FlatNet: Towards A Flatter Data Center Network

Dong Lin, Yang Liu, Mounir Hamdi and Jogesh Muppala

Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
{ldcse, liuyangcse, hamdi, muppala}@cse.ust.hk

***Abstract*—The design of the data center network that interconnects the massive number of servers is very important to ensure the agility and robustness of the data center to meet the requirements of the applications. In response to this challenge, the research community have begun exploring novel interconnect topologies including FatTree, DCell, BCube, HyPaC, etc. However, the solutions proposed so far either scale too slowly, suffer from performance bottlenecks, are server-location dependent, inherit poor availability, or can be too complex/expensive to construct. Motivated by these very important challenges, we propose a new data center interconnect called FlatNet that combines the advantages of previous architectures while avoiding their limitations. FlatNet is a cost-effective, high-performance, reliable and scalable interconnect with almost flat architecture. For example, given an equal-sized data center, the costs of a FlatNet in terms of number of links and switches are roughly 2/3 and 2/5 that of Portland, while still delivering comparable overall performance.**

Keywords-Data center, Network topology, Fault tolerance

I. INTRODUCTION

Mega data centers are being built around the world to provide various cloud computing services such as Web search, online social networking, online office and IT infrastructure out-sourcing for both individual users and organizations (e.g. Google, Microsoft, IBM, Amazon, eBay, and Yahoo are running data centers with at least 50,000 servers in each). As a result, data center networking has recently been a hot research topic in both academia and industry. A fundamental challenge is the design of the data center network that interconnects the massive number of servers, and provides an efficient and reliable platform to other applications [12]. In response to this challenge, the research community have begun exploring a plethora of interconnect topologies including FatTree, Clos, HyPaC, FiConn, DCell and BCube among a rapidly growing set of alternatives, many adapted from earlier solutions in the telecom and supercomputing areas.

The solutions proposed so far either scale too slow or suffer from performance bottlenecks, are server-location dependent, inherit poor availability, or can be too complex/expensive to be constructed. Motivated by these very important challenges, we propose a new type of data center interconnect that combines the advantages of previous architectures while avoiding their limitations. In particular, we develop a new interconnection network architecture, termed FlatNet that flattens/simplifies the data center network architecture. We then address and propose solutions to the following issues: 1) The investigation of the architectural and topological properties of FlatNet; 2) The design of routing algorithms for FlatNet taking into consideration their time complexity, fault-tolerance, and load-balancing properties.

The rest of the paper is organized as follows. In Section II, we briefly review the related work from the literature. In Section III, we describe the structure of FlatNet. We then propose routing, load-balancing, and fault-tolerant mechanisms for this new architecture. Some preliminary results are also presented in this section, including both theoretical analysis and simulation data. Finally, we conclude in Section IV.

This research has been supported by a grant from Huawei Technologies Co. Ltd. (Project code: HUAW10B15Z002.09/10PN).

II. RELATED WORK

Numerous proposals for identifying suitable network architectures for massive data centers have been investigated and implemented in both academia and industry. These architectures can be classified based on whether they have evolved from the field of parallel computing or from Internet switches and routers.

A. DCIs Evolving from Parallel Computing Interconnects

Several recently proposed scalable and fault-tolerant data center networking architectures, e.g., DCell [3], BCube [4], FiConn [6], Portland [14] and MDCube [8], are built upon the rich research literature on the interconnection networks proposed for parallel computing.

DCell is a recursively-defined architecture. Servers in DCell have multiple ports. Each server is connected with a single mini-switch and with many other servers via communication links. $DCell_0$ is the basic building block to construct a larger DCell. It consists of n servers connected to an n-port switch. $DCell_k$ is formed using $a_{k-1}+1$ $DCell_{k-1}$s, where a_{k-1} denotes the number of servers in a $DCell_{k-1}$. As a result, the DCell architecture scales double exponentially.

FiConn is another example of a recursive structure. It requires at most 2 ports on computers, while being able to scale up to millions of computers. The basic element of FiConn, the $FiConn_0$, is the same as $DCell_0$. Each computer in FiConn has a port connected to the switch in its $FiConn_0$, which is called the level-0 port. The backup port on the computer is left to connect to other computers. Assume that there are b available backup ports inside a $FiConn_{k-1}$, to construct a $FiConn_k$, one needs to connect $b/2$ available ports to $b/2$ other $FiConn_{k-1}$s. As a result, a $FiConn_k$ consists of $(b/2+1)$ FiConn $_{k-1}$s.

BCube is a server-centric network structure. There are two types of devices which form the BCube infrastructure - servers with multiple network ports and mini-switches which connect servers at different layers. A $BCube_k$ is constructed recursively from n $BCube_{k-1}$s and n^{k-1} n-port switches. In a BCube structure, switches never directly connect to other switches and they just do forwarding. The BCube uses a lot of wires and switches and has high cabling complexity that prohibits it from being scaled beyond a shipping container based modular data center (MDC).

The MDCube is an attempt at scaling up a BCube-based MDC to a large number of servers, using the MDC as a building block. In such a structure, each container is labeled with an m-tuple. A container is connected to all the containers that have only one different digit in their labels.

The Portland architecture is based on a complete binary tree called FatTree [14]. It consists of 3 levels (core, aggregation and edge). Unlike basic tree topology, all the switches and the total number of connections in each of the 3 levels are the same. The major drawback of Portland lies in the poor fault-tolerance topological properties. E.g., if the centralized fabric manager fails, the whole Portland scheme will fail.

B. DCIs Evolving from Internet Switches and Routers

Many data center interconnects are based on architectures originally designed for Internet switches and routers. These include VL2 [1], DOS [15], PETASW [10] and HyPaC [7].

A VL2 network is built from many switches arranged into a Clos topology to support large path diversity. VL2 provides uniform high capacity and performance isolation between any two servers. However, it is very expensive to be constructed.

DOS uses wavelength routing characteristics based on an Array Waveguide Grating Router (AWGR) to design a scalable optical switch for data centers. They take advantage of having multiple wavelengths to demonstrate that their architecture outperforms electronic switches for data center interconnects in terms of the bandwidth and the size of the switching fabric.

PETASW and HyPaC (viz. *c*-through) improve the latency performance of AWGR by introducing hybrid electronic and optic networking architectures. In PETASW, the line card serves as the bridge between electronical and optical switching, where individual packets are buffered and processed before being forwarded to an unbuffered optical switch. In HyPaC, electronic switching and optical switching are maintained as separate networks. The optical switching can be dynamically configured to temporarily boost the bandwidth between certain top-of-rack (ToR) electronic switches.

III. FLATNET

A. Motivation

Scalable data center network architectures must be capable of hosting millions of servers while minimizing cost in terms of the number of interface cards, switches, links, etc. Moreover, it must exhibit gradual scalability in a fault-tolerant environment, and provide high network bandwidth for various traffic intensive tasks. In general, data center network architectures can be compared in terms of performance, scalability, availability and complexity to achieve a comprehensive evaluation. In practice, these requirements are usually in conflict with each other and lead to many tradeoffs. In particular, scalability has significant influence on the performance and the cost of the data center network, and therefore is highlighted next.

Despite some differences, previous architectures/algorithms can be classified into four categories according to their scalabilities (viz. the number of servers under a certain configuration) : $O(c)$ [7][10], $O(n^c)$ [1][11][14], $O(n^k)$ [4] and $O(n^{(2^{\wedge}(k-1))})$ [3][6], where c refers to an arbitrary constant that is associated with some hardware limitation (e.g., the scale of a PETASW data center interconnect is restricted by the maximum size of the optical switching fabric); n is the port-count of switches/routers and k denotes the server degree (viz. the number of interface cards in each server) or the number of network layers. Given different application environments and variable parameter values, it is difficult to tell which category of data center network architecture actually outperforms the others. However, there are some clues that can help us to peer through the fog.

First, $O(c)$ or $O(n^c)$ – class of data center interconnects have their physical limitation. The scalability of these network architectures relies entirely on the development of the key components, such as the size of an optical switching fabric, the port count of switches/routers. For example, assuming the maximum port count of switches/routers is n; a Portland/VL2 data center network with a three-layer network can only host $n^3/4$ servers, which is insufficient for a large-scale data center.

Second, a "flat" network architecture is usually preferred because it allows the applications' performance to be independent of the physical locations of the servers. In February 2011, Juniper unveiled its data center fabric called QFabric [9] which is designed to be the foundation of data centers for the next decade. It collapses the traditional three-layer network (e.g. FatTree) down to a single, high-performance layer by using a full-mesh (logically) topology in its core network. By using QFabric, the entire data center interconnect can be simply regarded as a giant switch which simplifies the management and maintenance of a data center. However, due to the complex network topology, the scalability of a QFabric is limited to just several thousand servers. Moreover, in order to achieve high availability and avoid the single-point-failure of the core network, a redundancy backup retention policy has to be adopted which increases the overall cost dramatically.

The QFabric may not be as perfect a solution as it is claimed to be; however, it points the development direction of data center interconnects – towards a flatter and simple network architecture, where multiple network layers are undesirable. This contradicts those recursively defined network architectures, such as DCell and BCube, which exhibit good fault-tolerance but typically require more than three network layers to scale up to a large size. As shown in Fig.1, neither $O(n^k)$ nor $O(n^{(2^{\wedge}(k-1))})$ – class of data center interconnection networks scale to a large size until more than three network layers are considered. Given only two network layers, even a DCell that has a double exponentially increasing size can only yield a data center with $O(n^2)$ servers.

Such observations encourage us to seek a new type of network architecture that scales faster than double exponentialy when the data center network consists of only a few layers (e.g. <3). $O(n*(an)^{(k-1)})$ - class of data center interconnection network is the first one that comes into our mind, where a refers to an arbitrary constant. Hence, Fig. 1 indicates the region we intend to explore. It originates from a simple mathematical fact that $O(n*(an)^{(k-1)})$ is actually the fastest scaled polynomial with k<3. Accordingly, one of the major topics we focus on in this paper is "*how to design a large-scale data center network with* $O(n*(an)^{(k-1)})$ *scalability, so as to deliver a flatter network architecture with good performance while reducing the overall cost*".

B. FlatNet Network Topology

To explore the advantage of an $O(n*(an)^{(k-1)})$ - class of data center network architectures, we start from one special case where $a = n$. We propose a new interconnection network architecture called FlatNet to demonstrate the feasibility and the possible outcomes. It scales at a speed of n^3 with only 2 layers of network.

The first layer of the FlatNet contains n servers and one n-port switch. The second layer of the FlatNet consists of n^2 1-layer FlatNet. In particular, a 2-layer FlatNet can simply be regarded as an n^2 *n matrix, where each row is a 1-layer FlatNet which consists of n servers. Alternatively, it can be considered as having n columns where each column contains exactly n^2 servers which belong to n^2 1-layer FlatNet.

A column-based connection is introduced here to connect the n^2 servers located at the same column by using exactly n n-port switches. The connection patterns of these n switches are listed as shown in Table I. There are n different kinds of connection patterns. Fig. 2 demonstrates the network topology of a 64-server FlatNet constructed by using 4-port switches.

C. Key Features

Our preliminary investigation reveals that the FlatNet topology exhibits some good properties as shown in Table II.

a) Scalability and Cost per Server: a significantly larger data center network can be constructed with lower average cost by using a simple FlatNet architecture. By using identical n-port switches and only two-layer architecture, a FlatNet can host n^3 servers which are approximately 4 times that of a Portland/VL2 and n times that of a DCell/BCube. Besides excellent scalability, the FlatNet still maintains a modest average cost. The cost per server of a FlatNet is identical to a tiny 2-layer BCube. Moreover, each server in a FlatNet requires only 2/3 number of links and 2/5 number of switches as that of a (1/4)-sized Portland.

b) Bisection Bandwidth: the minimum number of links cut when a network is partitioned into two equal halves. The bisection bandwidth of a FlatNet is $n^3/4$. The ratio between the bisection bandwidth and the server number is a constant. In contrast, the bisection bandwidth of VL2 increases sub-linearly to the server number, when n increases.

c) Diameter: The maximum shortest distance between any servers. There are only three kinds of links for all data center networks, viz. the link between two servers, the link

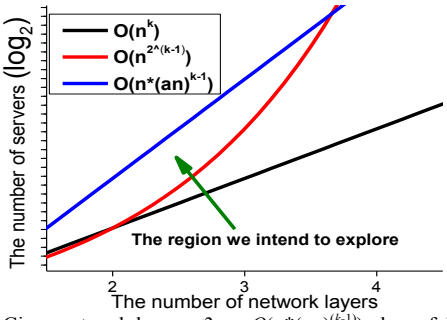

Figure 1. Given network layers <3, an $O(n*(an)^{(k-1)})$-class of data center interconnection network outperforms the others in terms of scalability.

TABLE I. THE DESTINATION ROW NUMBER OF THE CORRESPONDING SWITCH FOR THE $i-th$ COLUMN ($1 \le i \le n$)

	j-*th* port ($1 \le j \le (n/2+1)$)	j-*th* port ($(n/2+1) < j \le n$)
x-*th* switch ($1 \le x \le n$)	$[(i+j-1)+(x-1)n] \bmod n^2$	$[(i+n/2)+(j-n/2-1)\ (n+1)+(x-1)n] \bmod n^2$

* When the derived destination row number is "0", it actually means "n^2".

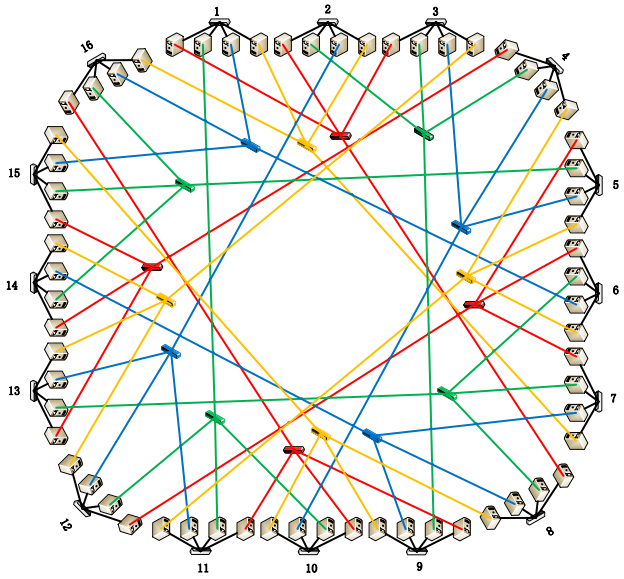

Figure 2. A 64-server FlatNet constructed by using 4-port switches. Each color of links/switches represents the connections of a "column".

TABLE II. PROPERTIES OF DIFFERENT NETWORK ARCHITECTURES

	Portland (3 layers)	VL2 (3 layers)	DCell (2 layers)	BCube (2 layers)	FlatNet (2 layers)
Degree	1	1	2	2	2
Servers Number	$\frac{n^3}{4}$	$\frac{(n-2)n^2}{4}$	$n(n+1)$	n^2	n^3
Links Number	$\frac{3n^3}{4}$	$\frac{(n+2)n^2}{4}$	$\frac{3n(n+1)}{2}$	$2n^2$	$2n^3$
per Server	3	$\frac{n+2}{n-2}$	$\frac{3}{2}$	2	2
Switches Number	$\frac{5n^2}{4}$	$\frac{3n}{2}+\frac{n^2}{4}$	$n+1$	$2n$	$2n^2$
per Server	$\frac{5}{n}$	$\frac{n+6}{n^2-2n}$	$\frac{1}{n}$	$\frac{2}{n}$	$\frac{2}{n}$
Bisection Bandwidth	$\frac{n^3}{8}$	$\frac{n^2}{4}$	$\frac{n^2}{4}+\frac{n}{2}$	$\frac{n^2}{2}$	$\frac{n^3}{4}$
per Server	$\frac{1}{2}$	$\frac{1}{n-2}$	$\approx\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$
Diameter	6	6	5	4	8

*To ensure a fair comparison, the number of ports of a switch is n by default for all architectures. Accordingly, we assume $n_{ToR}=n-2$ for VL2 which is also the default configuration in [1].

between a server and a switch, the link between two switches. In this paper, we assume that the costs of a packet transiting over these three types of links are identical and cost one *hop* exactly. As a result, the diameters of all architectures are recalculated for the sake of consistency. The diameter of a FlatNet is 8 hops. It is slightly bigger than the others, however still acceptable given its larger scale and lower average cost.

d) Incremental construction: The architecture of FlatNet is highly symmetric. Missing entire rows/columns does not reduce the system performance. For example, given absence of $1/n$ rows (e.g. those mod n=1), the entire network can still be regarded as a complete FlatNet which is constructed by using (n-1)-port switches. As a result, a FlatNet can be constructed incrementally, e.g., $1/n^2$ each time.

Given the above unique characteristics, we believe that the network topology of a FlatNet outperforms the previous architectures in terms of many aspects, including scalability, simplicity and average cost. In the rest of this paper, we will further prove that a FlatNet combined with a well-designed routing scheme can actually achieve its maximum theoretical performance, and therefore making it a competent candidate of the future data center network.

D. Broadcasting-based Routing

Despite being a flatter architecture, FlatNet is still a server centric scheme as that of DCell and BCube, where a server must be able to forward packets targeted to other servers. Such scheme increases the path length and the diameter of a data center network, however, minimizes the requirement of switches, which in turns reduces the cost of a data center [2][5].

In this paper, we propose two routing schemes which aim to maximize the performance of fault-tolerance and minimize the time complexity of routing respectively. We first present a broadcasting-based routing scheme in this part.

In the broadcasting-based routing scheme, each server only keeps the connectivity status of its nearby servers. A feasible routing path can be derived through recursive broadcasting / flooding. When the broadcasting packet reaches its destination, it will bounce back to the source through the same path that the packet travelled. Given multiple feasible paths, we choose the one with the minimal path length to guarantee a shortest path routing. In equal conditions, we choose randomly so as to achieve the optimal load-balancing.

To minimize the overhead of broadcasting, extensive experiments have been conducted, where all-to-all traffic pattern and Aggregation Bottleneck Throughput (ABT) are introduced for the sake of comprehensive evaluation.

In all-to-all communication, each server communicates with all servers (except itself) in two-way communication [4]. It simulates the most intensive network activities in a real-life data center network.

On the other hand, the ABT measures the overall throughput of all flows, assuming that the throughput of a flow is determined by the bottleneck throughput along its path [4]. In contrast to the bisection bandwidth, ABT reflects the actual performance of a data center network more accurately. Unless otherwise mentioned, in this paper, all the links are capable of two-way communication and the bandwidth of a link in one-way communication is "1" by default.

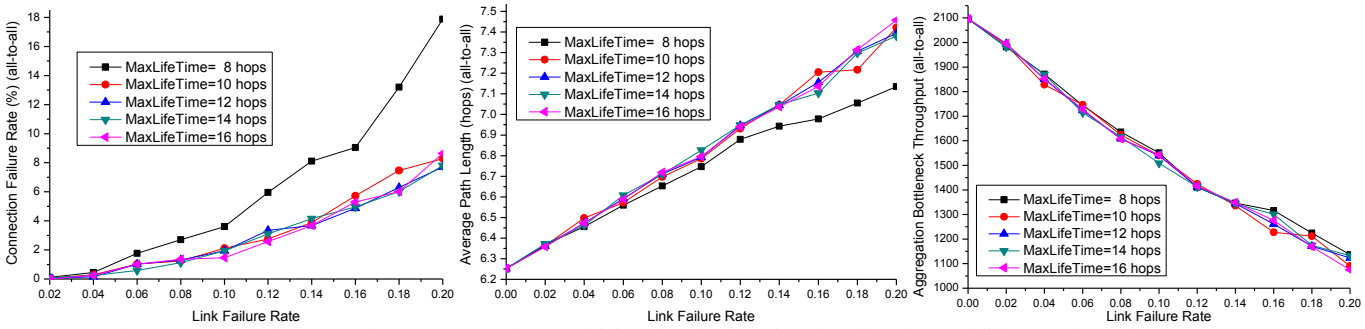

Figure 3. The performance of a 4096-server FlatNet with broadcasting-based routing algorithm and different values of *MaxLifeTime*.

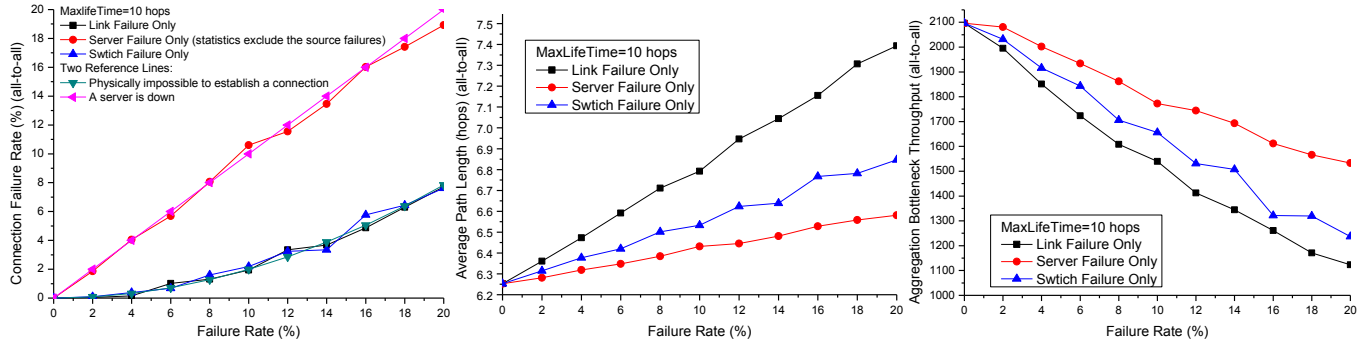

Figure 4. The performance of a 4096-server FlatNet under various faulty conditions with broadcasting-based routing scheme. (*MaxLifeTime* =10hops).

We define the *MaxLifeTime* as the maximum lifespan of a broadcasting packet in a FlatNet. It is implemented as a counter attached to each broadcasting packet. Once the prescribed count has elapsed, packet is discarded. As shown in Fig. 3, we find that a *Maxlifetime* > 10 hops improves the system performance very little. Thus, "10 hops" becomes the best tradeoff between performance and complexity. Setting *Maxlifetime* to 10 hops, Fig. 4 demonstrates the performance of a 4096-server FlatNet under various faulty conditions. The FlatNet successfully yields a gradually decaying performance as the failure rates increase, and the measured results are very close to the optimum, viz. the references lines derived from the stochastic mathematical model directly. For example, it is physically impossible to establish a connection, if either the source or the destination server is disconnected from the network completely. Assume the link failure rate is ρ, the probability of such isolation event is $1-(1-\rho^2)^2$.

TABLE III. THE CHARACTERISTICS OF A FLATNET WITH DIFFERENT SWITCHES (BROADCASTING-BASED ROUTING, FAULT-FREE CONDITION)

Switch Port-Count	No. of Servers	Average Path Length	Diameter
4	64	5.38	8
8	512	5.94	8
16	4'096	6.34	8
20	8'000	6.32	8
24	13'824	6.36	8
48	110'592	6.48	8
72	373'248	6.54	8
144	2'985'984	6.59	8

* For switch port-count larger than 72, sampling results are presented.

TABLE IV. ABT OF A 4096-SERVER FLATNET (BROADCASTING-BASED ROUTING, FAULT-FREE CONDITION)

All-to-All Communication	*Some(1/4)-to-All Communication*
2095 (12.8% of link capacity)	**1530 (73% of all-to-all)**

Table III lists the characteristics of a FlatNet with different switches in fault-free environment. The average path length (APL) of a FlatNet is slightly affected by the port-count of switches. For a 3M-server data center, the APL is only 6.6 hops.

Table IV presents the ABTs of a 4096-server FlatNet under both all-to-all and some-to-all (e.g., randomly select 1024 servers as the active sources instead of choosing all 4096 servers) traffic patterns. It reaches 2095 as its peak, which is roughly 12.8% of the overall link capacity (viz. the capacity of 8192 links in two-way communication). Given the APL of a 4096-server FlatNet is 6.34, "12.8%" is actually very close to the theoretical limitation (viz. 1/6.34≈15.8%). Besides, 73% of ABT under some-to-all traffic pattern also reveals the good load-balancing of a FlatNet.

E. Subsystem-oriented Routing

The broadcasting-based routing scheme works quite well except its relatively high overhead in broadcasting. In order to reduce the overhead, 1) a server only forwards the broadcasting packet owning the largest lifespan, when multiple broadcasting packets with the same pair of source and destination are received; 2) a source gradually increases the *Maxlifetime* starting from the known shortest distance; 3) a server avoids forwarding an obviously invalid packet if its remaining lifespan is shorter than the known shortest distance.

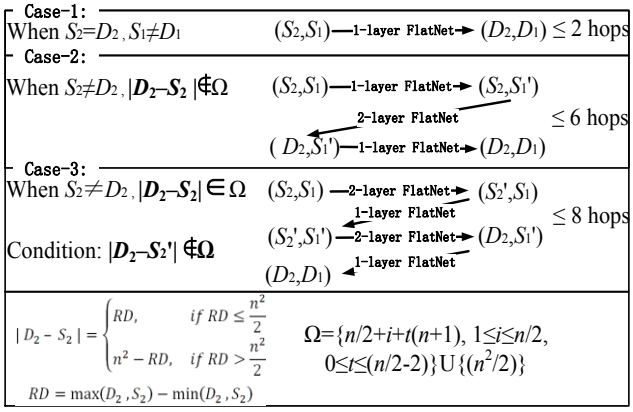

Figure 5. Three routing cases in a fault-free FlatNet.

TABLE V. THE CHARACTERISTICS OF A FLATNET WITH DIFFERENT SWITCHES (SUBSYSTEM-ORIENTED ROUTING, FAULT-FREE CONDITION)

Switch Port-Count	No. of Servers	Average Path Length	Diameter
4	64	5.588	8
8	512	6.220	8
16	4'096	6.626	8
20	8'000	6.700	8
24	13'824	6.748	8
48	110'592	6.874	8
72	373'248	6.916	8
144	2'985'984	6.958	8

* For switch port-count larger than 72, sampling results are presented.

TABLE VI. ABT OF A 4096-SERVER FLATNET (SUBSYSTEM-ORIENTED ROUTING, FAULT-FREE CONDITION)

All-to-All Communication	*Some(1/4)-to-All Communication*
1831 (11.2% of link capacity)	**1545 (84.4% of all-to-all)**

These optimizations help to reduce the overhead, however they cannot change the trend. The overhead of the broadcasting-based routing scheme still increases with the increasing number of servers. To enable efficient routing in a large-scale FlatNet, a subsystem-oriented routing scheme is proposed.

In the subsystem-oriented routing scheme, a server in a FlatNet is labeled using two coordinates (C_2, C_1), which denote that this server is the C_1-th server of the C_2-th subsystem in a FlatNet. Given a pair of servers, e.g., the source (S_2, S_1) and the destination (D_2, D_1), according to the values of$|D_2 - S_2|$, a path of 0~8 hops can be directly calculated. Fig.5 presents the routing cases and their corresponding solutions in a fault-free FlatNet. There are three cases in total where a Case-*i* can be gradually reduced to a simpler Case-(*i*-1).

Using a 64-server FlatNet as an example, we demonstrate the routing procedure from a source (1,3) to a destination (4,1) as follows.

First, "(1,3)→(4,1)" matches Case-3, as $\Omega = \{3,4,8\}$ and $|4-1| \in \Omega$, which represents the scenario that there is no direct connection between the source and destination subsystems. Thus, an immediate subsystem which has direct connections to both subsystems 1 and 4 should be chosen. According to the Case-3, the first step is a 2-layer transition making (6,3) and

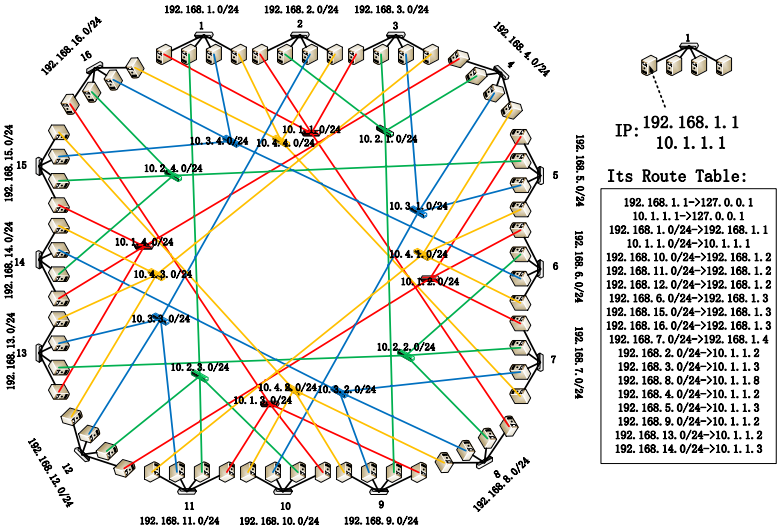

Figure 6. IP and routing table configurations of a 64-server FlatNet.

The IP addresses 192.168.*k*.*i* and 10.*i*.*x*.*k*. ($1 \le i \le n$, $1 \le x \le n$) are assigned to the *i*-th server of the *k*-th subsystem in a FlatNet.

$k=(i+j-1)+(x-1)n \bmod n^2 \qquad 1 \le j \le (n/2+1)$

$k=((i+n/2)+(j-n/2-1)\ (n+1)+(x-1)n) \bmod n^2 \qquad (n/2+1) < j \le n$

When k= "0", set k to "n^2".

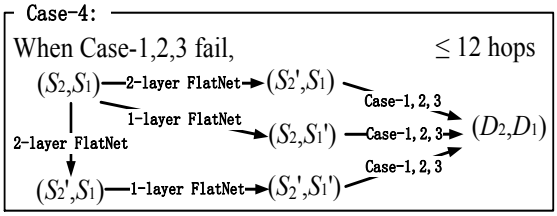

Figure 7. Additional Case-4 for fault-tolerant routing.

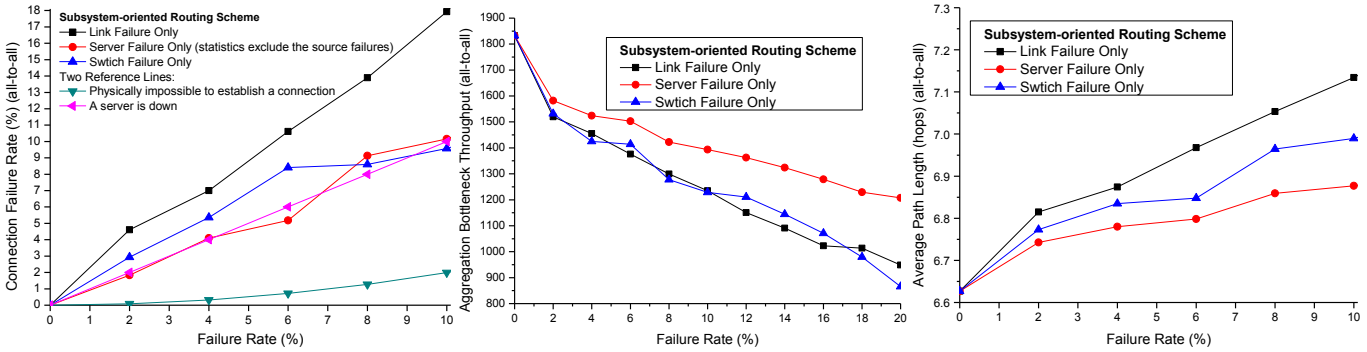

Figure 8. The performance of a 4096-server FlatNet under various faulty conditions with the subsystem-oriented routing scheme (Case-4 included).

(15,3) the candidates of next hop. Assume (6,3) is chosen in this step, the original routing problem (1,3)→(4,1) is now reduced to (6,3)→(4,1).

Second, "(6,3)→(4,1)" matches the Case-2, as $|4-6| \notin \Omega$, which indicates the existence of a direct connection between the source and destination subsystems, e.g., "(6,4)→(4,4)". As a result, "(6,3)→(4,1)" is now reduced to two Case-1 problems: "(6,3)→(6,4)" and "(4,4)→(4,1)". Finally, we get "(1,3)→(6,3)→(6,4)→(4,4)→(4,1)".

Unlike the broadcasting-based routing, the subsystem-oriented routing is not a shortest path routing. However, its APL is very close to (e.g. 4%-6% higher) that of a shortest path routing has achieved as illustrated in Table V.

Since the ABT under all-to-all traffic pattern is sensitive to APL, we observe a clear decline of ABT under all-to-all traffic pattern as expected. As shown in Table VI, with the subsystem-oriented routing, the ABT of a 4096-server FlatNet under all-to-all traffic pattern is only 1831, which is roughly 84.4% that the broadcasting-based routing does. On the contrary, the ABT under some-to-all traffic pattern improves slightly, mainly due to the compensation of an improved load-balancing.

The major advantage of the subsystem-oriented routing lies in the simplicity. There are only n^2 subsystems in a FlatNet. Since the routing scheme is subsystem oriented, n^3 n^2-entry routing tables stored at n^3 servers respectively can depict the routing behaviors of the entire FlatNet. In other words, the routing tables in a FlatNet need to be calculated only once and can then be used with $O(1)$-complexity.

Fig. 6 presents the IP and routing table configurations of a 64-server FlatNet, where each server behaves as a router and has a unique routing table that is derived according to the routing cases described in Fig. 5. In particular, the routing table of a server (viz., the first server in a 64-node FlatNet) is presented for your reference.

In the subsystem-oriented routing, the fault-tolerant routing and load-balancing can be implemented by changing the routing tables of individual server in real time. To calculate a proper routing table in a faulty environment, we introduce one additional routing case (viz. Case-4) as the supplement to Case-1,2,3. As shown in Fig.7, when Case-1,2,3 fail, the Case-4 enables the system to choose a reachable server in a radius of 2 hops as the new source so as to resume the previously failed routing Case-1,2,3.

To preserve the simplicity of the subsystem-oriented routing scheme to the hilt, Case-4 allows a new source only in a radius of 2 hops from the original source. Such restriction however brings certain negative effects on fault-tolerance.

Fig. 8 presents the performance of a 4096-server FlatNet under various faulty conditions with the subsystem-oriented routing scheme (Case-4 included). Although it still yields a gradually decayed performance as the increasing failure rates, the measured results are far from the optimum, except the scenario of server failure only.

A careful observation of Fig. 8 reveals that a FlatNet with the subsystem-oriented routing scheme still provides us with a upper-bound of connection failure rate of no more than 2ρ, when the server/switch/link failure rate is ρ. E.g., 10% link failure rate results in only 17.9% connection failure rate.

On the other hand, for those data center network architectures (e.g., Portland, VL2) with only one interface card at each server, the lower-bound of the connection failure rate is $1-(1-\rho)^2$, assuming that the server/switch/link failure rate is ρ.

As $\frac{1-(1-\rho)^2}{2\rho} \leq 0.95 \quad (\rho \leq 10\%)$, a FlatNet with subsystem-oriented routing scheme actually has a comparable fault-tolerant performance as that of a Portland/VL2, given the server/switch/link failure rate in a data center is no more than 10%. (Note that $\rho \leq 5\%$ in real-life data centers [13].)

F. Speedup of the First Layer FlatNet

In a FlatNet, a first layer link has higher utilization rate, thus more likely becomes a bottleneck. Fig. 9 shows that the ABTs of both broadcasting-based and subsystem-oriented routing schemes can be further improved with tiny speedup of the first layer network. 1.1~1.2 speedup factor to the first layer network makes the entire data center network become more cost-effective.

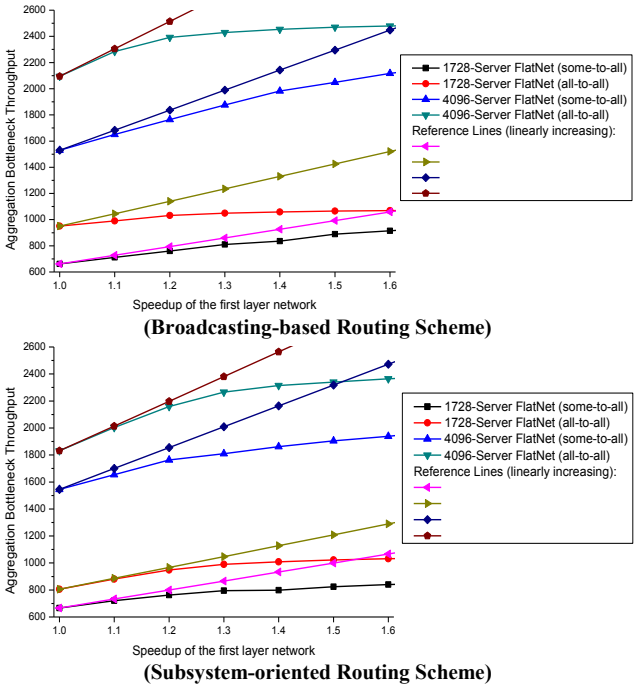

Figure 9. The speedup of the first layer network improves the ABT.

IV. CONCLUSION

In this paper, we presented the design and evaluation of a novel data center network architecture named FlatNet, to scale data center to large size with only two layers of network. FlatNet combines the advantages of previous architectures while avoiding their limitations. Given an equal sized data center, the costs of a FlatNet in terms of number of links and switches are roughly 2/3 and 2/5 that of a Portland, while still providing comparable overall performances. FlatNet is also fault-tolerant and load-balancing in nature due to its special structure and the low-time-complexity routing protocol on top of its network topology.

REFERENCES

[1] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Pat., "VL2: A Scalable and Flexible Data Center Network," In SIGCOMM 2009 on Data communication, Pages: 51-62, August 16-21, 2009, Barcelona, Spain.

[2] A. Greenberg, J. R. Hamilton, D. A. Maltz, P. Patel, "The cost of a cloud: research problems in data center networks", ACM SIGCOMM Computer Communication Review, Vol. 39, No. 1, Jan. 2009.

[3] C. Guo, H. Wu, K. Tan, L. Shiy, Y. Zhang, S. Luz, "DCell: A Scalable and Fault Tolerant Network Structure for Data Centers," In SIGCOMM 2008 on Data communication, Pages:75-86, August 17-22, 2008, Seattle, WA, USA.

[4] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, C. Tian, Y. Zhang, and S. Lu, "BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers," In SIGCOMM 2009 on Data communication, Pages: 63-74, August 16-21, 2009, Barcelona, Spain.

[5] Chayan Sarkar, "Seminar Report on Data Center Network Design", Technical Report, 2010, http://www.cse.iitb.ac.in/~chayan/seminar/final_report_dcn.pdf.

[6] D. Li, C. Guo, H. Wu, K. Tan, Y. Zhang, and S. Lu "FiConn: Using Backup Port for Server Interconnection in Data Centers", IEEE INFOCOM 2009, April 19-25, 2009, Rio de Janeiro, Brazil.

[7] G. Wang, David G. Andersen, M. Kaminsky, K. Papagiannaki, T. S. Eugene Ng, M. Kozuch, and M. Ryan, "c-Through: Part-time Optics in Data Centers," In Proc. SIGCOMM, 2010.

[8] H. Wu, G. Lu, D. Li, C. Guo, and Y. Zhang, "MDCube: A High Performance Network Structure for Modular Data Center Interconnection", in ACM SIGCOMM CoNEXT, December 2009.

[9] Juniper Network, " QFabric White Paper, 2011. http://www.juniper.net/us/en/local/pdf/whitepapers/2000384-en.pdf

[10] K. Xia, Y. Kao, M. Yang and J. Chao, "Petabit Optical Switch for Data Center Networks," Technical Report, 2010.

[11] M. Al-Fares, A. Loukissas, and A. Vahdat, "A Scalable Commodity Data Center Network Architecture," In SIGCOMM 2008 on Data communication, Pages: 63-74, August 17-22, 2008, Seattle, WA, USA.

[12] M. Armbrust, A. Fox, R. Griffith, et al. "Above the Clouds: A Berkeley View of Cloud Computing," UC Berkeley TRUCB/EECS-2009-28, 2009.

[13] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," In Proceedings of the ACM SIGCOMM 2011 conference (SIGCOMM '11). ACM, New York, NY, USA, 350-361.

[14] R. N. Mysore, A. Pamporis, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "PortLand: A Scalable, Fault-Tolerant Layer 2 Data Center Network Fabric," In SIGCOMM 2009 on Data communication, Pages: 39-50, August 16-21, 2009, Barcelona, Spain.

[15] X. Ye, Y. Yin, S.J.B.Yoo, P. Mejia, R.Proietti and V. Akella, "DOS – A Scalable Optical Switch for Datacenters", ACM ANCS'10, October 25-26, 2010, La Jolla, CA, USA.