Lecture #11: Data center & Cloud networks

WPI CS4516 Spring 2019 D term

Instructor: Lorenzo De Carli (ldecarli@wpi.edu)

Data center networking

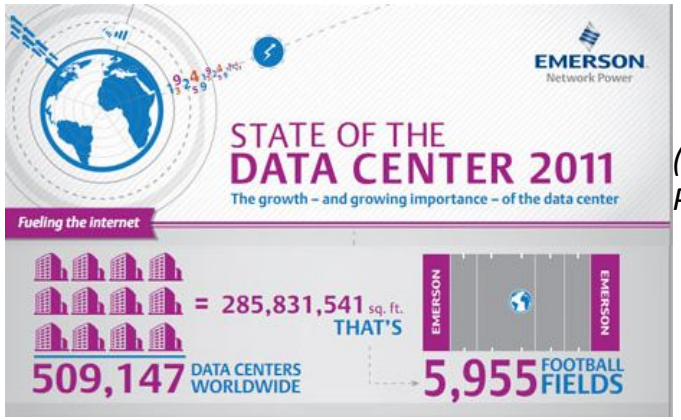
- **Relevant:** data centers are common and used to provide important services to large number of users
  - Campus/enterprise data centers: run small, well-defined set of services (business apps, email, storage, etc.)
  - Cloud: either heterogeneous (e.g., customer-driven applications), or homogeneous (e.g., mapreduce)
- **Different:** data center networking is fundamentally different from “traditional” data center networking

Structure of this lecture

- **Clos topologies:** an approach to provide high data center bandwidth w/ commodity, cheap equipment
- What do we talk about when we talk about data centers? Review of **data center measurement studies** from the last 10 years
- Review of **TCP pathologies in data centers**

Data center relevance

- Why do we care about data center networks?
  - Data centers host servers that **run many of the computations that organizations and individual users depend on**



(Source: Emerson Network Power, 2012)

  - In 2007, 1.5% of entire US electricity production went into data centers (source: EPA study)
    - Likely to be much higher now

Data center relevance - II

- Why is so much computation carried in data centers?
  - **Cloud computing:** many applications now require server backends that run in dedicated clouds
    - Think of GMail, Facebook, Outlook and pretty much any app published in the last few years
  - **Economies of scale:** densely packing servers in a space w/ dedicated power lines, cooling etc. is more convenient than having them scattered around
    - Even non-cloud-based apps (e.g. employee management system) are now moving towards data centers

Data center primer

- On-premise vs off-premise
  - **On-premise:** built, owned and run by an organization for that organization's purposes
  - **Off-premises:** run by third-party, and leased (fully or in part) to organizations needing computational muscle

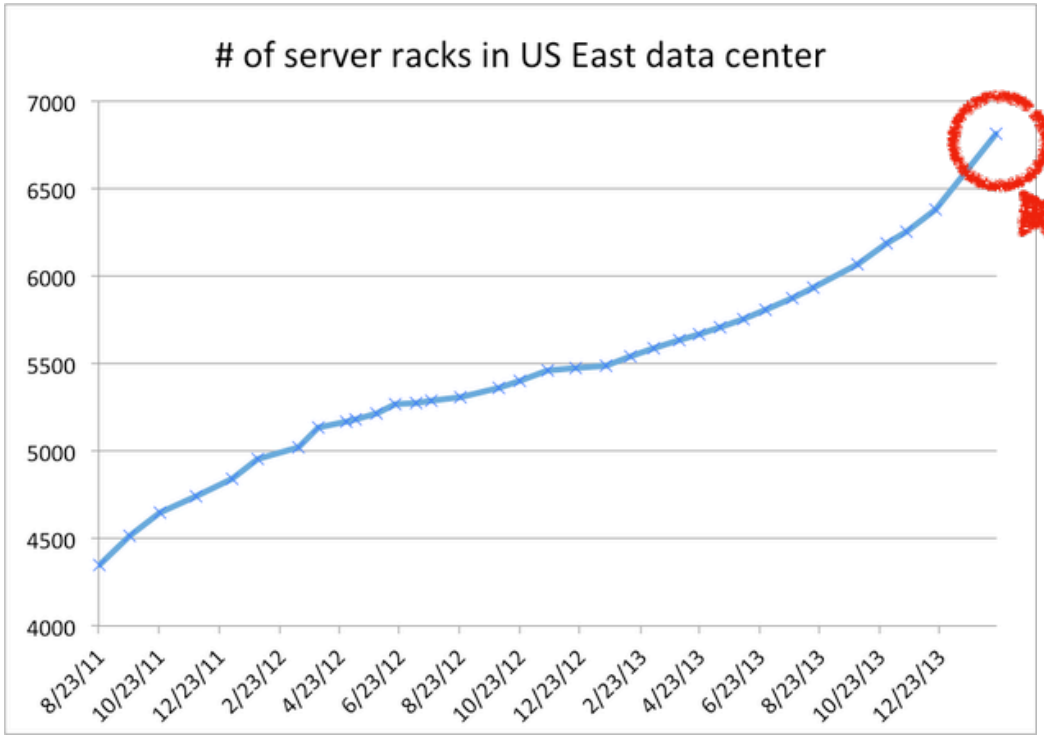
Data center vs cloud

- Easy to confuse the two concepts
- **Data center:** a physical place dedicated to store and run servers
- **Cloud computing:** a paradigm to provision and share computational resources while abstracting away the computational substrate (typically using virtualization)
  - Actual serves are **not visible to the users of the cloud**; virtualization is used to consolidate multiple users on same machine while giving the “illusion” of dedicated servers
  - Placement of virtual machines, network configuration etc. are all **managed transparently** to the user

Data center vs cloud

- Data center is an **entity providing computational resources**
- Cloud computing is a **paradigm for making use of those resources**
- Can have data center without cloud, but not the reverse!
- Clouds are typically used to manage and share off-premise data centers, but not always
  - Can have **private cloud**, run on privately-owned data center by a single organization

How big are these things, anyway?

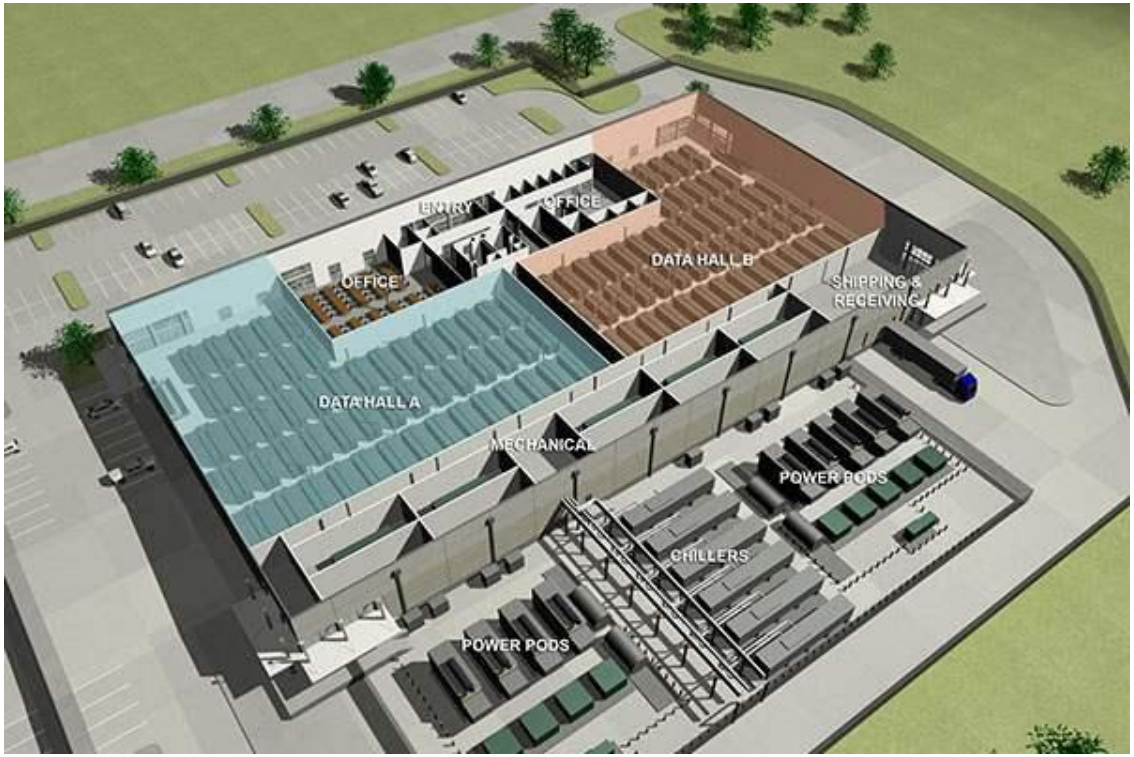


Amazon US East: est. 6382 racks (between 229K and 408K servers) at the end of 2013

Source of estimate: Huan Liu (https://huanliu.wordpress.com/2014/02/26/amazon-ec2-grows-62-in-2-years/)

Not all data centers are this big! (e.g. small campus/company/research data centers)

Data center structure



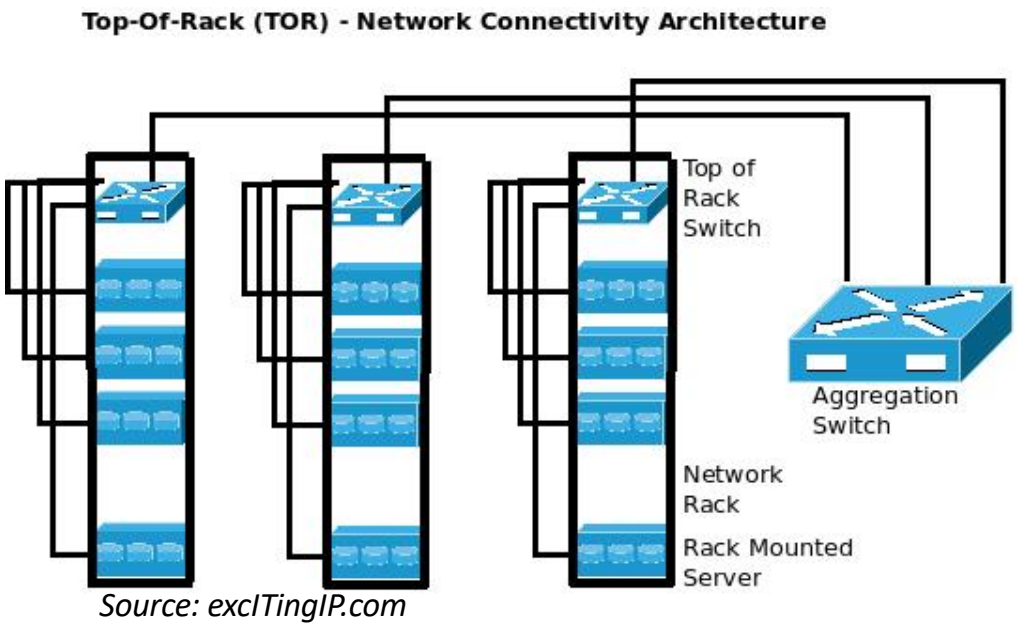
Source: CyrusOne

Data center structure - II

- Note: **physical layout** is different from **physical and logical topology**!
- **Physical layout:** how servers are placed within racks and racks are placed on the data center's floor
- **Physical topology:** how data center switches and servers are connected by cabling
- **Logical topology:** how routing policies select links and carry traffic from sources to destinations within the data center

Data center structure - III

- Data center organization principle: try to keep computation local (i.e., between servers who are close in terms of #hops)
- Typical local organization:



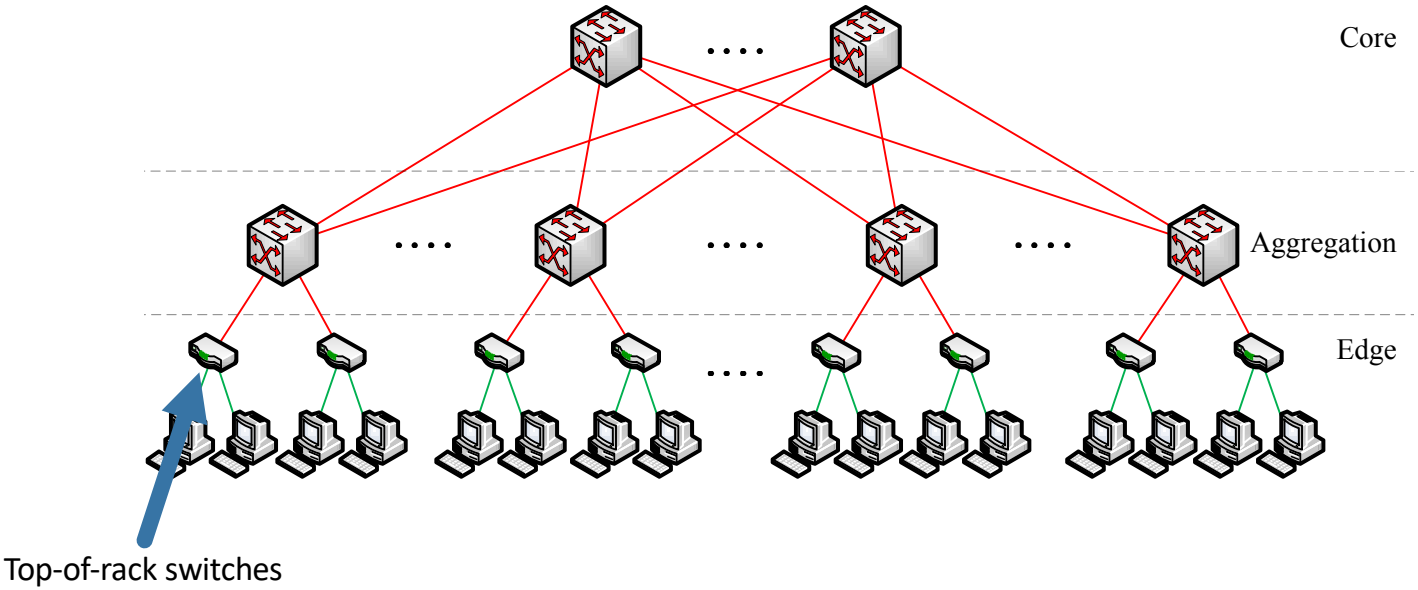
Source: exclTingIP.com

Servers in same rack are connected to dedicated rack switch and can talk to each other at full BW

Other organization: middle-of-row (one switch per set of racks)

Data center structure - IV

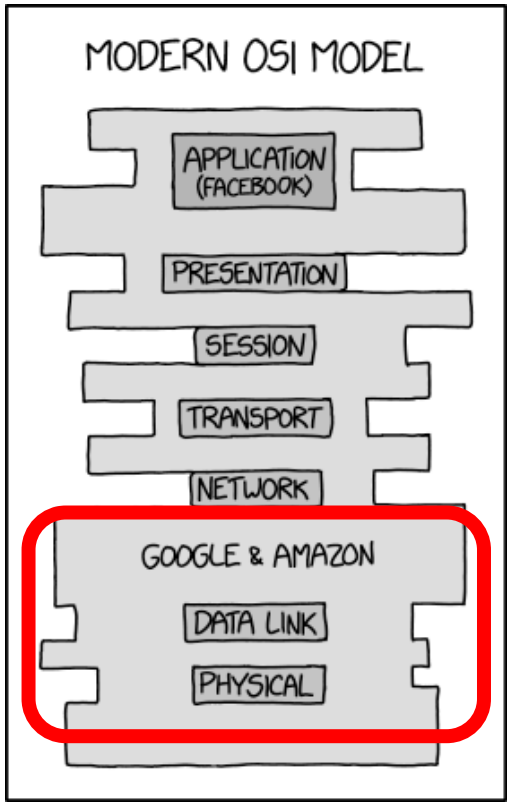
Typical global organization:



Note: smaller data center may lack the intermediate aggregation layer

How relevant is data center networking?

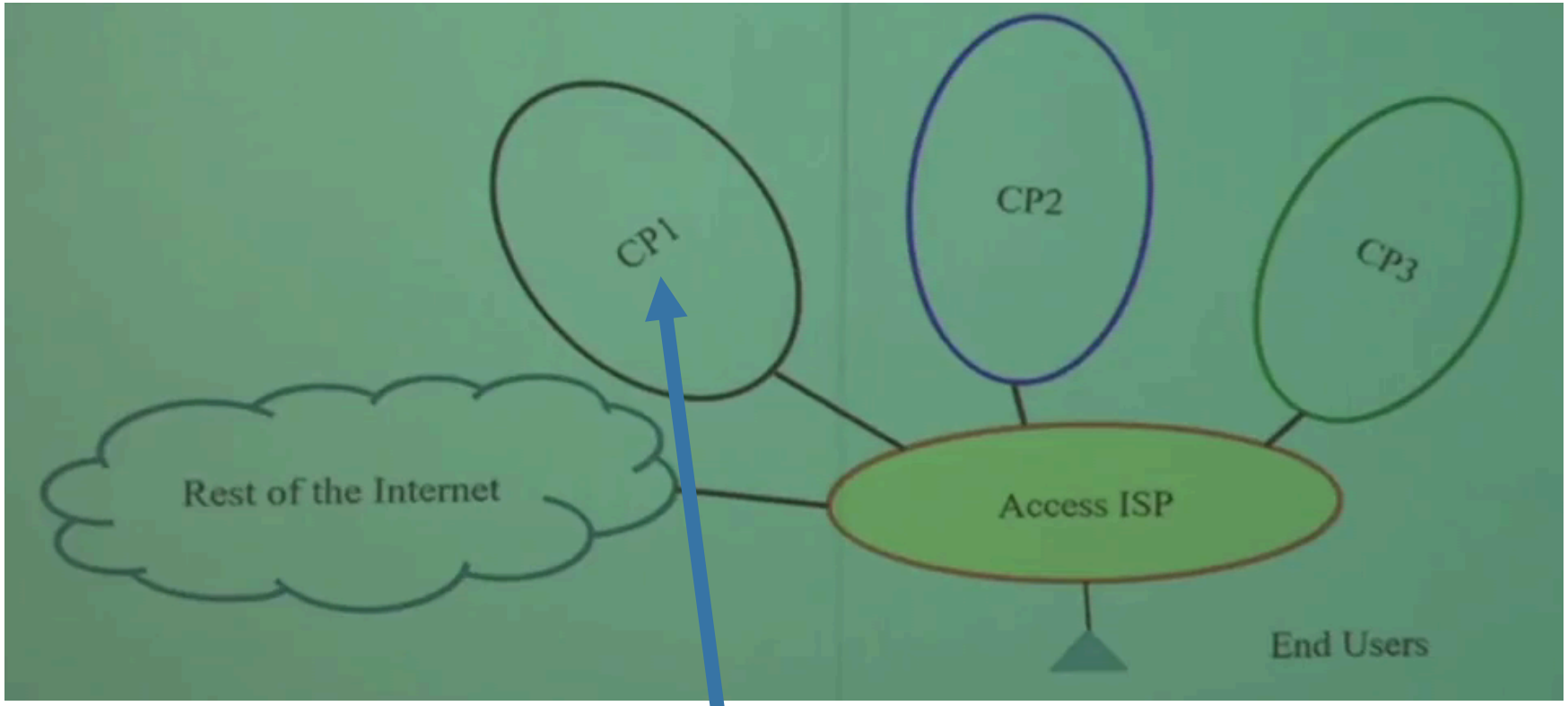
Well…



☺

(https://xkcd.com/2105/)

You are using data center networks without knowing it!



Source: Mostafa Ammar, "The Time-Traveling Computer Networking Researcher and Other Short Stories"

Large content providers (eg. Google) have direct links to/from many ISPs/when you connect to GMail, after a few hops your packets travel entirely within the Google network!

What makes data center networks unique?

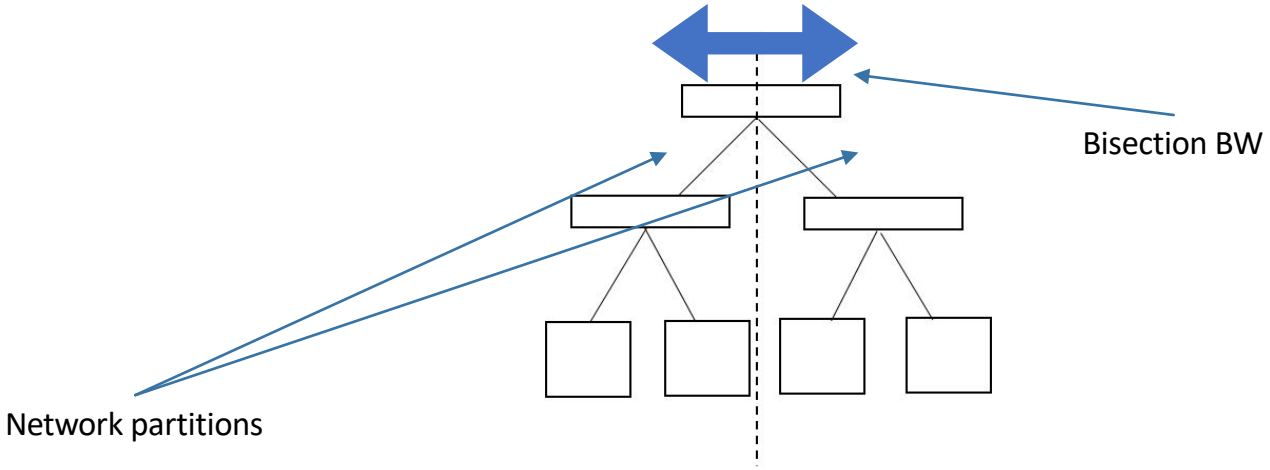
- **Very different settings** from wide area networks (WAN) and the general Internet!
- Under control of **single organization**:
  - **Backward-compatibility** not important
    - E.g. can deploy **customized TCP** without worrying about interaction w/ “traditional” TCP
  - Can really easily **upgrade to new protocols, hardware** etc.!
- **Homogeneity:** unified technology, oftentimes unified switch/server/OS vendor, etc.

What makes data center networks unique? - II

- **Very high bandwidth** between pairs of hosts (up to full Ethernet BW, 1Gbps/10Gbps)
- **Very low latency** (<250 µs w/o queuing)
  - May make RTT/RTO estimation noisy
- **Little statistical multiplexing:** limited numbers of flows per path

Bisection bandwidth

- Measure of the **capacity of a data center network between two partitions** connected through the core layer
- **Why is it important?** Core layer is a potential chokepoint as all communications between distant hosts must go through it



Source: Wikipedia

Data center topologies

A Scalable, Commodity Data Center Network Architecture

Mohammad Al-Fares
malfares@cs.ucsd.edu

Alexander Loukissas
aloukiss@cs.ucsd.edu

Amin Vahdat
vahdat@cs.ucsd.edu

Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0404

ABSTRACT

Today's data centers may contain tens of thousands of computers with significant aggregate bandwidth requirements. The network architecture typically consists of a tree of routing and switching elements with progressively more specialized and expensive equipment moving up the network hierarchy. Unfortunately, even when deploying the highest-end IP switches/routers, resulting topologies may only support 50% of the aggregate bandwidth available at the edge of the network, while still incurring tremendous cost. Non-uniform bandwidth among data center nodes complicates application design and limits overall system performance.

In this paper, we show how to leverage largely commodity Ethernet switches to support the full aggregate bandwidth of clusters consisting of tens of thousands of elements. Similar to how clusters of commodity computers have largely replaced more specialized SMPs and MPPs, we argue that appropriately architected and interconnected commodity switches may deliver more performance at less cost than available from today's higher-end solutions. Our approach requires no modifications to the end host network interface, operating system, or applications; critically, it is fully backward compatible with Ethernet, IP, and TCP.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Network topology; C.2.2 [**Network Protocols**]: Routing protocols

General Terms

Design, Performance, Management, Reliability

Keywords

Data center topology, equal-cost routing

1. INTRODUCTION

Growing expertise with clusters of commodity PCs have enabled a number of institutions to harness petaflops of computation power and petabytes of storage in a cost-efficient manner. Clusters consisting of tens of thousands of PCs are not unheard of in the largest institutions and thousand-node clusters are increasingly common in universities, research labs, and companies. Important applications classes include scientific computing, financial analysis, data analysis and warehousing, and large-scale network services.

Today, the principle bottleneck in large-scale clusters is often inter-node communication bandwidth. Many applications must exchange information with remote nodes to proceed with their local computation. For example, MapReduce [12] must perform significant data shuffling to transport the output of its map phase before proceeding with its reduce phase. Applications running on cluster-based file systems [18, 28, 13, 26] often require remote-node access before proceeding with their I/O operations. A query to a web search engine often requires parallel communication with every node in the cluster hosting the inverted index to return the most relevant results [7]. Even between logically distinct clusters, there are often significant communication requirements, e.g., when updating the inverted index for individual clusters performing search from the site responsible for building the index. Internet services increasingly employ service oriented architectures [13], where the retrieval of a single web page can require coordination and communication with literally hundreds of individual sub-services running on remote nodes. Finally, the significant communication requirements of parallel scientific applications are well known [27, 8].

There are two high-level choices for building the communication fabric for large-scale clusters. One option leverages specialized hardware and communication protocols, such as InfiniBand [2] or Myrinet [6]. While these solutions can scale to clusters of thousands of nodes with high bandwidth, they do not leverage commodity parts (and are hence more expensive) and are not natively compatible with TCP/IP applications. The second choice leverages commodity Ethernet switches and routers to interconnect cluster machines. This approach supports a familiar management infrastructure along with unmodified applications, operating systems, and hardware. Unfortunately, aggregate cluster bandwidth scales poorly with cluster size, and achieving the highest levels of bandwidth incurs non-linear cost increases with cluster size.

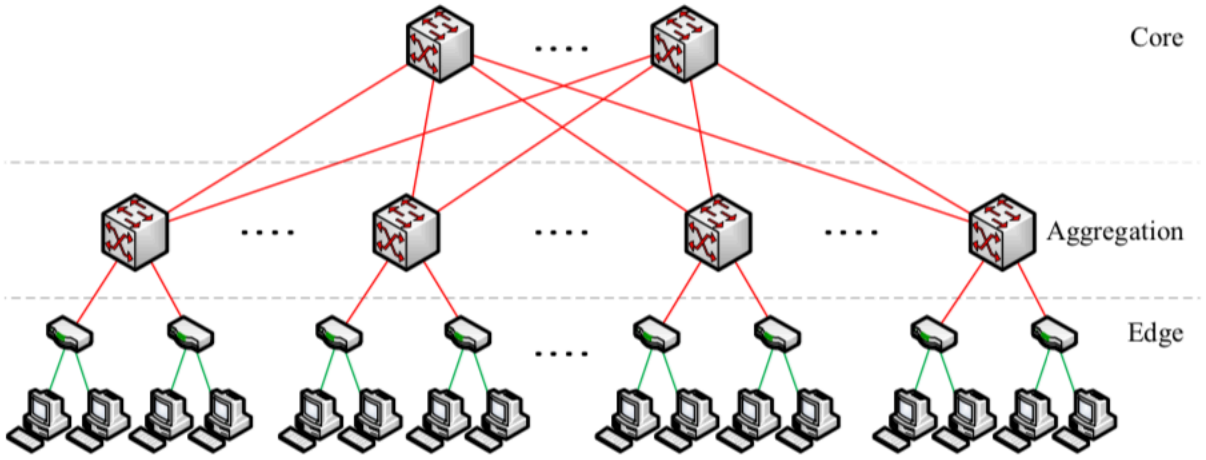
For compatibility and cost reasons, most cluster communication systems follow the second approach. However, communication bandwidth in large clusters may become oversubscribed by a significant factor depending on the communication patterns. That is, two nodes connected to the same physical switch may be able to communicate at full bandwidth (e.g., 1Gbps) but moving between switches, potentially across multiple levels in a hierarchy, may limit available bandwidth severely. Addressing these bottlenecks requires non-commodity solutions, e.g., large 10Gbps switches and routers. Further, typical single path routing along trees of interconnected switches means that overall cluster bandwidth is limited by the bandwidth available at the root of the communication hierarchy.

SIGCOMM'08, August 17–22, 2008, Seattle, Washington, USA.

Bandwidth issues

- Fundamental tension: **oversubscription** vs **cost**
  - **Oversubscription:** ratio of worst-case aggregate bandwidth to total bisection bandwidth
  - E.g.: 1:1 means that **all hosts can communicate to all other hosts at full BW**; 8:1 means that the bisection bandwidth is **1/8 of the worst-case bandwidth requirement**
- Oversubscribing by a large factor **limits bandwidth**, but enables use of **cheaper switches**, **less switches** and **simpler cabling** (substantial cost savings!)

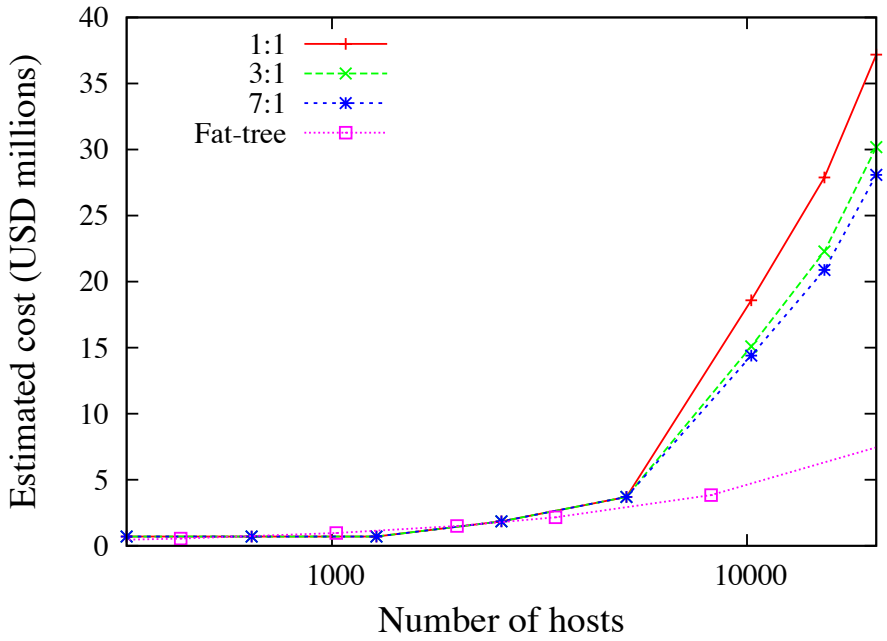
Traditional data center topologies



Issue #1: require expensive high-performance switches in the core

Traditional data center topologies - II

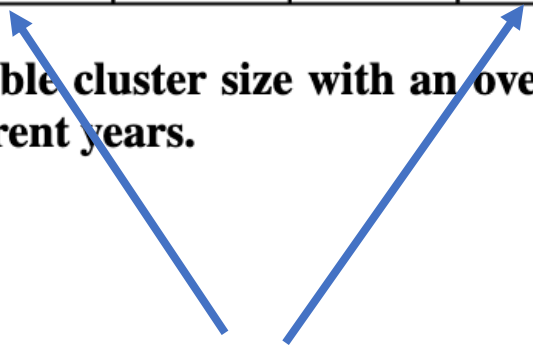
Issue #2 (related): low oversubscription in large data centers incurs prohibitively high costs



More on cost-benefit analysis

	Hierarchical design			Fat-tree		
Year	**10 GigE**	**Hosts**	**Cost/ GigE**	**GigE**	**Hosts**	**Cost/ GigE**
2002	28-port	4,480	$25.3K	28-port	5,488	$4.5K
2004	32-port	7,680	$4.4K	48-port	27,648	$1.6K
2006	64-port	10,240	$2.1K	48-port	27,648	$1.2K
2008	128-port	20,480	$1.8K	48-port	27,648	$0.3K

Table 1: The maximum possible cluster size with an oversubscription ratio of 1:1 for different years.

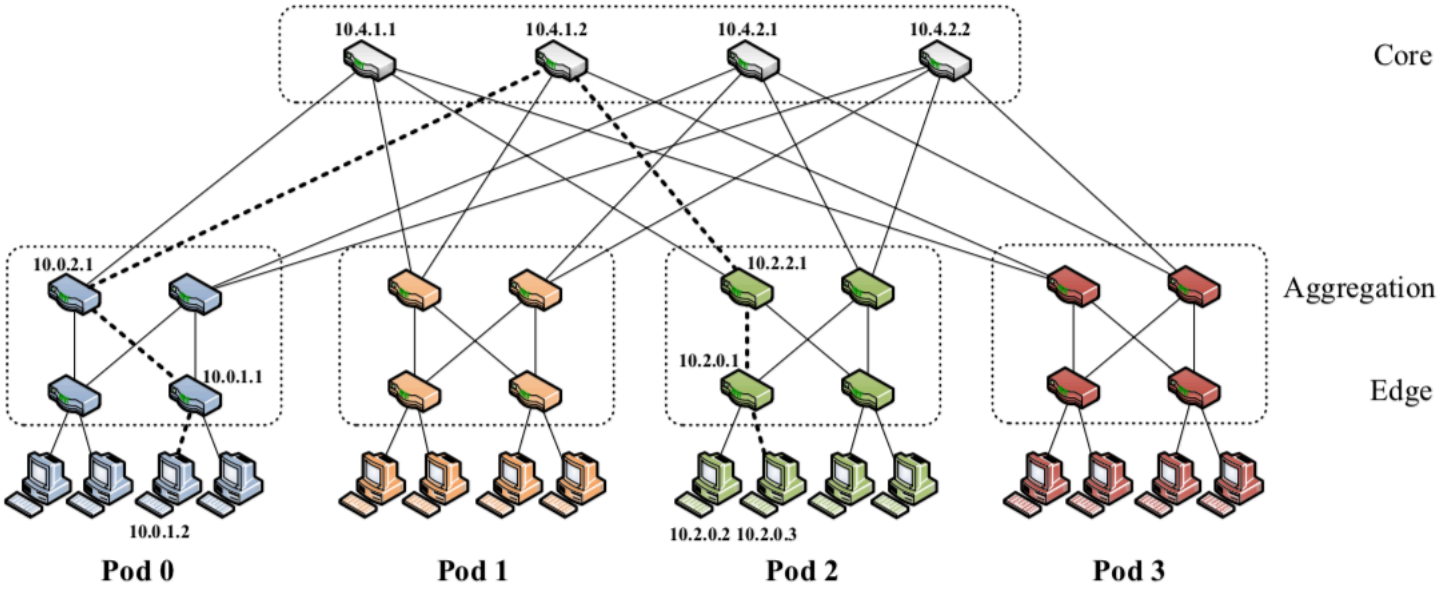


Why this difference?

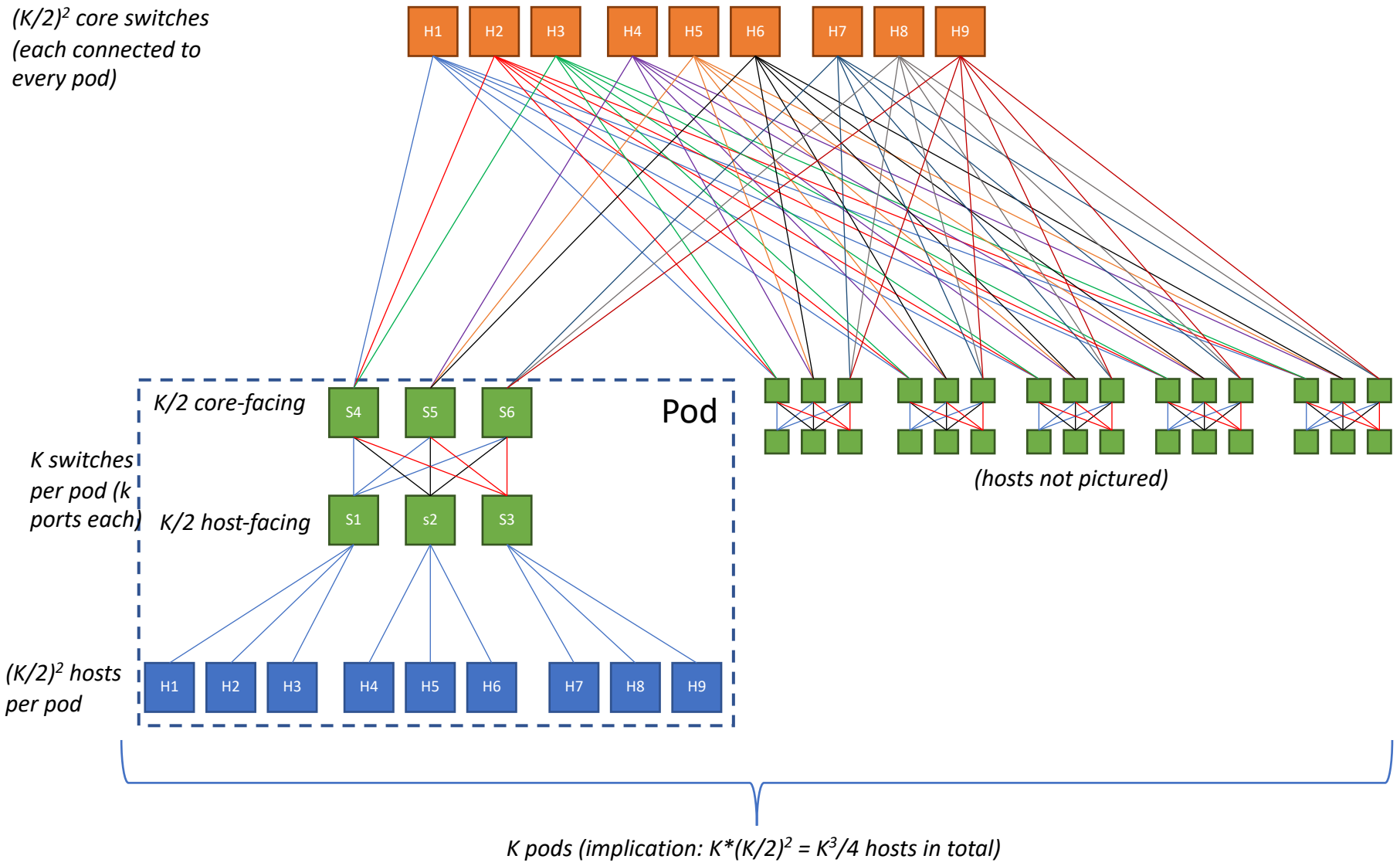
Fat trees

- Originate from work on optimized telephone networks (Charles Clos, 1953)
- Special instance of **Clos topology**
- **Rearrangeably non-blocking:** for arbitrary communication patterns, there exists a set of paths that allow servers to communicate at full bandwidth
- **High fan-out:** lots of alternative paths between the same endpoints (allows high bisection BW by statistical multiplexing of flows on paths)
- Achieving 1:1 oversubscription requires **multipath routing:** flows between same pair of switches must be distributed among multiple network paths

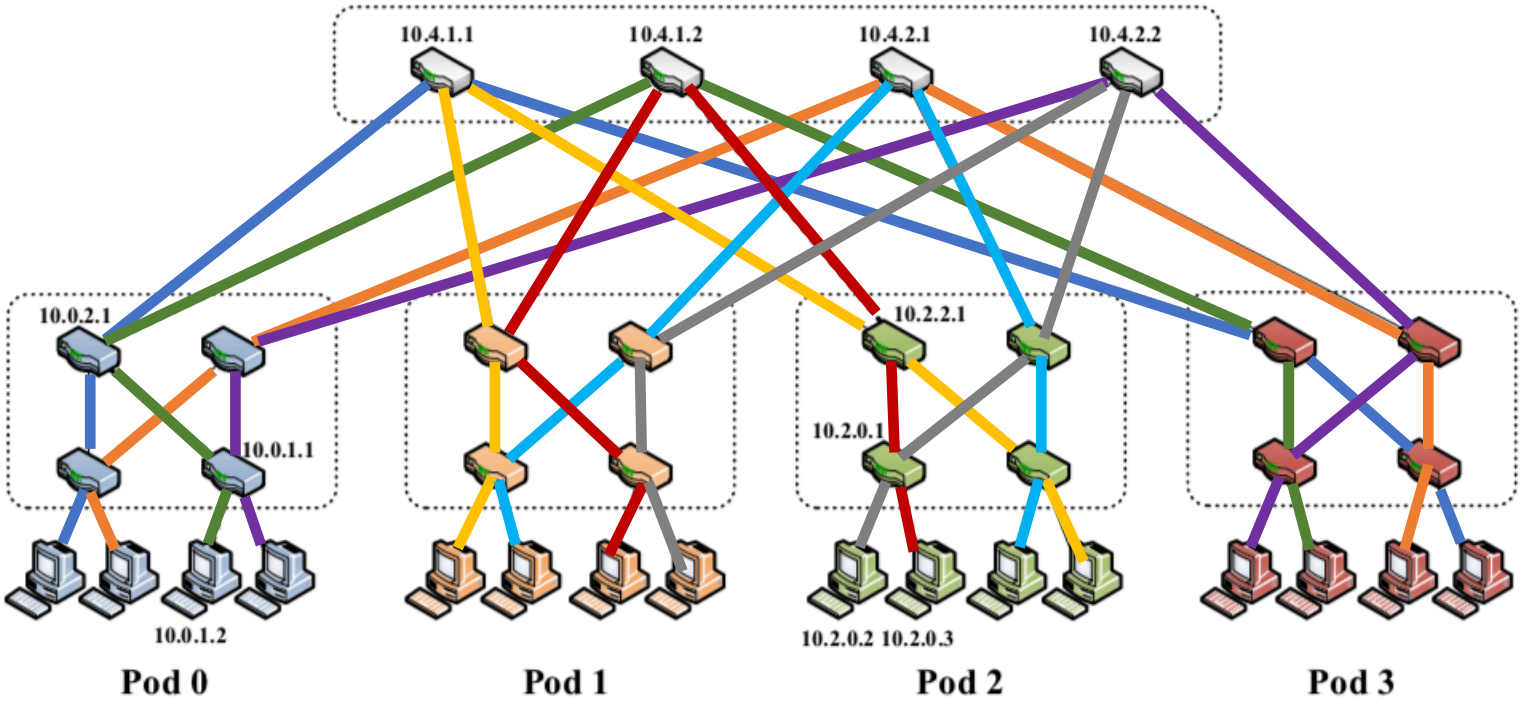
Fat tree topology - example



Another example with K=6



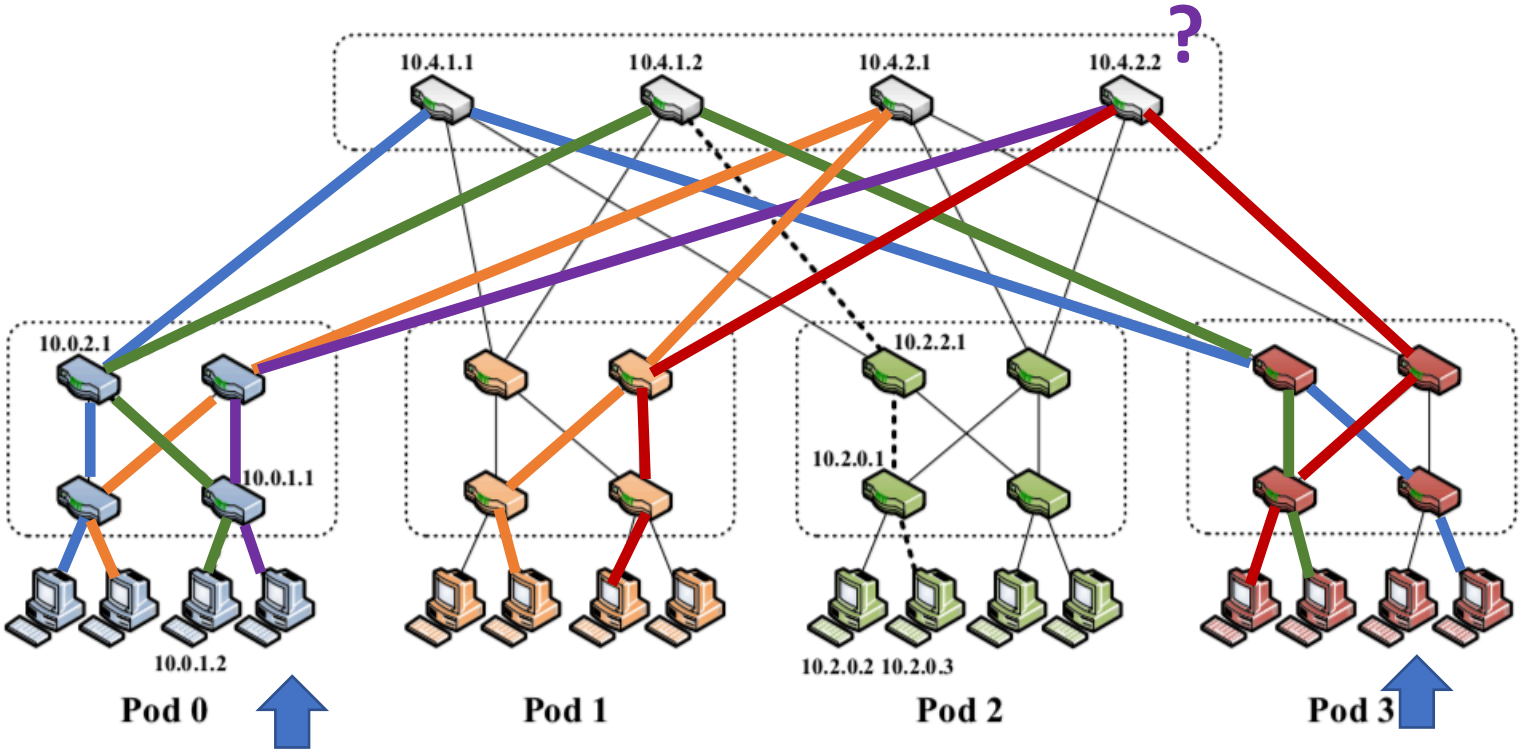
Rearrangeably non-blocking



All good then?

- Well, it’s tricky
- **We can’t know a priori the set of flows** that will be established over the life of the data center
  - So it is possible to make decisions that **will prevent to utilize full bandwidth** in the future
  - (that’s where “rearrangeably” comes into play)
- Also, differently from a telephone network, **a single host may establish multiple flows** with many other destinations

Bad decisions



How bad is it?

- In a **packet-switched network**, “bad decisions” are less problematic because **multiple flows can be multiplexed on the same path**
- Still, bad path allocation can **degrade bandwidth**
- One can:
  - Attempt to **guarantee that multiple flows rarely end up sharing a path** (routing algorithm 1)
  - Rearrange flows based on **local information** (r. a. 2)
  - Rearrange flows based on **global information** (r. a. 3)

Multipath routing

- Making good use of the available paths on a fat-tree topology requires **multipath routing**
  - **Traditional routing algorithms** are not designed for that – for every possible pair of communicating hosts there is **one unique communication path**
  - **ECMP (equal-cost multipath)** is an extension to traditional routing where **multiple paths of equal costs** could be installed in a forwarding table
  - Flows are **load-balanced** across paths

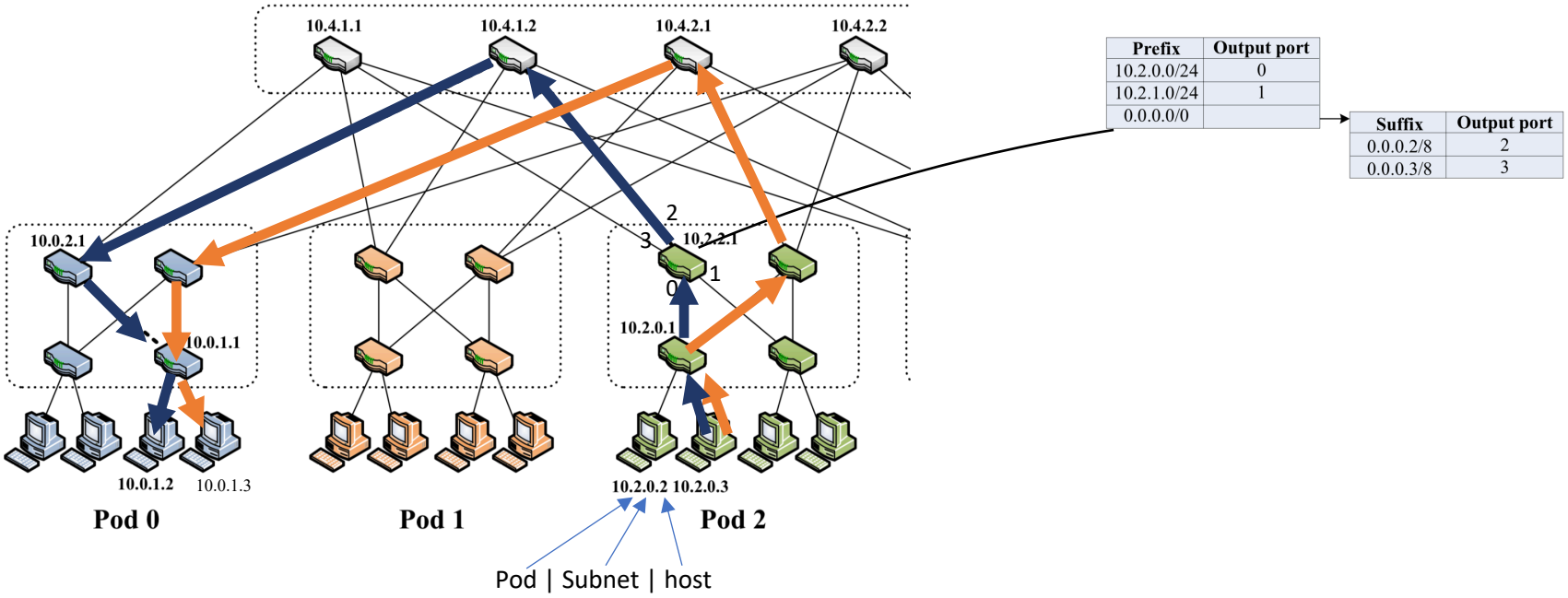
Problems with ECMP

- **Static** flow balancing
  - Typically, **hash of the flow five-tuple** (src IP, dst IP, src port, dst port, proto) defines path
  - Fast, stateless, but **load-ignorant**
- **Limited path diversity** (16 - seems a lot, but it is not for large data center)
- Requires **large routing tables** (1 prefix for each path to each destination)

Routing on fat tree topologies - I

- How can flows towards destinations in the same remote pod be **distributed** across multiple links?
  - While supporting multiple paths…
  - And **without running out of space** in the routing tables
- Solution: two-level routing table
  - **Level 1** routes traffic to local pod
  - **Level 2** used to spread traffic to remote servers based on last bits of destination IP address (statistical multiplexing)

Two-level routing: example



Example 1: from 10.2.0.3 to 10.0.1.2

Example 2: from 10.2.0.3 to 10.0.1.3

No overlap between paths!

Static routing - considerations

- **Simple**
  - Requires clever routing table generation algorithm and near-standard TCAM hardware
- **Not optimal**
  - Certain communication patterns can create ”hotspots” (lots of flows on the same path)
- Getting better performance requires **allocating flows dynamically** (i.e., periodically revise routing decisions)

Routing on fat-tree topologies - II

- A simple way to perform dynamic load-balancing is to have **switches track the bandwidth utilization** on each output port
- By default, packets are routed using pre-defined paths – but if a pod switch identifies imbalance between port utilization it can **reassign flows to different ports**
- **Core switches** have only one link to each pod so they **use static routing**

Issues with local flow scheduling

- Pod switches **do not have global view** of the network
- Decisions that are locally optimal **may interact poorly in the core**
  - If multiple switches decide to reassign flows to ports that happen to be connected to the same core switch, hotspots will form
- Can’t find optimal solution (Bin-packing, NP-hard)
  - Must use heuristic

Routing on fat-tree topologies - III

- **Centralized scheduler:** keep track of link utilization globally and reassign flows to optimize bandwidth
- Can't do that for every flow
  - Too much communication, computation overhead
- Solution: **schedule "elephant" flows**
- Switches report large flows, central scheduler find paths not already assigned to large flows

Comments on global flow scheduling

- **Global visibility** allows to find globally optimal scheduling
- Requires a **central scheduler**
  - Does it remind you of something?
- **If you can do it, enables the most efficient bandwidth utilization**

Routing paradigms - comparison

Test	Tree	Two-Level Table	Flow Classification	Flow Scheduling
Random	53.4%	75.0%	76.3%	93.5%
Stride (1)	100.0%	100.0%	100.0%	100.0%
Stride (2)	78.1%	100.0%	100.0%	99.5%
Stride (4)	27.9%	100.0%	100.0%	100.0%
Stride (8)	28.0%	100.0%	100.0%	99.9%
Staggered Prob (1.0, 0.0)	100.0%	100.0%	100.0%	100.0%
Staggered Prob (0.5, 0.3)	83.6%	82.0%	86.2%	93.4%
Staggered Prob (0.2, 0.3)	64.9%	75.6%	80.2%	88.5%
Worst cases:				
Inter-pod Incoming	28.0%	50.6%	75.1%	99.9%
Same-ID Outgoing	27.8%	38.5%	75.4%	87.4%

Table 2: Aggregate Bandwidth of the network, as a percentage of ideal bisection bandwidth for the Tree, Two-Level Table, Flow Classification, and Flow Scheduling methods. The ideal bisection bandwidth for the fat-tree network is 1.536Gbps.

Part II - what do we talk when we talk about data centers

Nature of data center traffic

- Not something companies like to disclose
  - Mix of customers/traffic, technologies, scale are typically kept private
  - **Large-scale studies of data center** traffic are few and far in between
  - We are going to review some of them

Data center traffic characteristics (2010)

Network Traffic Characteristics of Data Centers in the Wild

Theophilus Benson*, Aditya Akella* and David A. Maltz†
*University of Wisconsin–Madison
†Microsoft Research–Redmond

ABSTRACT

Although there is tremendous interest in designing improved networks for data centers, very little is known about the network-level traffic characteristics of current data centers. In this paper, we conduct an empirical study of the network traffic in 10 data centers belonging to three different types of organizations, including university, enterprise, and cloud data centers. Our definition of cloud data centers includes not only data centers employed by large online service providers offering Internet-facing applications, but also data centers used to host data-intensive (MapReduce style) applications. We collect and analyze SNMP statistics, topology, and packet-level traces. We examine the range of applications deployed in these data centers and their placement, the flow-level and packet-level transmission properties of these applications, and their impact on network utilization, link utilization, congestion, and packet drops. We describe the implications of the observed traffic patterns for data center internal traffic engineering as well as for recently-proposed architectures for data center networks.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Design studies; Performance attributes

General Terms

Design, Measurement, Performance

Keywords

Data center traffic, characterization

1. INTRODUCTION

A *data center* (DC) refers to any large, dedicated cluster of computers that is owned and operated by a single organization. Data centers of various sizes are being built and employed for a diverse set of purposes today. On the one hand, large universities and private enterprises are increasingly consolidating their IT services within on-site data centers containing a few hundred to a few thousand servers. On the other hand, large online service providers, such as Google, Microsoft, and Amazon, are rapidly building geographically diverse cloud data centers, often containing more than 10K servers, to offer a variety of cloud-based services such as E-mail, Web servers, storage, search, gaming, and Instant Messaging. These service providers also employ some of their data centers to run large-scale data-intensive tasks, such as indexing Web pages or analyzing large data-sets, often using variations of the MapReduce paradigm [6].

Despite the growing applicability of data centers in a wide variety of scenarios, there are very few systematic measurement studies [19, 3] of data center usage to guide practical issues in data center operations. Crucially, little is known about the key differences between different *classes* of data centers, specifically university campus data centers, private enterprise data centers, and cloud data centers (both those used for customer-facing applications and those used for large-scale data-intensives tasks).

While several aspects of data centers still need substantial empirical analysis, the specific focus of our work is on issues pertaining to a data center network's operation. We examine the sending/receiving patterns of applications running in data centers and the resulting link-level and network-level performance. A better understanding of these issues can lead to a variety of advancements, including traffic engineering mechanisms tailored to improve available capacity and reduce loss rates within data centers, mechanisms for improved quality-of-service, and even techniques for managing other crucial data center resources, such as energy consumption. Unfortunately, the few recent empirical studies [19, 3] of data center networks are quite limited in their scope, making their observations difficult to generalize and employ in practice.

In this paper, we study data collected from *ten data centers* to shed light on their network design and usage and to identify properties that can help improve operation of their networking substrate. The data centers we study include *three university campus data centers, two private enterprise data centers, and five cloud data centers*, three of which run a variety of Internet-facing applications while the remaining two predominantly run MapReduce workloads. Some of the data centers we study have been in operation for over 10 years, while others were commissioned much more recently. Our data includes SNMP link statistics for all data centers, fine-grained packet traces from select switches in four of the data centers, and detailed topology for five data centers. By studying different classes of data centers, we are able to shed light on the question of how similar or different they are in terms of their network usage, whether results taken from one class can be applied to the others, and whether different solutions will be needed for designing and managing the data centers' internal networks.

We perform a top-down analysis of the data centers, starting with

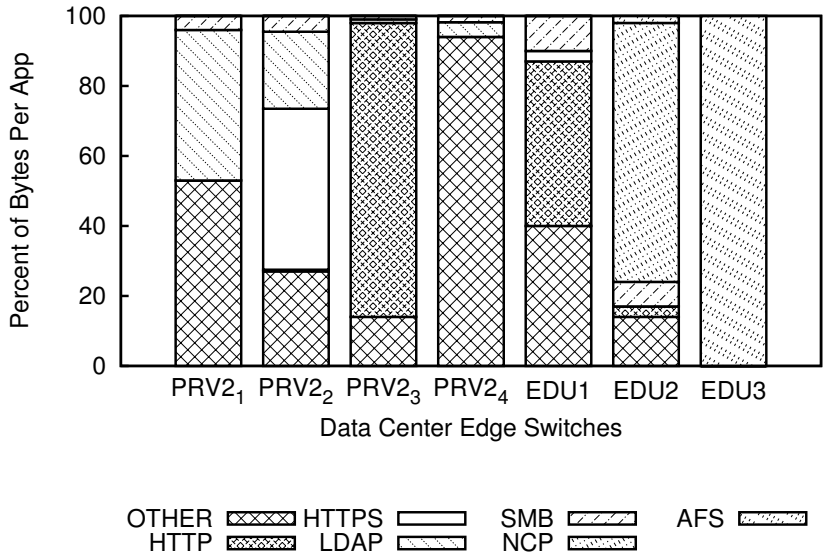
IMC'10, November 1–3, 2010, Melbourne, Australia.

Details of the study

- **Data sources:** 3 campus data centers, 2 private data center, 5 cloud data centers
- **Data types:**
  - Network topologies (incomplete)
  - Packet captures on sample links (incomplete)
  - SNMP data (switch bytes-in and bytes-out)

Some findings

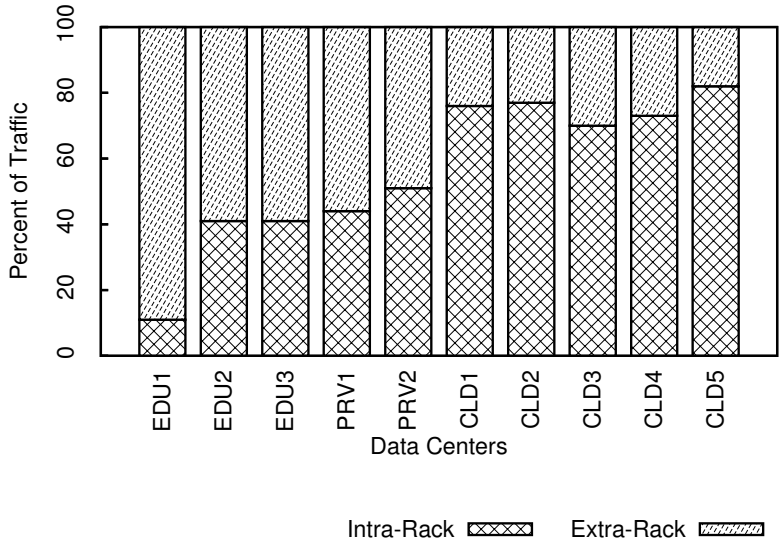
App breakdown for campus & private data centers:



Cloud data centers (not depicted) are either **dedicated to single workload** (mapreduce), or used to **run messaging/web apps/web portals** plus necessary services

Some findings - II

Intra-rack vs extra-rack traffic composition:



Why is it important? Intra-rack traffic enjoys full BW as servers are connected to the same switch; extra-rack traffic BW depends on topology, capacity and traffic pattern

Some findings - III

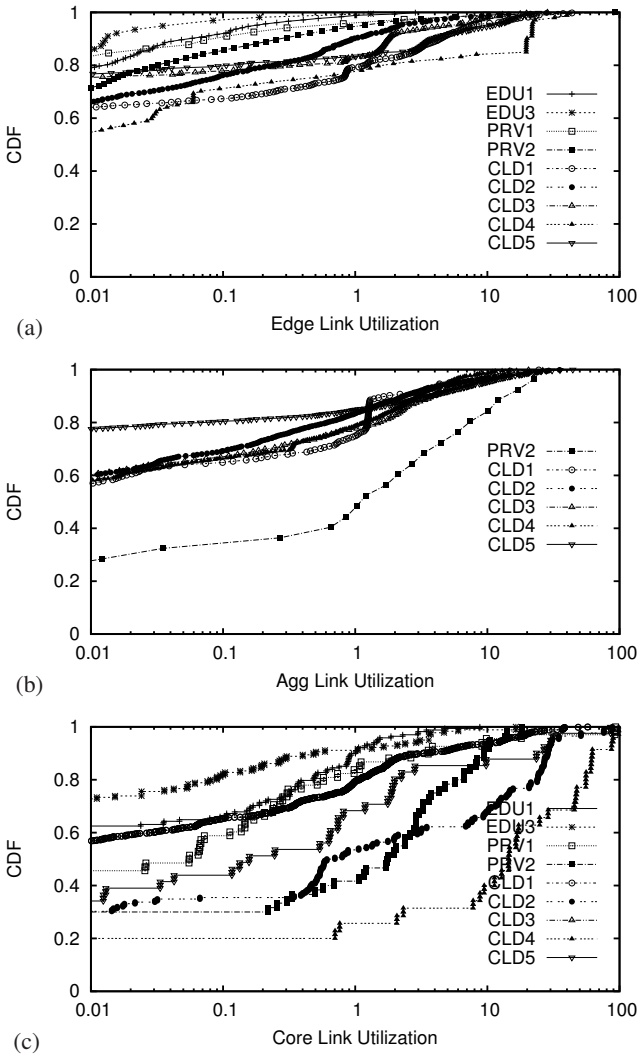


Figure 9: CDF of link utilizations (percentage) in each layer.

- For some data centers, highly utilized links (> 70% capacity) are frequent…
- But in others (e.g. EDU1) there are no highly utilized links!
- More in general: link utilization remains low except in the core…
- …and even in the core, no more than 25% of links is highly-utilized

Data centers @Google

Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network

Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, and Amin Vahdat
Google, Inc.
jupiter-sigcomm@google.com

ABSTRACT

We present our approach for overcoming the cost, operational complexity, and limited scale endemic to datacenter networks a decade ago. Three themes unify the five generations of datacenter networks detailed in this paper. First, multi-stage Clos topologies built from commodity switch silicon can support cost-effective deployment of building-scale networks. Second, much of the general, but complex, decentralized network routing and management protocols supporting arbitrary deployment scenarios were overkill for single-operator, pre-planned datacenter networks. We built a centralized control mechanism based on a global configuration pushed to all datacenter switches. Third, modular hardware design coupled with simple, robust software allowed our design to also support inter-cluster and wide-area networks. Our datacenter networks run at dozens of sites across the planet, scaling in capacity by 100x over ten years to more than 1Pbps of bisection bandwidth.

CCS Concepts

•**Networks** → **Data center networks;**

Keywords

Datacenter Networks; Clos topology; Merchant Silicon; Centralized control and management

1. INTRODUCTION

Datacenter networks are critical to delivering web services, modern storage infrastructure, and are a key enabler for cloud computing. Bandwidth demands in the datacenter are doubling every 12-15 months (Figure 1), even faster than the wide area Internet. A number of recent trends drive this growth. Dataset sizes are continuing to explode with more photo/video content, logs, and the proliferation of Internet-connected sensors. As a result, network-intensive data processing pipelines must operate over ever-larger datasets. Next, Web services can deliver higher quality results by accessing more data on the critical path of individual requests. Finally, constellations of co-resident applications often share substantial data with one another in the same cluster; consider index generation, web search, and serving ads.

Ten years ago, we found the cost and operational complexity associated with traditional datacenter network architectures to be prohibitive. Maximum network scale was limited by the cost and capacity of the highest end switches available at any point in time [24]. These switches were engineering marvels, typically recycled from products targeting wide area deployments. WAN switches were differentiated with hardware support/offload for a range of protocols (e.g., IP multicast) or by pushing the envelope of chip memory (e.g., Internet-scale routing tables, off chip DRAM for deep buffers, etc.). Network control and management protocols targeted autonomous individual switches rather than pre-configured and largely static datacenter fabrics. Most of these features were not useful for datacenters, increased cost, complexity, delayed time to market, and made network management more difficult.

Datacenter switches were also built as complex chassis targeting the highest levels of availability. In a WAN Internet deployment, losing a single switch/router can have substantial impact on applications. Because WAN links are so expensive, it makes sense to invest in high availability. However, more plentiful and cheaper datacenter bandwidth makes it prudent to trade cost for somewhat reduced intermittent capacity. Finally, switches operating in a multi-vendor WAN environment with arbitrary end hosts require support for many protocols to ensure interoperability. In single-operator dat-

SIGCOMM '15 August 17-21, 2015, London, United Kingdom

ACM ISBN 978-1-4503-3542-3/15/08.
DOI: http://dx.doi.org/10.1145/2785956.2787508

Take-aways

- **Clos topologies rule**!
  - “Recent work on alternate network topologies such as HyperX [1], Dcell [17], BCube [16] and Jellyfish [22] deliver more efficient bandwidth for uniform random communication patterns. However, to date, we have found that the benefits of these topologies do not make up for the cabling, management, and routing challenges and complexity.”
- Use of **custom switches** built from cheap off-the-shelf components
- SDN-like **centralized routing management**

More on data center traffic characteristics (2017)

High-Resolution Measurement of Data Center Microbursts*

Qiao Zhang
University of Washington
qiao@cs.washington.edu

Vincent Liu
University of Pennsylvania
liuv@cis.upenn.edu

Hongyi Zeng
Facebook, Inc.
zeng@fb.com

Arvind Krishnamurthy
University of Washington
arvind@cs.washington.edu

ABSTRACT

Data centers house some of the largest, fastest networks in the world. In contrast to and as a result of their speed, these networks operate on very small timescales—a 100 Gbps port processes a single packet in at most 500 ns with end-to-end network latencies of under a millisecond. In this study, we explore the fine-grained behaviors of a large production data center using extremely high-resolution measurements (10s to 100s of microsecond) of rack-level traffic. Our results show that characterizing network events like congestion and synchronized behavior in data centers does indeed require the use of such measurements. In fact, we observe that more than 70% of bursts on the racks we measured are sustained for at most tens of microseconds: a range that is orders of magnitude higher-resolution than most deployed measurement frameworks. Congestion events observed by less granular measurements are likely collections of smaller µbursts. Thus, we find that traffic at the edge is significantly less balanced than other metrics might suggest. Beyond the implications for measurement granularity, we hope these results will inform future data center load balancing and congestion control protocols.

CCS CONCEPTS

• **Networks → Network measurement**; **Data center networks**; *Network performance analysis*; *Network monitoring*; Social media networks;

KEYWORDS

Data center traffic, microbursts

ACM Reference Format:
Qiao Zhang, Vincent Liu, Hongyi Zeng, and Arvind Krishnamurthy. 2017. High-Resolution Measurement of Data Center Microbursts. In *Proceedings of IMC '17, London, United Kingdom, November 1–3, 2017,* 8 pages.
https://doi.org/10.1145/3131365.3131375

*Raw data for the distributions presented in the paper are available at https://github.com/zhangqiaorjc/imc2017-data

IMC '17, November 1–3, 2017, London, United Kingdom

ACM ISBN 978-1-4503-5118-8/17/11…$15.00
https://doi.org/10.1145/3131365.3131375

1 INTRODUCTION

Data center networks are defined by their scale. The largest of today's data centers are massive facilities that house up to hundreds of thousands of servers connected by thousands of network switches. The switches in turn have high (and rapidly growing) capacity, with state-of-the-art models able to process terabits of traffic per second at 100 Gbps per port.

In contrast to the massive aggregate bandwidth of these deployments is the minuscule timescales on which they operate: a 100 Gbps port processes packets in at most 500 ns, and a packet can traverse the entire network in tens to hundreds of microseconds. Unfortunately, much of what we know about data center traffic (and, in fact, most production monitoring systems) are either on the scale of minutes [7] or are heavily sampled [16]. While such measurements can inform us of long-term network behavior and communication patterns, in modern data center networks, coarse-grained measurements fail to provide insight into many important behaviors.

One example: congestion. In most prior work, large cloud network operators have observed that packet discards occur, but are uncorrelated or weakly correlated with observed link utilization, implying that most congestion events are too short-lived to be characterized by existing data sets. Coarse-grained measurements also make it difficult to answer questions about concurrent behavior like how many ports are involved in each congestion event or how effective the network is at load balancing. The design of data center switches, networks, and protocols depend on this type of fine-grained behavior.

Our primary contribution is to provide a high-resolution characterization of a production data center network. To do so, we developed a custom high-resolution counter collection framework on top of the data center operator's in-house switch platform. This framework is able to poll switch statistics at a 10s to 100s of microseconds granularity with minimal impact on regular switch operations.

With the framework, we proceed to perform a data-driven analysis of various counters (including packet counters and buffer utilization statistics) from Top-of-Rack (ToR) switches in multiple clusters running multiple applications. While our measurements are limited to ToR switches, our measurements and prior work [6, 9, 18] indicate that the majority of congestion occurs at that layer. More generally, we do not claim that our results are representative of all modern data center networks—they are merely a slice of one large operator's network, albeit at a heretofore unprecedented granularity. Our main findings include:

- µbursts, periods of high utilization lasting less than 1 ms, exist in production data centers, and in fact, they encompass most congestion events. The p90 burst duration is ≤200 µs.

Take-aways

- Data center traffic proceeds in **micro-bursts** (measurement study on Facebook infrastructure)
  - (**Traffic burst:** period of high link utilization preceded and followed by a period of low utilization)
  - Bursts are **extremely short** – most of them last < 200us

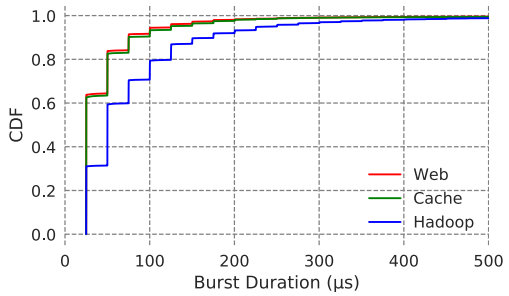


Figure 3: CDF of µburst durations at a 25 µs granularity.

  - Needs **highly optimized measuring code**!

More on traffic micro-bursts

- ECMP-based load-balancing is **too slow to deal with microbursts**
  - Load distribution that appears well-balanced at seconds scale **may not be at microsecond-scale**
  - May require **load-balancing at microflow level** (consider bursts separated by enough time to avoid reordering as the unit of scheduling)
- Most congestion control algorithms adapt to congestion in time ~ RTT
  - **Too slow!** (but no idea of what could be used instead)
- TCP is supposed to prevent bursts! Use ACK to pace sending of new packets
  - But **offload of segmentation and reassembly to NICs breaks it**!

Part 3 – TCP in data centers

TCP in data centers

- **99.91% traffic in data centers is TCP** (Alizadeh et al, SIGCOMM 2010)
- Data center switches tend to have **limited buffer space**
  - **Reduces cost** (buffer shared between ports require expensive multiported memory)
  - **Lower latency** (shorter queues, shorter delays)
  - …but **greater sensitivity to congestion**! (smaller buffers, more drops)
- Also: shared buffers mean that large, throughput sensitive flows can negatively affect short, delay-sensitive flows

TCP in data centers - II

- **Shallow buffers**, **low latencies** and **high throughput** create **problems**
  - TCP **reactive approach** to congestions means queue fill quickly and packets are dropped
- **Larger buffers** alleviate the issue, but **increase delay**
- **Proactively reacting** to congestion (i.e., before buffers are full) can work, but **reduces throughput if too aggressive**
- Ideal goal: **low delays** (= low queue utilization), **low packet drops**, and **high throughput**

More on TCP patologies

- **TCP incast:** multiple senders communicates w/ same destination in a synchronized fashion
  - **Several data center workloads can originate it**: web search queries, mapreduce,…

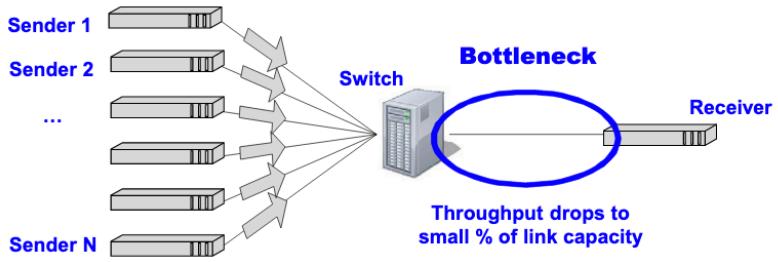


Figure 1: Simple setup to observe incast

From Chen et al., ;login:, 2006

Solutions?

- Introduce **jitter** in timing of requests/responses
  - By **adding some random delay to the establishment of connections**, synchronization between senders can be avoided
  - Works (somewhat) but hard to tune and reason about
  - Adds **delay** to communication! (the more delay, the better it works)
- More principled solutions?

Solutions? /2

- Google:
  - **QoS** (prioritize flows based on application)
  - **Bound TCP window size** (prevents senders to dump large amount of data in the network suddenly)
  - Perform **good network engineering** (measure bandwidth utilization, modify buffer allocation strategy to readily absorb temporary buffer growth, deploy ECMP-based load-balancing)
  - Use **ECN**
    - **Explicit congestion notification:** allow routers along the path to mark packets when congestion is arising (can be done at IP or TCP level)

More on TCP ECN

Data Center TCP (DCTCP)

Mohammad Alizadeh[‡†], Albert Greenberg[†], David A. Maltz[†], Jitendra Padhye[†],
Parveen Patel[†], Balaji Prabhakar[‡], Sudipta Sengupta[†], Murari Sridharan[†]

[†]Microsoft Research [‡]Stanford University
{albert, dmaltz, padhye, parveenp, sudipta, muraris}@microsoft.com
{alizade, balaji}@stanford.edu

ABSTRACT

Cloud data centers host diverse applications, mixing workloads that require small predictable latency with others requiring large sustained throughput. In this environment, today's state-of-the-art TCP protocol falls short. We present measurements of a 6000 server production cluster and reveal impairments that lead to high application latencies, rooted in TCP's demands on the limited buffer space available in data center switches. For example, bandwidth hungry "background" flows build up queues at the switches, and thus impact the performance of latency sensitive "foreground" traffic.

To address these problems, we propose DCTCP, a TCP-like protocol for data center networks. DCTCP leverages Explicit Congestion Notification (ECN) in the network to provide multi-bit feedback to the end hosts. We evaluate DCTCP at 1 and 10Gbps speeds using commodity, shallow buffered switches. We find DCTCP delivers the same or better throughput than TCP, while using 90% less buffer space. Unlike TCP, DCTCP also provides high burst tolerance and low latency for short flows. In handling workloads derived from operational measurements, we found DCTCP enables the applications to handle 10X the current background traffic, without impacting foreground traffic. Further, a 10X increase in foreground traffic does not cause any timeouts, thus largely eliminating incast problems.

Categories and Subject Descriptors: C.2.2 [Computer-Communication Networks]: Network Protocols
General Terms: Measurement, Performance
Keywords: Data center network, ECN, TCP

1. INTRODUCTION

In recent years, data centers have transformed computing, with large scale consolidation of enterprise IT into data center hubs, and with the emergence of cloud computing service providers like Amazon, Microsoft and Google. A consistent theme in data center design has been to build highly available, highly performant computing and storage infrastructure using low cost, commodity components [16]. A corresponding trend has also emerged in data center networks. In particular, low-cost switches are common at the top of the rack, providing up to 48 ports at 1Gbps, at a price point under $2000 — roughly the price of one data center server. Several recent research proposals envision creating economical, easy-to-manage data centers using novel architectures built atop these commodity switches [2, 12, 15].

Is this vision realistic? The answer depends in large part on how well the commodity switches handle the traffic of real data center applications. In this paper, we focus on soft real-time applications, supporting web search, retail, advertising, and recommendation systems that have driven much data center construction. These applications generate a diverse mix of short and long flows, and require three things from the data center network: low latency for short flows, high burst tolerance, and high utilization for long flows.

The first two requirements stem from the *Partition/Aggregate* (described in §2.1) workflow pattern that many of these applications use. The near real-time deadlines for end results translate into latency targets for the individual tasks in the workflow. These targets vary from ∼10ms to ∼100ms, and tasks not completed before their deadline are cancelled, affecting the final result. Thus, *application requirements for low latency directly impact the quality of the result returned and thus revenue.* Reducing network latency allows application developers to invest more cycles in the algorithms that improve relevance and end user experience.

The third requirement, high utilization for large flows, stems from the need to continuously update internal data structures of these applications, as the freshness of the data also affects the quality of the results. Thus, high throughput for these long flows is as essential as low latency and burst tolerance.

In this paper, we make two major contributions. First, we measure and analyze production traffic (>150TB of compressed data), collected over the course of a month from ∼6000 servers (§2), extracting application patterns and needs (in particular, low latency needs), from data centers whose network is comprised of commodity switches. Impairments that hurt performance are identified, and linked to properties of the traffic and the switches.

Second, we propose Data Center TCP (DCTCP), which addresses these impairments to meet the needs of applications (§3). DCTCP uses Explicit Congestion Notification (ECN), a feature already available in modern commodity switches. We evaluate DCTCP at 1 and 10Gbps speeds on ECN-capable commodity switches (§4). We find DCTCP successfully supports 10X increases in application foreground and background traffic in our benchmark studies.

The measurements reveal that 99.91% of traffic in our data center is TCP traffic. The traffic consists of query traffic (2KB to 20KB in size), delay sensitive short messages (100KB to 1MB), and throughput sensitive long flows (1MB to 100MB). The query traffic experiences the incast impairment, discussed in [32, 13] in the context of storage networks. However, the data also reveal new impairments unrelated to incast. Query and delay-sensitive short messages experience long latencies due to long flows consuming

SIGCOMM'10, August 30–September 3, 2010, New Delhi, India.

Some insights from the paper

- Uses **TCP ECN** (Explicit Congestion Notification) option
- Switches **mark packets** when buffer occupancy crosses threshold; receiver **mirrors marks** to sender
- However, **standard ECN** (which halves the congestion window on congestion notification) is found to be **too aggressive**
- **Paper proposal:** upon reception of marked packets, sender cuts congestion window using
  $cwnd \leftarrow cwnd \times (1 - \alpha/2)$
  where α is the (exponentially weighted) moving **average of the fraction of marked packets**

Advantages

- **Early reaction to congestion** - helps reducing queue buildup and therefore queuing delays
  - Also, **more buffer room** to deal with small transient bursts
- By preventing excessively long queues on congested ports**, DTCP reduces buffer pressure**, which means that **congested ports do not exhaust the buffer for everyone else**
- By gently reducing *cwnd* size, DTCP still **ensure high throughput** for connections that require it