existing works have adopted. As readers may have already noticed, SWKautz and SWdBruijn demonstrate an extremely high degree of similarity. In the following, we just choose SWKautz as the representative of the two, and omit discussions on SWdBruijn. Since SWCube and SWKautz are based on the generalized hypercube and Kautz graph, respectively, they are conjectured to have high degrees of

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

regularity, high width, good fault-tolerance, rich parallel paths, as well as other nice properties for DCNs. In our work, we focus on their properties related to routing.

9.1 Routing Properties of SWCube and SWKautz

Lemma 3: The shortest path length between two servers, $S = (S^1, S^2)$ and $D = (D^1, D^2)$, in an SWCube can be calculated by: $1 + \min\ \{hd(S^1, D^1),\ hd(S^1, D^2),\ hd(S^2, D^1),\ hd(S^2, D^2)\}$, where $hd()$ is the Hamming distance between two switches.

Proof: For a packet at server S to reach D, it must go through one of the two switches S^1 and S^2, and one of the two switches D^1 and D^2. In a generalized hypercube, the shortest path length between two nodes is their Hamming distance; thus, in the shortest path between any pair of two switches, say S^1 and D^1, there exist at most $hd(S^1, D^1)$ servers. Thus, the shortest path length between two servers S and D is $1 + \min\ \{hd(S^1, D^1),\ hd(S^1, D^2),\ hd(S^2, D^1),\ hd(S^2, D^2)\}$. □

Theorem 9: For two servers $S = (S^1, S^2)$ and $D = (D^1, D^2)$, if their shortest path length is $l \geq 2$, their exist at least $l-1$ server-disjoint shortest paths between them.

Proof: For $l \geq 2$, $\min\{hd(S^1, D^1),\ hd(S^1, D^2),\ hd(S^2, D^1),\ hd(S^2, D^2)\} \geq 1$, without lose of generality, we assume $\min\{hd(S^1, D^1),\ hd(S^1, D^2),\ hd(S^2, D^1),\ hd(S^2, D^2)\} = hd(S^1, D^1)$. According to the hypercube properties, there exist $hd(S^1, D^1)$ switch disjoint shortest paths between S^1 and D^1. Obviously, these paths are also server-disjoint. Therefore, there exist at least $hd(S^1, D^1) = l-1$ server-disjoint shortest paths between S and D. □

It has been shown that, there exist r node-disjoint paths between any pair of nodes in a KA(r, k) [23], and their lengths are no greater than $k + 2$.

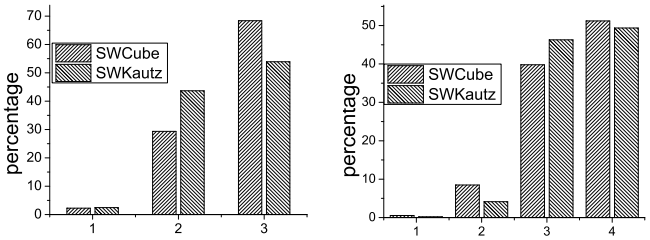
Theorem 10: There exist at least $n/2$ server-disjoint paths between any pair of servers in an SWKautz$(n/2, k)$, and their lengths are no greater than $k + 3$.

Proof: This theorem follows from the aforementioned fact. Plus 1 is imposed on $k + 2$ because the length of the path between two servers is less than or equal to the number of servers in a path between their home switches plus 1. □

As Theorems 9 and 10 have indicated, both SWCube and SWKautz have good fault-tolerance properties.

9.2 Average Path Length

For SWCube, the shortest path length between any pair of servers can be easily calculated. For SWKautz, considering two servers $S = (S^1, S^2)$ and $D = (D^1, D^2)$, we first calculate the shortest path length among the following paths as in the original Kautz graph: $S^1 \rightarrow D^1$, $D^1 \rightarrow S^1$, $S^1 \rightarrow D^2$, $D^2 \rightarrow S^1$, $S^2 \rightarrow D^1$, $D^1 \rightarrow S^2$, $S^2 \rightarrow D^2$, $D^2 \rightarrow S^2$ by the shortest path routing algorithm of Kautz graph [20]; denote this value as l_m. Then the shortest path length between S and D in SWKautz is $l \leq l' = 1 + l_m$. The actual shortest path length l may not be equal to l' because Kautz is a directed graph, while SWKautz is undirected. More details are omitted. We calculate shortest path lengths and l' values of all possible pairs of servers in SWCube and SWKautz, respectively. Fig. 12(a) shows the shortest path length distribution of SWCube$(13, 2)$ and l' value distribution of SWKautz(12,2). Fig. 12(b) shows the same distributions of SWCube$(9, 3)$ and SWKautz$(12, 3)$. In the figures, a bar represents the percentage of paths whose lengths are equal to the corresponding value. The average shortest path lengths of SWCube$(13, 2)$ and SWCube$(9, 3)$ are 2.66 and 3.42, respectively; the average l' values of SWKautz$(12, 2)$ and SWKautz$(12, 3)$ are 2.51 and 3.45, respectively. Since SWKautz$(12, 3)$ consists of 22,464 servers, and SWCube$(9, 3)$ only consists of 8,748 servers, one might conjecture that the SWKautz has a greater average shortest path length. However, the average l' values is just slightly greater than the average shortest path length of SWCube$(9, 3)$. Thus, the shortest path length in SWKautz$(12, 3)$ is also small.



(a) $d = 3$. (b) $d = 4$.

Fig. 12. Path length distributions. (a) SWCube$(13, 2)$ and SWKautz$(12, 2)$. (b) SWCube$(9, 3)$ and SWKautz$(12, 3)$.

9.3 Routing Simulation With Congestion

We develop a proprietary simulator to conduct routing simulations to evaluate SWCube's and SWKautz's performances, under different degrees of network traffic flow pressure. Notice that our emphasis is on evaluating the architectures themselves, instead of on designing the most efficient routing algorithms. Our simulations are time step (ts) based. We consider single-packet flows and a fixed packet size. Thus, we have a fixed transmission delay, which is considered as one unit of time, which is the value of one time step. We assume that each server can send a packet at each time step; however, it can send at most one packet at each time step. If more than one packet needs to be sent out, the packages will be queued by the First-In-First-Out (FIFO) scheme. If packets arrive at a server at the same time, the packet with a smaller flow index is assigned a higher priority. At each time step, only the packet at the server's queue head will be sent to this packet's next server, and other packet(s) should be delayed. These idealistic assumptions comply with the HOH assumption that delay at switches is negligible, compared to the delay at servers.

In SWCube, we adopt the shortest path for SWCube, which can be easily achieved according to Lemma 3; as

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

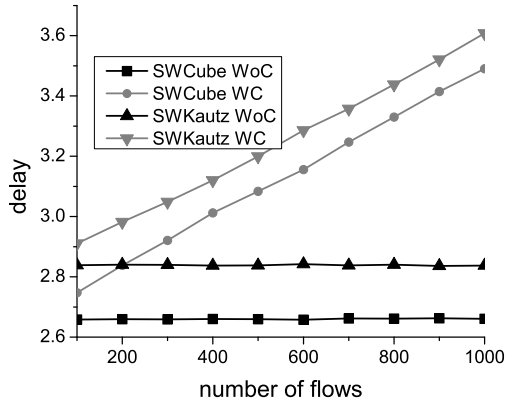


Fig. 13. Routing simulation for SWCube$(13, 2)$ and SWKautz$(12, 2)$ without and with congestion.

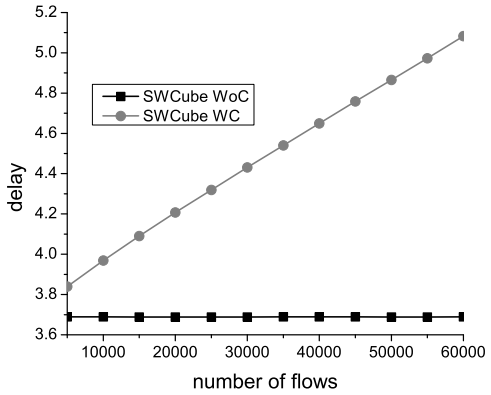


Fig. 14. Routing simulation for SWCube$(17, 3)$ without and with congestion.

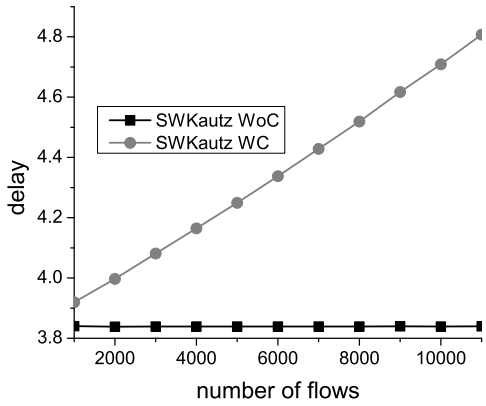


Fig. 15. Routing simulation for SWKautz$(12, 3)$ without and with congestion.

for selecting the shortest path from switch W^1 to W^2 in SWCube, the path that corrects the switch address string from the first bit to the last bit is selected. Since shortest path routing in Kautz graph may cause a severe congestion problem [20], we choose the long path routing algorithm for SWKautz. Each server chooses its left switch as its home switch; then, a path from a source server's home switch to a sink server's home switch can be constructed by the long path routing algorithm [20]. A flow's delay without congestion is just the flow's path length, while a flow's delay with congestion is the time step, ts's value, at which the flow arrives at its destination. Though our path selection is fixed, by simulating a sufficiently large number of flows, the path selection of all the servers in the network will tend to be randomized.

We conduct simulations for SWCube$(13, 2)$ and SWKautz$(12, 2)$ separately. SWCube$(13, 2)$ has $N_v = 2028$ servers, and SWKautz$(12, 2)$ has $N_v = 1872$ servers. Both of them use 24-port switches. We vary the number of flows as $100, 200, 300, \cdots, 1000$. For each number of flows, we randomly generate 100 sets of flows and calculate the average delay. In each set of the flows, flows' sources are in one half of all the servers, and their destinations are in the other half of all the servers. Partitioning the servers into two halves is based on servers' addresses. Specifically, each server has a unique address, which is converted into a unique integer, which can be from 0 to $N_v - 1$; then, the first half includes servers from 0 to $N_v/2 - 1$, and the second half includes servers from $N_v/2$ to $N_v - 1$. Fig. 13 shows the results of our simulation. "SWCube WoC", "SWCube WC", "SWKautz WoC", and "SWKautz WC" represent the average delays of SWCube without congestion, SWCube with congestion, SWKautz without congestion, and SWKautz with congestion, respectively. When the total number of flows is small, the average delay with congestion is almost the same as that of the average delay without congestion. For the case in which the number of flows is 100, the average delays with congestion are only 3.36% and 2.53% greater than the average delays without congestion for SWCube$(13, 2)$ and SWKautz$(12, 2)$, respectively. When the number of flows increases, the average delays with congestion only slightly increase linearly, even when about half of the servers initiate a flow at the same time. For the flow number equal to 1,000, the average delays with congestion are 31.17% and 27.15% greater than the average delays without congestion for SWCube$(13, 2)$ and SWKautz$(12, 2)$, respectively.

We also conduct simulations for SWCube$(17, 3)$ and SWKautz$(12, 3)$ separately. SWCube$(17, 3)$ uses switches with $n = 48$ ports, and has $N_v = 117,912$ servers. For SWCube$(17, 3)$, we vary the number of flows from 5,000 to 6,0000 with a step size of 5,000. The results for SWCube$(17, 3)$ are shown in Fig. 14. The average delay without congestion is about 3.6890 for all sets of flows. The average delay with congestion, when the number of flows is 5,000, is 3.8390, which is 4.06% greater than the average delay without congestion. When the number of flows is 60,000, the average delay with congestion is 5.0826, which is 37.78% greater than the average delay without congestion. SWKautz$(12, 3)$ uses switches with $n = 24$ ports, and has $N_v = 22,464$ servers. For SWKautz$(12, 3)$, we vary the number of flows from 1,000 to 11,000 with a step size of 1,000. The results for SWKautz$(12, 3)$ are shown in Fig. 15 The average delay

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

without congestion is about 3.8388 for all sets of flows. The average delay with congestion, when the number of flows is 1,000, is 3.9198, which is 2.11% greater than the average delay without congestion. When the number of flows is 11,000, the average delay with congestion is 4.8076, which is 25.24% greater than the average delay without congestion.

From the above simulations, we can see that both SWCube and SWKautz can efficiently handle network congestion. The simulation results for SWKautz show that, though SWKautz's long path routing has a greater average delay, its delay with congestion increases less significantly. Also, simulations for large-scale architectures, namely SWCube$(17, 3)$ and SWKautz$(12, 3)$ show that SWCube and SWKautz demonstrate good scalability.

10 Conclusion and Future Work

We consider the design and analysis of DCN architectures for interconnecting dual-port servers in this paper. Unlike all existing works, we propose distinguishing a sever-to-server-direct hop and a server-to-server-via-a-switch hop when calculating the distance of two servers, to unify the design of DCN architectures for interconnecting dual-port servers. Next, we aim to maximize the number of servers, given a network diameter and a switch port number. Motivated by the diameter/degree problem in traditional graph theory, we give an upper bound on this maximal number. Then, we propose three novel architectures which try to approximate this upper bound: SWCube, SWKautz, and SWdBruijn which are based on the generalized hypercube, Kautz graph, and de Bruijn graph, respectively. We compare our proposed architectures with three existing ones in various aspects. Results show that SWCube, SWKautz, and SWdBruijn demonstrate advantages in various aspects. Analysis and simulations on SWCube and SWKautz in detail reveal that they also maintain good properties for DCNs.

Our future work will focus on designing efficient routing algorithms for the proposed architectures. Another direction of our future work is to consider interconnecting servers with more than two NIC ports. Also, considering the design and analysis of DCNs with new technologies, such as wireless DCNs [24] and optical switching [25], deserves future research endeavor.

Acknowledgment

This work was supported in part by NSF grants ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167.

References

[1] D. Guo, T. Chen, D. Li, M. Li, Y. Liu, and G. Chen, "Expandable and cost-effective network structures for data centers using dual-port servers," *IEEE Trans. on Computers*, vol. 62, no. 7, pp. 1303–1317, 2013.

[2] Y. Liao, D. Yin, and L. Gao, "Dpillar: Scalable dual-port server interconnection for data center networks," in *Proc. of 19th Int'l. Conf. on Computer Comm. and Networks*, 2010, pp. 1–6.

[3] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *Proc. of the 19th ACM Symp. on Operating Systems Principles*, 2003, pp. 29–43.

[4] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: a distributed storage system for structured data," in *Proc. of the 7th USENIX Symp. on Operating Systems Design and Implementation - Volume 7*, 2006, pp. 15–15.

[5] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[6] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in *Proc. of the 2nd ACM SIGOPS/EuroSys European Conf. on Computer Systems*, 2007, pp. 59–72.

[7] Y. Zhang and N. Ansari, "On architecture design, congestion notification, tcp incast and power consumption in data centers," *Communications Surveys Tutorials, IEEE*, vol. 15, no. 1, pp. 39–64, 2013.

[8] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *Proc. of the ACM SIGCOMM Conf. on Data Comm.*, 2008, pp. 63–74.

[9] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "Vl2: a scalable and flexible data center network," in *Proc. of the ACM SIGCOMM 2009 Conf. on Data Comm.*, pp. 51–62.

[10] J. H. Ahn, N. Binkert, A. Davis, M. McLaren, and R. S. Schreiber, "Hyperx: Topology, routing, and packaging of efficient large-scale networks," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, Nov. 2009, pp. 41:1–41:11.

[11] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," in *Proceedings of the 37th Annual International Symposium on Computer Architecture*, 2010, pp. 338–347.

[12] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: a scalable and fault-tolerant network structure for data centers," in *Proc. of the ACM SIGCOMM Conf. on Data Comm.*, 2008, pp. 75–86.

[13] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "Bcube: a high performance, server-centric network architecture for modular data centers," in *Proc. of the ACM SIGCOMM Conf. on Data Comm.*, 2009, pp. 63–74.

[14] D. Li, C. Guo, H. Wu, K. Tan, Y. Zhang, and S. Lu, "Ficonn: Using backup port for server interconnection in data centers," in *Proc. of IEEE INFOCOM*, 2009, pp. 2276–2285.

[15] C. Wang, C. Wang, Y. Yuan, and Y. Wei, "Mcube: A high performance and fault-tolerant network architecture for data centers," in *Int'l. Conf. on Computer Design and Applications,*, vol. 5, 2010, pp. V5–423–V5–427.

[16] A. Curtis, T. Carpenter, M. Elsheikh, A. Lopez-Ortiz, and S. Keshav, "Rewire: An optimization-based framework for unstructured data center network design," in *Proc. of IEEE INFOCOM*, 2012, pp. 1116–1124.

[17] N. Biggs, *Algebraic Graph Theory*. Cambridge: Cambridge University Press, 1974.

[18] L. Bhuyan and D. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," *Computers, IEEE Trans. on*, vol. C-33, no. 4, pp. 323–333, 1984.

[19] W. H. Kautz, "The design of optimum interconnection networks for multiprocessors," *Architecture and Design of Digital Computer, NATO Advances Summer Institute*, pp. 249–277, 1969.

[20] G. Panchapakesan and A. Sengupta, "On a lightwave network topology using kautz digraphs," *IEEE Trans. on Computers*, vol. 48, no. 10, pp. 1131–1138, 1999.

[21] F. Leighton, *Introduction to Parallel Algorithms and Architectures*. Morgan Kaufmann, 1992.

[22] J. Rolim, P. Tvrdik, J. Trdlicka, and I. Vrto, "Bisecting de bruijn and kautz graphs," in *Proc. of the 2nd Colloquium on Structural Information and Communication Complexity*, June 1995.

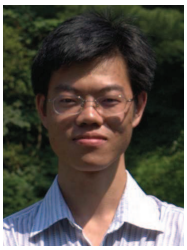
[23] G. J. M. Smit, P. Havinga, and P. Jansen, "An algorithm for generating node disjoint routes in kautz digraphs," in *Proc. of the 5th Int'l. Parallel Processing Symp.*, 1991, pp. 102–107.

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2015.2389847, IEEE Transactions on Computers

[24] Y. Cui, H. Wang, X. Cheng, and B. Chen, "Wireless data center networking," *Wireless Communications, IEEE*, vol. 18, no. 6, pp. 46–53, December 2011.

[25] I. Cerutti, P. G. Raponi, N. Andriolli, P. Castoldi, and O. Liboiron-Ladouceur, "Designing energy-efficient data center networks using space-time optical interconnection architectures," *Selected Topics in Quantum Electronics, IEEE Journal of*, vol. 19, no. 2, Mar. 2013.



Dawei Li is a PhD candidate in the Department of Computer and Information Sciences at Temple University since September 2011. He got the Bachelor's degree from the Advanced Class, Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, People's Republic of China. His research interest includes energy-aware task scheduling on multicores/multiprocessors, network-on-chip and data center networks.

Jie Wu is the chair and a Laura H. Carnell Professor in the Department of Computer and Information Sciences at Temple University. Prior to joining Temple University, he was a program director at the National Science Foundation and Distinguished Professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly published in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Computers, IEEE Transactions on Service Computing, and Journal of Parallel and Distributed Computing. Dr. Wu was general co-chair/chair for IEEE MASS 2006 and IEEE IPDPS 2008 and program co-chair for IEEE INFOCOM 2011. Currently, he is serving as general chair for IEEE ICDCS 2013 and ACM MobiHoc 2014, and program chair for CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.

0018-9340 (c) 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.