POLITECNICO DI MILANO

Computing Infrastructures





Networking

The topics of the course: what are we going to see today?

HW Infrastructures:

System-level: Computing Infrastructures and Data Center Architectures, Rack/Structure;

Node-level: Server (computation, HW accelerators), Storage (Type, technology), Networking (architecture and technology);

Building-level: Cooling systems, power supply, failure recovery

SW Infrastructures:

Virtualization: Process/System VM, Virtualization Mechanisms (Hypervisor, Para/Full virtualization)

Computing Architectures: Cloud Computing (types, characteristics), Edge/Fog Computing, X-as-a service

Methods:

Reliability and availability of datacenters (definition, fundamental laws, RBDs)

Disk performance (Type, Performance, RAID)

Scalability and performance of datacenters (definitions, fundamental laws, queuing network theory)

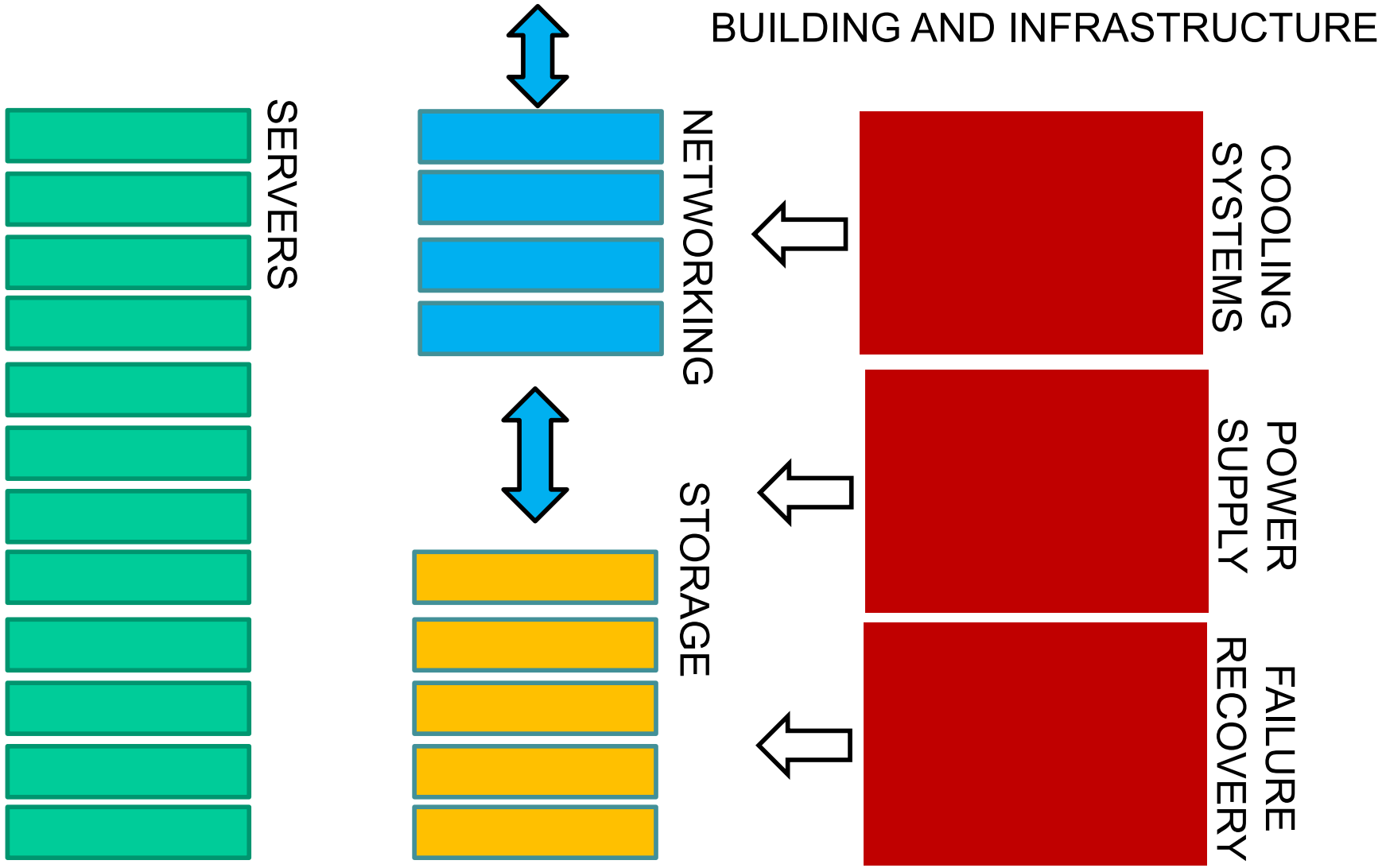
Outline

- Fundamental concepts
- Switch-centric architectures
  - Classical 3-tier architecture
  - Leaf-Spine architectures
- Server-centric and hybrid architectures

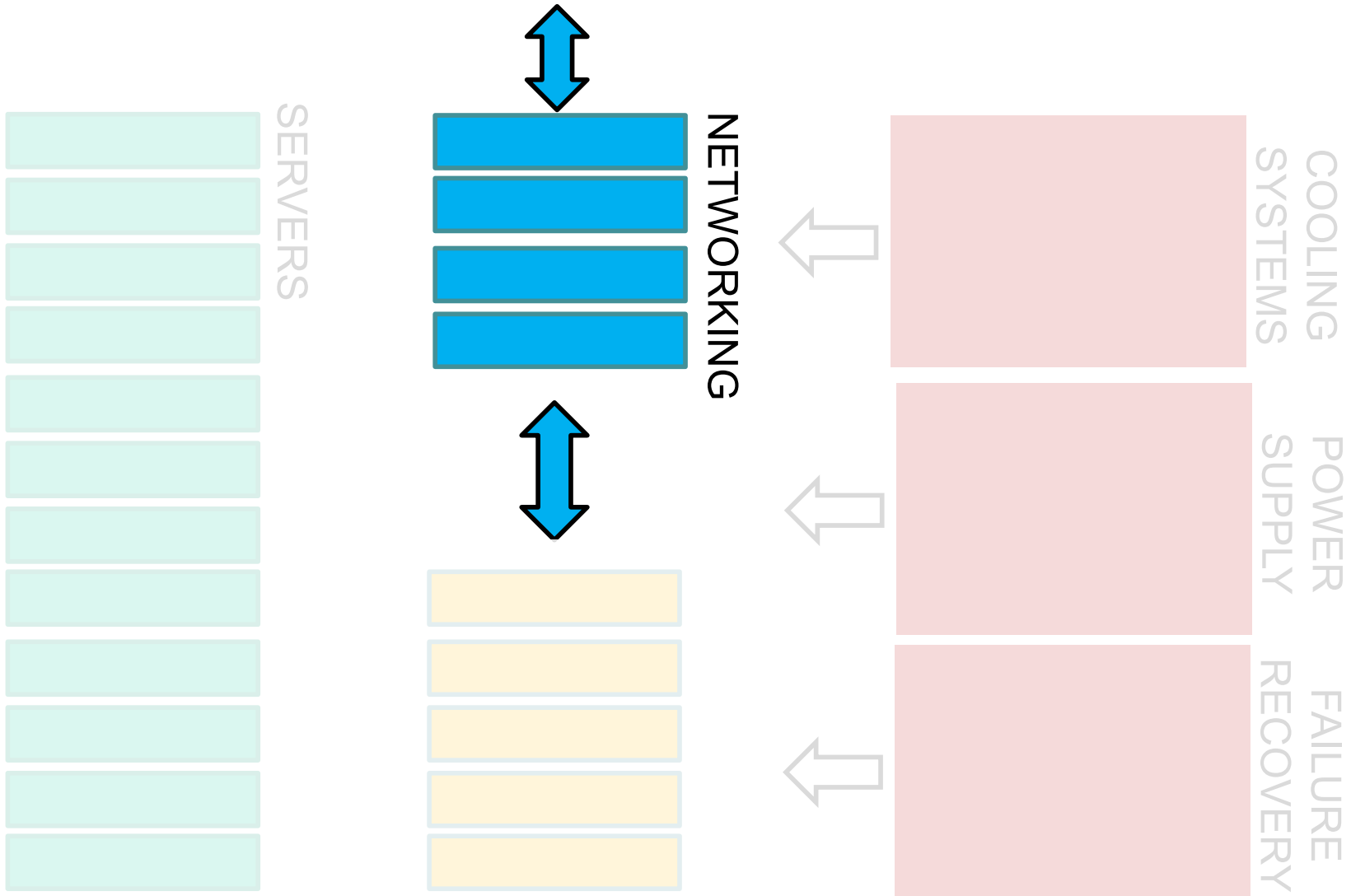
Outline

- Fundamental concepts
- Switch-centric architectures
  - Classical 3-tier architecture
  - Leaf-Spine architectures
- Server-centric and hybrid architectures

Data center infrastructure

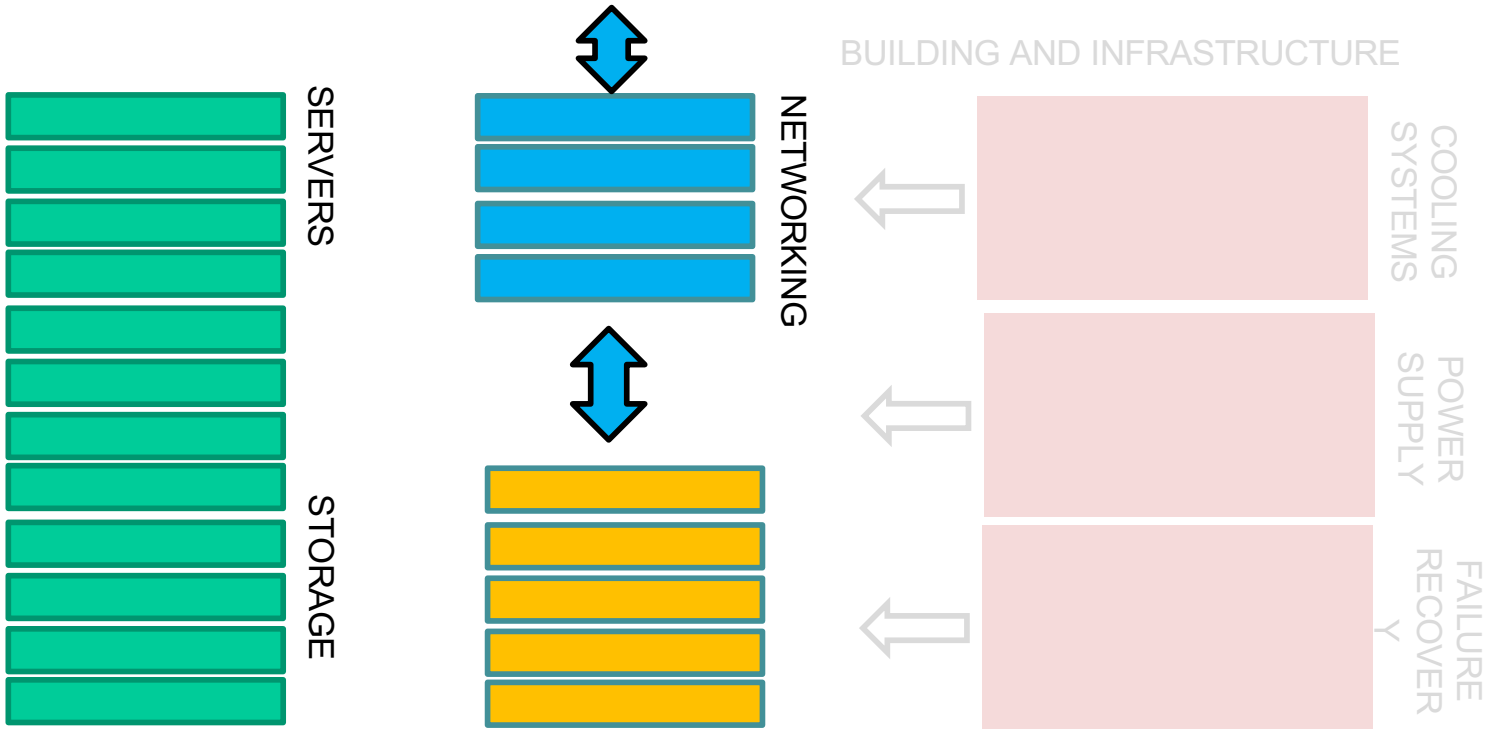


Intra-Datacenter networking
(aka DataCenter [interconnection] Network - DCN)



The need for effective networking in WSCs

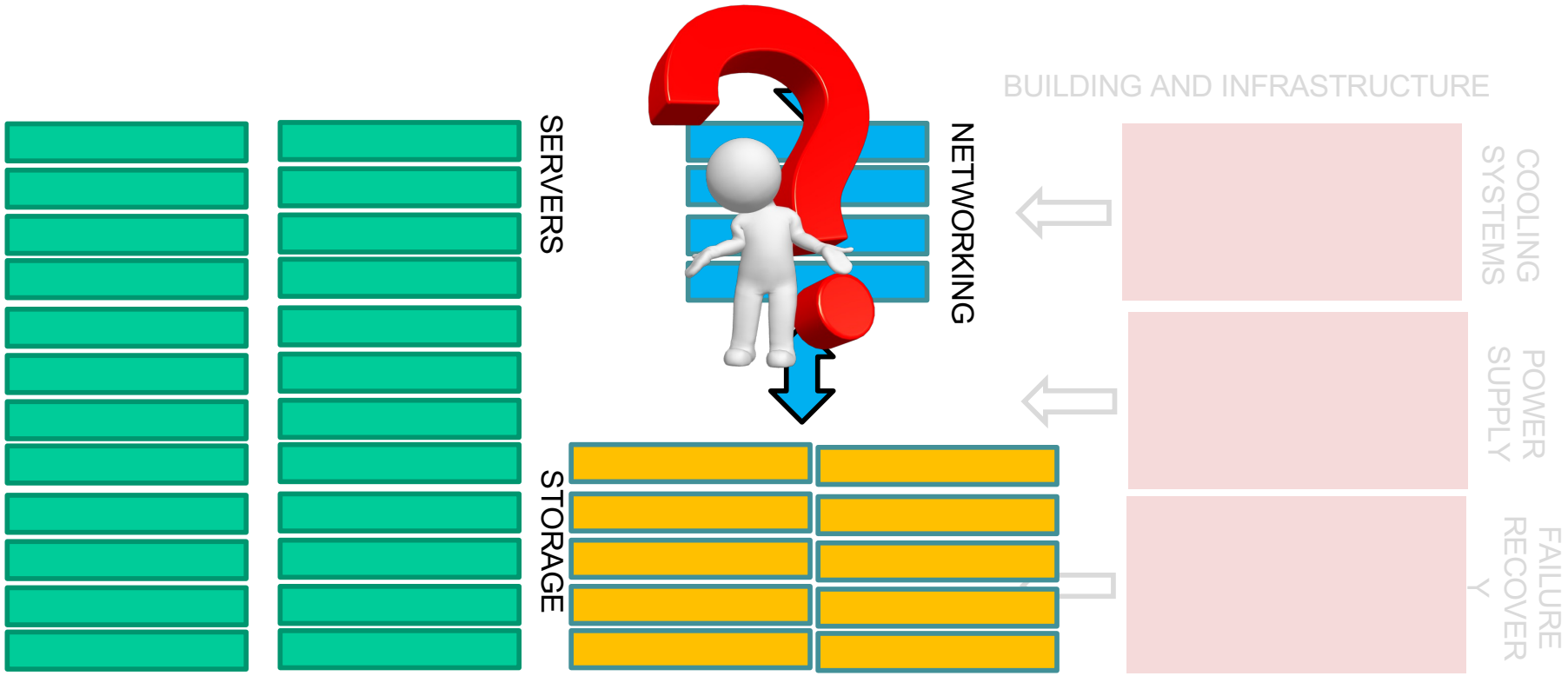
- The performance of servers increases over time, the demand for inter-server bandwidth naturally increases as well!!!



The need for effective networking in WSCs

- The performance of servers increases over time, the demand for inter-server bandwidth naturally increases as well!!!

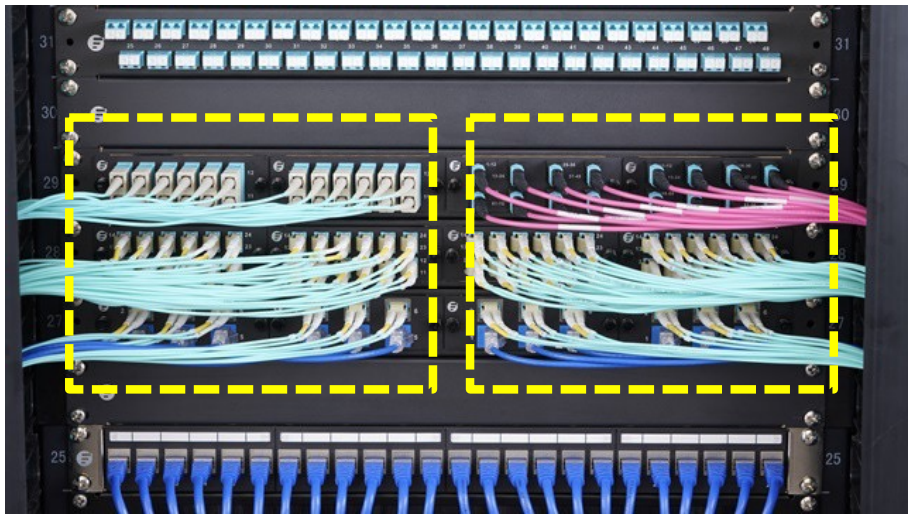
  ⇒ We can double the aggregate compute capacity or the aggregate storage simply by doubling the number of compute or storage elements



The need for effective networking in WSCs

- Networking has no straightforward horizontal scaling solution.
- Doubling leaf bandwidth is easy:
  - with twice as many servers, we'll have twice as many network ports and thus twice as much bandwidth.

x 1

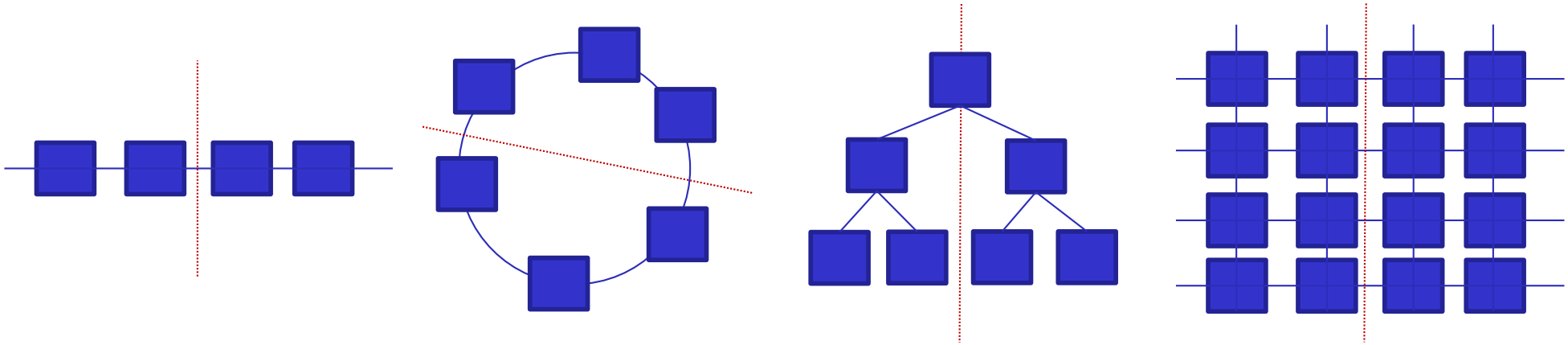


x 2

- But if we assume that every server needs to talk to every other server, we need to deal with bisection bandwidth

Bisection bandwidth

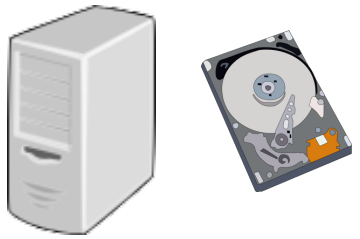
- The bandwidth across the narrowest line that equally divides the cluster into two parts
- Characterizes network capacity since randomly communicating processors must send data across the “middle” of the network

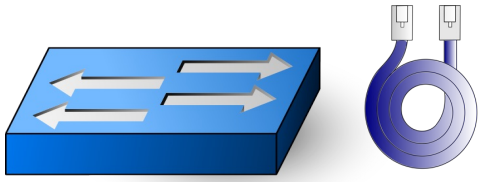


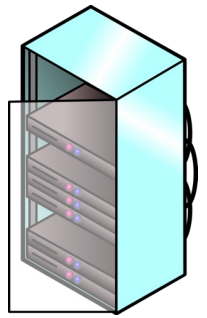
If we assume that every server needs to talk to every other server, we need to double not just leaf bandwidth but *bisection bandwidth*

The design of the data center network

- Design principles
  - very **scalable** in order to support a very large number of servers
  - **minimum cost** in terms of basic building blocks (e.g., switches)
  - **modular** to reuse simple basic modules
  - **reliable** and resilient
  - may exploit novel/proprietary technologies and protocols not compatible with legacy Internet







Classes of DCN

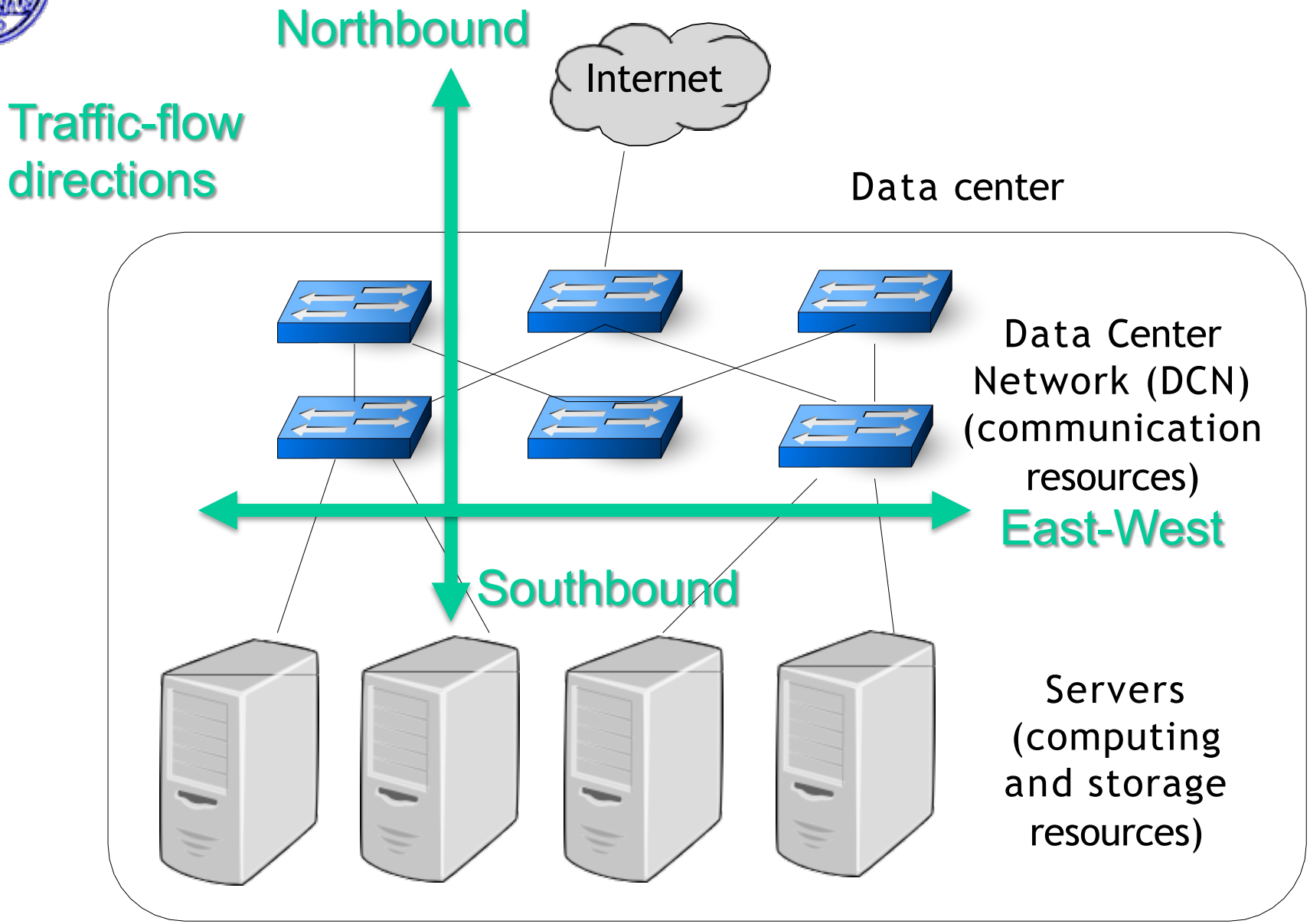
DCNs can be classified into three main categories:

- **Switch-centric** architectures
  - Uses switches to perform packet forwarding
- **Server-centric** architecture
  - Uses servers with multiple Network Interface Cards (NICs) to act as switches in addition to performing other computational functions
- **Hybrid** architectures
  - Combine switches and servers for packet forwarding

Outline

- Fundamental concepts
- Switch-centric architectures
  - Classical 3-tier architecture
  - Leaf-Spine architectures
- Server-centric and hybrid architectures

Switch-centric architectures

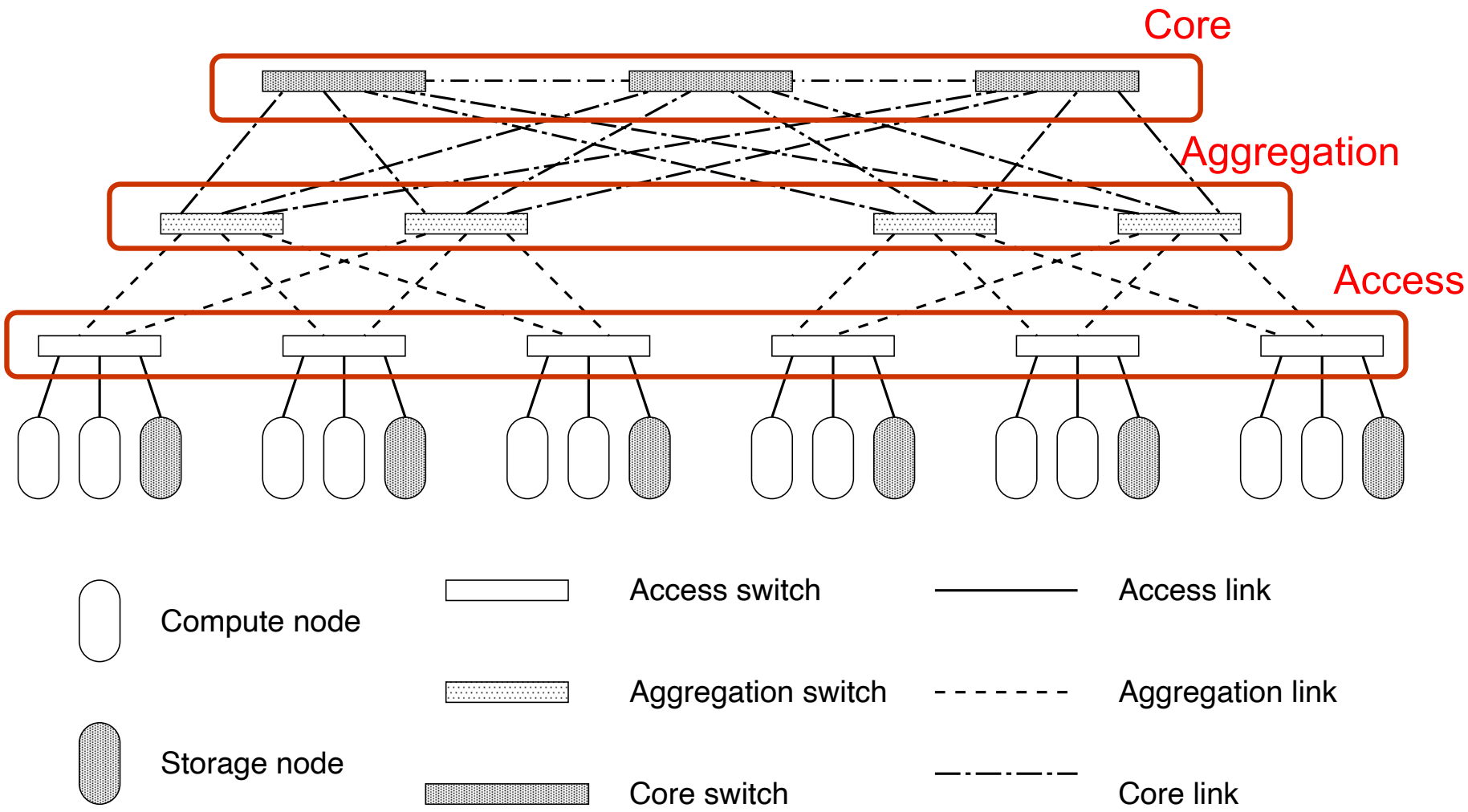


East-West traffic

- East-West traffic
  - storage replication (few flows, many data)
    - in Hadhoop distributed filesystem, at least 3 copies of the same data, usually two in the same rack and one in another rack
  - VM migration
  - Network Function Virtualization (NFV)
    - data is processed through a sequence of VMs (e.g., firewall, web server, parental control, accounting server)
- East-West traffic usually larger than North-South traffic
  - A **1 byte transaction in North-South** traffic **generates on average a 100 bytes transaction in East-West traffic**
  - According to Cisco's Global Cloud Index (http://blogs.cisco.com/security/trends-in-data-center-security-part-1-traffic-trends, May 2014):
    - In a data center: East-West traffic (76%), North-South traffic (17%), inter-data center traffic (7%).
    - In campus networks: North-South traffic (>90%).

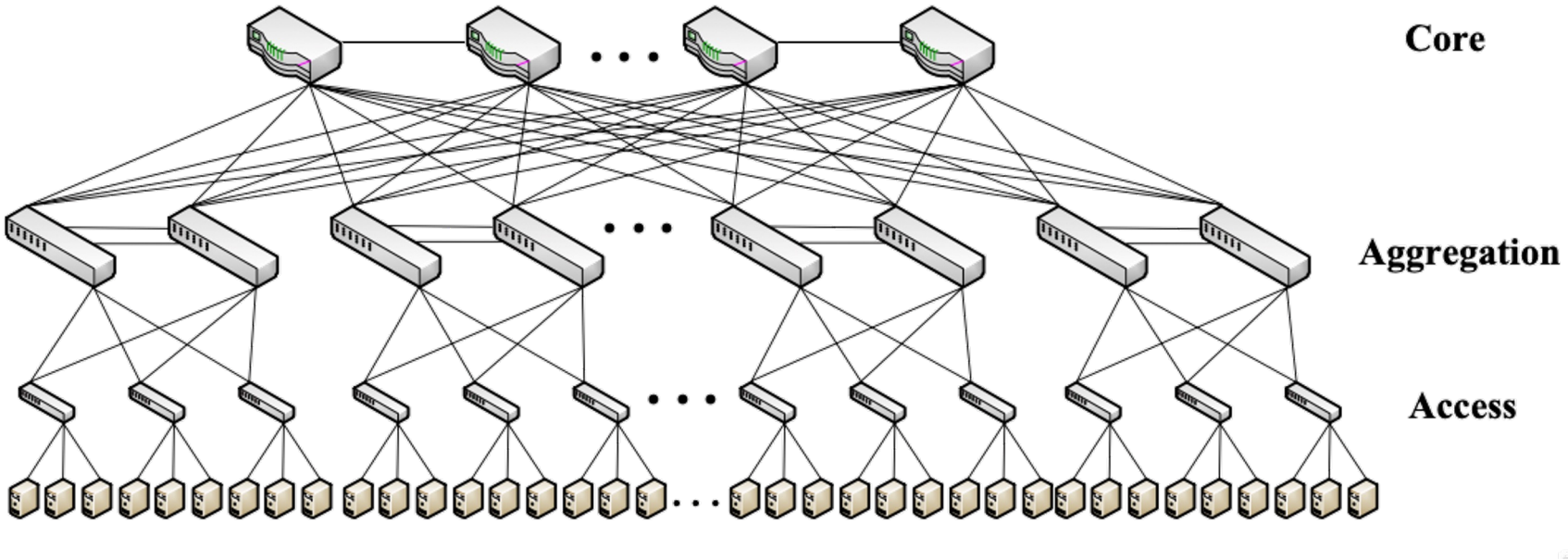
Three-Tier (or layer) "Classical" Network

- Three layer architecture configures the network in three different layers:



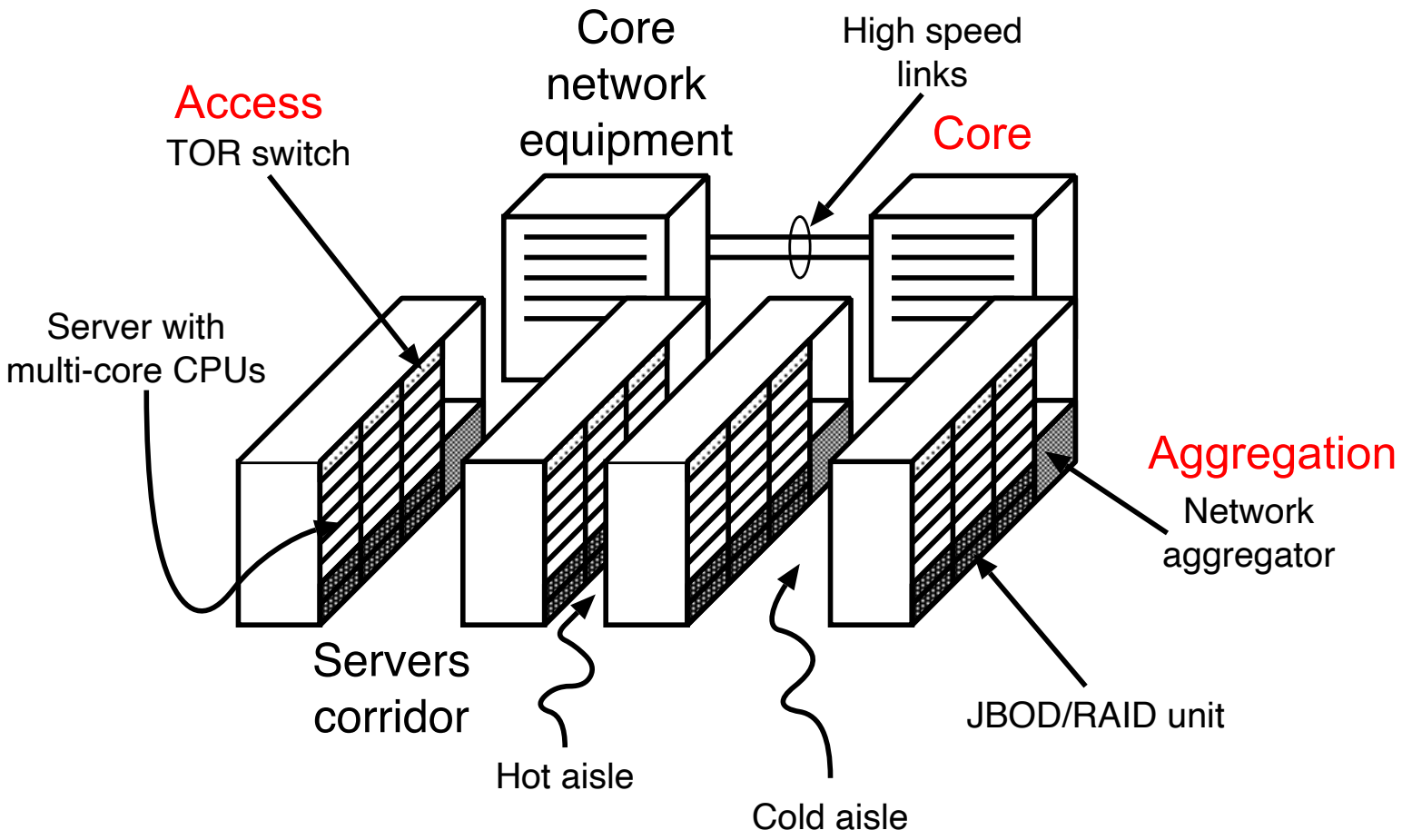
Three-Tier (or layer) "Classical" Network

- A simple DCN topology
- Servers are connected to the DCN through access switches.
- Each access-level switch is connected to at least two aggregation-level switches.
- Aggregation-level switches are connected to core-level switches (gateways).



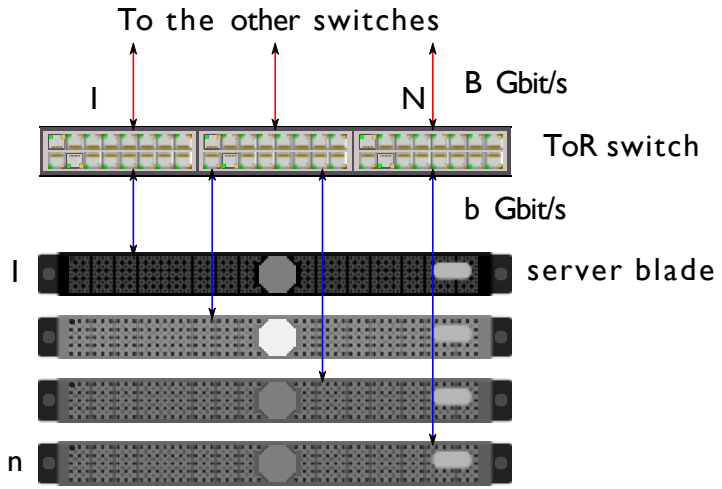
Data-center network architectures

- Three layer architecture reflects the topology of the data center:



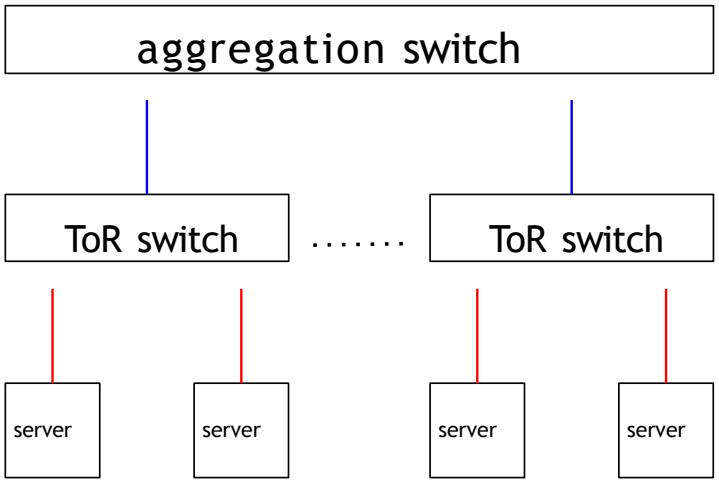
Server packing in a rack

- Standard 19 inch rack 42 EIA Units (pizza box)
  - 40 server blades
    - possible single /26 subnet
  - 1 ToR (Top of Rack) switch
- without oversubscription: NB = nb
  - example
    - 40 ports @ 1 Gbit/s to the servers
    - 4 ports @ 10 Gbit/s to the other switches
- with oversubscription: NB < nb
  - example with oversubscription 1:4
    - 40 ports @ 1 Gbit/s to the servers
    - 1 ports @ 10 Gbit/s to the other switches

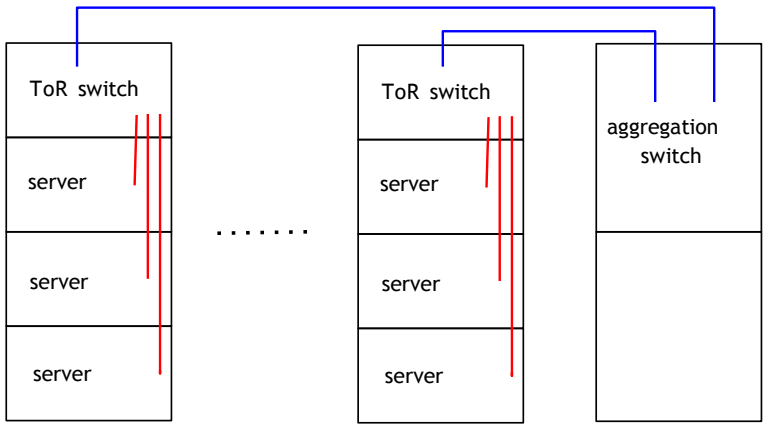


ToR vs EoR architectures

- ToR (Top-of-Rack) architecture
  - All servers in a rack are connected to a ToR access switch within the same rack
  - Aggregation switches are in dedicated racks or in shared racks with other ToR switches and servers
  - The number of cables is limited → simpler cabling. The number of ports per switch is also limited (lower costs)
  - Higher complexity for switch management



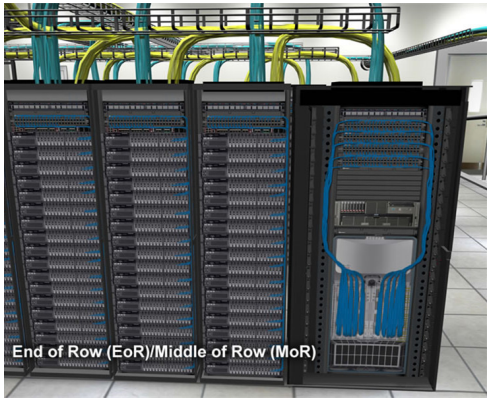
logical topology

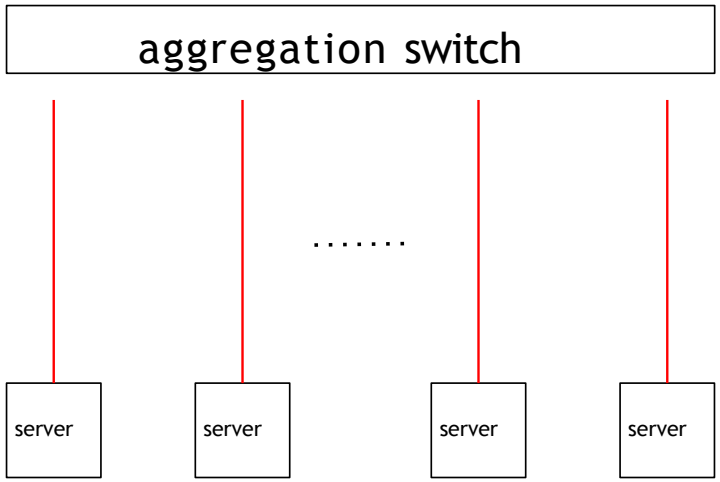


physical topology

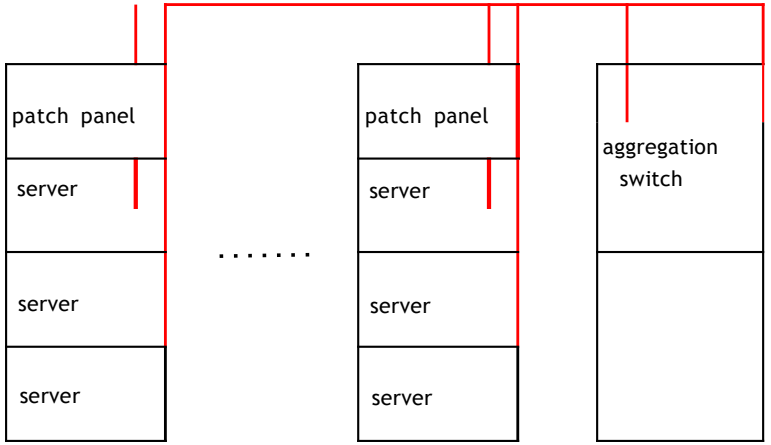
ToR vs EoR architectures

- EoR (End-of-Row) architecture
  - Aggregation Switches are positioned one per corridor, at the end of a line of rack.
  - servers in a racks are connected directly to the aggregation switch in another rack
  - Aggregation switches must have a larger number of ports
  - more complex cabling, longer cables are required (higher costs)
  - Patch panel to connect the servers to the aggregation switch
  - Simpler switch management





logical topology



physical topology

Three-Tier (or layer) “Classical” Network

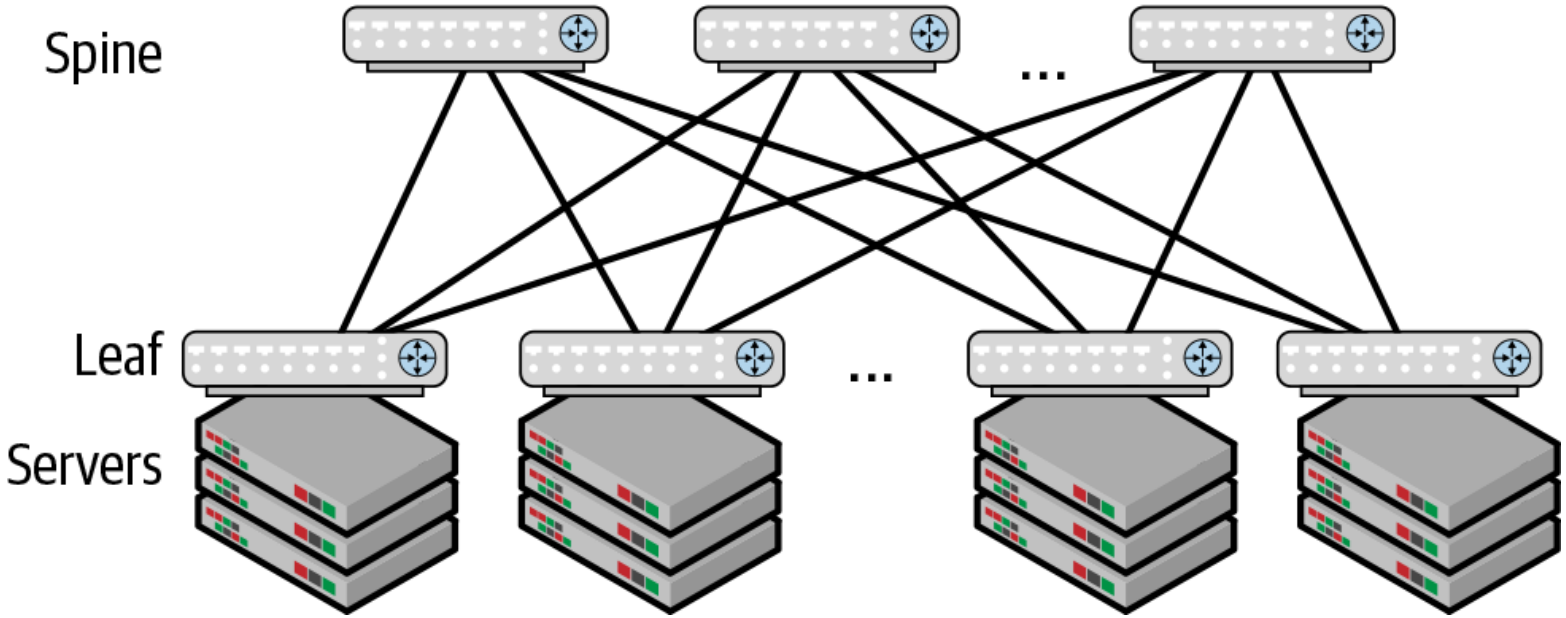
- This solution is very simple, but can be very expensive in large data-centers since:
  - Upper layers require faster network equipments.
  - For example:
    - 1 GB Ethernet at the access layer
    - 10 GB Ethernet at the aggregation layer
    - 25 GB Optical connections at the core layer
  - The cost in term of acquisition and energy consumption can be very high

Outline

- Fundamental concepts
- Switch-centric architectures
  - Classical 3-tier architecture
  - Leaf-Spine architectures
- Server-centric and hybrid architectures
- Addressing and routing in DCN

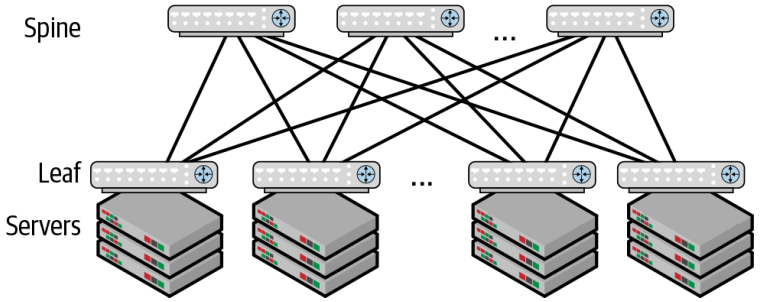
DCN Clos topology (a.k.a. Leaf-Spine)

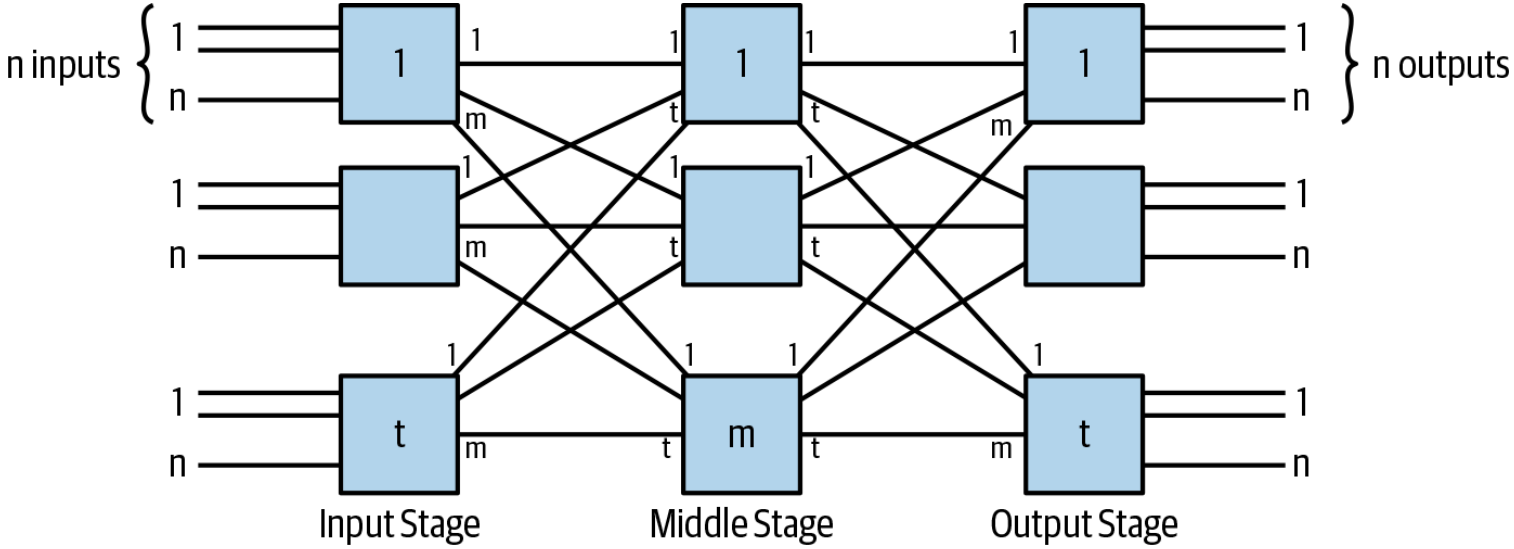
- Two stage interconnections
  - Leaf: ToR switch
  - Spine: dedicated switches (aggregation switches)
- In practice: servers have two interfaces connected to two ToR switches to provide fault-tolerance



Clos networks

- Spine-leaf topologies are borrowed from the telephone world
- Non-folded Clos structure

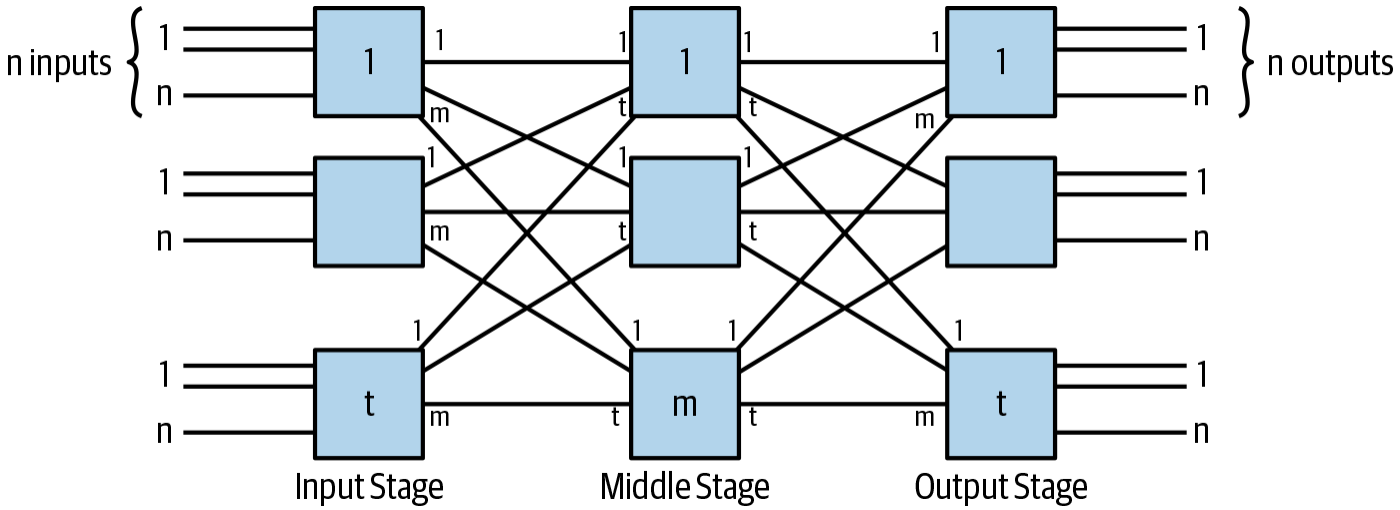




Clos networks in the telephone world

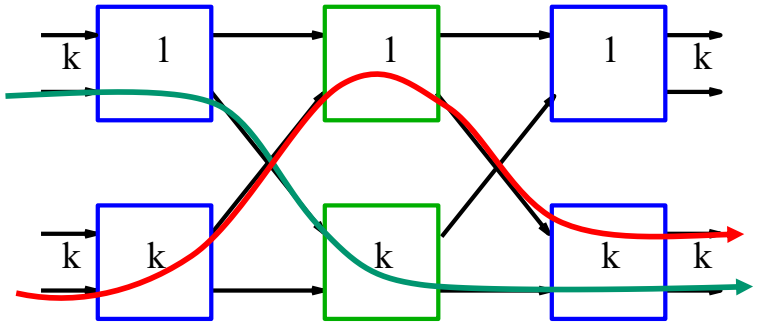
- Let m be the number of middle stage switches
- Let n be the number of input and outputs
  - If m ≥ n there is always a way to rearrange communications to free a path between any pair of idle input/output
  - If m ≥ 2n - 1 there is always a free path between any pair of idle input/output

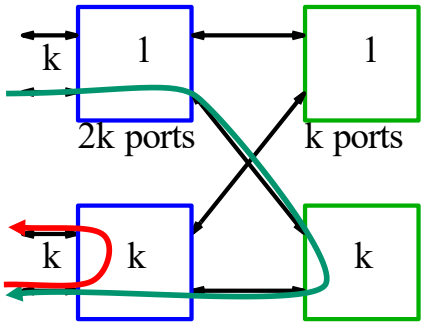
- But a DCN is a PACKET-SWITCHED network!!



From Clos to "Leaf and Spine" topology

- Clos topology (n = m = k)
  - each switching module is unidirectional
    - k input + k output ports per module
  - each path traverses 3 modules
- Leaf and spine topology
  - each switching module is bidirectional
    - Leaf: 2k bidirectional ports per module
    - Spine: k bidirectional ports per module
  - each path traverses either 1 or 3 modules

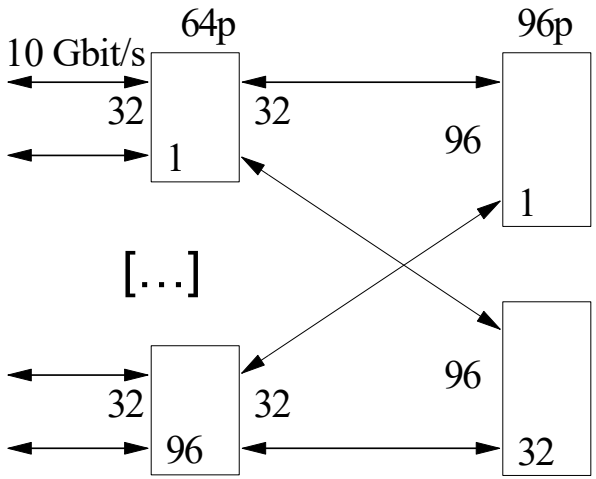


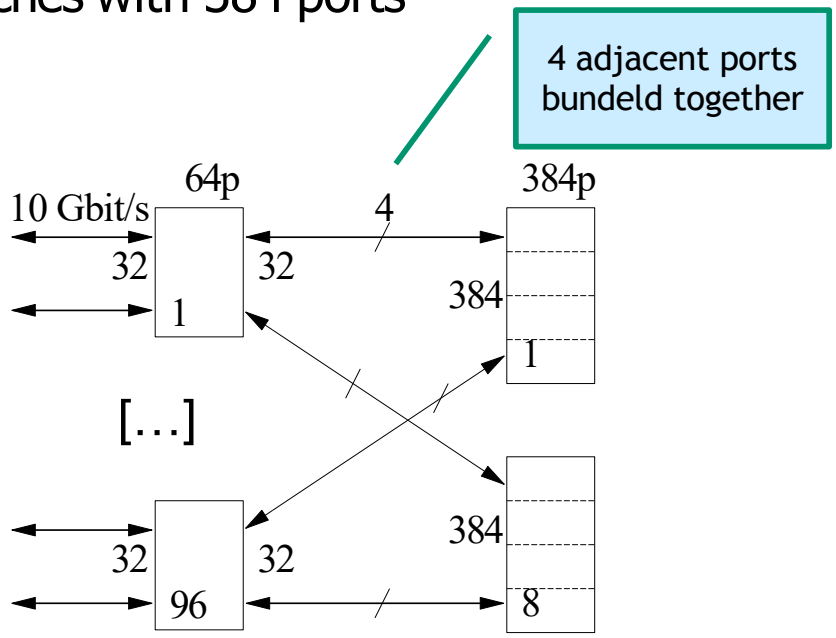


Examples of DCN design

- 3072 servers
  - 3072 ports at 10 Gbit/s ⇒ 30.72 Tbit/s

    alternative designs
    - 96 switches with 64 ports and 32 switches with 96 ports
    - 96 switches with 64 ports and 8 switches with 384 ports

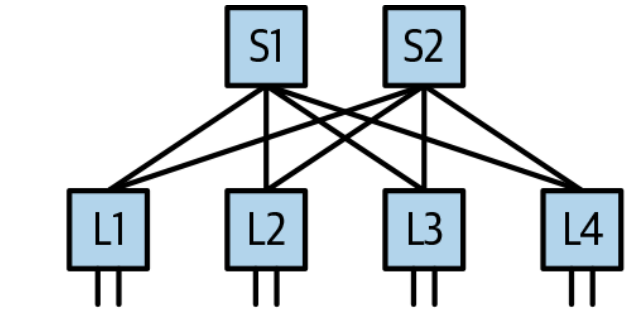




Example taken from “Cisco’s Massively Scalable Data Center”, 2009

Examples of DCN design

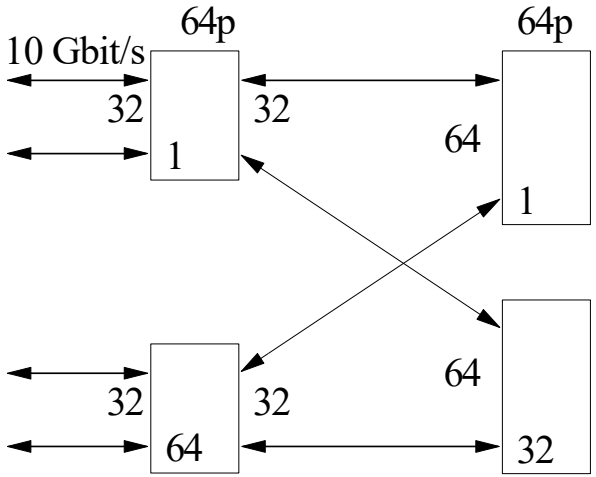
- Interesting case: all switches have an equal number of ports p = 2k
  - Leaves: k x k switches (2k ports), k servers per TOR
  - Spine: 2k-port switches; # spine switches = k
  - → # leaves = 2k → $2k^2$ servers



(a) Two-tier Clos using four-port switches

- k = 2 → 8 servers
  - 8 ports at 10 Gbit/s ⇒ 80 Gbit/s
  - 4 switches with 4 ports and 2 switches with 4 ports

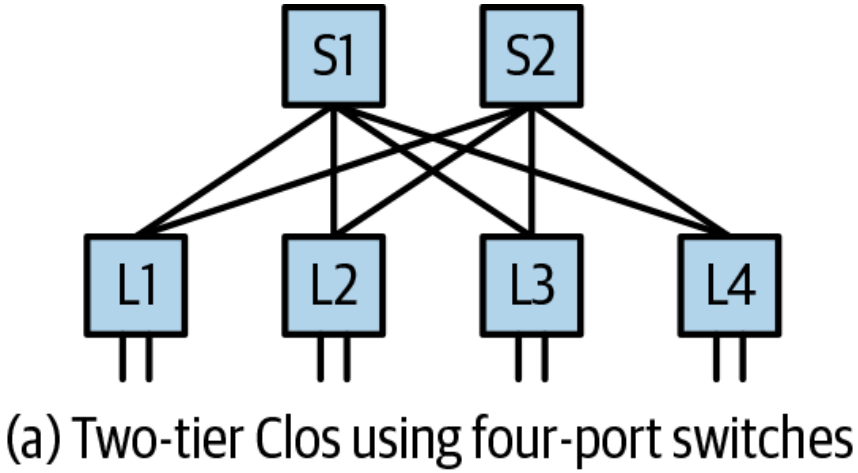
- k = 32 → 2048 servers
  - 2048 ports at 10 Gbit/s ⇒ 20.48 Tbit/s
  - 64 switches with 64 ports and 32 switches with 64 ports



Clos design in DNC

- Advantages
  - Use of homogeneous equipment
  - Simple Routing
  - The number of hops is the same for any pair of nodes
  - Small blast radius

- What About Scaling further?
- Can we scale to multi-tier designs?



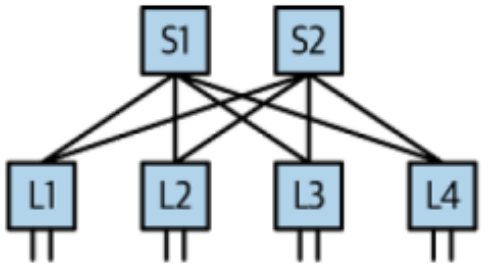
(a) Two-tier Clos using four-port switches

Start with a two-tier network and add an additional row of switches

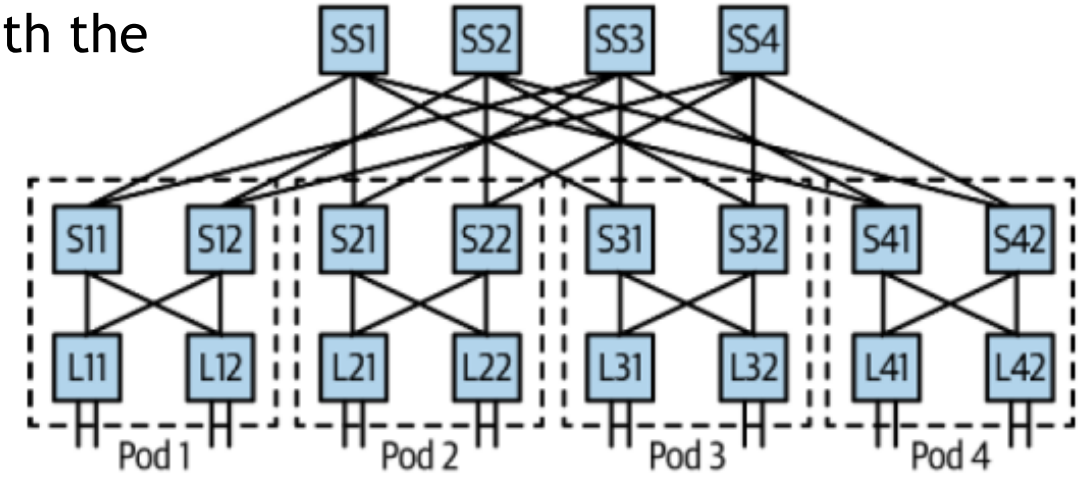
Scaling to a three-tier network

Pod-based model, aka the Fat Tree

- An option: transform each spine-leaf group into a «pod» and add a super-spine tier
  - POD: Point Of Delivery
- A highly scalable and cost-efficient DCN architecture that aims to maximize bisection bandwidth.
- It can be built using commodity Gigabit Ethernet switches with the same number of ports.
- Used by Microsoft, Amazon



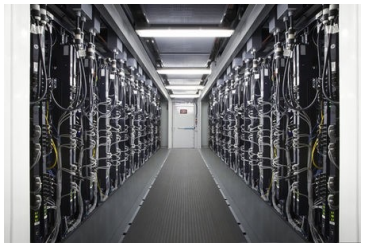
(a) Two-tier Clos using four-port switches

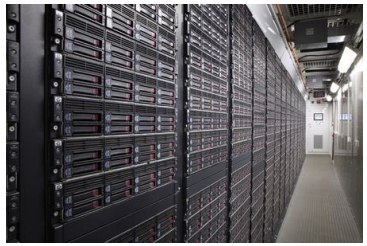


(c) Pod-based three-tier Clos using four-port switches

Point Of Delivery (POD)

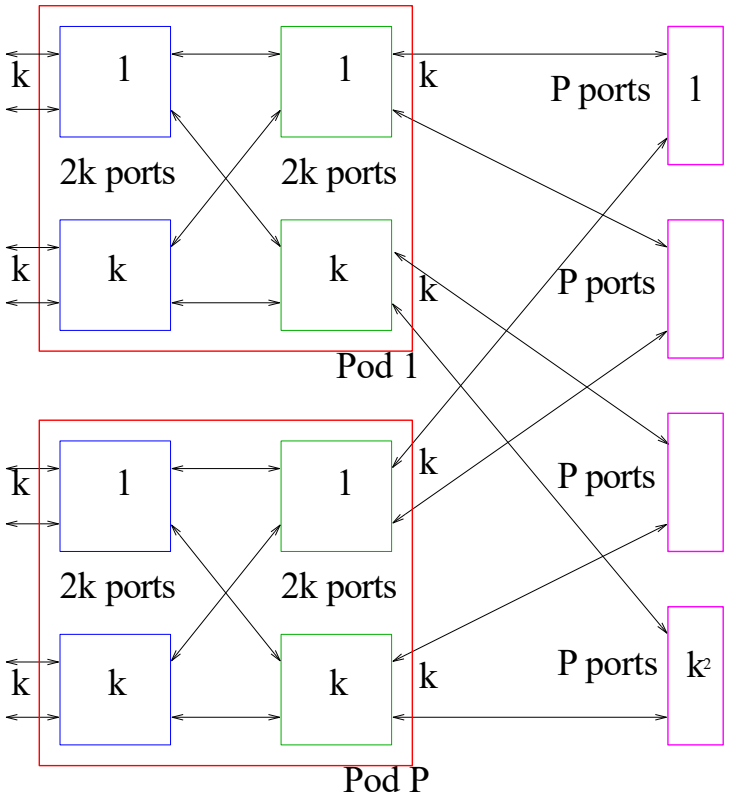
- A module or group of network, compute, storage, and application components that work together to deliver a network service
- The PoD is a repeatable pattern, and its components increase the modularity, scalability, and manageability of data
  - (taken from Wikipedia)





Intra-pod and inter-pod communications

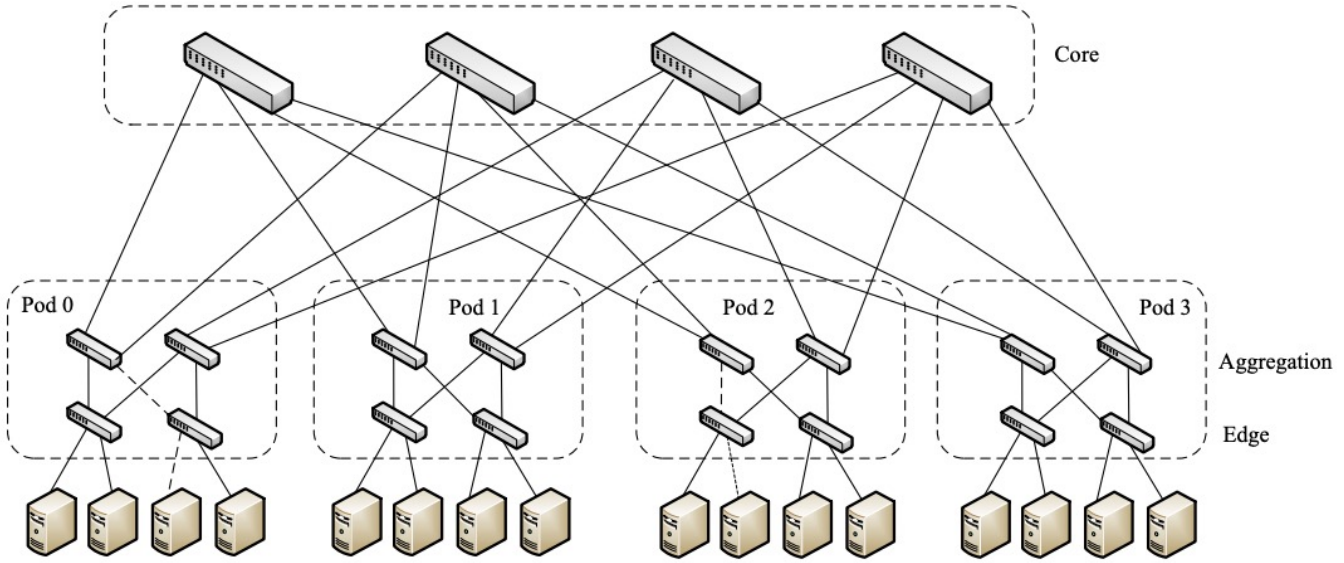
- k^2P servers
  - $2kP$ switches with $2k$ ports
  - k^2 switches with P ports
- Fat-tree: choose $P = 2k$
  - $2k^3$ servers
  - $(k^2 + 2k \cdot 2k) = 5k^2$ switches with $2k$ ports



The Fat-Tree Network

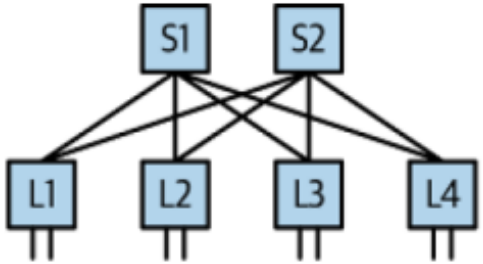
At the edge layer, there are 2*k* pods (groups of servers), each with k^2 servers.

- Each edge switch is directly connected to k servers in a pod and k aggregation switches.
- A fat-tree network with 2*k*-port commodity switches can accommodate $2k^3$ servers in total
- k^2 core switches with 2*k*-port each, each one connected to 2*k* pods
- Each aggregation switch is connected to *k* core switches
  - Note the partial connectivity at switch level

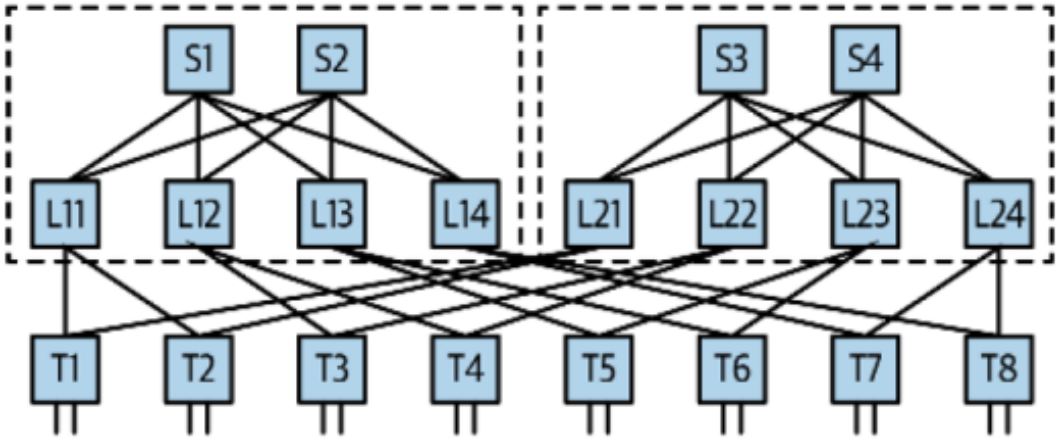


k = 2
4 pods
16 servers
20 switches

Virtual chassis model (e.g. Facebook)



(a) Two-tier Clos using four-port switches



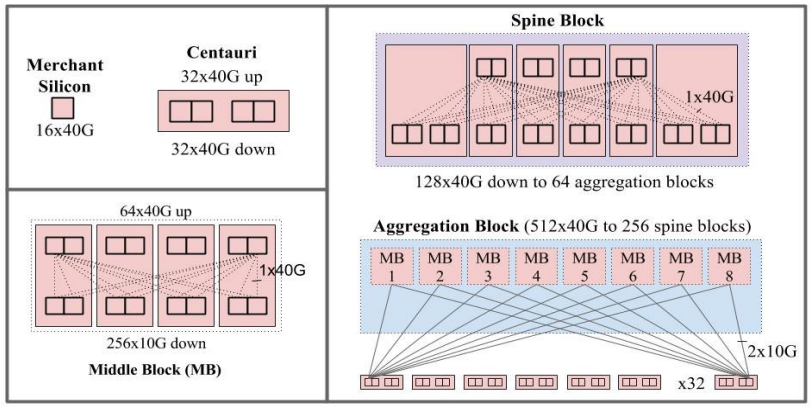
(b) Virtual chassis-based three-tier Clos using four-port switches

Google scenario → Goolge Datacenter

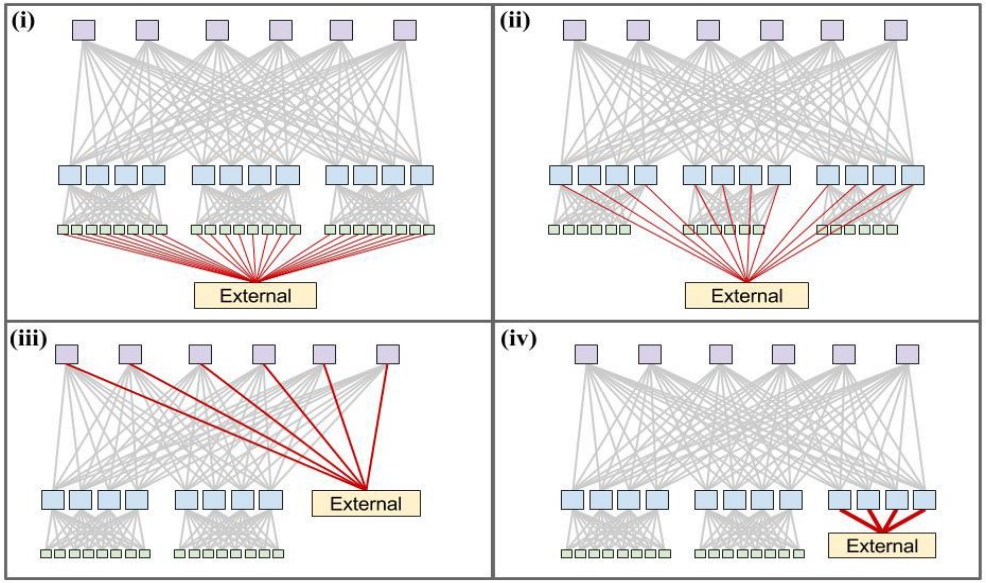
- Worldwide coverage with tens of sites
- Data center traffic bandwidth demand doubles every 12-15 months (faster than Internet)
  - larger datasets (photo/video, logs, Internet-connected sensors, etc.)
  - web services
  - internal applications (index generation, web search, serving ads, etc.)
- **Google Design approach**
  - multistage Clos topologies on commodity switch silicon
  - centralized control
    - one configuration pushed to all the switches
    - SDN approach
  - modular hardware design with simple, robust software
  - general-purpose, off-the-shelf switch components

A. Singh, et al., "Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network", ACM SIGCOMM Computer Communication Reviews, Oct. 2015

Jupiter topology



Building blocks used in the Jupiter topology.



Options to connect to the external network layer.

Jupiter Middle blocks housed in racks.

Figures reproduced from [*Google*].

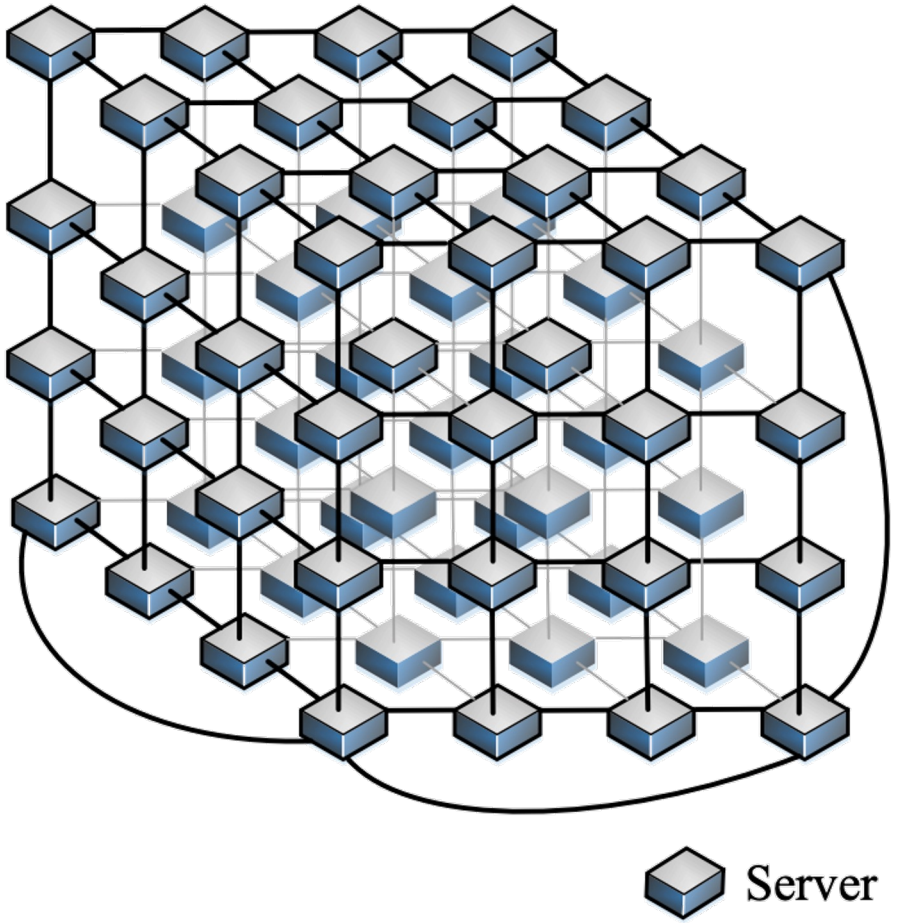
Outline

- Fundamental concepts
- Switch-centric architectures
  - Classical 3-tier architecture
  - Leaf-Spine architectures
- Server-centric and hybrid architectures

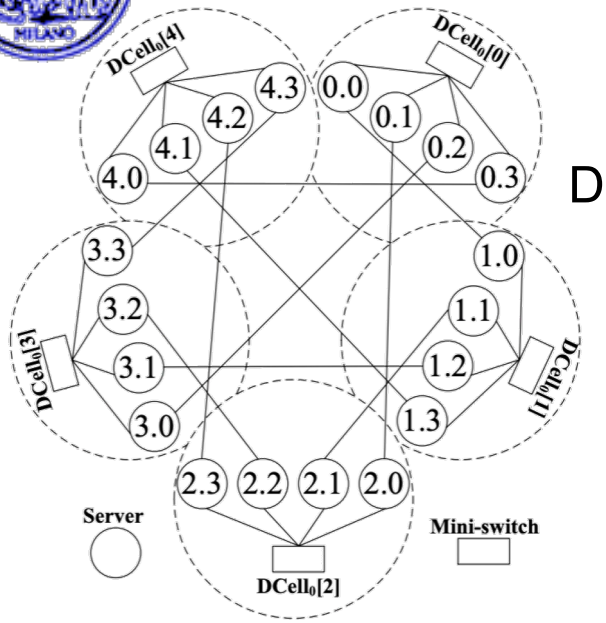
Server-centric architectures: CamCube

- A server-centric architecture proposed for building container-sized data centers.
- It may reduce implementation and maintenance costs by using only servers to build the DCN.
- It uses a 3D-Torus topology to interconnect the servers directly.
- As a torus-based architecture, it exploits network locality to increase communication efficiency.
- **Drawbacks**: CamCube requires servers with multiple NICs to assemble a 3D Tours network, long paths, and high routing complexity

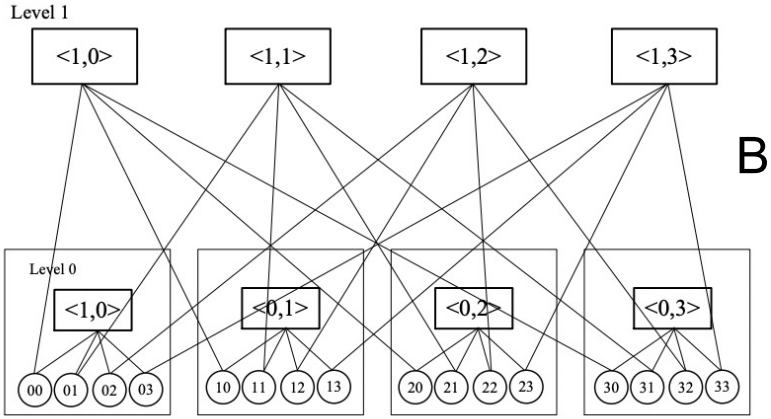
3D Torus with 64 servers



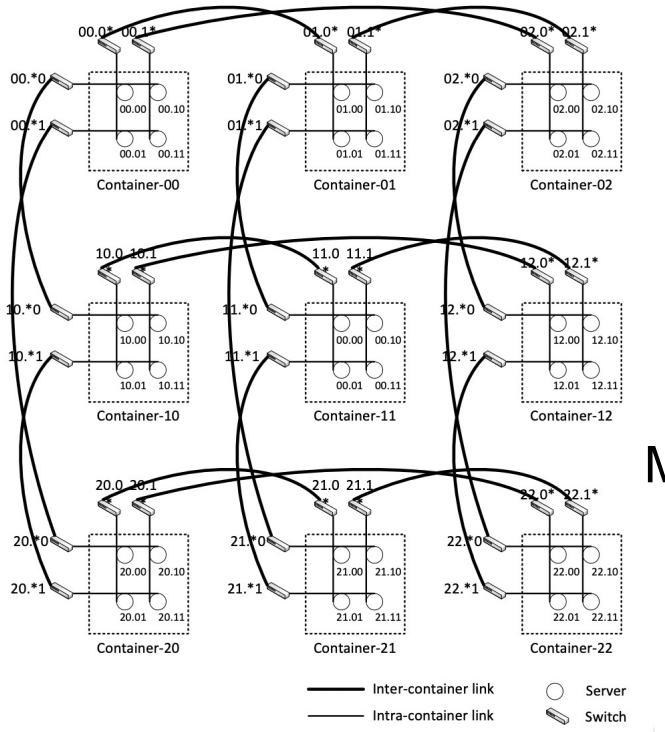
Hybrid architectures



DCell



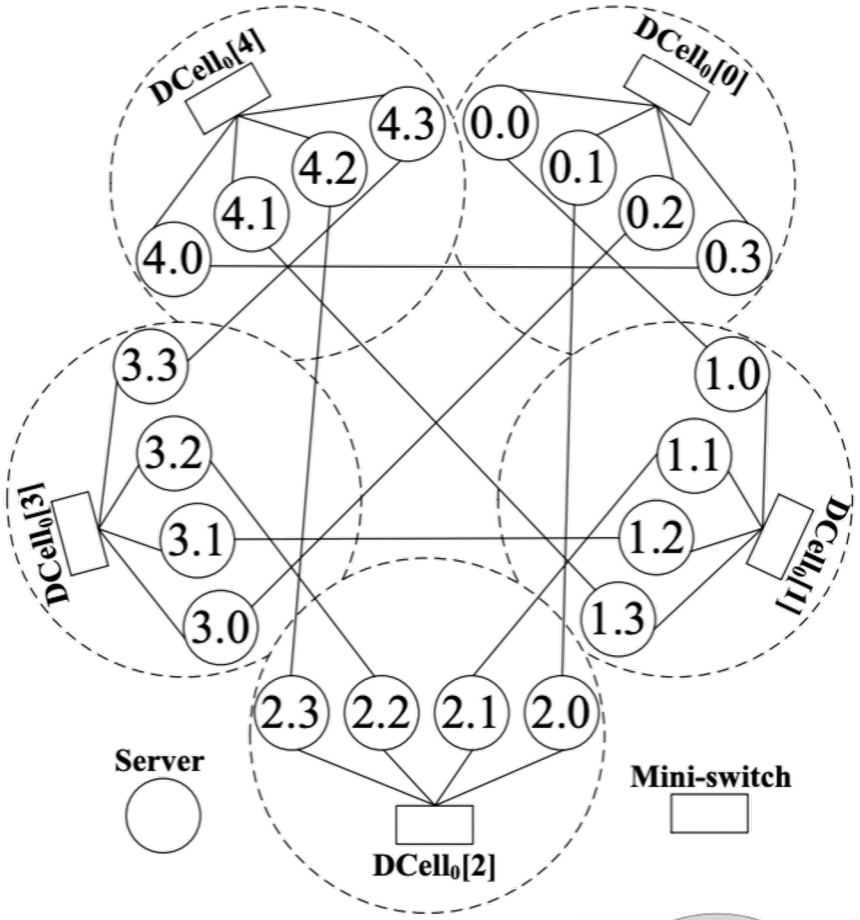
BCube

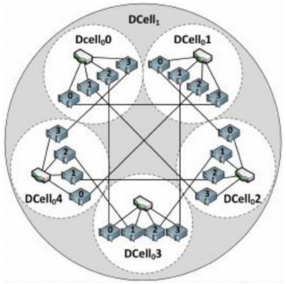


MDCube

Hybrid architectures: DCell

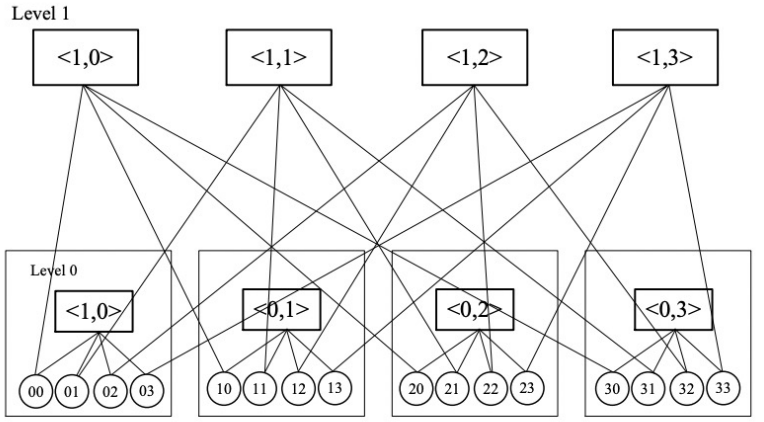
- A scalable and cost-efficient hybrid architecture that uses switches and servers for packet forwarding.
- **Recursive Architecture.** It uses a basic building block called $DCell_0$ to construct larger DCells.
- $DCell_k$ (k>0) denotes a level-k DCell constructed by combining n+1 servers in $DCell_0$.
- A $DCell_0$ has n (n<8) servers directly connected by a commodity switch
- Drawbacks: Long communication paths, many required NICs, and increased cabling costs



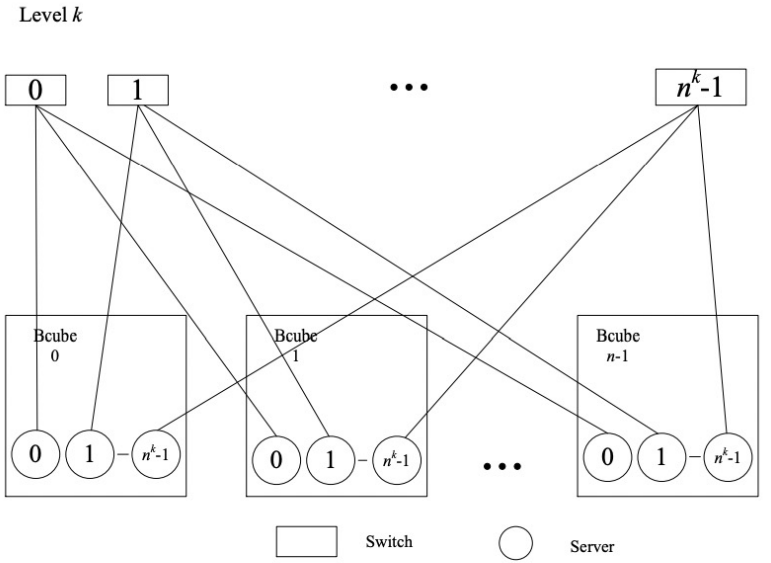


Hybrid architectures: BCube

- A hybrid and cost-effective architecture that can scale up through recursion.
- It provides high bisection bandwidth and graceful degradation of throughput.
- It uses BCube as a building block, which consists of *n* servers connected to an *n*-port switch.
- A $BCube_k$ (*k*>0) is constructed with *n* $BCube_{(k-1)}$ s and n^k *n*-port switches.
- In a $BCube_k$ there are $n^{(k+1)}$ *k*+1-port servers and *k*+1 layers of switches.
- Drawbacks: limited scalability and high cabling costs, many required NICs.



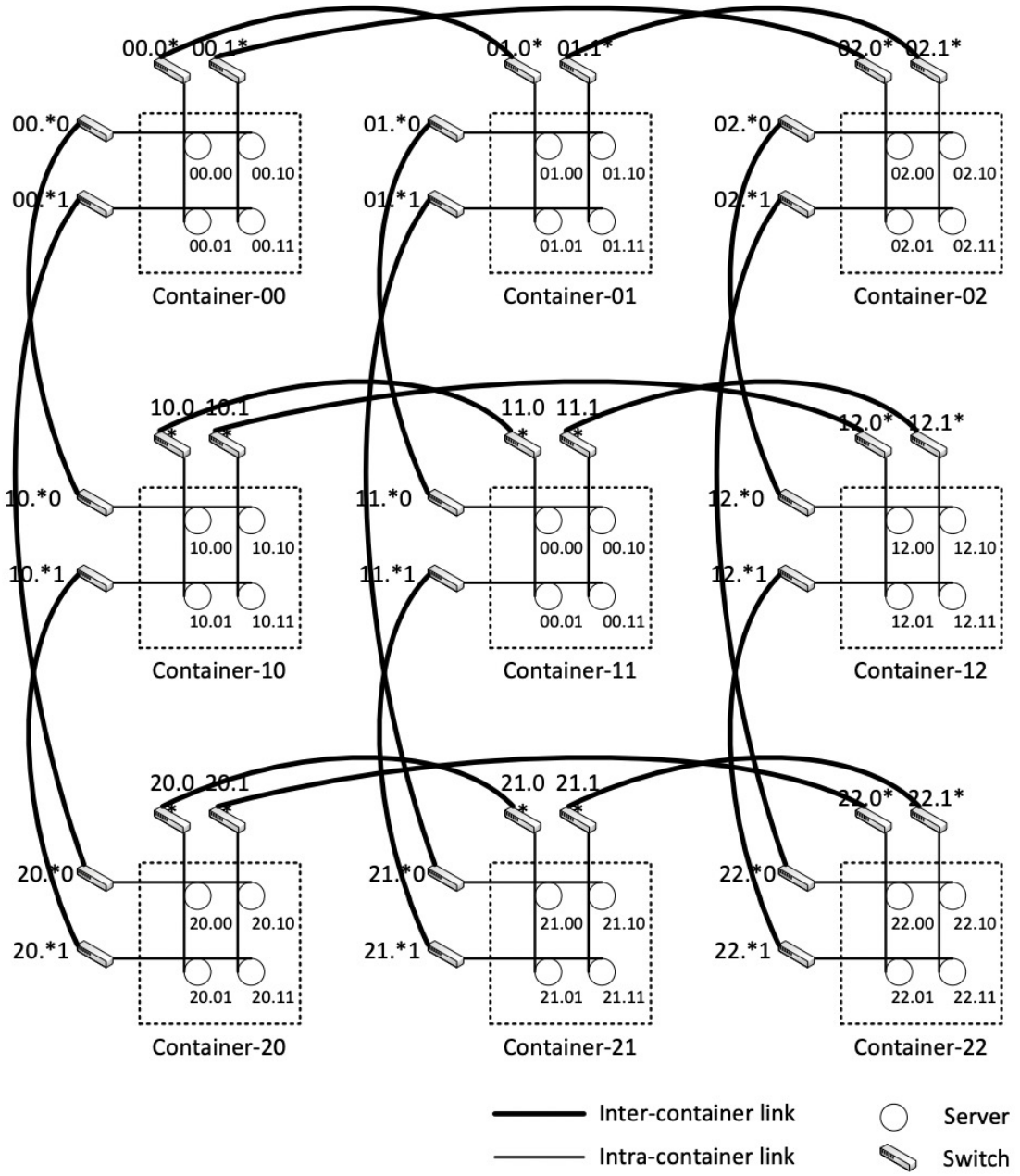
(a) Request phase.



(b) Grant phase.

Hybrid architectures
MDCube

- It is designed to reduce the number of cables in the interconnection of containers
- Each container has an ID mapped to a multidimensional tuple.
- Each container is connected to a neighbor container with a different tuple in one dimension.
- There are two types of links: Intra-container and high-speed inter-container links



The interplay of storage and networking technology

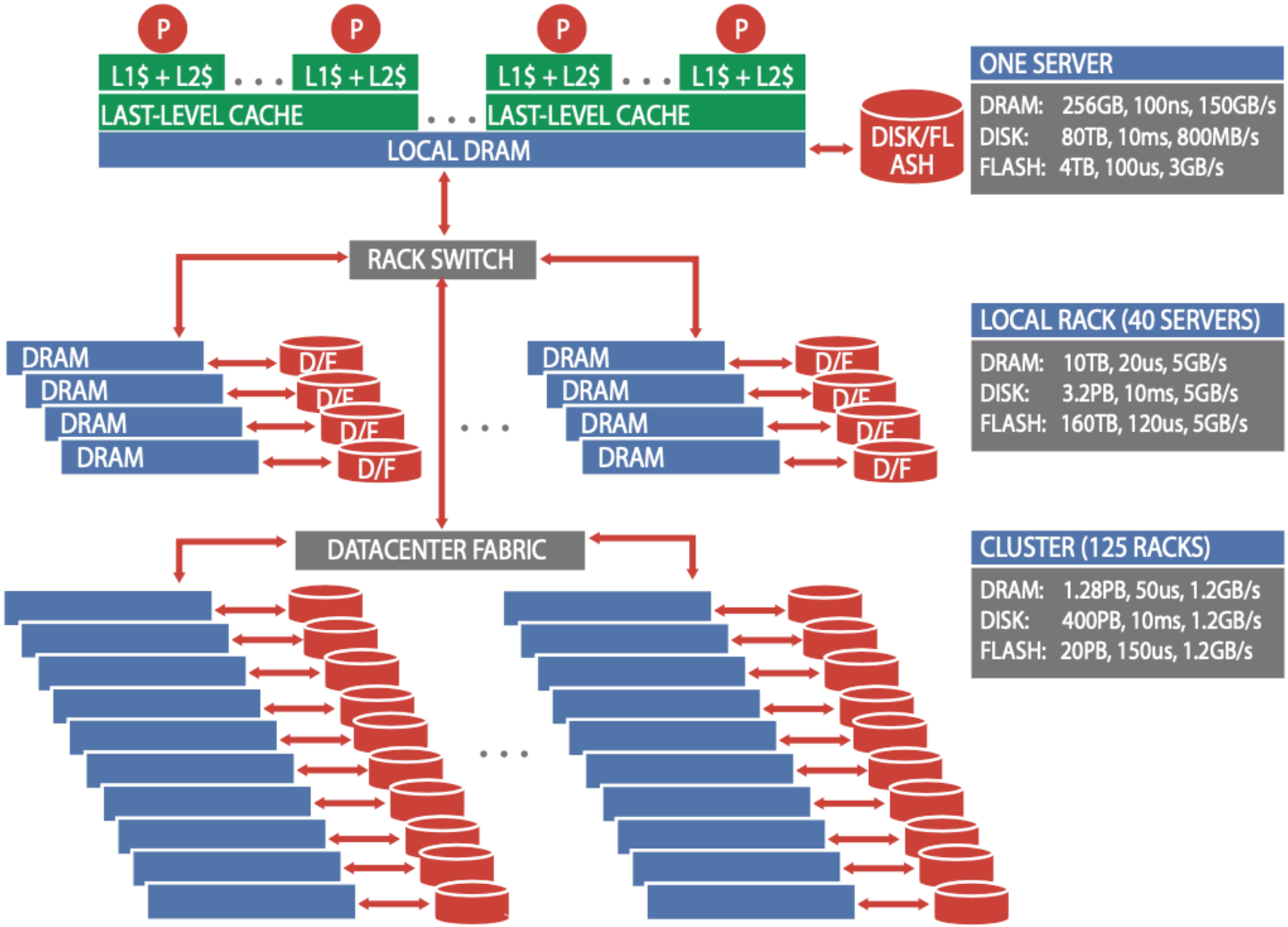
Balanced design

- The success of WSC distributed storage systems can be partially attributed to the evolution of data center networking fabrics:
  - **disk locality is no longer relevant in intra-data center computations.**
- This observation enables dramatic simplifications in the design of
  - ✓ distributed disk-based storage systems
  - ✓ utilization improvements,

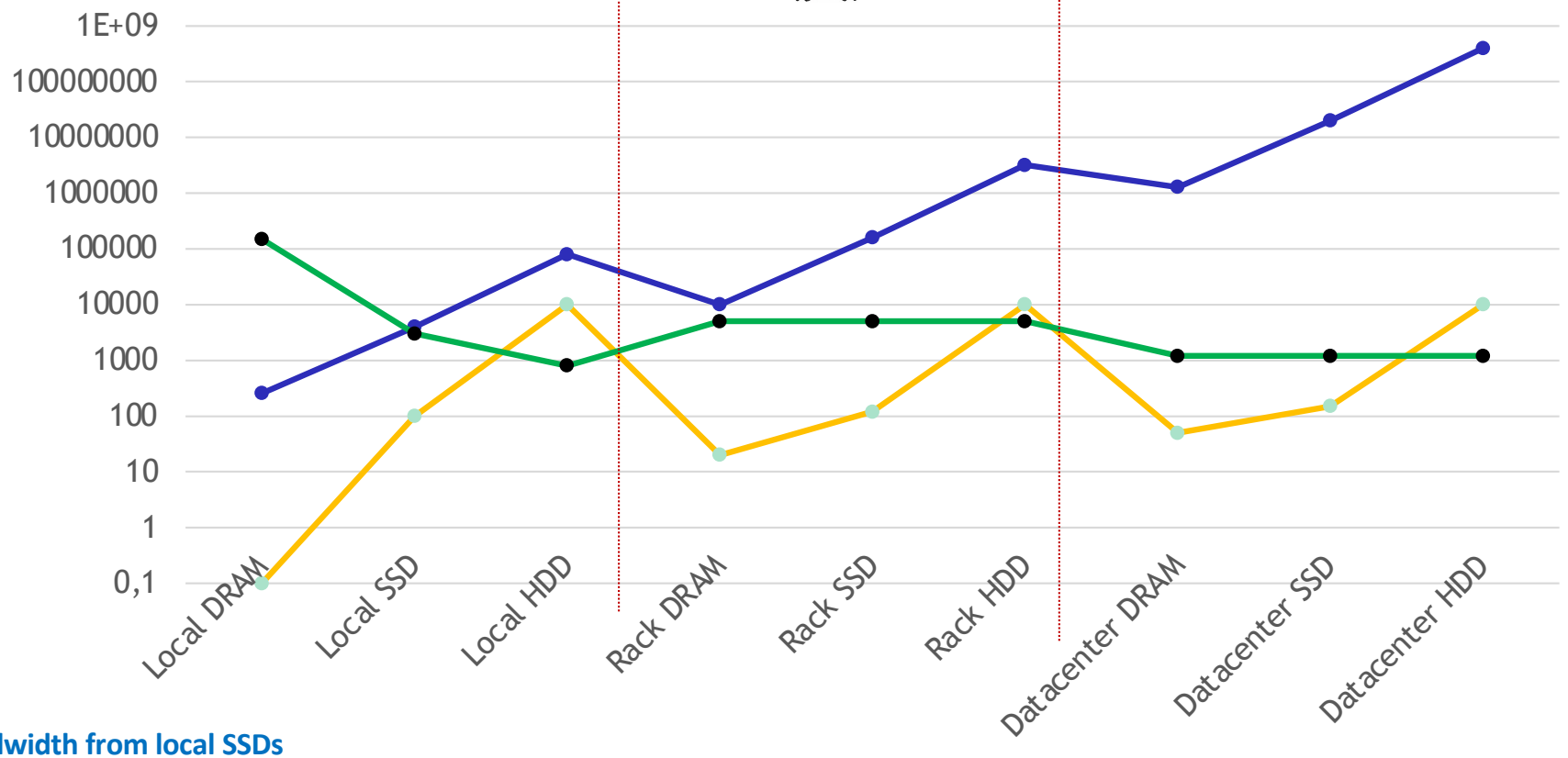
  since any disk byte in a WSC facility can, in principle, be utilized by any task regardless of their relative locality

Computer architects are trained to solve the problem of finding the right combination of performance and capacity from the various building blocks that make up a WSC

Storage hierarchy of a WSC _ (Part2)



Quantifying Latency, Bandwidth and Capacity _ (Part3)



The bandwidth from local SSDs is 3 GB/s, whereas the one from off-rack SSDs is 1.25 GB/s

Total disk storage in the cluster is more than one million times larger than local DRAM.

- **A large application** that requires servers in many racks to operate **must deal effectively with large discrepancies in latency, bandwidth, and capacity**.

- These discrepancies are much larger than those seen on a single machine, making it more difficult to program a WSC:
  - **A key challenge for architects of WSCs** is to smooth out these discrepancies in a cost-efficient manner.
  - **A key challenge for software architects** is to build SW infrastructure and services that hide this complexity