P-FatTree: A Multi-channel Datacenter Network Topology

William M. Mellette, Alex C. Snoeren, and George Porter

Department of Computer Science and Engineering
University of California, San Diego

Abstract

The bandwidth and latency requirements of next-generation datacenter networks stress the limits of CMOS manufacturing. A key trend in their design will be a move from single-channel links and switches to multi-channel links and switches. Today's network topologies erase this distinction, providing the illusion of a unified network fabric. In this work we propose P-FatTree, which is a FatTree topology designed specifically for the future multi-channel reality. P-FatTree requires fewer switch chips and as a result has lower cost, power consumption, and latency than existing approaches. Furthermore, by embracing the parallel nature of the network itself, it enables compelling new ways to better manage and deliver application traffic.

1. INTRODUCTION

Over the past decade, the increasing availability of large-scale processing and storage in datacenters has driven the development of new classes of applications [4]. Scale-out data processing frameworks such as MapReduce [8] and Spark [30], and low-latency infrastructure services such as Memcached [23], provide a powerful interface to these underlying computing resources. Yet ensuring that those software layers are performant imposes stringent requirements on the underlying network fabric in terms of bandwidth, latency, power, and cost.

Researchers have developed scale-out datacenter network topologies [2, 14] which permit upgrading the network fabric by replacing older, slower switches with newer, faster switches. This trend has been enabled by merchant silicon switch chips, which can, with each new generation of CMOS fabrication process, forward increasing amounts of data. The result has been a seamless transition from 1- to 10-, and now 40-Gb/s fabrics [11, 28].

However, as network demands increase, fundamental limitations in CMOS manufacturing have begun to derail this trend. In particular, the per-port data rate of merchant silicon switches and network links has not been able to keep up with end-host and network bandwidth demands. At issue is the fact that the fundamental channel rate of commodity Ethernet has grown relatively slowly: first 1 Gb/s, then 10, and now 25 Gb/s [1]. This slow-growing channel rate is largely a function of limitations in CMOS serialization and deserialization (SerDes) hardware. While faster SerDes rates are possible with higher-order modulation formats, maintaining power efficiency becomes increasingly difficult [6, 19]. Yet per-port network bandwidth has grown much more quickly: from 1 to 40 and now to 100 Gb/s, with 200 and 400 Gb/s in the standardization process [9]. SerDes channel rates simply cannot keep up with increasing bandwidth demands, which has led network device vendors to move away from single-channel designs into *multi-channel* designs. A multi-channel link or switch simply "gangs together" multiple lower-speed channels to form a high-speed link or switch port. For example, commercially-available 100-Gb/s Ethernet links are actually made up of four parallel 25-Gb/s underlying channels [9], and 100-Gb/s switches actually devote four 25-Gb/s ports on the internal switch chip to each external-facing 100-Gb/s port. Thus, the basic building blocks of network fabrics—links and switches—are in fact quickly becoming multi-channel components.

Despite this sea change in the way that network components are designed and built, datacenter network designs have largely remained unchanged. We argue that the move to multi-channel links and switches has significant ramifications to the overall network architecture. Because a multi-channel link "uses up" multiple switch ports, as links move from single- to multi-channel designs, it is as if the number of hosts in the network increases by a factor of 4, 8, or 16× (described further in Section 3.2). Not only do network designers need to keep up with increases in bandwidth and the number of end hosts, but with multi-channel links, each end host effectively requires many more ports. Designers have already moved away form single-switch-chip architectures to multi-chip *chassis-based* designs to mask the ef-

HotNets-XV, November 9–10, 2016, Atlanta, GA, USA

© 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-4661-0/16/11…$15.00

DOI: http://dx.doi.org/10.1145/3005745.3005746

fects of this trend [10, 28]. However, conventional multi-channel network designs are unsustainable, and as we show, threaten the decade-long cost- and energy-effectiveness of folded-Clos topologies (also known as "FatTrees").

In this paper we describe how multi-channel network components affect the cost- and energy-effectiveness of FatTree networks by dramatically increasing the needed port count of the network. We then propose *P-FatTree*, which is a FatTree designed to support multi-channel links. We argue that P-FatTree is more scalable, and yet simpler, than existing designs, requiring significantly fewer components resulting in consummate reductions in power and cost. This increase in scalability comes from exposing the multi-channel structure of the network to end points, which does alter the network delivery model. Far from being a strict disadvantage, however, we discuss how this richer delivery model can improve network management and performance, making it easier to achieve desirable high-level network properties.

2. BACKGROUND

We begin with an overview of traditional folded-Clos ("FatTree") network designs in Section 2.1 and then describe an important shift in their construction to reduce cost and cabling complexity in Section 2.2.

2.1 Traditional FatTrees

Originally due to Charles Clos [7], the observation that large switch fabrics can be composed of relatively small and inexpensive switches became relevant in datacenter network architecture with the advent of merchant silicon switch chips [2]. The structure of a FatTree can be characterized by the number of tiers of switch chips that it requires, and how the chips are packaged into boxes. These design choices then dictate the number of hops a packet must traverse, as well as the number of fiber links and optical transceivers required to connect the fabric. Each additional tier incurs cost, power, latency, and cabling complexity, making it desirable to use the largest radix commodity switches available.

Figure 1(a) shows a small-scale illustration of a traditional folded-Clos topology built from 4-port switches, with 32 end hosts and four tiers. As a more realistic—but difficult to illustrate—example, the components required to build an 8,192-end-host network out of 32-port switches are listed in the first row of Table 1. While small compared to today's largest networks, we use an 8,192-end-host exemplar network throughout this paper because it allows for an "apples to apples" comparison between designs.

In addition to fully-provisioned networks, we analyze oversubscribed networks, which typically reduce network cost. A common practice is to oversubscribe the top of rack (ToR) switches to provide full intra-rack bandwidth, but reduced inter-rack bandwidth. The end result is lower network cost and lower bisection bandwidth compared to a fully-provisioned network. The bracketed entries in the first row of Table 1 show the components required to build a 24,576-end-host network with a 3:1 oversubscription ratio at the ToR layer (as employed in production datacenters [28]). This oversubscription allows $3\times$ the number of end hosts with only a 40–60% increase in hardware components.

As shown in the first row of Table 1, both fully-provisioned and oversubscribed traditional FatTree designs have several shortcomings at scale: 1) the large number of expensive and power-hungry optical transceivers required in between tiers, 2) the deployment and maintenance overhead of many long fiber-optic cables, and 3) the replication of packaging and ancillary hardware (e.g. CPU, PHY chips, power supply, etc.) within each discrete switch box.

2.2 Chassis-based FatTrees

The disadvantages of traditional FatTree designs have led several industrial players to design and build chassis-based FatTree topologies [29] in which multiple switch chips are integrated into a common box, known as a chassis, and connected using energy-efficient copper backplane traces. By increasing the density of switching capacity, this architecture requires fewer optical transceivers and long fiber runs which reduces hardware, power, and deployment costs. Figure 1(b) illustrates a 32-host FatTree built with a chassis architecture, including how switch chips are connected in a 3-stage Clos within a chassis. This chassis-based architecture has been a critical factor enabling large-scale datacenter deployments, where the hardware and management costs of a traditional FatTree would be infeasible [10, 28].

The second row of Table 1 shows the components required to build an 8,192-node network out of 128-port chassis, with each chassis using 24 16-port chips in a 3-stage Clos. To facilitate comparison to the traditional architecture, we can assume two 16-port chips are implemented with one 32-port chip. Only two tiers of switch chassis are required, reducing the fiber cabling and number of optical transceivers by 1/3. The number of discrete switch boxes is reduced by almost an order of magnitude. Similar savings are realized for a 3:1 oversubscribed network using chassis switches. The downside of the chassis architecture is that more switch chips are required and the worst-case number of hops (and latency) through the network is larger. Chassis power consumption also presents a scaling challenge as network speeds increase.

3. MULTI-CHANNEL FATTREES

As mentioned above, modern link speeds are achieved by combining an ever-increasing number of parallel underlying channels. The same is true for merchant silicon switch chips; switching capacity is scaling through the addition of more SerDes channels. While current designs aggregate these channels to switch multi-channel links, the switch chips can also be configured to instead expose individual channels as external ports, yielding large switch radices. Instead of grouping underlying channels together to form higher-bandwidth links, we propose a new P-FatTree topology that logically partitions the network among the channels.

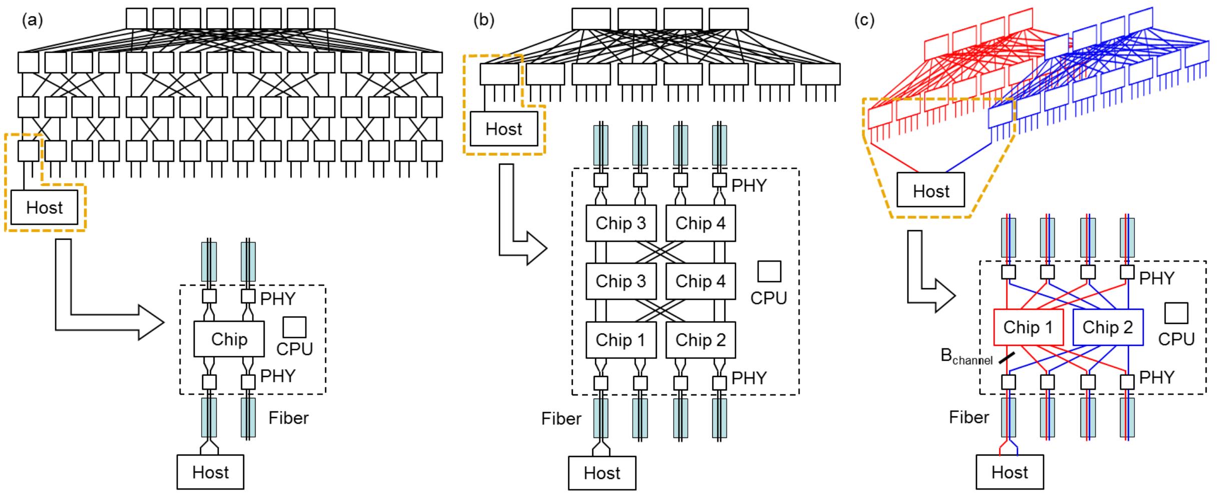

Figure 1: A (a) single-channel traditional, (b) single-channel chassis-based, and (c) multi-channel FatTree (P-FatTree).

Architecture	# Tiers	# Hops	# Trx	# Switch chips	# Sw. boxes	# Fibers
Traditional [3:1]	3 [3]	5 [5]	49 k [82 k]	1,280 [1,792] × 32-port	1,280 [1,792]	25 k [41 k]
Multistage chassis [3:1]	2 [2]	9 [9]	33 k [66k]	2,304 [3,840] × 32-port (4,608 [7,680] ×16-port)	192 [320]	16 k [33 k]
Multi-channel [3:1]	2 [2]	3 [3]	33 k [66k]	768 [1,280] × 32-port (1,536 [2,560] × 16 port)	192 [320]	16 k [33 k]

Table 1: Component counts for three different network architectures to support 8,192 end-hosts at full bisection bandwidth (unbracketed) and 24,576 end-hosts with a 3:1 oversubscription ratio at the ToR layer (bracketed). Each design is built from the same underlying 32-port switch chips.

3.1 P-FatTree design

For an N-channel link technology, P-FatTree implements N entirely disjoint, parallel FatTree networks, as shown in Figure 1(c). Figure 1(c) shows how an 8-port switch chassis can be composed of two 8-port switch chips each operating at $B_{channel} = \frac{1}{2}B_{port}$. Instead of grouping channels at each switch chip to increase port bandwidth, P-FatTree instead distributes the channels among parallel switch chips. This multi-channel design has the same switching capacity as the multistage chassis in Figure 1(b), but with fewer switch chips and backplane traces. P-FatTree's key insight is that by forgoing the abstraction that each path through the network be a single channel at the end-host link rate, it uses the underlying parallelism in links and switches to reduce network cost, energy consumption, and latency.

Cost: The third row of Table 1 shows the component counts for a 8,192-node P-FatTree cluster built from the same switch chips as the multistage chassis FatTree, but now configured with 8× ports each at 1/8th the bandwidth per port. Where 24 chips composed each multistage chassis, only 8 chips are required for a multi-channel P-FatTree chassis. Our multi-channel architecture maintains the lower cabling complexity and transceiver cost of the multistage chassis approach while further decreasing switch chip cost. Similar savings can be seen for the 3:1 oversubscribed network.

Power: In addition to hardware savings, P-FatTree reduces network power consumption relative to the multistage chassis FatTree. Figure 2 compares the power consumption of the 128-port switch chassis used in the 8,192-node cluster examples (second and third rows of Table 1), constructed using the conventional multistage chassis and P-FatTree's multi-channel chassis architectures. The SerDes and switch chips are the primary chassis power draws, respectively consuming 10 mW/Gb/s and 50 W per component, and we approximate the total chassis power consumption as the sum of these two factors. P-FatTree's multi-channel chassis requires 1/3rd the number of switch chips (8 instead of 24), half the number of backplane traces, and half the number of SerDes as the traditional multistage chassis. As the aggregate chip (and chassis) switching capacity increases, the SerDes power necessary to interconnect the switch chips begins to dominate the total chassis power consumption, with the multi-channel chassis being about twice as energy efficient.

Latency: P-FatTree also has lower network latency compared to both the traditional and multistage chassis FatTrees, respectively due to fewer inter-switch hops and intra-chassis hops. Because the number of tiers scales inversely with switch radix, maximizing the number of channels in P-FatTree minimizes packet head latency. However, splitting each link into too many channels increases packet serialization latency. Because total latency is the sum of head and serialization latencies, we must consider both to determine the net effect of moving to a multichannel network.

At zero network load, each switch chip introduces a port-to-port delay t_s associated with packet processing. The zero-

COTS product:	Broadcom Trident	Broadcom Tomahawk	Barefoot Tofino	(not released)	(not released)
Capacity:	1.28 Tb/s	3.2 Tb/s	6.4 Tb/s	12.8 Tb/s	25.6 Tb/s
Max. port BW:	32 p $\times$ 40 Gb/s	32 p $\times$ 100 Gb/s	64 p $\times$ 100 Gb/s	32 p $\times$ 400 Gb/s	32 p $\times$ 800 Gb/s
Max. radix:	128 p $\times$ 10 Gb/s	128 p $\times$ 25 Gb/s	256 p $\times$ 25 Gb/s	256 p $\times$ 50 Gb/s	512 p $\times$ 50 Gb/s

Table 2: Commercial off-the-shelf (COTS) switch chips and expected future chips.

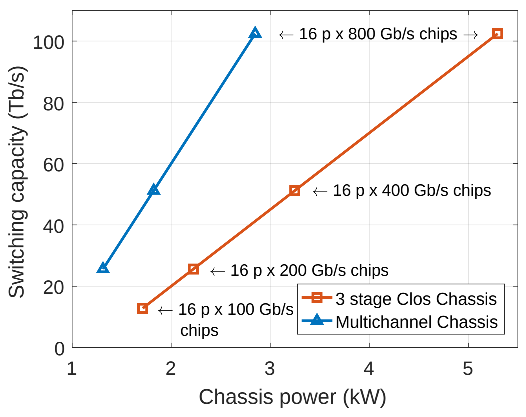

Figure 2: Chassis power consumption and switching capacity for 128-port multistage and multi-channel switch chassis. There is no multi-channel solution for standard 16 port $\times$ 100 Gb/s chips because 8 parallel channels are necessary for the 128-port chassis, and 100-Gb/s links can only be broken into 4 $\times$ 25 Gb/s channels.

load head latency is the product of t_s and the number of hops. The number of hops is proportional to the number of tiers: $\text{hops} = 2(\text{tiers}) - 1$, where $\text{tiers} = \log_{k/2}(H/2)$, k is the switch radix, and H is the number of hosts. Each link is split into N channels, with $N = 1$ corresponding to the traditional FatTree architecture. The packet serialization latency is NL/b where L is the packet length and b is the link rate. As an example, consider a cluster with H = 8,192 hosts, b = 400-Gb/s links, t_s = 200 ns, and switch chips with capacity C = 12.8 Tb/s; each chip is configured with $k = NC/b$ ports. In such a configuration, 1,500-byte packets experience minimum latency when traversing a P-FatTree composed of between 4 and 8 parallel networks. Serialization latency becomes more significant by 16 parallel networks, resulting in roughly the same latency as the traditional FatTree.

The above analysis applies to an unloaded network. Under load, we expect the queueing latency to outweigh serialization latency due to packet buffering resulting from port contention. Because P-FatTree has fewer switch chip hops relative to the traditional and multistage FatTrees, we expect it to have lower queueing latency and lower latency variance.

3.2 Multi-channel links and switches

Our design takes advantage of the fundamental shift in the way increasing link speeds are being achieved, namely the move from single-channel designs to multi-channel designs.

Link rate (Gb/s)	SerDes rate (Gb/s)	Num. channels	Notes
1	1	1	IEEE 802.3ab
10	10	1	IEEE 802.3ae
40	10	**4**	IEEE 802.3ba
100	25	**4**	IEEE 802.bj
200	50	**4**	{SR4,LR4}[†] [9, 17]
400	25	**16**	SR16* [5, 9, 18]
400	50	**8**	LR8* [9]

Table 3: Number of Ethernet channels required to support a variety of next-generation link speeds. [†] indicates proposals under consideration, and * indicates proposals in the standardization process.

A list of current and pending multi-channel Ethernet link designs is shown in Table 3, with references included to any current or pending standardization efforts. Note that for all future and pending link standards on the horizon, between four and sixteen channels are required to meet the desired link rate.

Just as links are moving to a multi-channel design, so are network switches. Each switch port is limited by the SerDes bandwidth (e.g., 25 Gb/s), and so to provide faster switches, each external switch port must connect to multiple internal ports on the switch chip itself. For example, a 100-Gb/s link might connect to a switch via an external QSFP28 connector, which internally splits out into four 25-Gb/s channels, connecting in turn to four 25-Gb/s switch chip ports. In this example, to provide B_{port} bandwidth to each end-host, given a SerDes bandwidth that is a quarter the desired per-port bandwidth ($B_{channel} = \frac{1}{4}B_{port}$), four channels are assigned to each port to achieve the desired port bandwidth. This reduces the effective switch radix by a factor of four. Consider Broadcom's Tomahawk chip, which has a 3.2-Tb/s capacity and uses 25-Gb/s SerDes. It can be operated with 128 ports at 25 Gb/s/port or with 32 ports at 100 Gb/s/port by ganging together four SerDes per port, reducing its radix by a factor of four from 128 to 32. Table 2 depicts a number of current (and potentially pending) merchant silicon switches, highlighting the maximum port bandwidth and a configuration exposing the highest radix (at a lower channel bandwidth).

4. END-HOST INTERFACE

Both the single-channel traditional (Figure 1a) and single-channel chassis-based (Figure 1b) FatTrees export a very simple delivery model, which is the abstraction of a single link running at $N \times B_{channel}$ bits per second. From the point of view of the end host, it is connected to a unified network fabric with a single link. In contrast, our proposed multi-channel network exports N separate links to each end point, each operating at $B_{channel}$ bits per second. As a result, each end point needs N separately addressable interfaces to N logical, disjoint network fabrics. Here we discuss several ramifications of this change.

4.1 The link layer

Each end point requires N separate link-level MAC addresses, one for each logical FatTree. NICs already support multiple MAC addresses to enable network virtualization, often with hardware support for in-NIC forwarding tables and in some cases vSwitch acceleration [22]. Multi-queue NIC drivers and hardware are a natural fit for mapping outgoing packets to different logical FatTrees, and RSS (receive-side scaling) could be used to steer incoming packets from a logical FatTree to specific cores, VMs, or containers. SR-IOV support enables guest VMs or containers to directly connect to one or more logical FatTrees with minimal overhead.

4.2 The IP layer

One question that arises is whether each connection to a logical FatTree should have its own unique IP address, (and thus be independently addressable from the host), or whether a host should only have a single IP address (and rely on e.g., ECMP [16] to stripe packets across the logical FatTrees).

The most straightforward approach would be to have a unique IP address per physical network, which could be assigned similarly to existing networks, or perhaps specially constructed (as in Portland [24]). Here each logical FatTree could use a disjoint portion of the IP address space.

Switches within each logical network would not need to be modified, as they would simply handle forwarding as they do today, however end-hosts and hypervisors will need to support $N\times$ larger forwarding and routing tables. This is because each end host would need to keep state for all N logical networks. Thanks to the prevalence of network virtualization, modern NIC hardware has some assists to help with those tables. For example, Mellanox's most recent ConnectX-5 has special tables for doing hypervisor forwarding for vSwitch [22].

4.3 Transport and congestion control

A goal of a P-FatTree-compatible transport layer is to maximally use up bandwidth across the N logical networks, while ensuring that traffic is not congested on one logical network when slack capacity is available on another. There are three main ways to utilize the bandwidth available across all N logical networks. The first, and most expedient, would be to rely on N-way ECMP to stripe packets across each of the networks. Such an approach could be implemented within the NIC in a straightforward way (by e.g., round-robin scheduling multiple TX queues, each corresponding to one logical network). However in the event of a failure in one of the logical networks, this ECMP mechanism would have to be updated, requiring a control plane that does not currently exist between the network and the end hosts. A second option would be to rely on MPTCP [26] to probe and spread traffic across the N logical networks. One disadvantage of MPTCP is that it is somewhat slow to converge, requiring multiple RTTs to probe for slack bandwidth.

New opportunities: A third option would be to use a very low-latency control plane to assign packets and flows to logical networks, such as pFabric [3], EyeQ [20], or pHost [13]. These approaches attempt to support a mixture of traffic—including a mixture of elephants and mice—on a single fabric. In P-FatTree, the existence of multiple physically disjoint logical networks would make deploying systems such as pFabric easier, since entire classes of traffic could be split out and controlled separately. This applies to other recent proposals, such as the "Jump the Queue" work [15] which segregates traffic into priority classes. P-FatTree would provide a physical partitioning of traffic classes into different logical parallel networks.

5. DESIGN IMPLICATIONS

Here we discuss several potential network management benefits that arise from the parallel nature of P-FatTree.

5.1 Network control plane

Depending on how network routing is managed (centralized or decentralized), the impact on network routing and forwarding varies. For distributed deployments relying on e.g., ISIS or OSPF, each logical FatTree can be managed independently, adding no additional complexity to the control plane. ToRs would need to participate in $N\times$ more routing protocol implementations.

Centralized and software-defined control planes will require more resources to support P-FatTree. The number of IP addresses, ports, and links that a centralized scheme needs to manage increases by a factor of N. There are ways of mitigating this increase in state, such as partitioning subsets of logical networks across different SDN controllers.

5.2 Middleboxes

Middleboxes are prevalent in real networks—some networks have as many middleboxes as routers [27]. P-FatTree presents both advantages and disadvantages to deploying middleboxes within the network. One advantage of P-FatTree is that it exposes channel-based parallelism to end points (and implicitly to middleboxes), rather than simply presenting a "fat pipe". This makes packet processing easier, since individual ports on a middlebox need only service e.g., 25 or 50 Gb/s of data, rather than a full 100 Gb/s, 400 Gb/s, or more. In this way, a single link's worth of bandwidth can be split across multiple physical middleboxes if needed, which is challenging in chassis-based FatTree designs.

On the other hand, the multi-channel nature of P-FatTree means that traffic from a single end host no longer travels on a single, unified fabric, but rather is split across multiple logical networks. For middleboxes supporting intrusion detection and security applications, ensuring that they "see" a unified view of end-host traffic becomes harder. It should be possible to manage the flow of traffic across multiple logical FatTrees to help mitigate this effect, if required.

5.3 Fault tolerance

A key feature of FatTree networks is that if a link or switch fails, there are many other paths to route traffic around that failure. For failures of a link or switch, P-FatTree provides yet another degree of freedom: rerouting traffic across entirely different logical FatTrees. On failure, the end host has a decision to make, which is whether to wait until the failed network recovers, or to migrate traffic to a different logical network during that convergence period. Depending on the time it takes to reconverge, it may be advantageous to fail over to another logical network to restore connectivity.

This improvement in fault tolerance only exists if the logical FatTrees are also physically uncorrelated. As a counterexample, imagine that the N logical FatTrees are identically deployed, so that each physical fiber or cable has all N channels of a single logical link, and that each chassis box has all N switch chips making up the N logical networks. In such a case, losing a single fiber or chassis box would result in losing *all* N channels, or all N switch chips. This could be avoided by permuting the assignment of logical links and switches to the physical network. As inspiration, the F10 [21] topology shows that "shuffling" the assignment of links to core switches improves fault tolerance. By choosing an appropriate embedding of logical single-channel networks to physical multi-channel networks, losing a link or chassis would result in losing N uncorrelated links or N uncorrelated switches, providing a potential avenue for improving network-wide fault tolerance.

5.4 Ongoing work

Future work will address a number of open questions identified in this initial investigation of P-FatTree, including: (1) How much additional switch buffer memory is required? (2) How best to mitigate the increase in network state? (3) How many parallel networks are feasible? (4) What are the tradeoffs in partitioning the network for fault tolerance?

5.5 Related work

There has been interest in addressing datacenter scalability with reconfigurable circuit-switched topologies, particularly using optical switching [12, 25]. While these proposals have significant potential impact, they have not been adopted in practice due to architectural and physical-layer challenges. Architecturally, circuit-switched approaches tend to require centralized control to compute optimal circuit assignments based on real-time network-wide demand. Such a tight control loop may be impractical for large-scale networks. At the physical layer, circuit switches have not been shown to scale to the port count and reconfiguration speeds necessary for large-scale networks.

P-FatTree may not realize the performance gains possible with an optical circuit-switched network, but still provides considerable advantages over conventional networks—and does so without requiring entirely new hardware or control-plane approaches. Because P-FatTree allows datacenters to scale more cost effectively, it may provide time for reconfigurable topologies to overcome the architectural and hardware challenges currently hampering their adoption.

6. CONCLUSION

Increases in the aggregate bandwidth demands of next-generation hosts and switches exceeds the increases in channel bandwidth enabled by new generations of CMOS manufacturing. As a result, networks are moving from a single-channel design to a multi-channel design. Today's network topologies erase this distinction, providing the illusion of a unified network fabric, with ever-increasing difficulty and cost. In this work we propose P-FatTree, which is a FatTree topology designed specifically for the future multi-channel reality. P-FatTree requires fewer switch chips and as a result has lower cost and power requirements than existing approaches. Furthermore, by embracing the parallel nature of the network itself, it enables compelling new ways to better manage and deliver application traffic.

Acknowledgements

This work is funded in part by the National Science Foundation under grants CNS-1314921 and CNS-1564185, and by a gift from Facebook.

7. REFERENCES

[1] 25/50G Ethernet Consortium. 25 and 50g ethernet specification, ver 1.4. http://25gethernet.org/.

[2] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity, data center network architecture. In *Proceedings of the ACM SIGCOMM Conference*, Seattle, WA, Aug. 2008.

[3] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker. pFabric: Minimal near-optimal datacenter transport. In *Proceedings of the ACM SIGCOMM Conference*, Hong Kong, China, Aug. 2013.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the clouds: A Berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, UC Berkeley, Feb 2009.

[5] CDFP MSA. CDFP 400 Gb/s MSA Consortium. http://www.cdfp-msa.com/.

[6] P.-C. Chiang, H.-W. Hung, H.-Y. Chu, G.-S. Chen, and J. Lee. 2.3 60Gb/s NRZ and PAM4 transmitters for 400GbE in 65nm CMOS. In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pages 42–43. IEEE, 2014.

[7] C. Clos. A Study of Non-blocking Switching Networks. *Bell System Technical Journal*, 32(2), 1953.

[8] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. In *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[9] Ethernet Alliance. 2016 Ethernet Roadmap. http://www.ethernetalliance.org/roadmap/.

[10] Facebook. Introducing "6-pack": the first open hardware modular switch. https://code.facebook.com/posts/717010588413497-/introducing-6-pack-the-first-open-hardware-modular-switch/.

[11] Facebook. Introducing data center fabric. https://code.facebook.com/posts/360346274145943/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/.

[12] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat. Helios: A hybrid electrical/optical switch architecture for modular data centers. In *Proceedings of the ACM SIGCOMM Conference*, New Delhi, India, Aug. 2010.

[13] P. X. Gao, A. Narayan, G. Kumar, R. Agarwal, S. Ratnasamy, and S. Shenker. pHost: Distributed near-optimal datacenter transport over commodity network fabric. In *Proceedings of ACM CoNEXT*, Heidelberg, Germany, Dec. 2015.

[14] 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. *Commun. ACM*, 54(3):95–104, Mar. 2011.

[15] M. P. Grosvenor, M. Schwarzkopf, I. Gog, R. N. M. Watson, A. W. Moore, S. Hand, and J. Crowcroft. Queues don't matter when you can jump them! In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation*, NSDI'15, pages 1–14, Berkeley, CA, USA, 2015. USENIX Association.

[16] IEEE. 802.1Qbp - equal cost multiple paths. http://www.ieee802.org/1/pages/802.1bp.html.

[17] IEEE 802.3bs. Adopted changes to 802.3bs project objectives. http://www.ieee802.org/3/bs/NGOATH_3bs_Objectives_16_0122.pdf.

[18] IEEE P802.3bs. 400 gb/s 100m MMF reach objective draft baseline proposal. http://www.ieee802.org/3/bs/public/14_11/king_3bs_02a_1114.pdf.

[19] IEEE P802.3bs - 400GbE Task Force Meeting. http://www.ieee802.org/3/bs/public/14_11/healey_3bs_01_1114.pdf.

[20] V. Jeyakumar, M. Alizadeh, D. Mazières, B. Prabhakar, C. Kim, and A. Greenberg. EyeQ: Practical network performance isolation at the edge. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, NSDI'13, pages 297–312, Berkeley, CA, USA, 2013. USENIX Association.

[21] V. Liu, D. Halperin, A. Krishnamurthy, and T. Anderson. F10: A fault-tolerant engineered network. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, NSDI'13, pages 399–412, Berkeley, CA, USA, 2013. USENIX Association.

[22] Mellanox. ConnectX-5 EN. http://www.mellanox.com/page/products_dyn?product_family=260&mtag=connectx_5_en_card.

[23] Memcached. Memcached. http://www.memcached.org/.

[24] R. N. Mysore, A. Pamboris, 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 *Proceedings of the ACM SIGCOMM Conference*, Barcelona, Spain, Aug. 2009.

[25] G. Porter, R. Strong, N. Farrington, A. Forencich, P. Chen-Sun, T. Rosing, Y. Fainman, G. Papen, and A. Vahdat. Integrating microsecond circuit switching into the data center. In *Proceedings of the ACM SIGCOMM Conference*, Hong Kong, China, Aug. 2013.

[26] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley. Improving datacenter performance and robustness with multipath TCP. In *Proceedings of the ACM SIGCOMM Conference*, Toronto, Ontario, Canada, Aug. 2011.

[27] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar. Making middleboxes someone else's problem: Network processing as a cloud service. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, Helsinki, Finland, Aug. 2012.

[28] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, A. Kanagala, J. Provost, J. Simmons, E. Tanda, J. Wanderer, U. Hölzle, S. Stuart, and A. Vahdat. Jupiter rising: A decade of Clos topologies and centralized control in Google's datacenter network. In *Proceedings of the ACM SIGCOMM Conference*, London, United Kingdom, Aug. 2015.

[29] A. Vahdat, M. Al-Fares, N. Farrington, R. N. Mysore, G. Porter, and S. Radhakrishnan. Scale-out networking in the data center. *IEEE MICRO*, 30(4):29–41, Aug. 2010.

[30] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 2–2, Berkeley, CA, USA, 2012. USENIX Association.