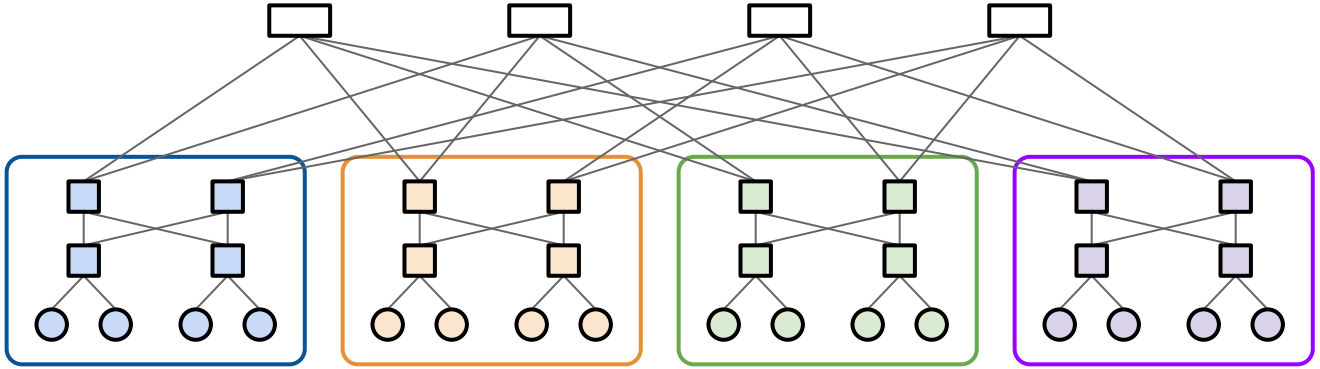


Lecture 19 (Datacenters 1)

# Datacenter Topology

CS 168, Fall 2026 @ UC Berkeley

# What is a Datacenter?

Lecture 19, CS 168, Fall 2026

**Designing Datacenters**

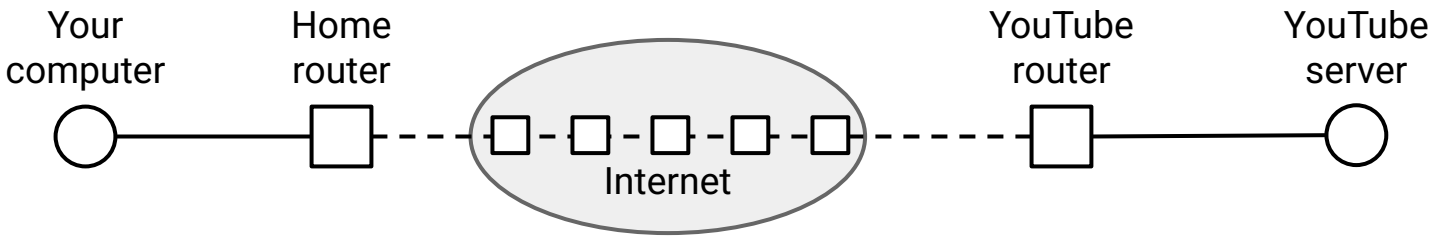
- **What is a Datacenter?**
- Inside a Real Datacenter
- Bisection Bandwidth
- Topologies (Clos Networks)

Congestion Control in Datacenters

# What is a Datacenter?

Our model of the Internet so far: Your computer exchanges packets with a server.

Is YouTube really a single machine serving videos to the entire world? No.

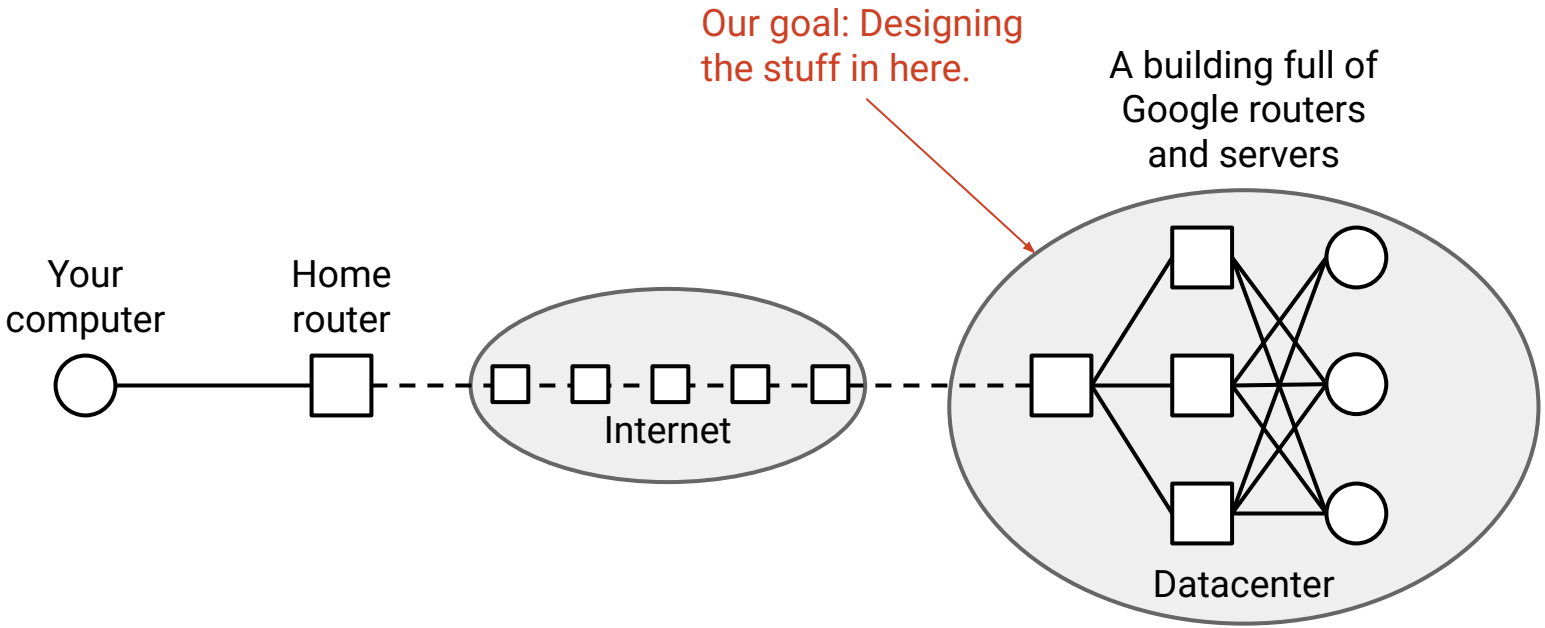


# What is a Datacenter?

In reality: Applications (e.g. YouTube, Facebook) are hosted in **datacenters**.

Datacenter: A building of inter-connected machines, hosting applications.

Our goal: How do we build networks to connect machines in the datacenter?



# General-Purpose Networks vs Datacenters

General-purpose networks.

- Example: UC Berkeley's network.
  - Hosts: Students' computers, servers in research labs.
  - Network infrastructure: Routers and links on campus.

Datacenter networks.

- Example: Google's datacenter.
  - Hosts: Servers running Google search, YouTube, Google Maps, etc.
  - Network infrastructure: ??? ← Our goal: Designing this.

Unlike general-purpose networks we've seen so far, datacenter networks have unique constraints and specialized solutions.

## Why are Datacenters Different?

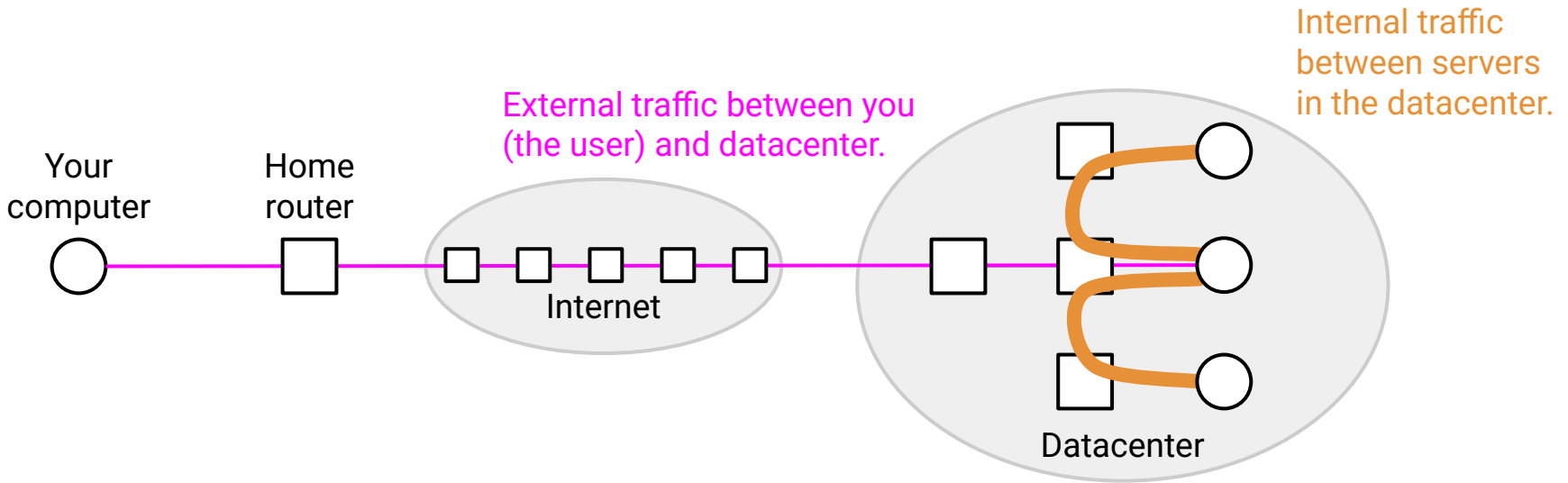
What makes datacenter network infrastructure different?

- Exists in a **single physical location** (e.g. a building).
- **Centralized control**: Operated by a single organization.
  - The operator has more control over the network and hosts
- **Homogenous**: Every server and switch can be built and operated identically.
  - Made possible because of centralized control.
- **High performance requirements**: Need extremely high bandwidth, reliability, etc.
  - We'll focus on modern *hyperscale* datacenters (e.g. Google, Facebook, Amazon), but the concepts can scale down.

# Datacenter Traffic Patterns

High performance is critical because the datacenter has lots of *internal* traffic.

- Suppose you load the Facebook home page.
- The one server you contact probably doesn't have all of Facebook's data.
- That server needs to exchange traffic with other servers to answer your request.
  - Example: Fetch ads, photos, posts, etc. from other servers with that data.



# Datacenter Traffic Patterns

High performance is critical because the datacenter has lots of *internal* traffic.

- A single request from the user might trigger lots of backend requests between datacenter servers.
  - Example: Loading a Facebook page.
  - Example: Running a Google search query.
- There's significantly more internal traffic (between machines) than external traffic (machine to user).
- Sometimes, there's no user-facing traffic at all.
  - Example: A server backing up its data on another server.

**Scaling Memcache at Facebook** Link

*Rajesh Nishtala et. al.* *USENIX NSDI, 2013*

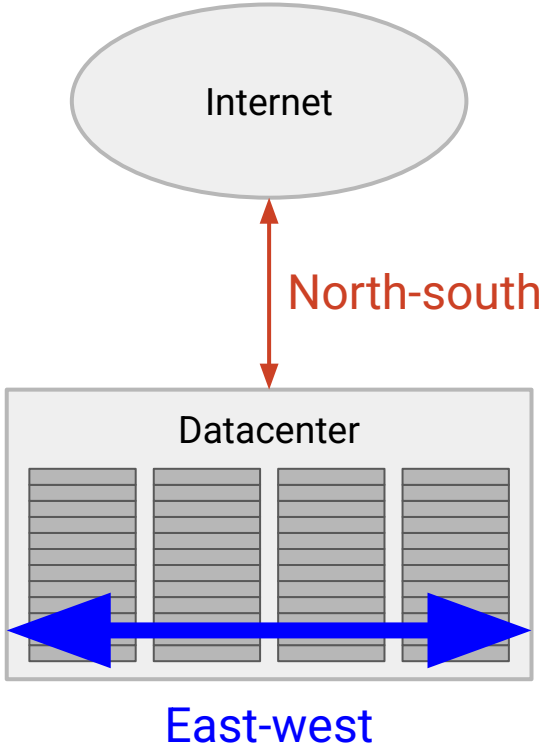
Loading one popular page requires 521 internal loads on average.

# Datacenter Traffic Patterns

Terminology:

- **East-west** traffic is internal, between machines.
- **North-south** traffic is external, from the datacenter to elsewhere.

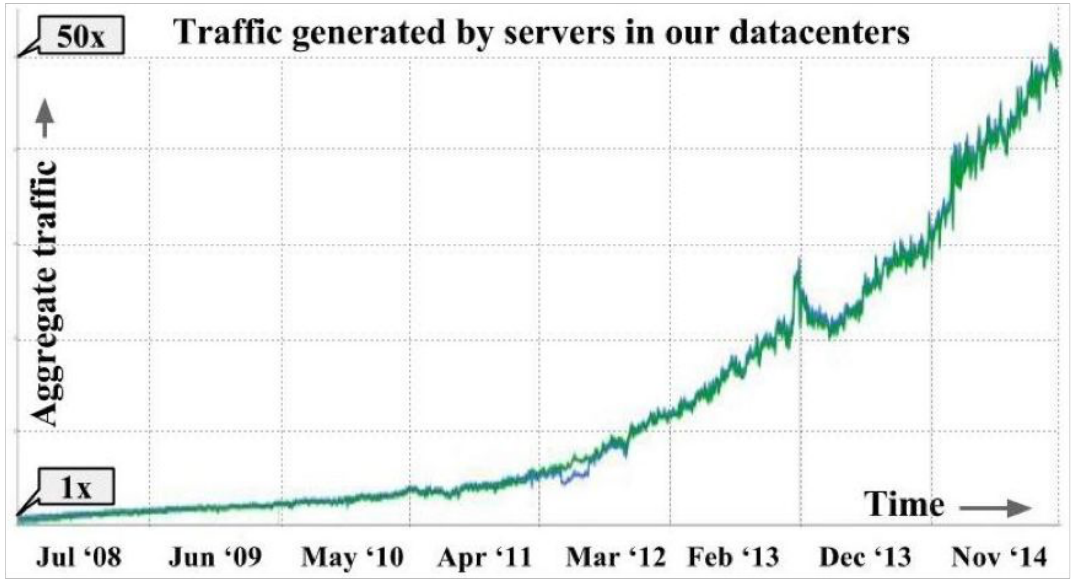
East-west traffic is several orders of magnitude larger than north-south traffic.



# Datacenter Traffic Patterns

Datacenter traffic demand has increased significantly.

- 50x increase between 2008 and 2014!




"Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network", Arjun Singh et al. @ Google, ACM SIGCOMM'15

# Inside a Real Datacenter

Lecture 19, CS 168, Fall 2026

**Designing Datacenters**

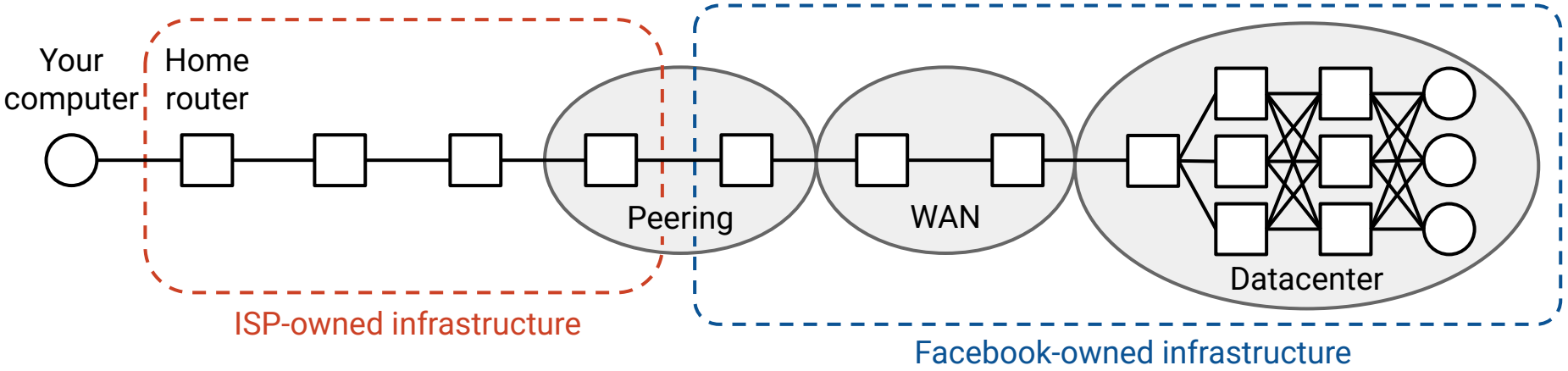
- What is a Datacenter?
- **Inside a Real Datacenter**
- Bisection Bandwidth
- Topologies (Clos Networks)

Congestion Control in Datacenters

# Accessing an Application in a Datacenter

When you send a packet to the service (e.g. Facebook):

- Your ISP forwards the packet through ISP-operated routers and links.
- Eventually, the packet reaches a peering location.
  - The ISP hands the packet off to a Facebook-operated router.
- Facebook forwards the packet through its own infrastructure to a datacenter.

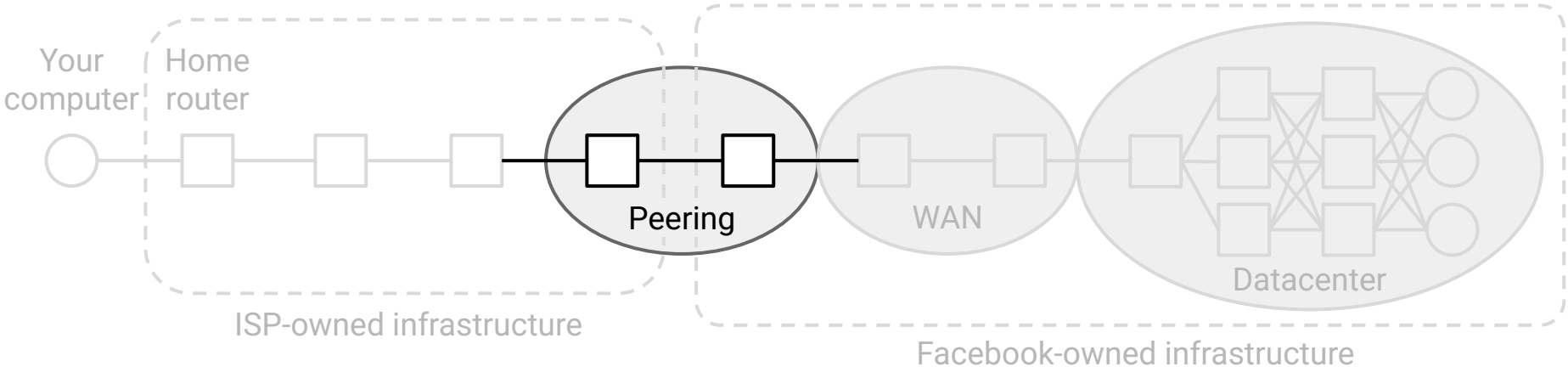


# Internet Infrastructure: Peering Locations

**Peering locations** host network interconnection infrastructure.

- Needs to be near other networks.
- Often found in cities.
- Applications and ISPs rent space here to install their routers.



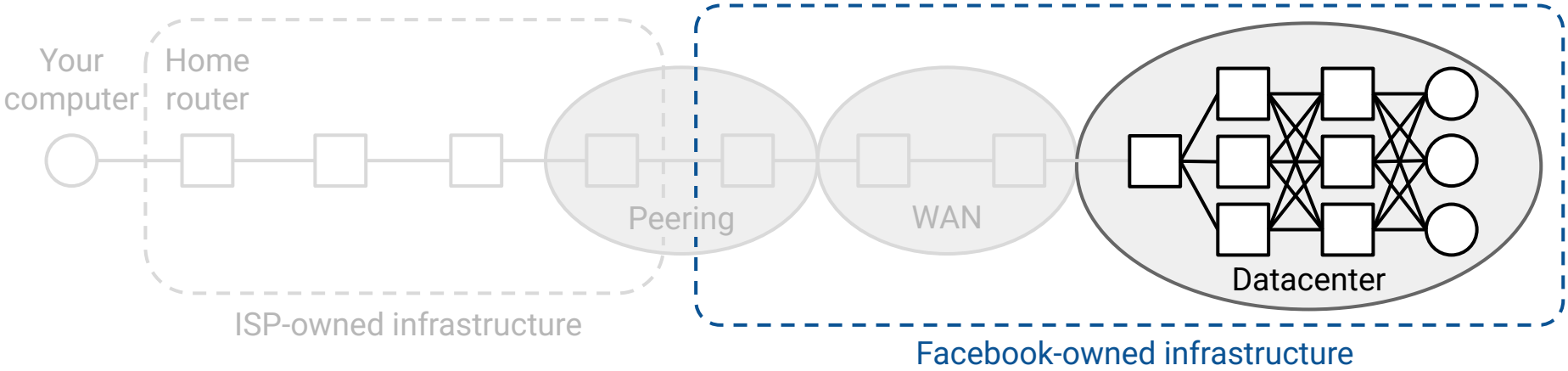


# Internet Infrastructure: Datacenters

**Datacenters** host the servers and application infrastructure.

- Doesn't need to be near other networks.
- Often found in less-populated areas.
- Constraints: Power, cooling, physical space.





# Internet Infrastructure: The Big Picture

**Datacenters** host the servers and application infrastructure.

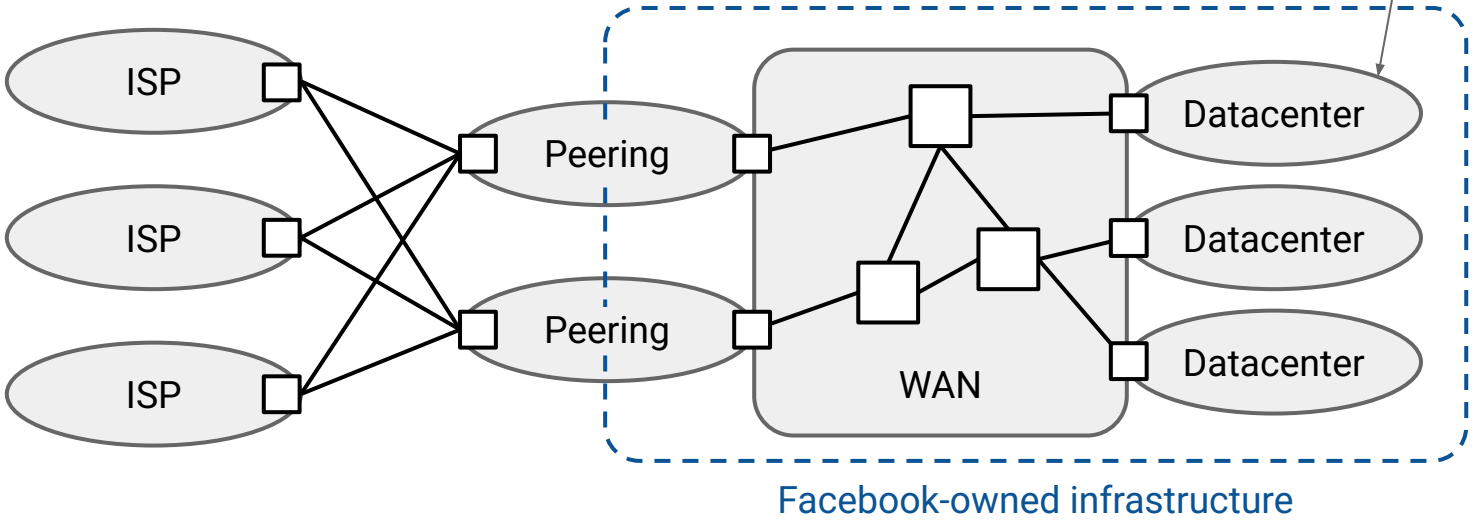
**Peering locations** host network interconnection infrastructure.

- Facebook and the ISPs all install routers here.

Facebook's **wide area network** (WAN) connects its locations together.

- The peering locations and datacenters could all be in different cities.

Our special datacenter networks connect servers inside here.



# Internet Infrastructure: Datacenters

Datacenter locations are often chosen based on constraints:

- Cooling: Near rivers.
- Power: Near a power station.
- Physical space: In less populated areas.



Infrastructure for cooling inside a Google datacenter.

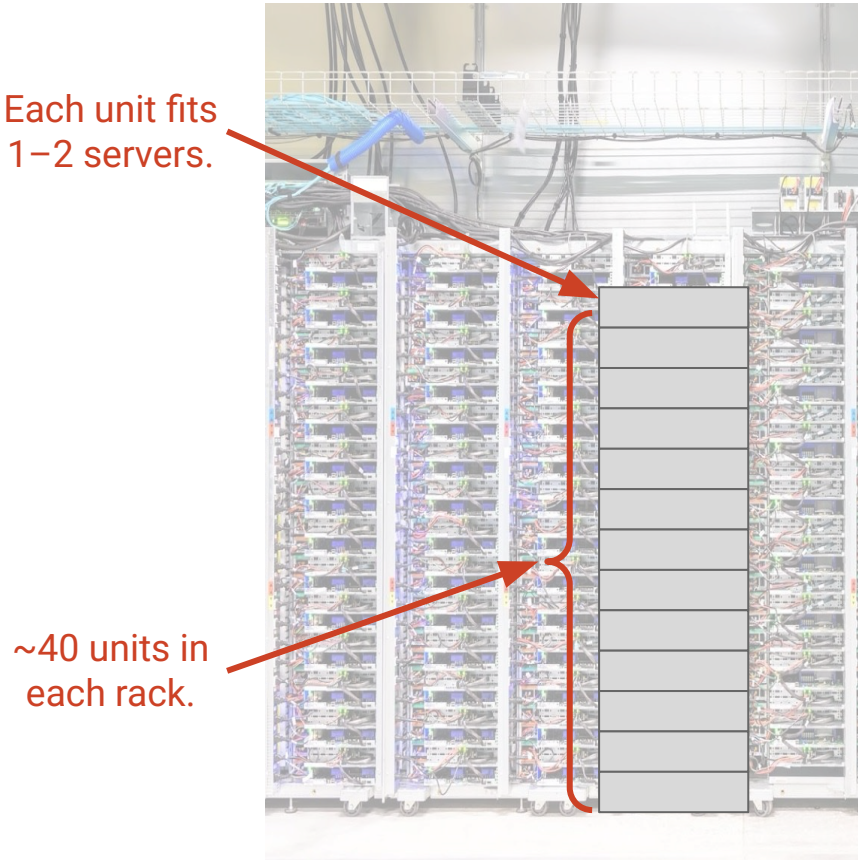


Ultimately, a datacenter is just a building with a ton of servers. Our job is to build the network that connects the servers.

# Inside a Datacenter: Racks

Inside a datacenter, servers are organized in physical **racks**.

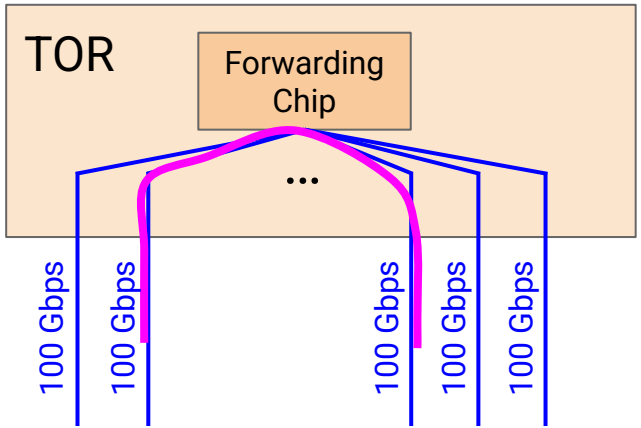
- Each rack has ~40 units (slots).
- Each unit can fit 1–2 servers.

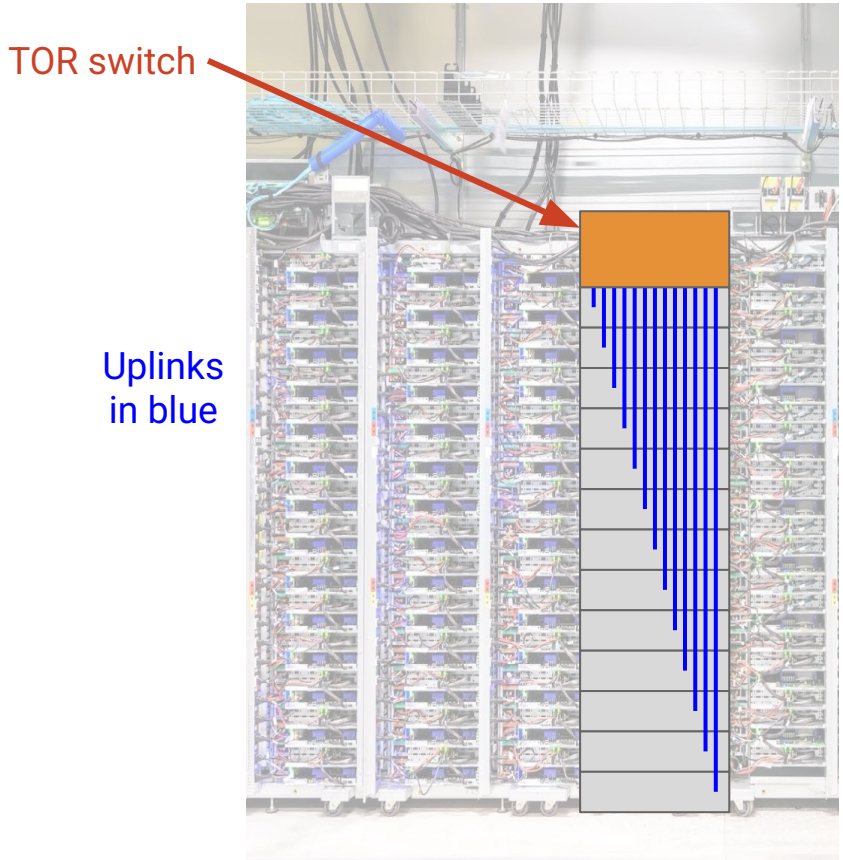


# Inside a Datacenter: Top-of-Rack Switches

To connect all the servers in the same rack:

- Each rack has a **top-of-rack (TOR) switch**.
- Every server in the rack has an **access link (uplink)** connecting to that switch.
- Each server uplink is ~100 Gbps.



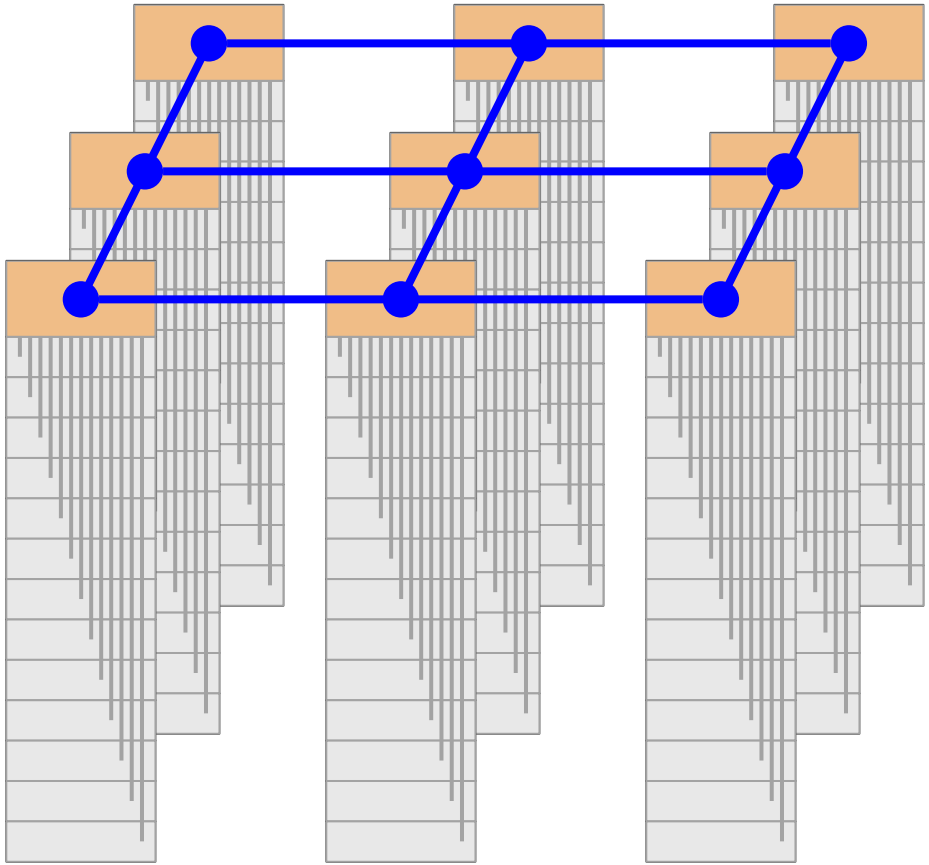



# Inside a Datacenter: Connecting Racks?

We built a rack of 40–80 servers, and connected all the servers in the rack.

Each datacenter has many racks.

- How do we connect the racks together?
- But first...how do we measure if our chosen topology is good?



# Bisection Bandwidth

Lecture 19, CS 168, Fall 2026

**Designing Datacenters**

- What is a Datacenter?
- Inside a Real Datacenter
- **Bisection Bandwidth**
- Topologies (Clos Networks)

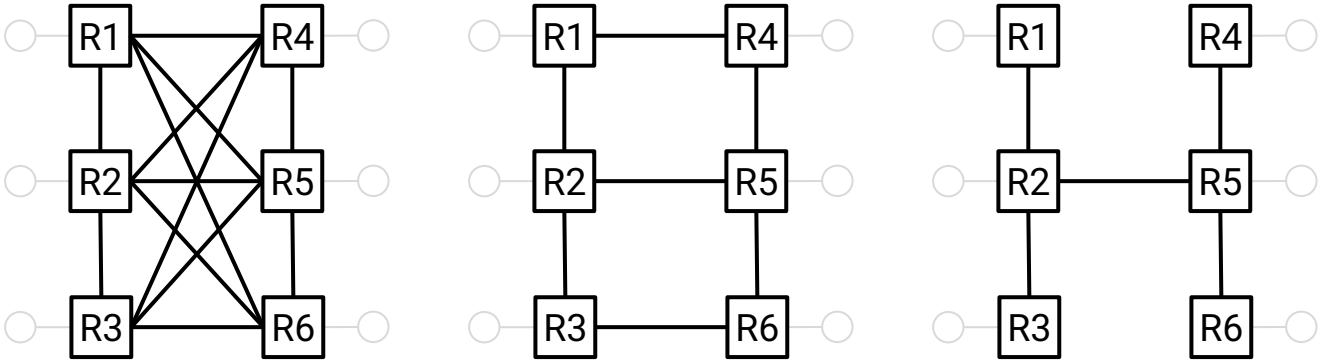
Congestion Control in Datacenters

# Measuring Network Connectivity

Before we think about connecting racks, we need a way to measure connectedness.

- All 3 topologies below are fully-connected.
- 1 is "more connected" than 2, and 2 is "more connected" than 3.

**Bisection bandwidth** is a way to formally measure how connected a network is.



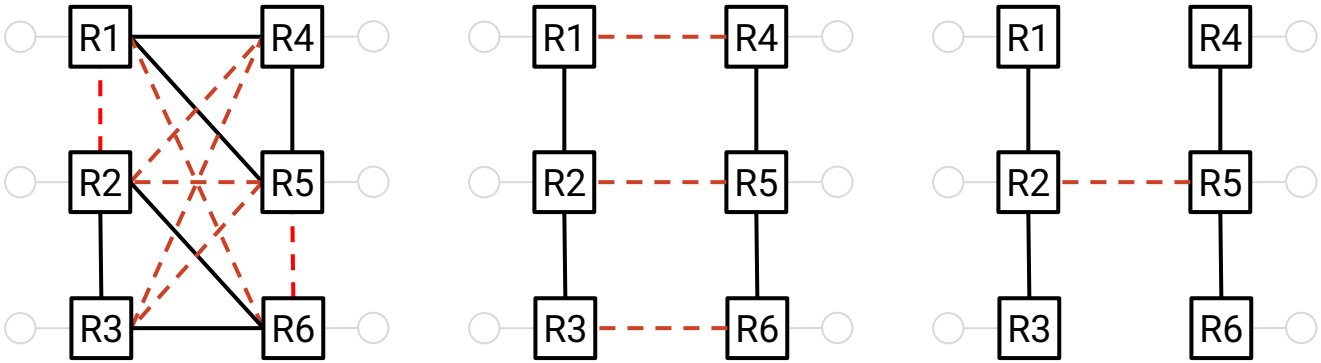

Topology 1 Topology 2 Topology 3

# Bisection Bandwidth

To compute bisection bandwidth:

- Delete links until the network is partitioned into two halves.
  - Delete the minimum number of links necessary (don't just delete everything).
- Add up the bandwidths on all the deleted links.

Intuition: If the network is more connected, we have to delete more links to split it.




Assume each link is 1 Gbps.

7 Gbps

3 Gbps

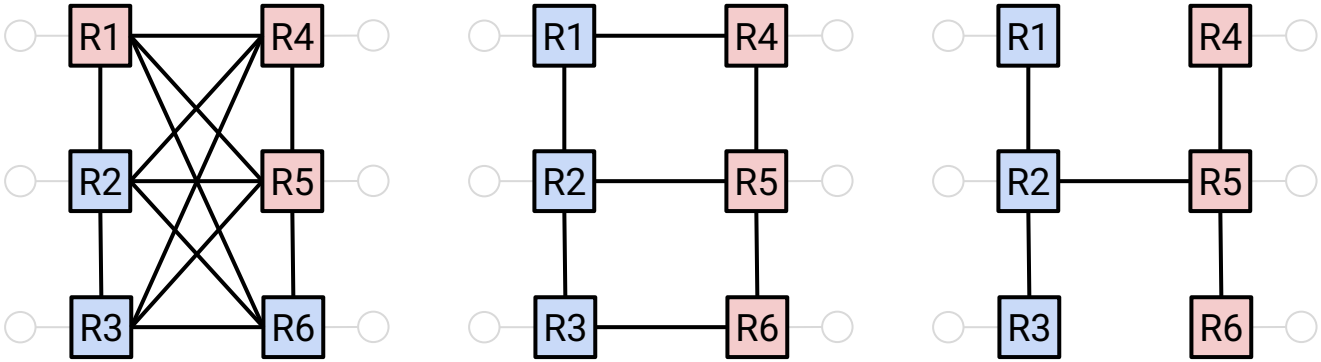
1 Gbps

# Bisection Bandwidth

Another equivalent definition of bisection bandwidth:

- Partition the network into two halves.
- All hosts in one half want to simultaneously send data to the other half.
- In the worst-case, what is the maximum bandwidth the nodes can collectively send at?
  - Worst-case → worst-case partition into two equal halves

Intuition: This is the bottleneck bandwidth of the network.




Assume each link is 1 Gbps.

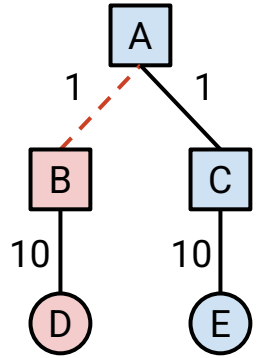
# Full Bisection Bandwidth1

**Full bisection bandwidth**: All hosts in one partition can send at full rate

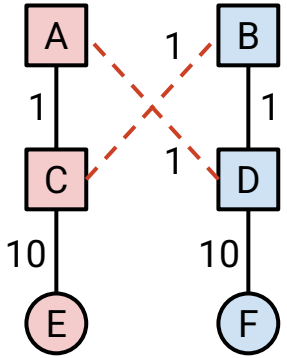
- If *N* nodes can each send at *R*, full bisection bandwidth = *N*/2 × R.

**Oversubscription** is a measure of how overloaded the network is.

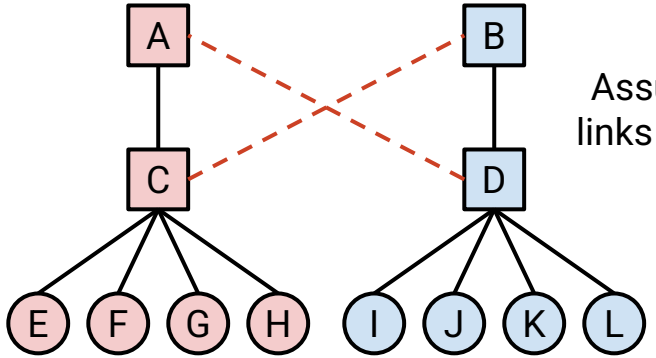
- Ratio of full bisection bandwidth, to actual bisection bandwidth.
- Intuition: This tells us how far from the full bisection bandwidth we are.




Full: 10 Gbps
Bisection: 1 Gbps
Over: 10x




Full: 10 Gbps
Bisection: 2 Gbps
Over: 5x




Full: 4 Gbps
Bisection: 2 Gbps
Over: 2x

Assume all links 1 Gbps.

# Datacenter Topologies (Clos Networks)

Lecture 19, CS 168, Fall 2026

**Designing Datacenters**

- What is a Datacenter?
- Inside a Real Datacenter
- Bisection Bandwidth
- **Topologies (Clos Networks)**

Congestion Control in Datacenters

# Maximizing Bisection Bandwidth

Back to our original problem:

- We were looking for a way to connect racks to each other.
- As we've seen, bisection bandwidth is a function of network topology.
- In the datacenter, we can choose the topology relatively easily.
  - Remember: A single operator controls the network.
  - They can build whatever cables they want.
- What topology do we build to maximize connectivity (bisection bandwidth)?

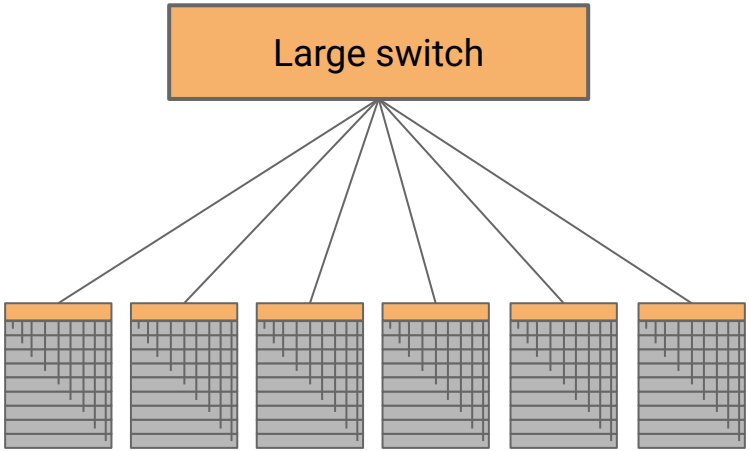
# Topology #1: Big Switch

Problem 1: High switching speed needed.

- In a datacenter, 100,000 servers × 100 Gbps = 10 petabits ($10^{16}$) per second!

Problem 2: Large **radix**. The switch needs a large number of (physical) ports.

- 1 port per rack. In a modern datacenter, ~2500 racks.



# Topology #1: Big Switch

A single big switch scales poorly.

- Impossible to build a switch with the needed capacity.
- Even if we could build this switch, it would be way too expensive.
- If the switch goes down, everything stops working.

Google tried to build this in ~2004.
It didn't go so well.

*via Urs Hölzle (Google) on LinkedIn* Link

But what we needed was a 10,000 port switch that cost $100/port. So, almost exactly 20 years ago, we sent this 5-page RFP to four different switch vendors (IIRC: Cisco, Force10, HP, and Quanta) and tried to interest them in building such a switch. They politely declined because "nobody is asking for such a product except you", and they anticipated margins to be low.

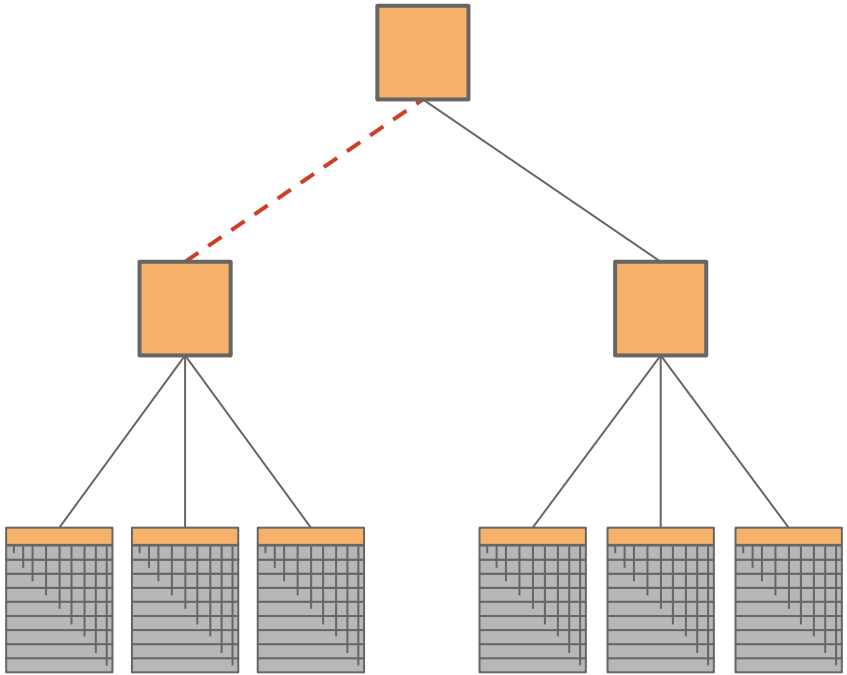
# Topology #2: Tree

A tree topology solves our problems:

- Problem 1: Reduced bandwidth.
- Problem 2: Reduced radix.

Problem: Low bisection bandwidth.

- Bottleneck is 100 Gbps.
- Full bisection bandwidth is 300 Gbps.
- Network is 3x oversubscribed.



100 Gbps links

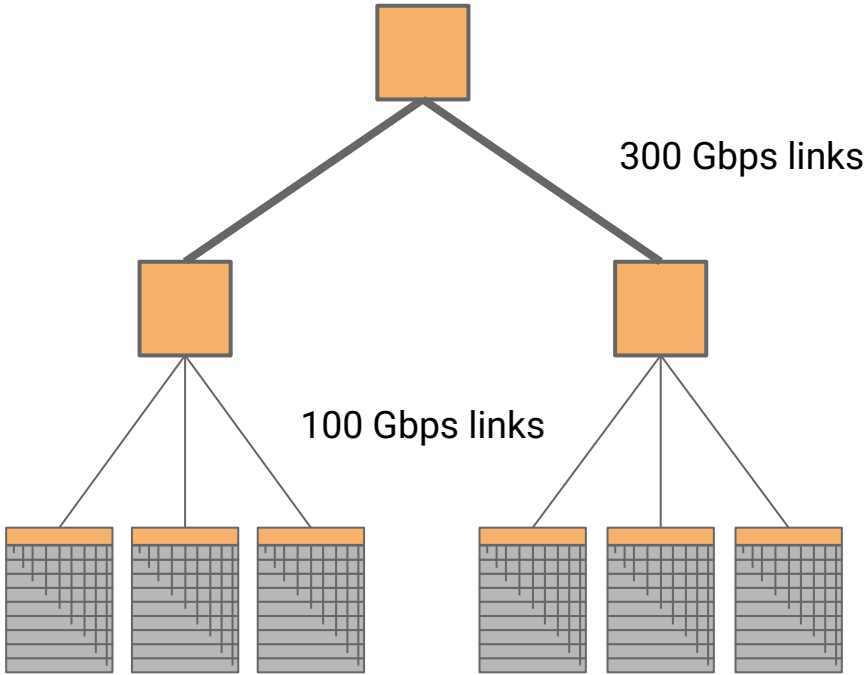
# Topology #3: Fat Tree

Increase link bandwidth at the top layer.

- Problem 1 returns: Top switch needs high link speed and switching capacity.
- Problem 2 solved: Reduced radix.

This can be used.

- But the top switch will be expensive and scale poorly.



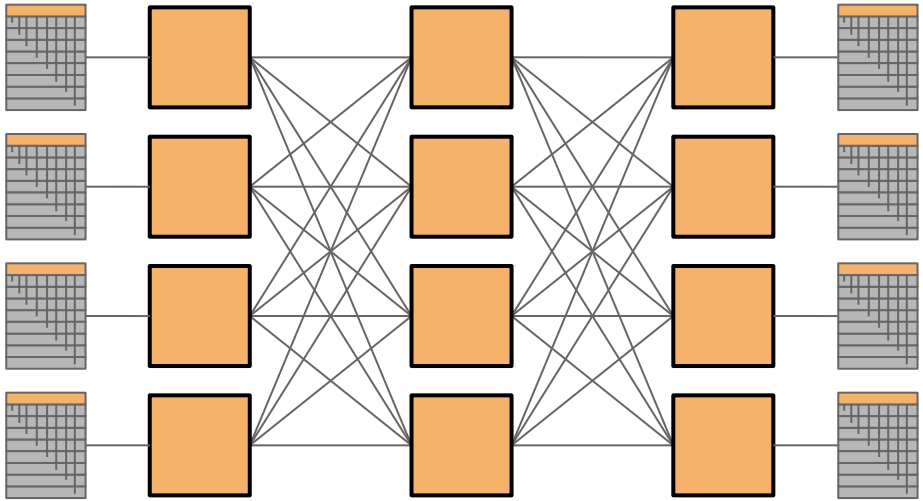
# Inspiration: Clos Networks

Build a big switch from lots of smaller, cheaper switches

- No ultra high radix switches
- No ultra high speed links

Example of a “scale out” approach

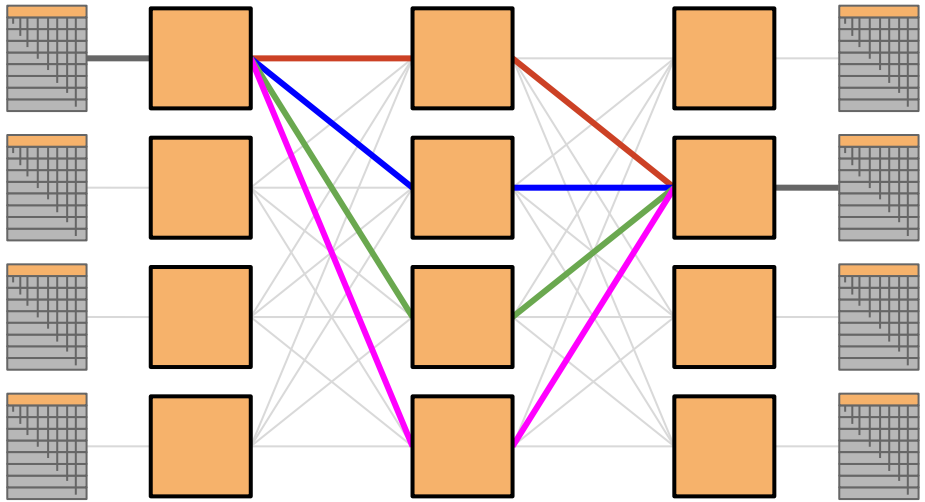
- Instead of “scale up”



# Inspiration: Clos Networks

High performance comes from creating many *paths*.

Ideally, any two hosts talking can use a dedicated path.



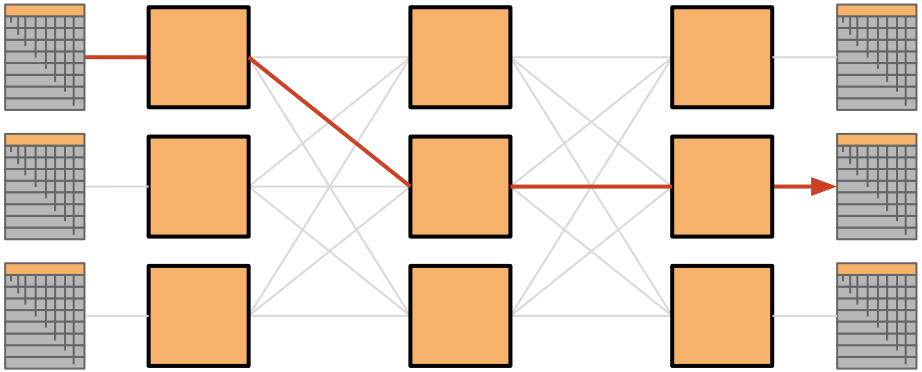
# Folded Clos Networks

So far, we've drawn sender racks on the left, and recipient racks on the right.

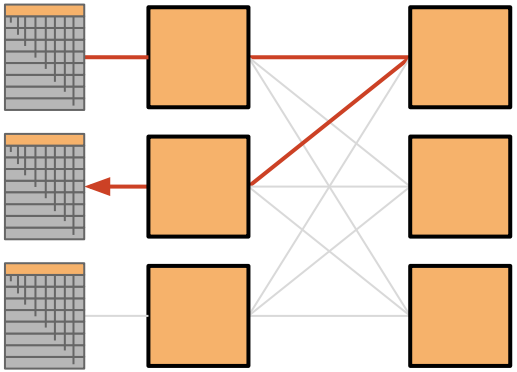
In real networks, each rack could be both sender and recipient.

In a **folded Clos network**, traffic flows into the network, and then back out.

- Links are bidirectional.



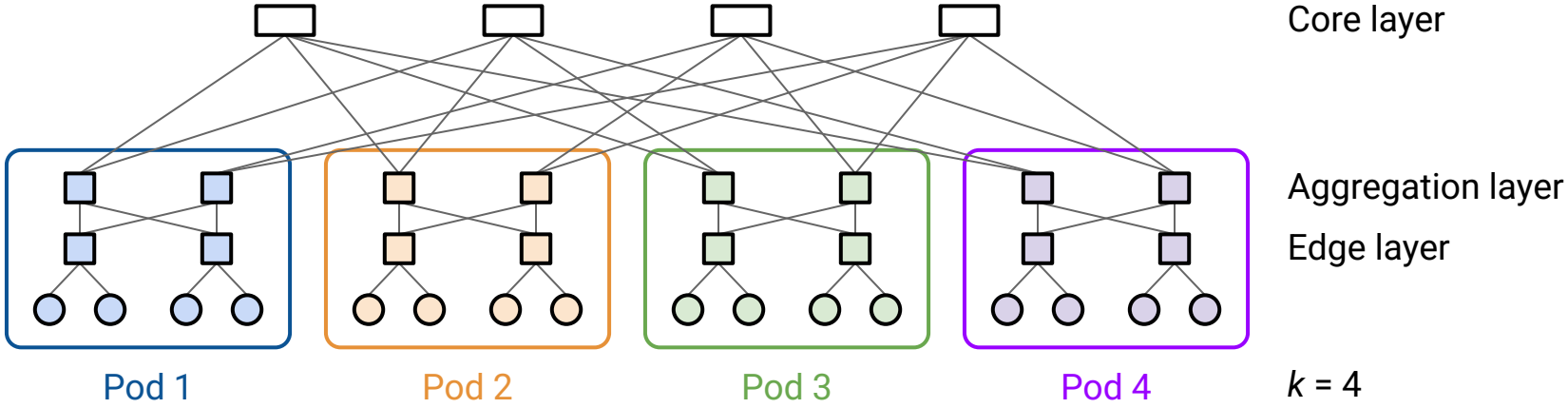
Clos Network



Folded Clos Network

# Topology #4: Fat Tree Clos Network

Challenge: providing high bisection bandwidth between ***N*** servers (each sending at rate R) using ***k***-port switches (with link speeds R) where ***k << N***

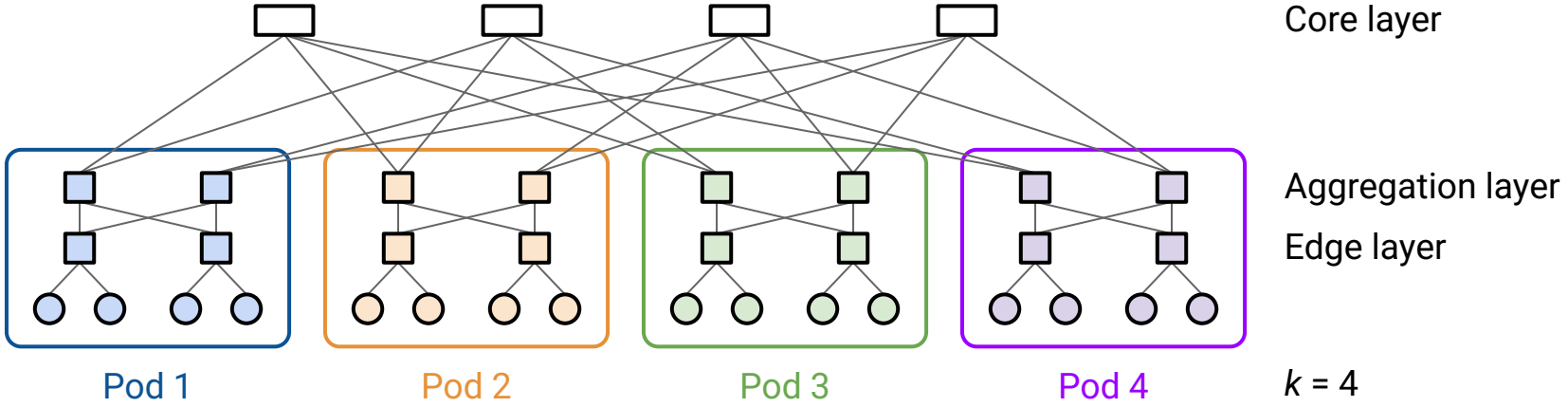


# Topology #4: Fat Tree Clos Network

Idea: Combine the tree and the Clos network.

**A Scalable, Commodity Data Center Network Architecture** Link

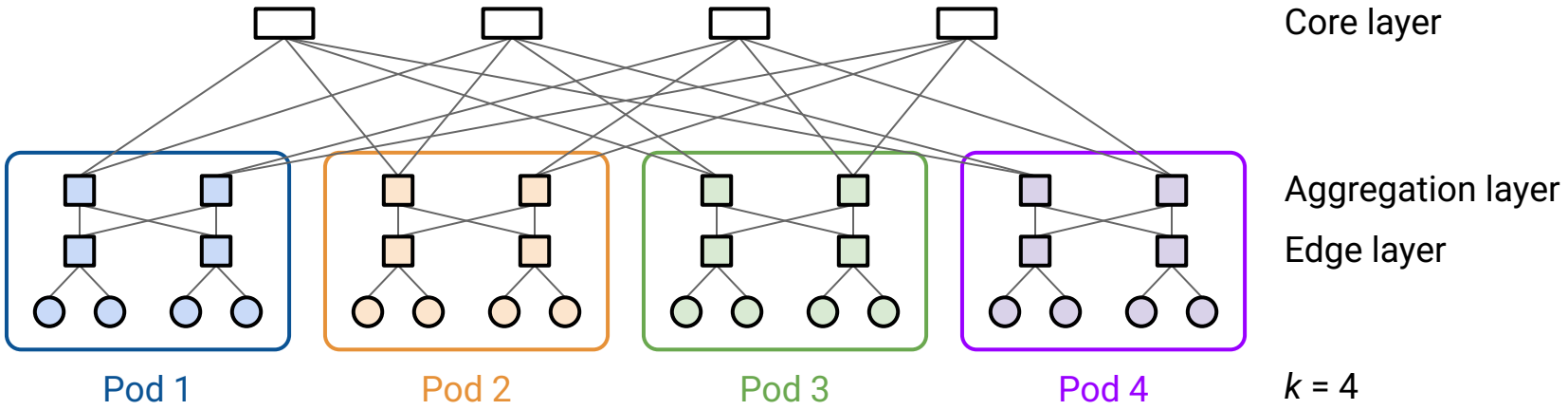
*Mohammad Al-Fares, Alexander Loukissas, Amin Vahdat* *SIGCOMM 2008*



# Topology #4: Fat Tree Clos Network

A *k*-ary fat tree has *k* pods.

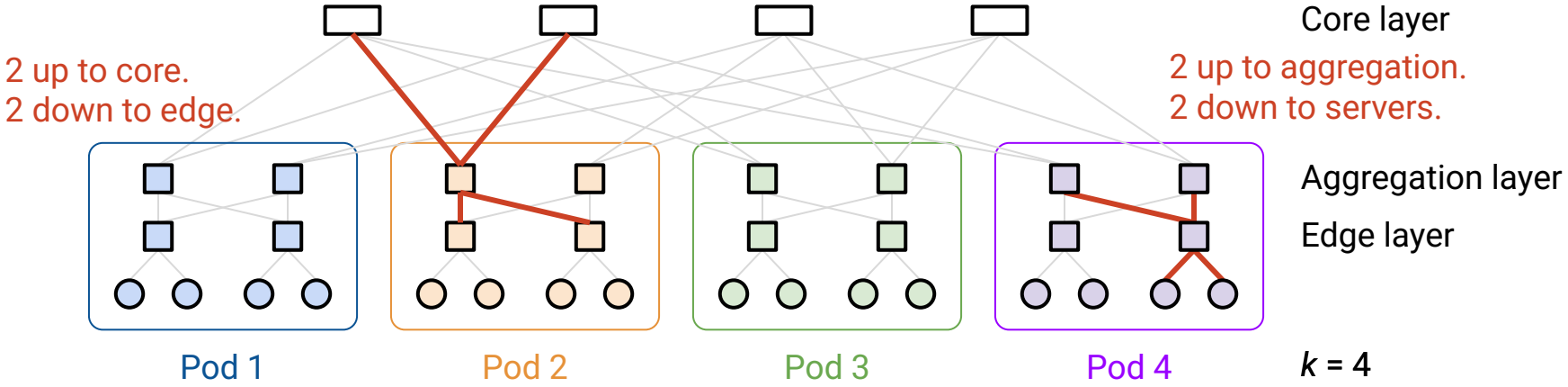
- Each pod has *k* switches.
  - *k*/2 switches in the upper aggregation layer.
  - *k*/2 switches in the lower edge layer.



# Topology #4: Fat Tree Clos Network

Each switch has *k* links. Half connect up, and half connect down.

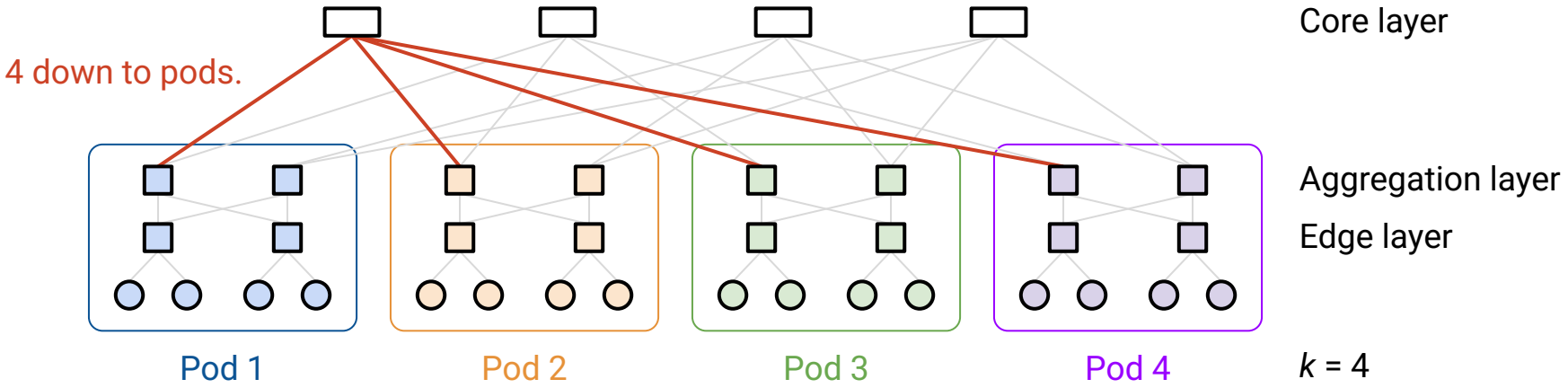
- Edge layer:
  - *k*/2 links connect up to all aggregation switches in the same pod.
  - *k*/2 links connect down to servers.
- Aggregation layer:
  - *k*/2 links connect up to core layer.
  - *k*/2 links connect down to all edge switches in the same pod.



# Topology #4: Fat Tree Clos Network

A *k*-ary fat tree has:

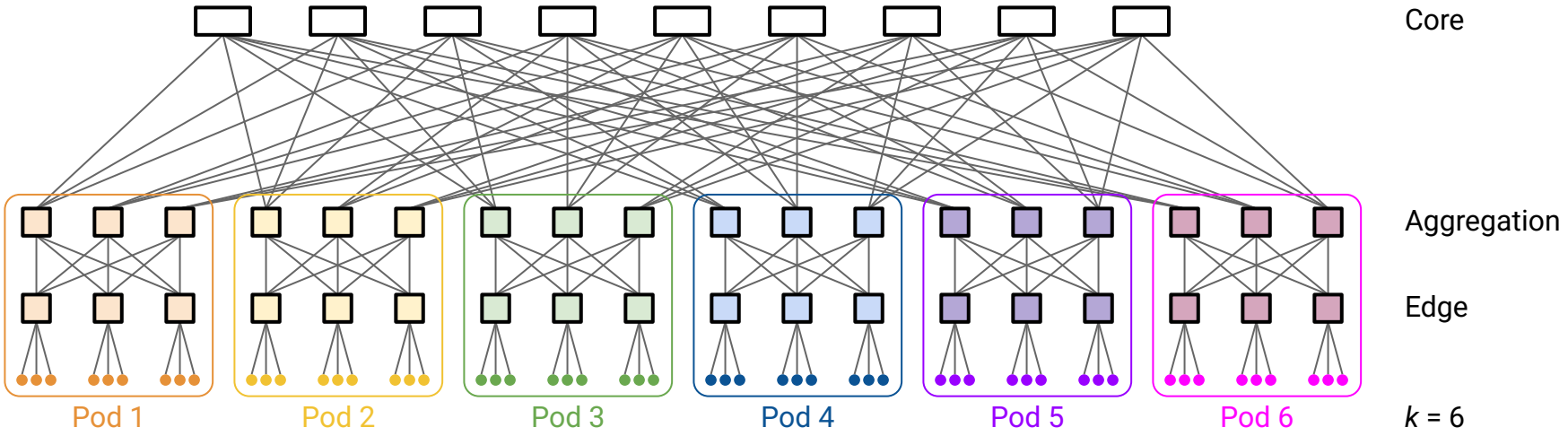
- $(k/2)^2$ servers per pod. *(k/2 edge switches, each has k/2 links down to servers.)*
- $(k/2)^2$ core switches. Each has *k* links down to pods.



# Topology #4: Fat Tree Clos Network

A *k*-ary fat tree has:

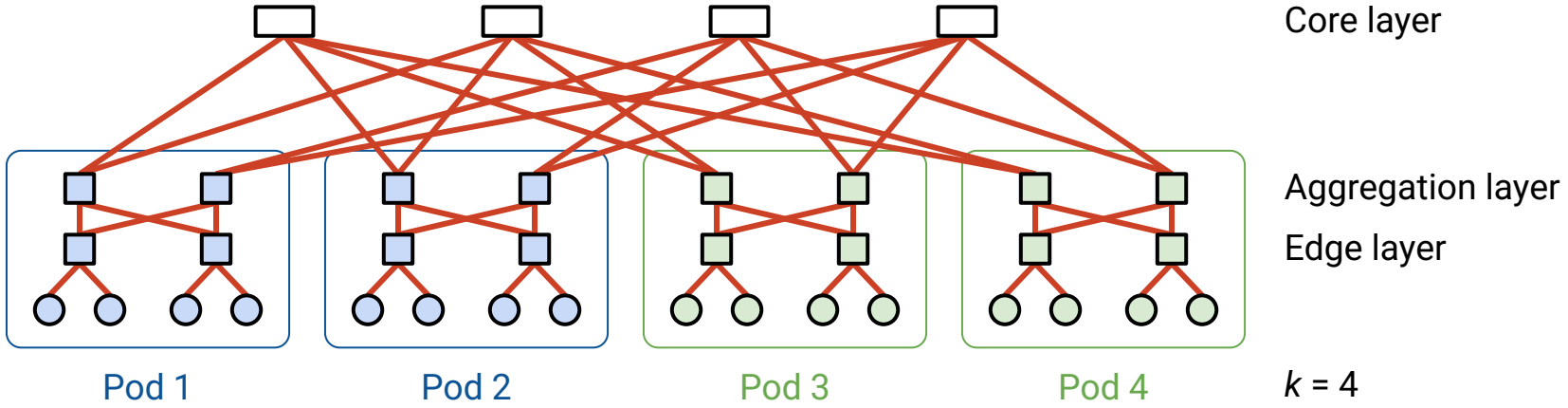
- *k* pods. Each pod has *k* switches, half in each layer.
- Each switch in the pod has *k* links, half up and half down.
- $(k/2)^2$ servers per pod.
- $(k/2)^2$ core switches. Each has *k* links down to pods.



# Topology #4: Fat Tree Clos Network – Bisection Bandwidth

Achieves full bisection bandwidth.

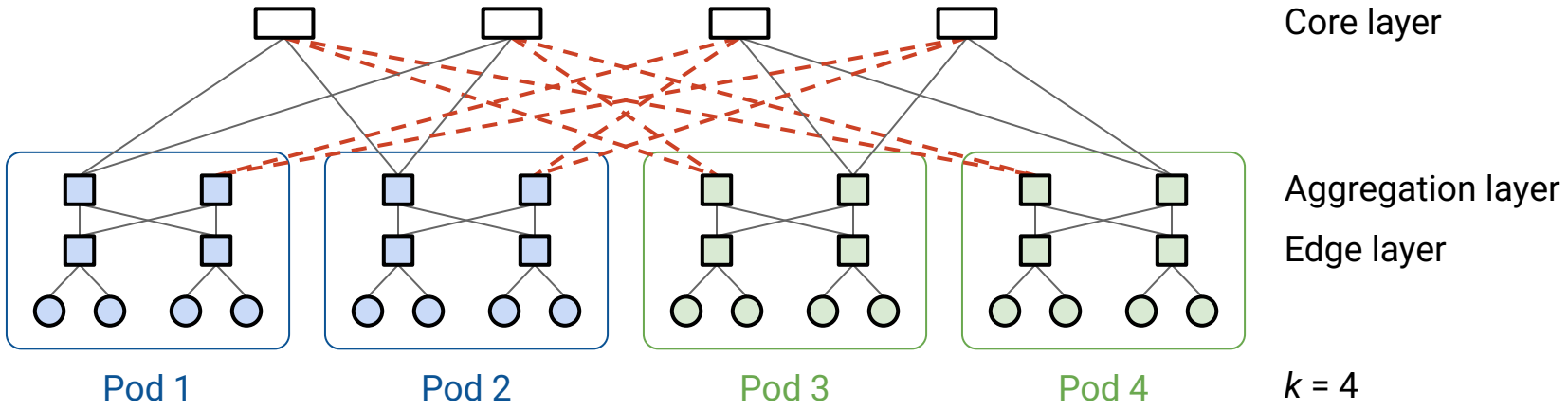
- Each host in left half has a dedicated path to each host in right half.



# Topology #4: Fat Tree Clos Network – Bisection Bandwidth

Achieves full bisection bandwidth.

- Must cut 8 links to disconnect network.
- 8 hosts in left each use one of those links to 8 hosts in right.



# Topology #4: Fat Tree Clos Network – Racks

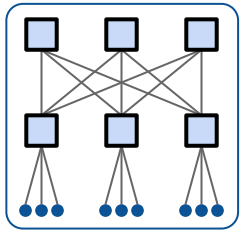
The servers and switches inside a pod can be organized into racks.

$k$ = 48:

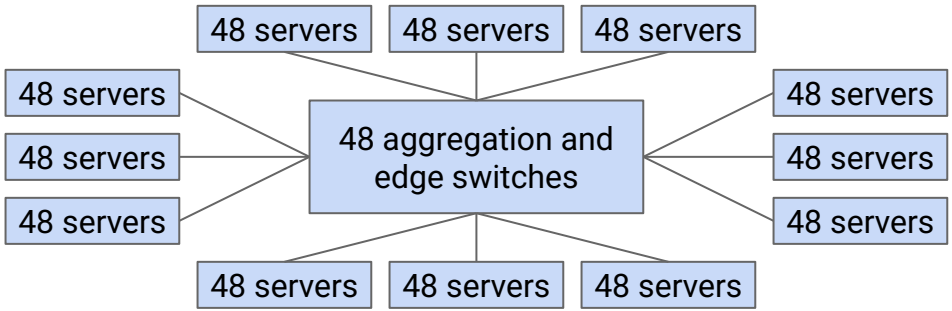
$k/2$ = 24 aggregation switches

$k/2$ = 24 edge switches

$(k/2)^2$ = 576 servers per pod

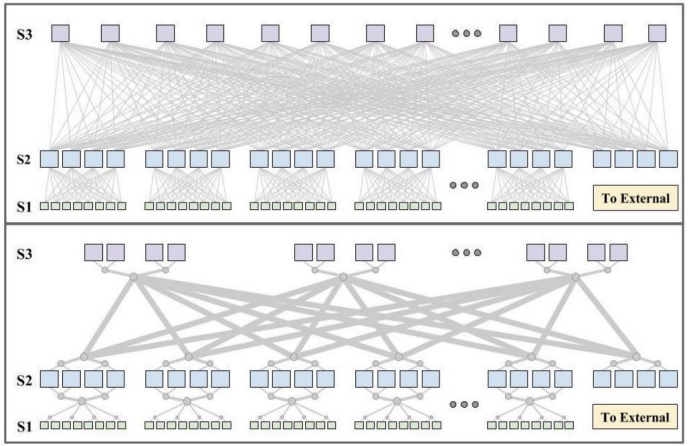


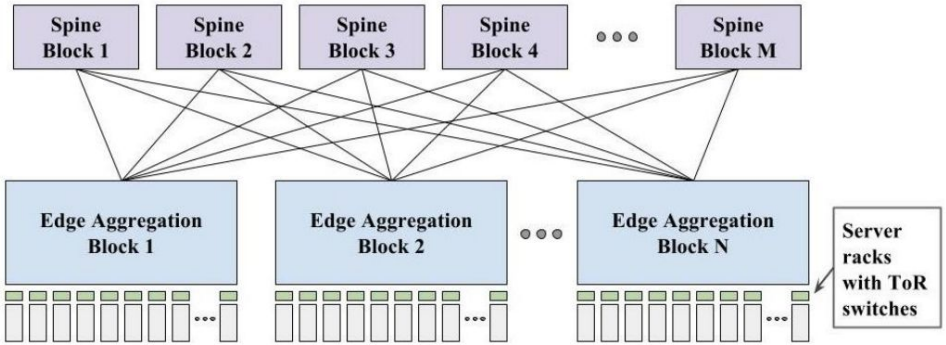
Pod




See the paper for a more detailed description.

# Evolution of Clos Networks for DC



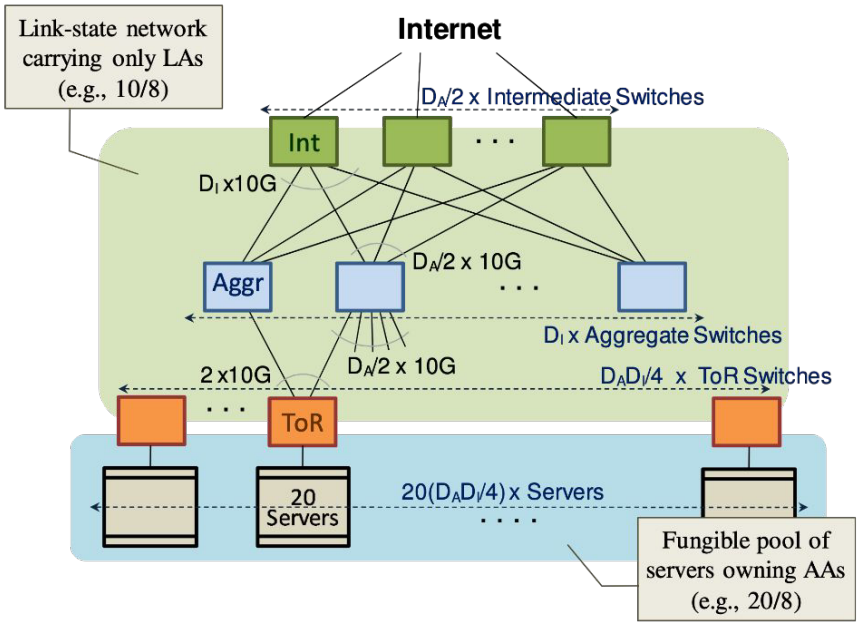




**Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network**

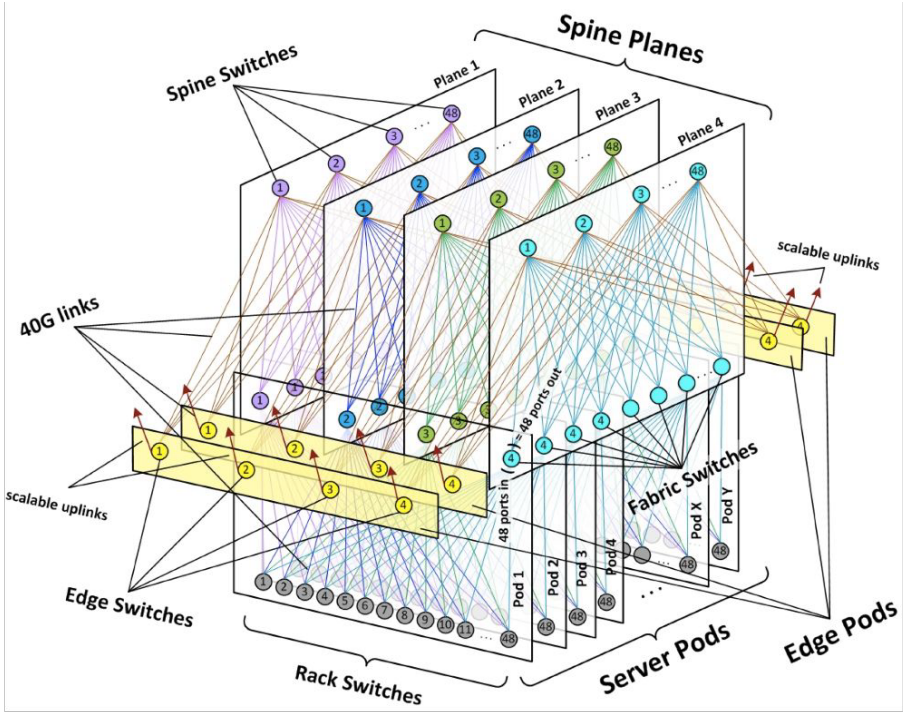
ACM SIGCOMM 2015

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

# Design Variants are Common




VL2 @ **Microsoft**, ACM SIGCOMM'09
Greenburg, Hamilton, Jain, Kandula, Kim, Lahiri, Maltz, Patel, Sengupta




"Introducing data center fabric, the next-generation **Facebook** data center network", Alexey Andreyev, 2015

# Congestion Control in Datacenters

Lecture 19, CS 168, Fall 2026

Designing Datacenters

- What is a Datacenter?
- Inside a Real Datacenter
- Bisection Bandwidth
- Topologies (Clos Networks)

**Congestion Control in Datacenters**

## Why are Datacenters Different? – Queuing Delay

Recall: Packet delay is the sum of 3 values.

- Transmission delay:
  - Small in datacenters. High-capacity links.
  - Larger in the general Internet.
- Propagation delay:
  - Very small in datacenters. Everything in the same building.
  - Much larger in the general Internet (e.g. undersea cables).
- Queuing delay:
  - In the wider Internet, usually insignificant compared to the other delays.
  - **In datacenters, queuing is often the dominant source of delay.**

## Why are Datacenters Different? – Queuing Delay

Example:

- Transmission delay = 0.8 μs per hop.
  - Assume 1000-byte packets and 10 Gbps links.
- Queuing delay = 40 μs.
  - Assume 10 packets in queue, on average; 5 hops per path
  - 10 packets × 0.8 μs transmission per packet = 8 μs spent in queue.
  - 8 μs per queue × 5 queues = 40 μs.
- Propagation delay:
  - In the general Internet, 100ms.
  - In a datacenter, 10 μs.

Total delay:

- In the general Internet: 0.8 μs + 40 μs + 100 ms → propagation dominates.
- In a datacenter: 0.8 μs + 40 μs + 10 μs → queuing dominates.

## Why are Datacenters Different?

Congestion control in datacenters is special because:

- Queuing is the dominant source of delay

Problem: TCP deliberately tries to fill up queues.

- Recall: TCP increases rate until the queue is full and packets are dropped.
- As a result packets suffer high queueing delays

What we want: senders stop increasing their rate *before* the queue gets too full

- Avoids queueing delay
- Hard to achieve if *loss* is the only indication of congestion

# Datacenter Congestion Control

Recall: Datacenters are constrained environments.

- Single operator means we have more control over the hosts and networks.
- Leads to the opportunity for innovation to exploit the network characteristics.

Datacenter-specific congestion control algorithms:

1. BBR: React to delay, not loss.
2. DCTCP: React to router feedback

# Datacenter Congestion Control (1/2) – BBR

BBR (Google) reacts to delay instead of loss.

- Measure RTTs. When RTTs increase, slow down.
- We can detect congestion before the queues fill up.

# Datacenter Congestion Control (2/2) – DCTCP

DCTCP (Microsoft, 2010) reacts to explicit feedback from routers.

- Recall the ECN (Explicit Congestion Notification) bit:
  - Routers mark packets when queue length exceeds a threshold.
  - Senders slow down if the bit is set.
  - Not widely-deployed in the Internet.
- Well-suited for datacenters:
  - Requires a fairly simple change at hosts and routers.
  - Datacenter control ensures we can update all hosts and routers to use ECN.
- Additional modifications:
  - Routers start marking packets earlier. Allows senders to adapt earlier.
  - Senders cut rate in proportion to number of packets with ECN markings. Allows senders to adapt more gently.

# Datacenter Congestion Control (2/2) – DCTCP Performance

We can measure performance with **FCT** (flow completion time):

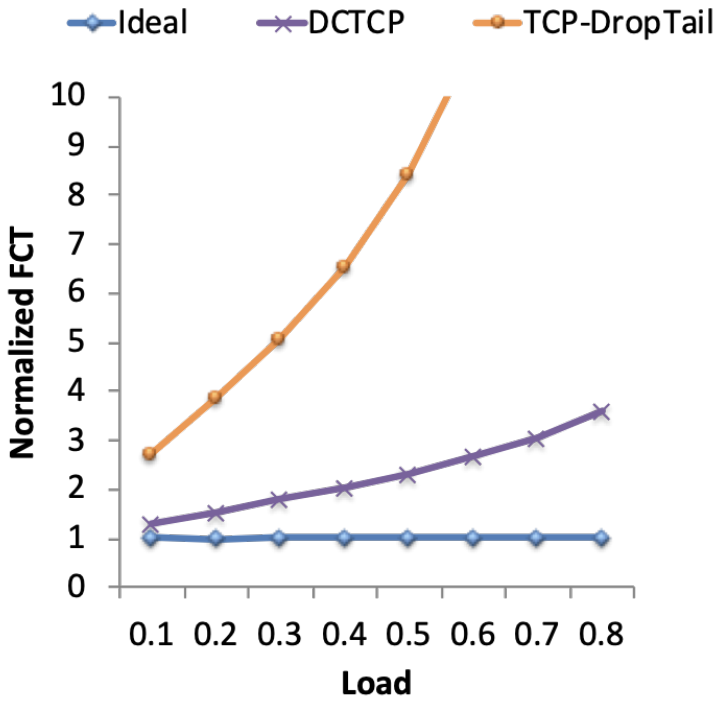
- Time from flow starting, to last byte being received at the destination.

Our benchmark is the ideal FCT:

- FCT using a omniscient scheduler with global knowledge, and schedules flows to minimize FCT.

Normalized FCT = FCT / ideal FCT.

- How much longer am I than ideal?



## Summary: Datacenters

- Datacenters are single organization, multi-application environments.
- A key criteria is high any-to-any bandwidth.
  - We characterize this as bisection bandwidth.
- The topology of the datacenter must be designed to both be scalable, and cost-efficient.
- Some technologies (e.g. congestion control) can be optimized based on the characteristics of datacenters.