On the Design and Analysis of Data Center Network Architectures for Interconnecting Dual-Port Servers

Dawei Li and Jie Wu
Department of Computer and Information Sciences
Temple University, PA
{dawei.li, jiewu}@temple.edu

***Abstract*—We consider the design and analysis of Data Center Network (DCN) architectures for interconnecting dual-port servers. Unlike existing works, we propose the concept of Normalized Switch Delay (NSD) to distinguish a server-to-server-direct hop and a server-to-server-via-switch hop, to unify the design of DCN architectures. We then 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. Two novel architectures are proposed: SWCube and SWKautz, based on the generalized hypercube and Kautz graph, respectively, which in most cases accommodate more servers than BCN [1], which was claimed to be the largest known architecture. Compared with three existing architectures, SWCube and SWKautz demonstrate various advantages. Analysis and simulations also show that SWCube and SWKautz have nice properties for DCNs, such as low diameter, good fault-tolerance, and capability of efficiently handling network congestion.**

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

I. Introduction

Data centers provide various internet-based/online services, such as search, 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 [2]–[5]. As the increasing of the service demand will never end, the number of servers in today's 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 [6]. In switch-centric designs [7], [8], switch functionality is extended to meet the interconnection need, while servers do not need to be modified for interconnection purposes. Thus, high-end switches are needed, which significantly increases the interconnection cost. In server-centric designs [1], [9]–[13], switches are only used as cross-bars while servers act as both computing nodes and packet relay nodes. Though packet relay overhead on the servers is introduced, server-centric designs have the advantage of using only low-end layer-2 switches, thus reducing cost; also, by putting the interconnection intelligence on servers, they provide a higher degree of programmability. Considering the number of NIC ports used on servers, architectures in the second category can be further classified into two sub-categories. In the first sub-category, servers can have more than 2 ports; in the second sub-category, servers only have 2 ports. Since COTS servers in data centers often only have 2 NIC ports [11]. 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 with the case when more than 2 ports are used on servers.

In this paper, we consider server-centric DCN architecture design, where servers only have 2 NIC ports, and 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, 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], [9]–[12], the lengths of a server-to-server-direct hop and a server-to-server-via-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 [14]. Thus, design and comparison based on this assumption may be inappropriate. For example, in FiConn [11] and BCN [1], there exist server-to-server-direct hops, while in DPillar [12], 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 of DCN architectures. Specifically, we assume that a server-to-server-via-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* [15], which provides the upper bound on the number of nodes in a graph given a node degree and diameter, 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 BCN under the same configurations. Besides, the numbers of dual-port servers that BCN and DPillar can accommodate show big gaps between the upper bound.

Third, we propose two novel DCN architectures which try to approximate the upper bound. The first one is called SWCube, which is based on the generalized hypercube. SWCube accommodates a comparable number of servers as 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 second one, called SWKautz, which is based on the Kautz graph, always accommodates 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 hardware interconnection cost per server, and the influences of c (NSD) on different architectures under the HEH assumption. Analysis and simulations on SWCube and SWKautz reveal that they also have nice properties for DCNs such as high degree of regularity, high bisection width, good fault-tolerance, and efficient handling of network congestion.

The rest of the paper is organized as follows. Some basic definitions are given in Section II. In Section III, 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 two novel architectures that try to approximate this upper bound, which are defined in Sections IV and V, respectively. Three existing architectures are reviewed in Section VI. In Section VII, we compare our two proposed architectures with the three existing ones in various aspects. Section VIII evaluates our two proposed architectures in detail. Conclusions and future work are sketched in Section IX.

II. 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 do not consider switch-to-switch-direct connection in our design and analysis, because this type of connection generally forms a tree-like structure, in which the top switches have to be high-end to handle the larger traffic in higher levels. 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-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-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 practical 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. Distance between servers and network diameter are critical factors on the communication latency in DCN, which is an important metric for DCN design.

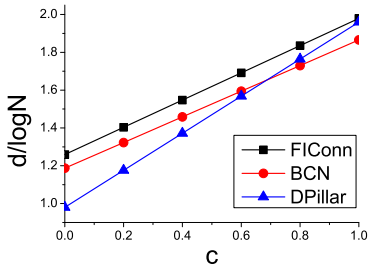


Fig. 1. Scaled diameter of FiConn, BCN, DPillar for different c values.

To give a preview of the influence of NSD, we compare the diameter of three existing architectures, namely, FiConn, BCN, and DPillar, when c chooses different values, given the same switch port number $n = 48$, server degree 2, and approximately equal numbers of servers. Readers may refer to Section VI 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 much. 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. 1, 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-switch hops, while in BCN, lots of server-to-server-direct hops exist. As c increases, server-to-server-via-switch hops will contribute more to the diameter.

III. 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. Though 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, 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

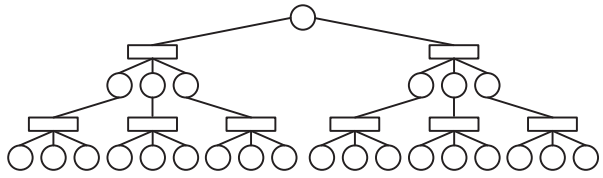


Fig. 2. The architecture greedily constructed to maximize the number of servers has a diameter $d = 2(1 + c)l, l = 2$.

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 $N \leq 1+\delta+\delta(\delta-1)+\cdots+\delta(\delta-1)^{d-1} = 1+\delta\sum_{i=1}^{d-1}(\delta-1)^i$ [15].

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

Proof: For c = 0, the lengths of a server-to-server-direct hop and a server-to-server-via-switch hop are equal. We consider the maximum number of other servers that a server S can reach within distance d. Within distance 1, S has two choices to reach other servers: the first one is to connect 2 other servers directly, and the second one is to connect to two switches, each of which connects $n-1$ other servers, resulting in a total of $2(n-1)$ servers. Obviously, the second choice is better because S reaches more other servers and more servers has one port remaining for further expansion. Within distance 2 of S, based on the second choice, the $2(n-1)$ servers connect to $2(n-1)$ switches, each of which connects $n-1$ other servers, resulting in another $2(n-1)^2$. Extending to distance d, S can reach at most $2(n-1)+2(n-1)^2+\cdots+2(n-1)^d$ other servers. Plus the original server S itself, the maximum number of dual-port servers that any network can accommodate is: $N_v \leq N_v^{ub} = 1 + 2(n-1) + 2(n-1)^2 + \cdots 2(n-1)^d = (2(n-1)^{d+1} - n)/(n-2)$. ■

Next, we consider the HEH assumption, where $0 \leq c \leq 1$.

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

Proof: Refer to the Appendix. ■

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-switch hops as shown in Fig. 2. Notice that we consistently use rectangles to represent switches, and circles to represent servers in all DCN architectures. The network in Fig. 2 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 $r^d + r^{d-1}$. These facts motivate us to design large order DCN architectures, based on the generalized hypercube and the Kautz graph. The following two sections present our two novel DCN architectures: SWCube and SWKautz. When calculating the diameter of SWCube and SWKautz, we assume $c = 0$. Since server-to-server hops are all server-to-server-via-switch hops in SWCube and SWKautz, the actual diameter is $1 + c$ times the diameter calculated assuming $c = 0$.

IV. SWCube

A. The Generalized Hypercube and SWCube Construction

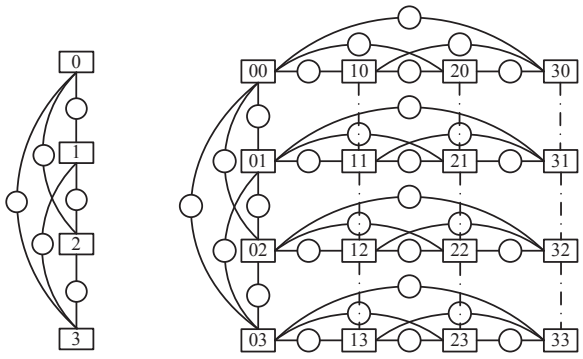
We denote a k-dimensional generalized hypercube [16] by $H^k_{r_1\times r_2\times\cdots\times r_k}$. The ith dimension is with radix r_i, $\forall i = 1, 2, \cdots, k$. A node W is represented by a k-tuple: $W = w_1w_2\cdots w_k$, where $0 \leq w_i \leq r_i - 1$, $\forall i = 1, 2, \cdots, k$. Two nodes are connected directly by a link if and only if their addresses differ at one bit. We call 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 link that connects two switches. The resulting DCN architecture is named SWCube because the SWitches form a generalized hyperCube. Fig. 3(a) and Fig. 3(b) show a 1-dimensional and a 2-dimensional SWCube, respectively, where $r_1(= r_2) = 4$. Note that in Fig. 3(b), the interconnections of switches and servers along the 2nd, 3rd, and 4th columns are represented by dotted lines.

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 $V^1 = v_1^1v_2^1\cdots v_k^1$ and $V^2 = v_1^2v_2^2\cdots v_k^2$ 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 $v_i^1 \neq v_i^2$.

B. Properties of SWCube

The number of switches in an SWCube 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



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

Fig. 3. SWCube.

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: $N_v = (\prod_{i=1}^{k} r_i)(\sum_{i=1}^{k}(r_i - 1)/2)$.

For symmetry and regularity, we can choose $r_1 = r_2 = \cdots = r_k = r$. We denote the constructed architecture as SWCube(r, k). 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. 3(a) and Fig. 3(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 consists of 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. Plus 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 3: The diameter of an SWCube(r, k) is $d = k+1$.

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

TABLE I. 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

Theorem 4: In terms of network diameter and switch port number, the number of servers in an SWCube(r, k) is $N_v = n(n/(d-1)+1)^{d-1}/2$.

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 $d = k + 1$, $r = n/(d-1) + 1$. 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 I 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 desired numbers of servers and to meet other requirements.

V. SWKAUTZ

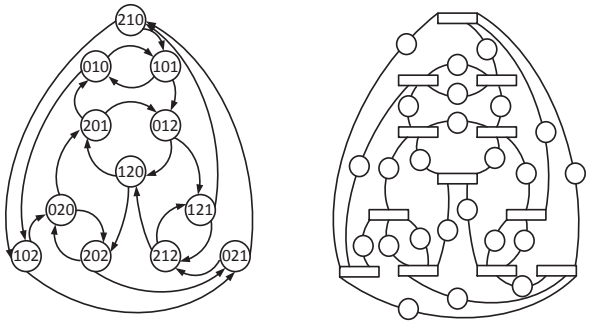
A. The Kautz Graph and SWKautz Construction

A k-dimensional Kautz directed graph [17], [18] 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 $w_1 w_2 \cdots w_k$ can represent a node in a KA(r, k) if $w_i \in Z, \forall 1 \leq i \leq k$ and $w_i \neq w_{i+1}$ $\forall 1 \leq i \leq k-1$. 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., $w_2^1 w_3^1 \cdots w_k^1 = w_1^2 w_2^2 \cdots w_{k-1}^2$, and $w_k^2 \neq w_{k-1}^2$. A 3-dimensional Kautz graph with 3 symbols is illustrated in Fig. 4(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 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 another 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 logically constructed as follows. We replace each node in the original KA$(n/2, k)$ graph with an n-port switch. After that, we remove the direction of all of the edges and insert a server into each edge. The architecture constructed as such is named SWKautz because the SWitches form a Kautz graph. Fig. 4(b) shows a SWKautz$(2, 3)$, whose base Kautz graph is KA$(2, 3)$ shown in Fig. 4(a). In SWKautz$(2, 3)$, the switch port number is 4.

Since the SWKautz 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



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

Fig. 4. Kautz graph and SWKautz.

switch is represented by a k-digit string $w_1w_2\cdots w_k$, where, $w_i \in Z, \forall 1 \leq i \leq k$ and $w_i \neq w_{i+1}, \forall 1 \leq i \leq k-1$. Note that there may be two servers connecting two 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^1 and W^2 are the server'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 from (W^1, W^2).

B. Properties of SWKautz

Theorem 5: 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 S connects, and D^1 and D^2 are the switches that D connects. 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 6: In terms of network diameter and switch port number n, the number of servers in an SWKautz$(n/2, k)$ is $N_v = (n/2)^d + (n/2)^{d-1}$.

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: $N_v = n((n/2)^k + (n/2)^{k-1})/2 = (n/2)^{k+1} + (n/2)^k = (n/2)^d + (n/2)^{d-1}$. ∎

VI. Related Existing Works

In this section, we review three main DCN architectures that also consider interconnecting dual-port servers. Notice that MCube [13] 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, $c = 0$.

A. FiConn

FiConn [11] 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) $(k > 0)$, the number of 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. 5 shows a FiConn$(4, 2)$, which consists of four FiConn$(4, 1)$'s.

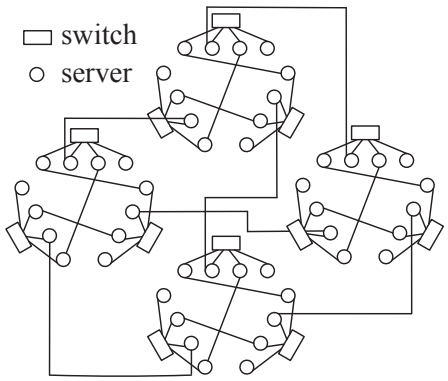


Fig. 5. FiConn$(4, 2)$.

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 II provides some feasible k and d values for FiConn.

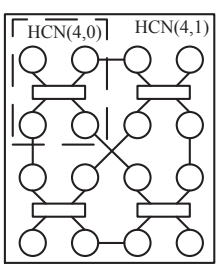
TABLE II. k, d VALUES FOR FICONN

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

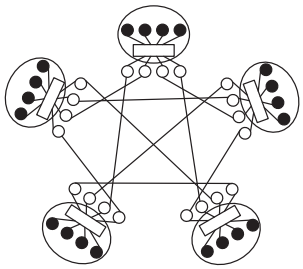
B. HCN & BCN

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. 6(a) shows an HCN$(4, 1)$ which connects four HCN$(4, 0)$'s.

A BCN architecture can be represented by BCN$(\alpha, \beta, h, \gamma)$, 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$(\alpha, \beta, h, \gamma)$ 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



(a) HCN



(b) BCN

Fig. 6. HCN and BCN.

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$(\alpha, \beta, h, \gamma)$ 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 III provides some feasible h, γ and d values for BCN. Fig. 6(b) shows a BCN$(4, 4, 0, 0)$.

TABLE III. 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

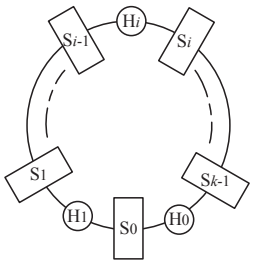
C. DPillar

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. 7(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 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. 7(b) shows a DPillar$(4, 3)$.

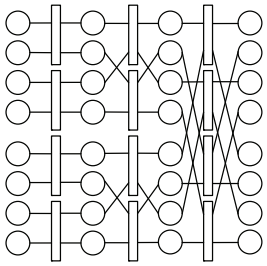
The diameter, the number of switches in a DPillar(n, k) are $d = k + \lfloor k/2 \rfloor$, $N_w = k(n/2)^{k-1}$, respectively. The number of servers in a DPillar(n, k) is $N_v = (2d/3)(n/2)^{2d/3}$, when k is even and $N_v = ((2d+1)/3)(n/2)^{2d/3}$, when k is odd. Every server's degree is 2, because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

TABLE IV. 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



(a) vertical view



(b) horizontal view

Fig. 7. DPillar.

VII. 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 interconnection cost per server in an architecture. Forth, we investigate the influence of c values on the architectures.

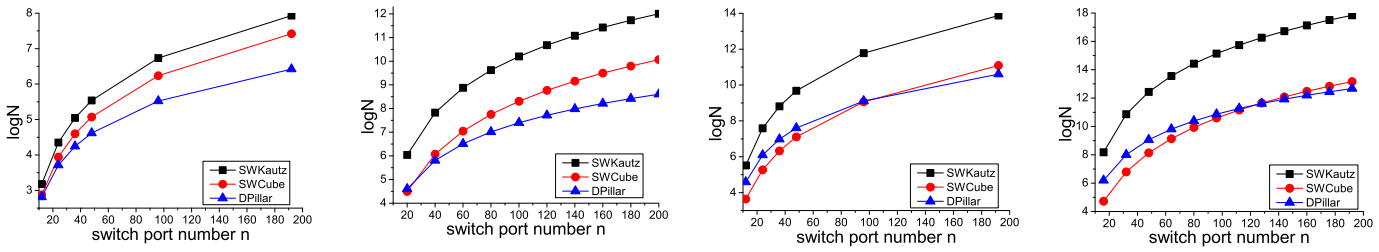
A. Design Flexibility

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_i's can be different 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 allows the most flexible choice of network diameters because it can choose arbitrary positive integers independent of switch port number, and its architecture itself does not incur additional constraints.

B. The Number of Servers Given d and n

We have also calculated the number of servers for all architectures. As we have shown above, FiConn and BCN allow very limited choices of 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 SWCube, DPillar, and SWKautz. The number of servers BCN accommodates is less than $n^{2\log_2((d+1)/2)+1}$, which is also less than that of SWCube, DPillar, and SWKautz for most n and d values. We do not include FiConn and BCN for comparison in Fig. 8

We choose four typical diameter values for comparing SWCube, DPillar and SWKautz, $d = 4, 6, 7, 9$. 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. 8(a), Fig. 8(b), Fig. 8(c), and Fig. 8(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 almost always outperforms DPillar and SWCube in terms of maximizing the number of servers. Also notice that, the number of servers that SWKautz accommodates is the most close to the upper bound even under the HEH assumption when $c \neq 0$.



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

Fig. 8. Number of servers given network diameter and various switch port numbers.

TABLE V. HARDWARE INTERCONNECTION COST COMPARISON

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

C. 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 V. 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 is twice that of FiConn and BCN; which is the main reason why DPillar, SWCube and SWKautz can accommodate more servers than FiConn and BCN. With only twice the cost, DPillar, SWCube, and SWKautz provide a large order of increase on the number of servers.

D. Influence of c on Various Architectures

We investigate the influence of c (NSD) on various architectures. For this purpose, we choose 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. 9(a), Fig. 9(b), Fig. 9(c) and Fig. 9(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 diameter tends to be comparable with that of DPillar. SWKautz always has the lowest scaled diameter because its network order is much greater than that of all the others.

VIII. 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. Since the two architectures are based on the generalized hypercube and Kautz graph, they are conjectured to have high degree of regularity, high bandwidth, 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.

A. 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 7: 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) [19], and their lengths are no greater than $k + 2$.

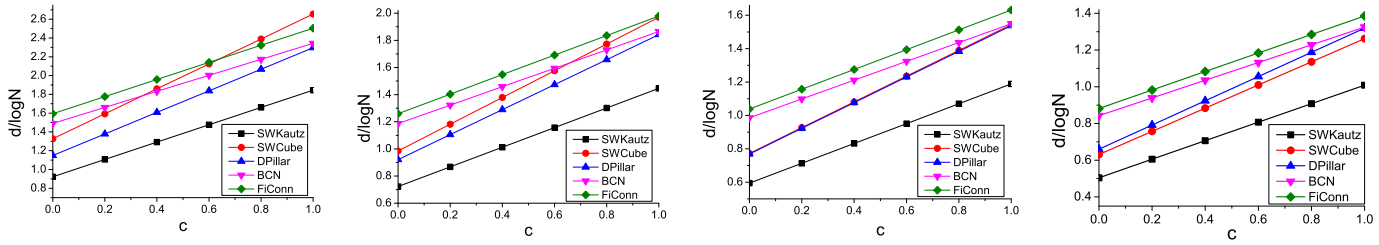
Theorem 8: There exist at least $n/2$ server-disjoint paths between any pair of servers in an SWKautz$(n/2, k)$, and their lengths are no greater than $k + 3$.

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 7 and 8 have indicated, both SWCube and SWKautz have good fault-tolerance properties.

B. 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: $S^1 \to D^1$, $D^1 \to S^1$, $S^1 \to D^2$, $D^2 \to S^1$, $S^2 \to D^1$, $D^1 \to S^2$, $S^2 \to D^2$, $D^2 \to S^2$ by the shortest path routing algorithm of Kautz graph [18]; 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



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

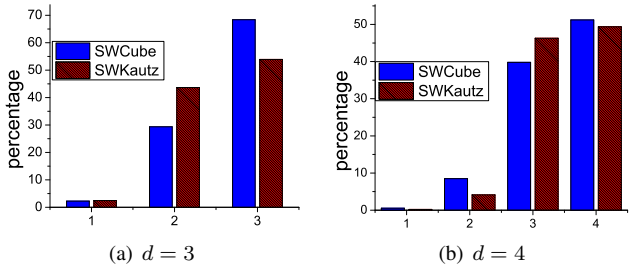
Fig. 9. Scaled diameter of FiConn, BCN, DPillar, SWCube, SWKautz for different c values.

path lengths and $l^{'}$ values of all possible pairs of servers in SWCube and SWKautz, respectively. Fig. 10(a) shows the shortest path length distribution of SWCube$(13, 2)$ and $l^{'}$ value distribution of SWKautz(12,2). Fig. 10(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.

C. Routing Simulation With Congestion

We design simulations to evaluate SWCube's and SWKautz's routing performance, 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. All randomly generated flows are imposed on the network at the same time step $ts = 0$. We assume that each server can send a packet and/or receive 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 packages 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 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 [18], 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 [18]. 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.



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

Fig. 10. Path length distribution $(n = 24)$.

We conduct simulations for SWCube$(13, 2)$ and SWKautz$(12, 2)$, which have $N_v = 2028$ and $N_v = 1872$ servers, respectively. 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. Fig. 11 shows the results of our simulation. "SWCube WoC", "SWCube WC", "SWKautz WoC", and "SWKautz WC" represent the average delays of SWCube without congestion, with congestion, SWKautz without congestion and 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 number of flows equal to 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. Thus, both SWCube and SWKautz can efficiently handle network congestion. As we can also see, though SWKautz's long path routing has a greater average delay, its delay with congestion increases less significantly.

IX. 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-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

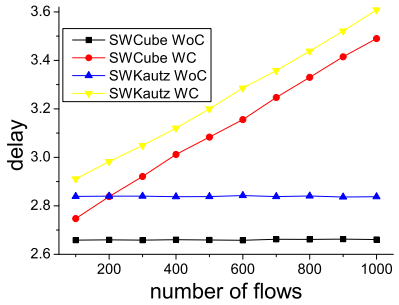


Fig. 11. Routing simulation for SWCube and SWKautz without and with congestion.

upper bound on this maximal number. Then, we propose two novel architectures which try to approximate this upper bound: SWCube and SWKautz, which are based on the generalized hypercube and Kautz graph, respectively. We compare our two proposed architectures with three existing ones in various aspects. Results show that SWCube and SWKautz demonstrate advantages in various aspects. Analysis and simulation on SWCube and SWKautz in detail reveal that they also maintain good properties for DCNs. Our future work will be designing efficient routing algorithms for the two architectures. Another direction of our future work is to consider interconnecting servers with more than two NIC ports.

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] 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.

[3] 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.

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

[5] 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.

[6] 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.

[7] 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.

[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] 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.

[10] 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 2009 Conf. on Data Comm.*, pp. 63–74.

[11] 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.

[12] 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.

[13] 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.

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

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

[16] 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.

[17] 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.

[18] 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.

[19] 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.

APPENDIX

Proof of Theorem 2: Consider the number of servers that a server S in a DCN can reach at most within distance d. For $1 \leq d < 1+c$, S can reach at most 2 other servers through server-to-server-direct hops; $\lceil d/(1+c) \rceil = 1$; the theorem holds. For $d \geq 1+c$, we consider three choices of S to reach as many other servers as possible within two hops (server-to-server-direct hop(s) and/or server-to-server-via-switch hop(s)). The first one is to reach other servers only by server-to-server-direct hops; in this case, it can reach at most four other servers (if possible), 2 of which have one port remaining for further outreaching, and S's remaining outreaching distance is $d-2$. The second choice is connecting S's two ports to two switches; by doing this, it can reach $2(n-1) > 4$ other servers, all of which have one port remaining for further outreaching, and S's remaining outreaching distance is $d-(1+c) > d-2$. Thus, compared with the first choice, the second one is always better. The third choice is to connect S's two ports to two other servers first; next, the two new servers connect to two switches, each of which connecting $n-1$ other servers, if $d \geq 1+(1+c)$. By the third choice, S can reach at most $2+2(n-1) = 2n$ other servers, of which $2(n-1)$ have one port remaining. However, if the next step of the third choice is possible, i.e. $d \geq (1+c)+1$, in the second choice, the $2(n-1)$ servers can also connect to $2(n-1)$ other servers within distance $(1+c)+1$. The second choice results in $4(n-1) > 2n$ new servers; there are also $2(n-1)$ servers with one port remaining. Thus, the second choice is also better than the third one. Based on the analysis of these three choices, we can see that S should always try to reach other servers via server-to-server-via-switch hops, if the remaining outreaching distance allows it to do so. Within $\lfloor d/(1+c) \rfloor$ server-to-server-via-switch outreaching hops, S can reach at most $2(n-1)+2(n-1)^2+\cdots+2(n-1)^{\lfloor d/(1+c) \rfloor}$ other servers. Exploiting the remaining outreaching distance $d-(1+c)\lfloor d/(1+c) \rfloor$, S can reach at most another $2(n-1)^{\lfloor d/(1+c) \rfloor}$ servers, if possible. Thus, the maximal number of servers in any network with diameter less than or equal to d is $N_v \leq (2(n-1)^{\lfloor d/(1+c) \rfloor+1}-n)/(n-2)+2(n-1)^{\lfloor d/(1+c) \rfloor} \leq (2(n-1)^{\lceil d/(1+c) \rceil+1}-n)/(n-2) = N_v^{ub}$.