NEW YORK INSTITUTE OF TECHNOLOGY

INCS 775
Data Center Security

Data Center Network Architecture

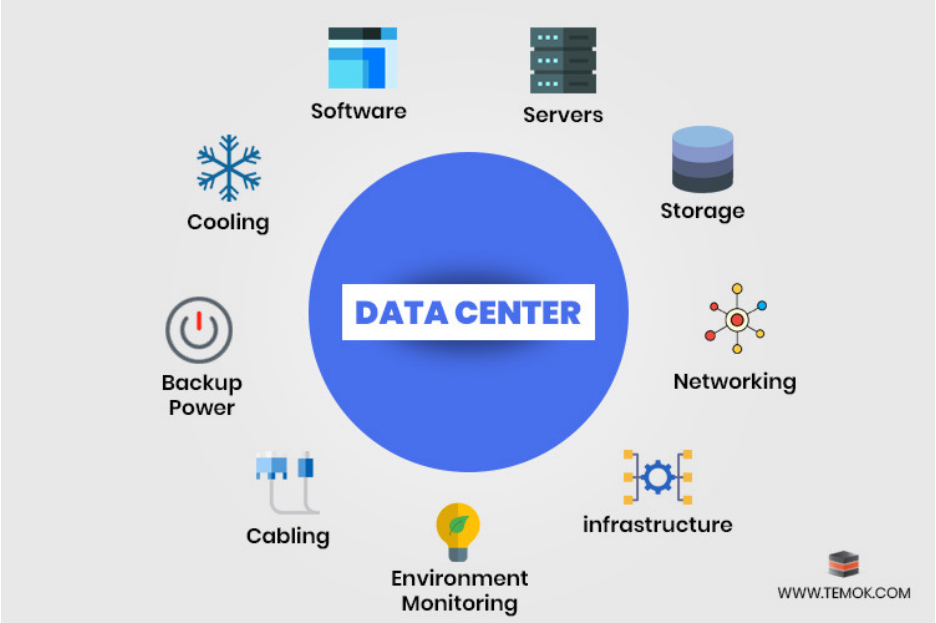
Dr. Zakaria Alomari
zalomari@nyit.edu

Today's Objectives

- ❑ Background of current DCN Architectures
- ❑ Desired properties in a DC Architecture
- ❑ Three-Layer Topology
- ❑ Clos-Based Networks
- ❑ Fat-tree Topology
- ❑ BCub Topology
- ❑ DCell Topology
- ❑ Comparing Data Center Topologies

Data Centre Component

- **A Data Center consists of:**
  - Servers
  - Networking
  - Storage
  - Software
  - Cabling Infrastructure
  - Power Infrastructure
  - Cooling Infrastructure
  - Backup Power
  - Environment Monitoring
  - Physical Security



Data Centre Component

❑ A Data Center consists of:

- **Servers:**
  - These are powerful computers responsible for processing and storing data.
  - Servers can host applications, websites, and databases.
  - They form the core of the data center, providing the computing power for various tasks.

- **Networking:**
  - This refers to the systems that connect the servers and other equipment in the data center.
  - It includes routers, switches, firewalls, and other devices that enable communication between servers and allow external access to the data center (e.g., internet connectivity).

Data Centre Component

- **Storage:**
  - This is the part of the data center where data is stored.
  - It includes hard drives, solid-state drives (SSDs), and network attached storage (NAS) devices.
  - Storage systems are designed for scalability, ensuring that large volumes of data can be saved, accessed, and retrieved quickly and efficiently.

- **Software:**
  - The software in a data center includes the operating systems, virtualization tools, management software, and applications that run on the servers.
  - These tools help manage workloads, automate tasks, and ensure that the servers and storage are operating efficiently.

Data Centre Component

- **Cabling infrastructure:**
  - Data centers rely heavily on cables to connect servers, storage systems, and network devices.
  - These cables typically include power cables, Ethernet cables for data transmission, and fiber-optic cables for high-speed data connections.

- **Power infrastructure:**
  - In a data center ensures a continuous and reliable power supply to all systems. It includes:
    - **Backup Generators**: Offer long-term power if utility power is lost.
    - **Power Distribution Units (PDUs)**: Distribute power to servers and equipment.
    - **Power Monitoring**: Tracks power usage and ensures system health.

Data Centre Component

- **Cooling infrastructure:**
  - Data centers generate a lot of heat due to the high number of servers and other equipment.
  - Cooling systems (such as air conditioning, liquid cooling, and fans) are critical for maintaining an optimal temperature, ensuring that equipment does not overheat, which could lead to hardware failures or performance issues.

- **Backup Power:**
  - Ensures uptime during power failures using UPS (Uninterruptible Power Supply) and generators.
  - Reduces the risk of data loss and service disruptions.

Data Centre Component

- **Environment Monitoring:**
  - Tracks temperature, humidity, and airflow to prevent overheating and system failures.
  - Use IoT sensors, alarms, and automated management systems.

- **Physical security:**
  - It protects data centers from unauthorized access, theft, and disasters through:
    - **Access Control –** Biometric authentication, keycards, and restricted entry.
    - **Environmental Protection –** Fire suppression, earthquake, and flood protection.

Data Center (DC) Infrastructure and Connectivity

- **Internal Connectivity**
  - Connects the DC to departments, branches, and remote offices within the organization.
  - Supports internal employees with access to corporate applications, databases, and collaboration tools.
  - Ensures smooth intranet operations for business efficiency.

- **External Connectivity**
  - Links the DC to the internet, third-party services, partners, and customers.
  - Supports cloud platforms, SaaS applications, and external integrations.
  - Enables customers and end-users to access web applications and online services.

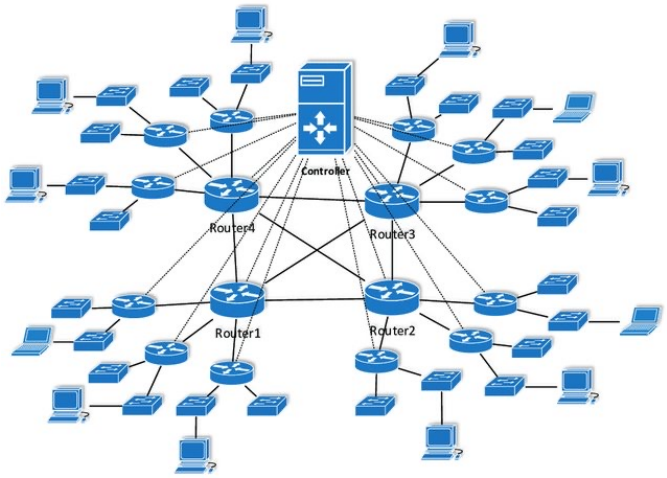
Data Center (DC) Infrastructure and Connectivity

- **Importance of Dual Connectivity**
  - **Seamless Data Exchange**: Ensures smooth communication between users and applications.
  - **Operational Efficiency**: Facilitates secure and efficient IT operations.
  - **Scalability and Security**: Supports growth while maintaining data protection.

- **Role of IP-based Networking in DCs**
  - IP protocol suite is the foundation for modern Data Center networking.
  - Hosts essential services like DNS, email, web servers, and cloud platforms.
  - Implements firewalls and security appliances for cyber protection.

Data Center Network

- Communications infrastructure refers to the framework that enables data transmission. It can be described by:
  - **Topology:** The network layout (e.g., Star, Mesh, Ring)
  - **Routing / Switching equipment:** Devices like routers and switches that direct traffic.
  - **Protocols:** Rules governing communication (e.g., TCP/IP, HTTP, BGP)



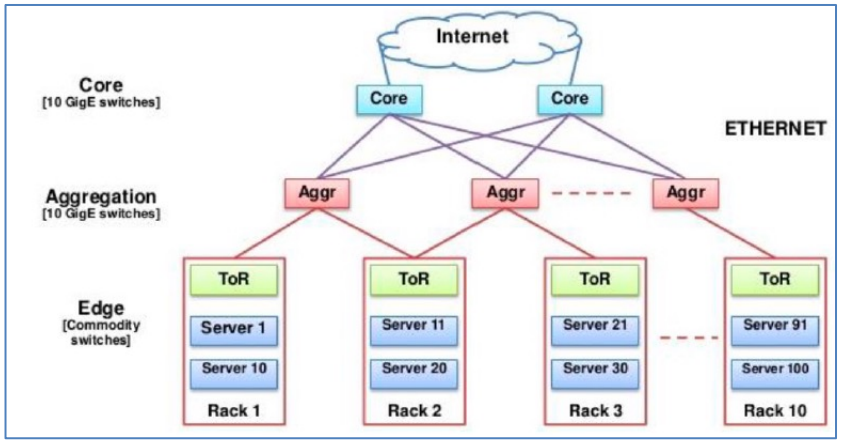
Data Center Network

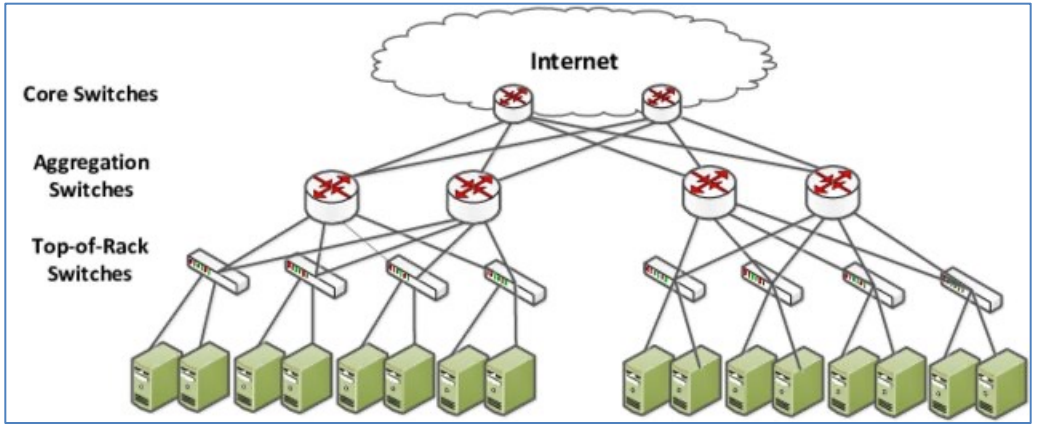
- A **network topology?**

  It is the physical and logical arrangement of nodes and connections in a network. Nodes usually include devices such as switches, routers and software with switch and router features. Network topologies are often represented as a graph.

Data Center Network

- A conventional three-layer data center network architecture consists of:
  - **Access layer (ToR witches)**: Connects servers and devices in racks to the network.
  - **Aggregation layer**: Aggregates traffic from the access layer and forwards it to the core layer.
  - **Core layer**: Acts as the high-speed backbone, connecting the data centre to external networks and ensuring inter-connectivity.





Data Center Network

- **Core layer:**
  - It provides the high-speed packet switching backplane for all flows going in and out of the data center.
  - The core layer provides connectivity to multiple aggregation modules and provides a resilient Layer 3 routed fabric with no single point of failure.

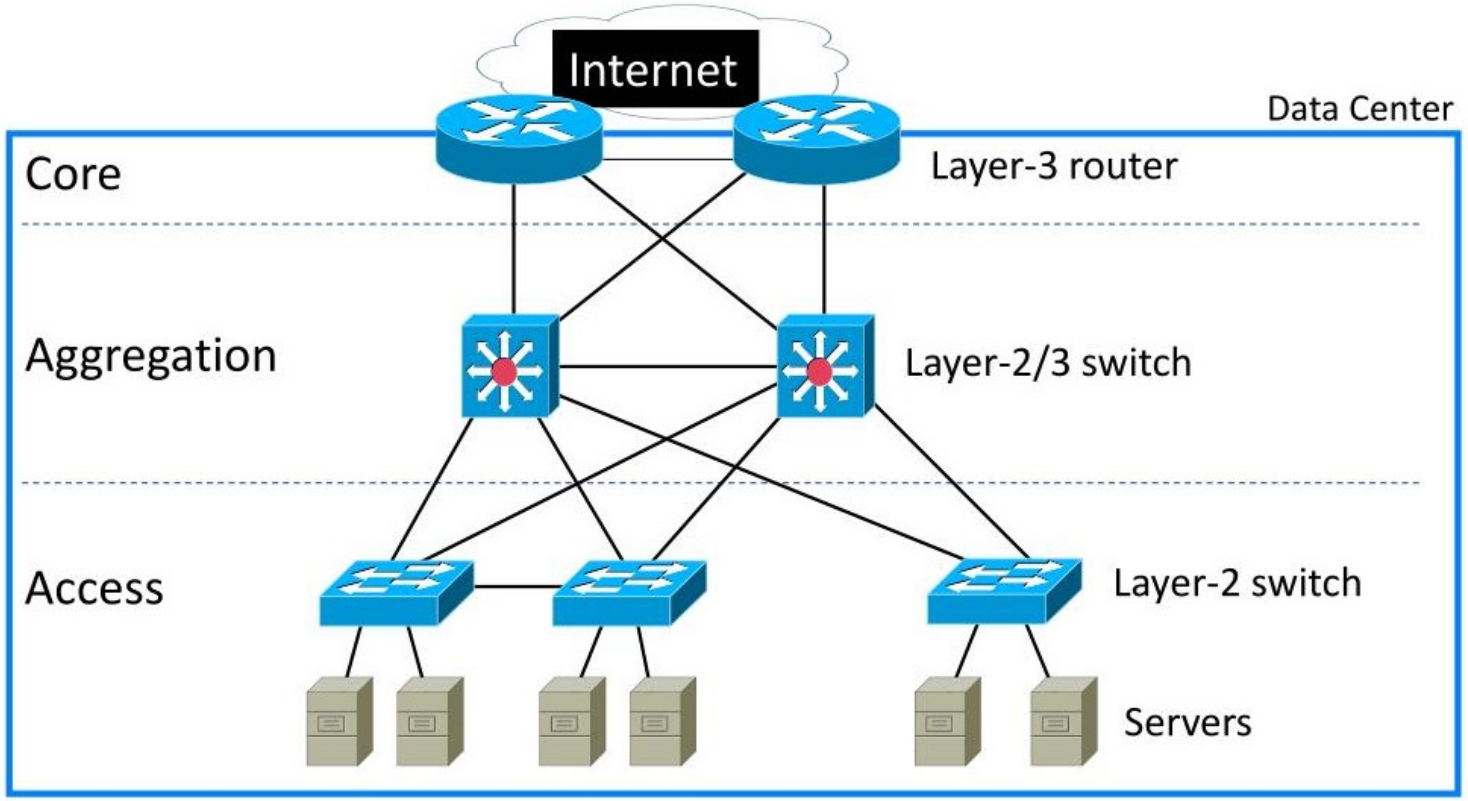
- **Aggregation layer:**
  - It aggregates the uplinks from the access layer to the data center core. This layer is the critical point for control and application services.

Data Center Network

- **Access layer:**
  - It is the last layer of three-tier architecture of a datacenter. The actual servers are connected to this layer.
  - The access layer communicates with its upper layer using several switches (like Layer 2 and Layer 3) and hubs.

Data Center Network

Common Data Center Topology



Data Center Network

- **Layer 2 – Data Link Layer:**
  - This layer is responsible for the node-to-node delivery of data packets between directly connected nodes in a network.
  - It deals with physical addressing, error detection, and flow control.
  - The most common protocol operating at this layer is **Ethernet.**
  - **Switches** operate at layer 2, making forwarding decisions based on **MAC addresses**.

Data Center Network

- **Layer 3 – Network Layer:**
  - This layer is responsible for routing packets from the source to the destination across multiple nodes.
  - It provides logical addressing, which helps in uniquely identifying devices on a network.
  - The most common protocol used for this purpose is **IP (Internet Protocol).**
  - Routing devices such as **routers** operate at Layer 3. They make forwarding decisions based on **logical addresses (IP addresses)**.

Oversubscription

- **Network oversubscription** occurs when the aggregate bandwidth of the network's connections **exceeds** the capacity of the core network elements, such as switches or routers.

- Here's a simplified example to illustrate **network oversubscription:**

  - Let's say you have a data center with servers connected to a switch (Access), and that switch is connected to another switch (Aggregation), which in turn connects to a router (Core), and finally, the router connects to the internet. Each of these devices has a **maximum capacity of 1 Gbps** (Gigabit per second).

Oversubscription

- Now, let's assume you have **10 servers**, each capable of transmitting data at **1 Gbps**.

- In a perfectly non-oversubscribed network, the aggregate capacity of all servers (10 Gbps) should match the capacity of the switch (10 Gbps), the capacity of the router (10 Gbps), and the internet connection (10 Gbps), ensuring no congestion.

- However, let's introduce oversubscription.
  - The switch connecting the servers can only handle **5 Gbps** of traffic *due to budget constraints*.
  - This means that while each server can still theoretically transmit at **1 Gbps**, the switch can only handle **5 Gbps** of incoming data.
  - So, if all servers try to transmit at their maximum capacity simultaneously, the switch becomes overwhelmed, leading to **congestion** and **dropped packets**.

Oversubscription

- **In previous example**, the network is oversubscribed by a factor of **2:1** at the switch level (**10 Gbps aggregate server capacity** vs. **5 Gbps switch capacity**).

- This oversubscription creates a potential bottleneck in the network, where traffic congestion and performance issues can occur, especially during peak usage times.

- To **mitigate oversubscription**, network administrators can either upgrade network components to handle higher capacities, implement Quality of Service (QoS) policies to prioritize traffic, or employ traffic shaping techniques to regulate bandwidth usage.

Oversubscription

❑ Example of Network Oversubscription

- **Scenario Setup**
  - **Servers:** 1000 servers, each with a 1 Gbps NIC (Network Interface Card).
  - **ToR Switches (Access Layer):** Each ToR switch connects 20 servers, with each ToR switch having 20 x 1 Gbps = 20 Gbps total server-facing bandwidth.
  - **Aggregation Layer:** Each aggregation switch connects 10 ToR switches. Each ToR switch uplinks to the aggregation switch with a 10 Gbps connection.
  - **Core Layer:** Each core switch connects 5 aggregation switches, each with a 10 Gbps uplink.

Oversubscription

- **Bandwidth Calculation**
  - **Access Layer:**
    1. Each ToR switch has 20 servers x 1 Gbps = 20 Gbps total server-facing bandwidth.
    2. Uplink from each ToR to Aggregation is 10 Gbps.
    3. **Oversubscription Ratio:** $\frac{\boldsymbol{20\ Gbps\ (downlink)}}{\boldsymbol{10\ Gbps\ (uplink)}} = 2:1$

Oversubscription

- **Aggregation Layer:**
  1. Each aggregation switch connects 10 ToR switches x 10 Gbps uplink = 100 Gbps total uplink from access layer.
  2. Uplink from each aggregation switch to Core is 10 Gbps.
  3. **Oversubscription Ratio:** $\frac{\mathbf{100}\ \boldsymbol{Gbps}\ (\boldsymbol{downlink})}{\mathbf{10}\ \boldsymbol{Gbps}\ (\boldsymbol{uplink})} = 10:1$

Oversubscription

- **Core Layer:**
  1. Each core switch connects 5 aggregation switches x 10 Gbps uplink = 50 Gbps total uplink from aggregation layer.
  2. External uplink or backbone connectivity is not specified, *so let's assume it has 10 Gbps as an example*.
  3. **Oversubscription Ratio:** $\frac{\mathbf{50\ Gbps\ (downlink)}}{\mathbf{10\ Gbps\ (uplink)}} = 5:1$

Oversubscription

- Understanding Oversubscription Impact
  - **Access to Aggregation:**
    - Servers have a (2:1) oversubscription at the ToR switch level.
    - This means that in the **worst case**, only half of the servers' traffic can be simultaneously sent to the aggregation layer without congestion.
  - **Aggregation to Core:**
    - The oversubscription ratio is higher (10:1) indicating a potential bottleneck at the aggregation layer.
    - Traffic from servers may experience significant congestion and latency as it moves up to the core layer.

Oversubscription

- **Core Layer:**
  - The core layer, with an oversubscription ratio of (5:1), suggests that there may be contention for bandwidth as data moves towards external networks or across different aggregation switches.

- Network administrators need to carefully monitor network usage, analyze traffic patterns, and adjust oversubscription ratios to ensure that the network can handle the demands placed on it without sacrificing performance.

Oversubscription

- **Network Oversubscription** occurs when incoming traffic (ingress) exceeds the available outgoing bandwidth (egress), potentially causing congestion.

- **Bandwidth** is the maximum data transfer rate in a network, which can be set differently for incoming (ingress) and outgoing (egress) traffic to control data flow efficiently.

Oversubscription

- Oversubscription in network design refers to the ratio of total downlink bandwidth (*access layer to distribution switch*) to total uplink bandwidth (*distribution switch to core layer*).

- A higher ratio means more bandwidth is available at the access layer than the core can handle, relying on **the assumption that not all devices will use their maximum bandwidth simultaneously**.

- For example, with 48 Gbps downlink and 10 Gbps uplink, the oversubscription ratio is 4.8:1. Too much oversubscription can lead to congestion and performance issues.

Oversubscription

- In a ***leaf-and-spine network***, each access switch has 48 ports, each supporting 1 Gbps, totaling 48 Gbps of potential traffic. To avoid network bottlenecks, the connection between the core switch and each access switch must be at least 48 Gbps, ensuring it can handle the full traffic load from the access switches.

Oversubscription

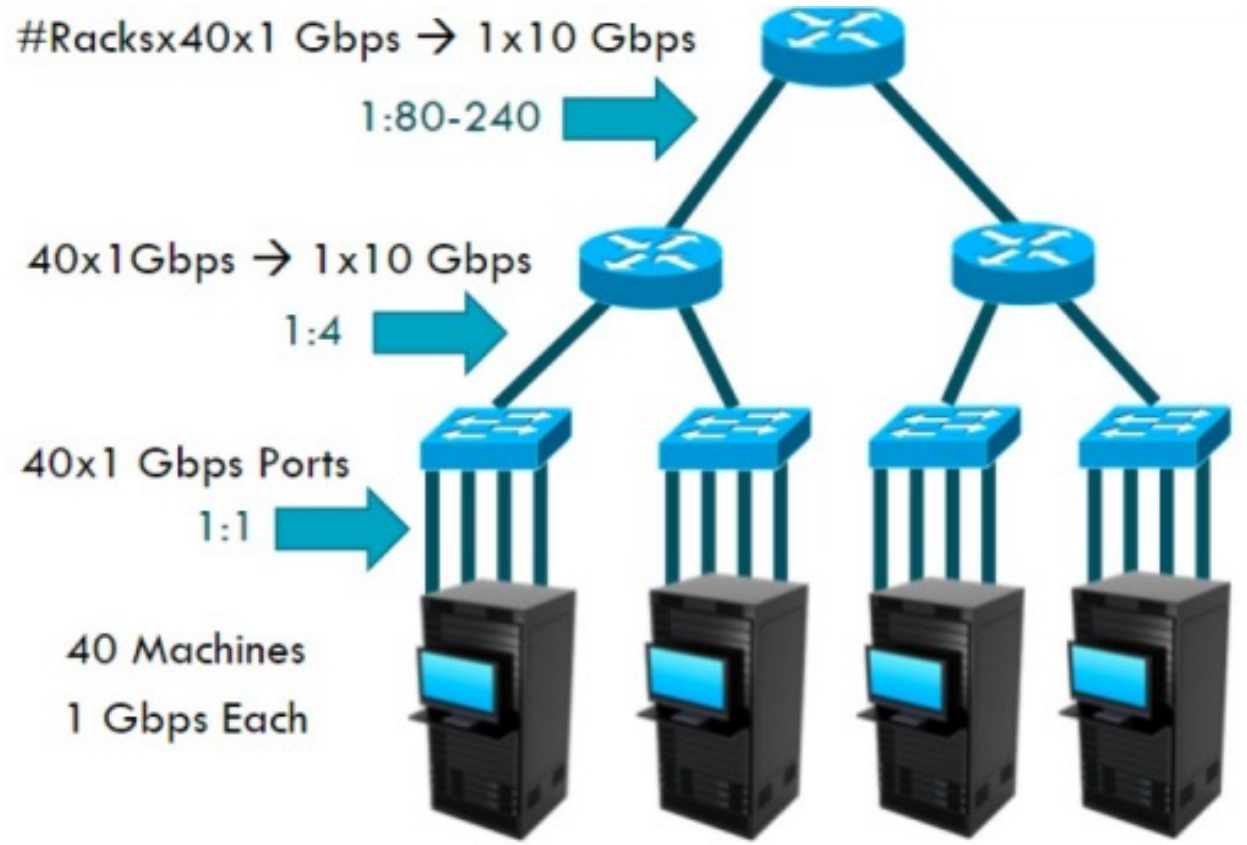
- Most often, this would be wasteful, because in practice you will never encounter a situation ***where all ports receive traffic at their maximum rate at the same time***.

- **Over-subscription:** Access switch with 48 x 1 Gbps ports and a 10 Gbps uplink results in a 4.8:1 over-subscription (48 Gbps / 10 Gbps).

- **Link Aggregation (LAG):** Using 2 x 10 Gbps links in a LAG provides a 20 Gbps uplink.

- **New Over-subscription Ratio**: With LAG, the over-subscription is reduced to 2.4:1 (48 Gbps / 20 Gbps).

Oversubscription



Oversubscription

- **When we use it and when not to use?**
  - As you can see it is almost always used when you have *several switch layers*.

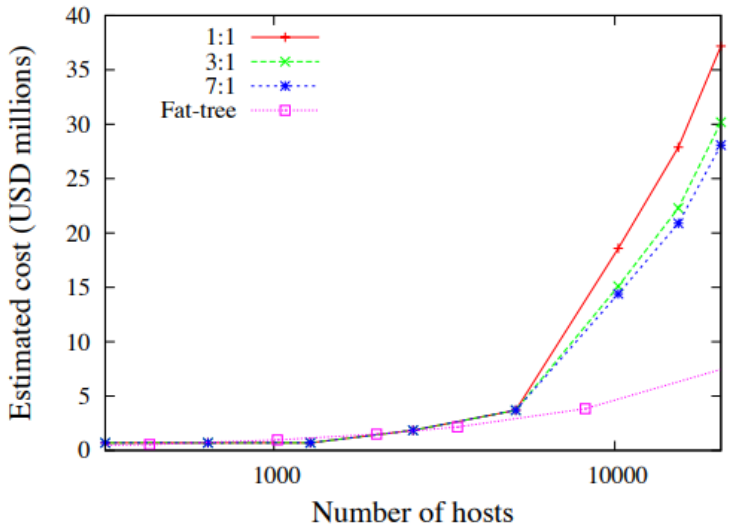
- **You don't use it:**
  - When you have only one switch layer (very small networks)
  - When you have very specific requirements and want the full bandwidth available on all ports at any time (and enough money to do so)

Oversubscription

- **How do we calculate these Value?**
  - As in the example above, the over-subscription ratio is the ratio between the upstream bandwidth and the downstream capacity.
  - As for how to decide which final ratio to attain when designing / upgrading a network, it can be tricky.
  - But the correct values for a given network *highly depend on the traffic pattern*.
  - For existing network, *a close monitoring of the bandwidth used on each port should give enough insight*. You can also use ***NetFlow / sFlow*** *to analyze further what use the bandwidth.*
  - When designing a new network you need to *assess the expected traffic.*

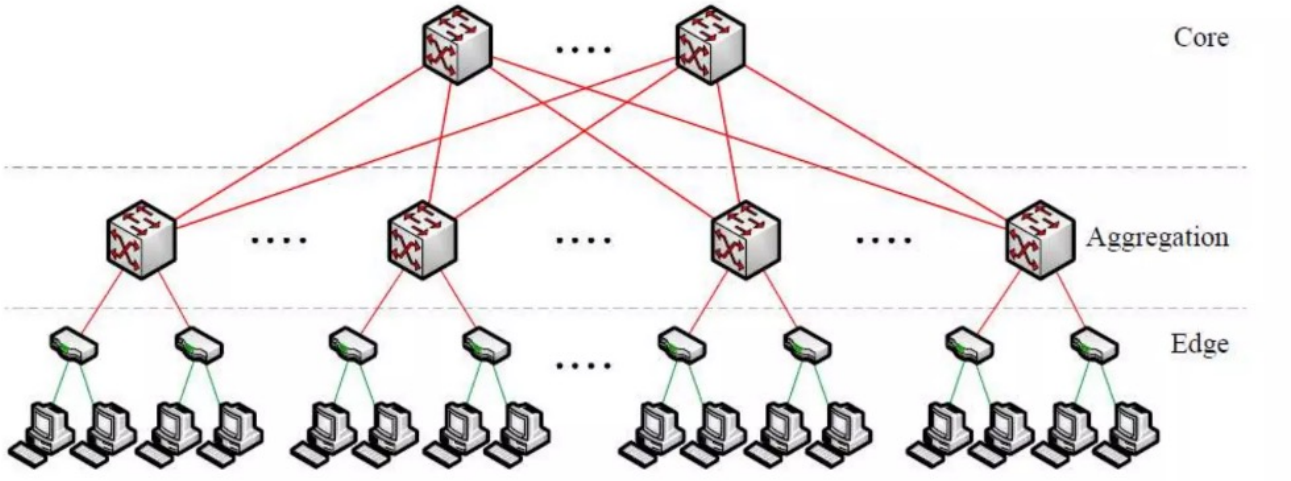
Cost

- We assume a cost of $7,000 for each 48-port GigE switch at the edge
- and $700,000 for 128-port 10 GigE switches in the aggregation and core layers.
- We do not consider cabling costs in these calculations.



1:1 there are no network bottlenecks; thus, leaf switches forward traffic with no packet loss.

Problem With common DC topology



- Single point of failure
- Over-subscription of links higher up in the topology

Well-known Low-cost Data Center Topologies

- Conventional Data Center network architecture, employing high-end equipment, suffers from prohibitive costs for large network sizes.
  - Traditional DCN architecture associated with high-end components suffers from cost constraints for large scale networks

- High-end equipment
  - Refers to powerful, often expensive hardware components that are designed for handling large-scale computing tasks, high-speed data processing, and robust network operations.
  - These components are typically required in traditional data center setups to meet performance, scalability, and reliability demands.

Well-known Low-cost Data Center Topologies

- Examples of such high-end equipment include
  - **High-Performance Servers**
    - These servers, often used as part of the compute layer, are equipped with powerful CPUs, large amounts of memory (RAM), and storage to handle significant processing tasks.
  - **Enterprise-Grade Switches and Routers**
    - These networking devices support high throughput, low latency, and advanced networking protocols.
    - Examples include **switches** that support 40G or 100G Ethernet or **routers** that can handle millions of packets per second.

Well-known Low-cost Data Center Topologies

- **Fat-Tree**, **BCube**, and **DCell** are scalable, fault-tolerant network topologies for data centers, using low-cost hardware.
- They offer modular designs that grow easily, providing high bandwidth and redundancy.
- **Low-cost commodity hardware** refers to standard, inexpensive equipment commonly available in the market, used to build networks or systems without requiring specialized or custom-built components. Example: Standard servers (e.g., Dell, HP, or Lenovo servers)

Well-known Low-cost Data Center Topologies

- Modern DCNs are more advanced and flexible than traditional networks, designed to handle rapidly growing data, traffic, and performance demands with **specialized, scalable architectures**.

- The term performance of DCN is not only limited to one factor rather it is a combination of various key values such as **scalability**, **cost**, **energy**, **latency** and **routing** etc. which are described as follows:

  1. Expandability: Due to rapidly growth in web applications and use of internet, Data centers have to occupy and control millions of servers in parallel. For example, Microsoft is controlling over 1 million servers in around 100 data centers worldwide.

Well-known Low-cost Data Center Topologies

2. Power Consumption: the annual power consumption of a data center of USA was predicted greater than 100 billion kWh in 2011, and the cost was approximately 7.4 billion.
   - Mainly, DCN works on maximum performance as they are high capacitated networks which *force them to consume more power to fulfil the requirement.*

3. High Bandwidth: Data centers networks should be designed to handle high capacity of traffic to fulfil the bandwidth requirement of various services such as Map Reduce and GFS. The main issue arises with the bandwidth is **oversubscription** which invites the problem of congestion.

Well-known Low-cost Data Center Topologies

4. Virtualization: Virtualization is the best method to make the full utilization of shared resources, hence results in increase the DCN performance.

5. Fault Tolerance: The term fault tolerance refers to the proper functioning of present task even after the occurrence of failures. It is observed that fault tolerance may be the reason of poor network performance at core and aggregation layer due to the lack of hardware connection.

Well-known Low-cost Data Center Topologies

- Divide DCN topologies:
  1. Switch centric refers to those where routing is controlled by switches only.
  2. Server Centric means routing is handled by servers.
  3. Hybrid which uses both switch and server as routing and transferring packets.

Well-known Low-cost Data Center Topologies

- The **conventional** DC topologies **Clos-Based Networks** and **Fat-tree** are switch-centric, *where only switches forward packets*.

- **BCube** and **DCell** are server-centric topologies, where *servers also participate in packet forwarding*.

Clos Networks

- A **Clos network** is a multi-stage, non-blocking network topology commonly used in telecommunication and parallel computing to optimize switch fabric efficiency.

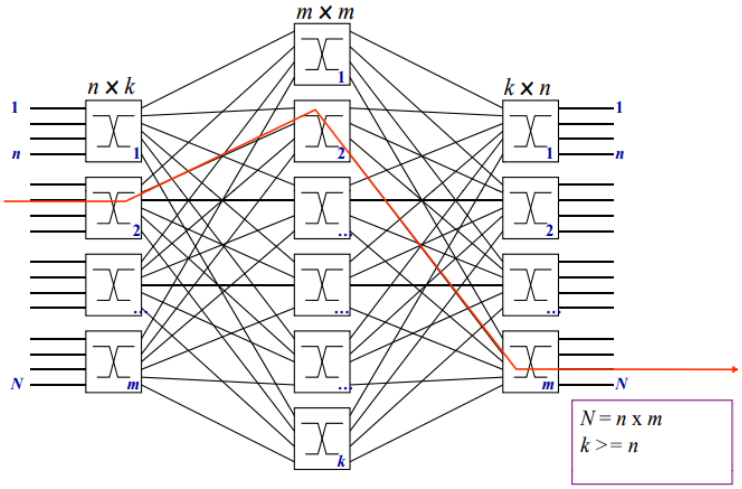
- It was introduced by Charles Clos in 1953 and provides a structured way to connect a large number of inputs to outputs with minimal hardware while ensuring scalability and redundancy.

Clos Networks

- **Clos networks** are categorized **into three or more stages of interconnected switching elements**:
  - **Input Stage** – Connects input devices to middle-stage switches.
  - **Middle Stage** – Provides interconnections between input and output switches.
  - **Output Stage** – Connects middle-stage switches to output devices.
- This topology ensures high bandwidth and avoids single points of failure.

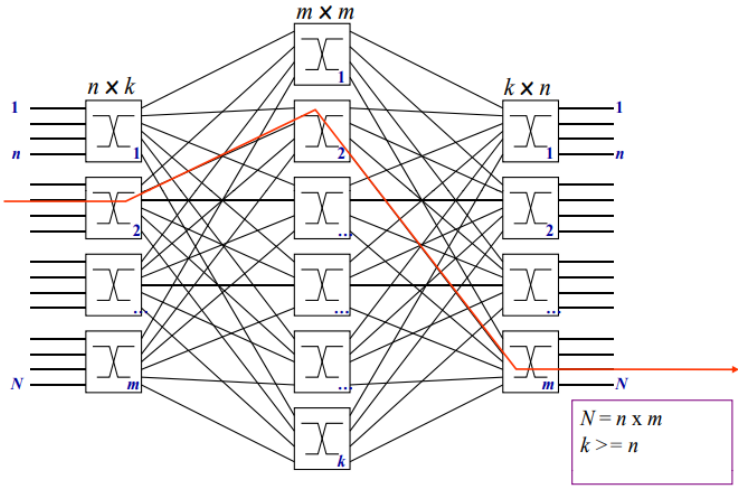
Clos Networks

- **The given figure represents a 3-stage Clos network with multiple switching elements at each stage. Here's how it is structured:**
  - First Stage (Input Stage)
    1. The leftmost switches are labeled as $\boldsymbol{n} \times \boldsymbol{k}$ (each handling $\boldsymbol{n}$ inputs and $\boldsymbol{k}$ outputs)
    2. These switches serve as the **entry points** for the network.
    3. Each input switch is connected to multiple middle-stage switches.



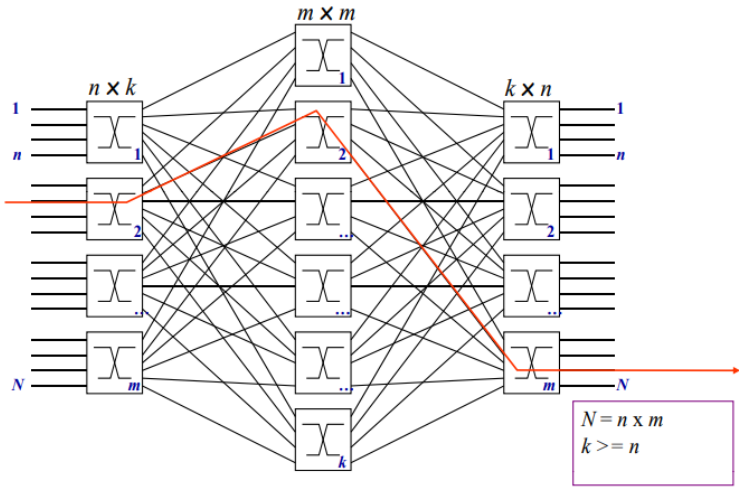
Clos Networks

- Second Stage (Middle Stage)
  1. The middle set of switches are labeled $\boldsymbol{m} \times \boldsymbol{m}$ (square switches, meaning they have an equal number of inputs and outputs).
  2. These switches facilitate cross-communication between the input and output stages.
  3. Every input switch is connected to multiple middle-stage switches, ensuring multiple routing paths.



Clos Networks

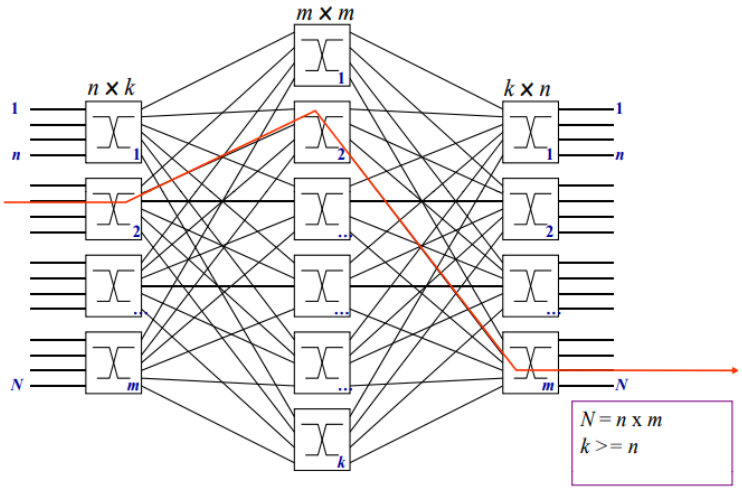
- Third Stage (Output Stage)
  1. The rightmost switches are labeled as $\boldsymbol{k} \times \boldsymbol{n}$ (each handling $\boldsymbol{k}$ inputs and $\boldsymbol{n}$ outputs).
  2. These serve as the **exit points** of the network, connecting to the final destinations.



- The red line in the figure represents a possible path of data flow through the network, demonstrating how a connection is routed from an input to an output.

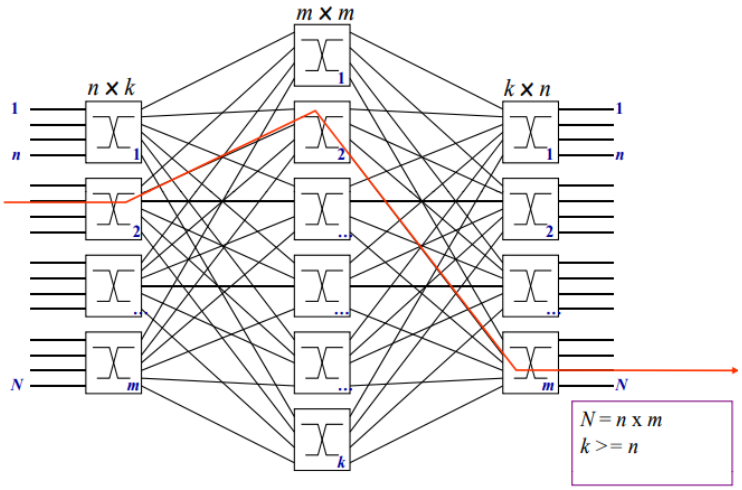
Clos Networks

- **Key Properties of the 3-Stage Clos Network:**
  - **Non-blocking**: The structure ensures that multiple connections can be established simultaneously without interference.
  - **Scalability:** Increasing the number of switches allows for expanding the network without degrading performance.
  - **Fault Tolerance:** Multiple paths exist between inputs and outputs, ensuring continued operation even if one path fails.



Clos Networks

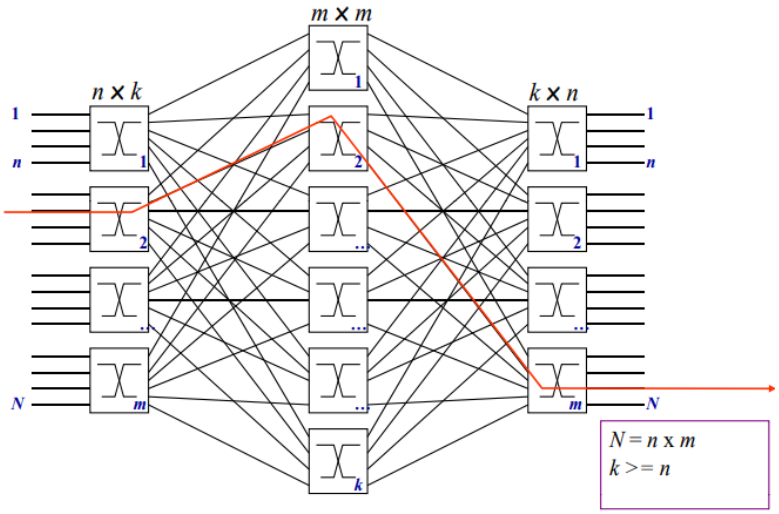
- **Key Properties of the 3-Stage Clos Network:**
  - **Efficient Resource Utilization:** Unlike a full crossbar switch (which would require a massive number of direct connections), the Clos network optimizes interconnections while maintaining full connectivity.
    - A **full crossbar switch** is a network switch where every input is directly connected to every output via a dedicated switching element.
    - It provides **non-blocking, high-performance switching** but is **expensive and hard to scale** due to the large number of crosspoints ($n \times m$).



Clos Networks

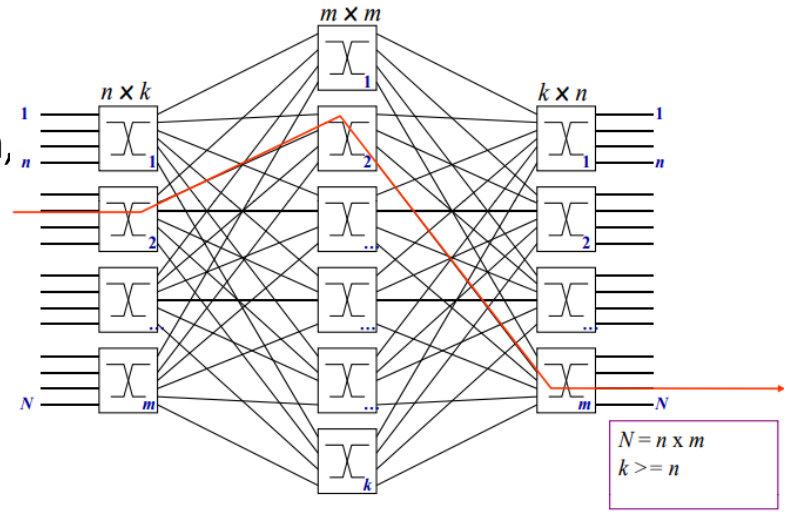
❑ Bipartite Matching in Clos Networks

A **Clos network** achieves a **bipartite matching** between input and output ports by appropriately configuring the switches in all stages, ensuring efficient and non-blocking communication.



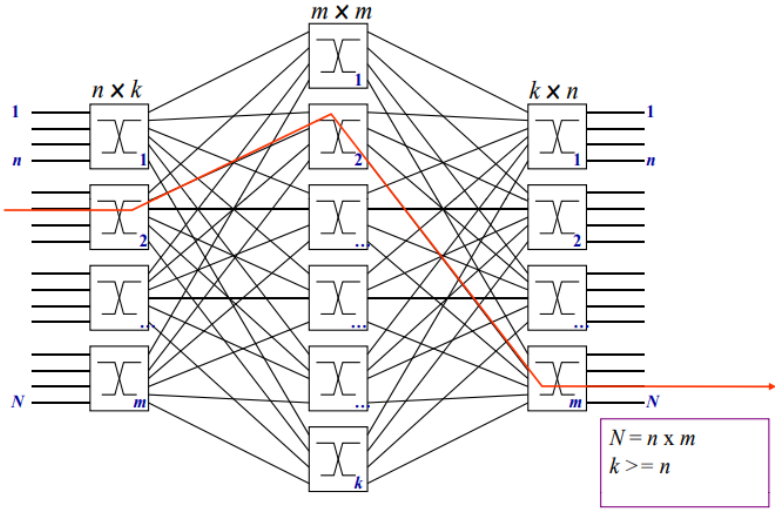
Clos Networks

- **In the given figure:**
  - n represents the number of input sources feeding into each of the m ingress-stage crossbar switches.
  - Each **ingress-stage switch** is connected **exactly once** to every **middle-stage switch**, ensuring a well-structured interconnection.
  - Similarly, each **middle-stage switch** is connected **exactly once** to every **egress-stage switch**, creating a balanced and distributed routing framework.



Clos Networks

- It can be demonstrated that when $\boldsymbol{k} \geq \boldsymbol{n}$, the Clos network operates as a **non-blocking switch**, similar to a **full crossbar**.
- This means that for any arbitrary input-output matching, it is always possible to configure an arrangement of paths through the middle-stage switches, ensuring seamless connectivity and avoiding contention.
- This structured connectivity enables efficient traffic distribution, scalability, and high performance in networking and switching applications.



Fat-Tree Based Networks

- Fat-tree networks were designed using a non-blocking folded-clos topology to mitigate challenges such as congestion due to oversubscription, increased failure rates, scalability limitations, and the high cost of *core switches*, *aggregation switches*, and *routers*.

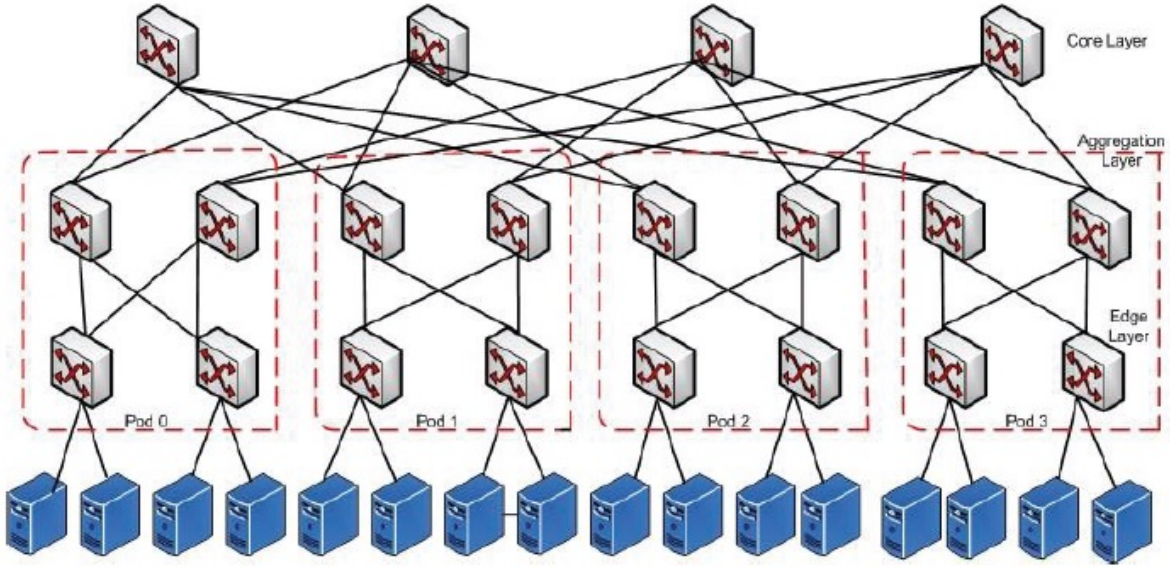
- As conventional multi-tier tree network, fat-tree also have three layers topology, i.e. **core layer**, **aggregation layer** and **edge layer**.

- Top of Rack (ToR) switches and aggregation switches are arranged in blocks knows as ***pods*** and pods are interconnected with the Core switches.

Fat-Tree Based Networks

- Every fat tree consist of ***k*** pods and contains $(k/2)^2$ core switches and $k^2/2$ aggregation and edge switches.

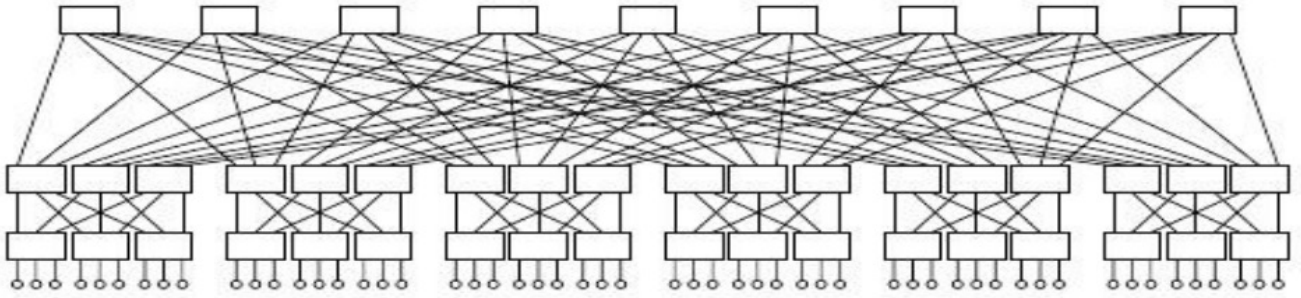
- A fat tree can support $k^3/4$ hosts or servers using $5k^2/4$ number of switches with $k^3/2$ links.



Fat-Tree Based Networks

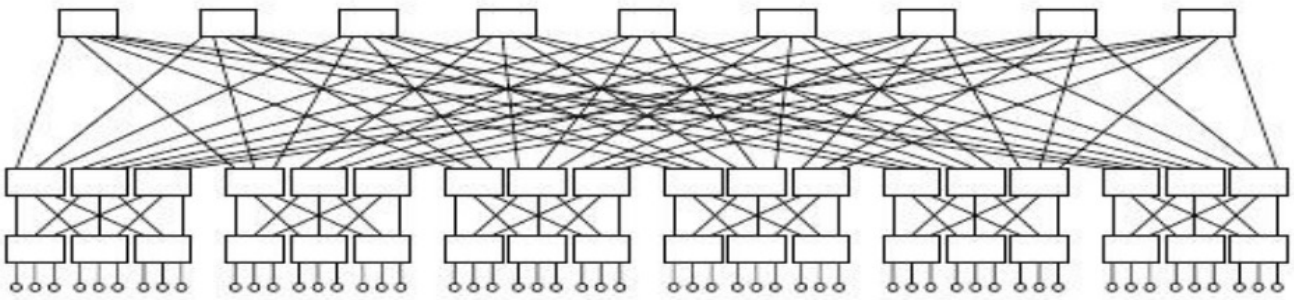
- Why Fat-Tree?
  - Fat tree has identical bandwidth at any bisections.
  - Each layer has the same aggregated bandwidth.

- Can be built using cheap devices with uniform capacity
  - Each port supports same speed as end host
  - All devices can transmit at line speed if packets are distributed uniform along available paths



Fat-Tree Based Networks

- **Great scalability: k-port switch supports $k^3/4$ servers(hosts).**
  - Each **k-port switch** contributes to a hierarchical structure where:
    - Each **edge switch** supports **k/2 servers**.
    - Each **pod** (group of interconnected switches) consists of **k switches**, and each pod connects **$k^2/4$ servers**.
    - The entire network, including all pods, can support **($k^3/4$) servers**.



Fat-Tree Based Networks

- **Overall number of core switches**: (k/2) *(k/2) = (k/2)^2
- **Overall number of servers**: k *(k/2)* (k/2)= (k^3)/4
- **Overall number of switches**: k* k + (k/2)^2= (k^2)*5/4
- **Overall number of links :** k* (k/2)* k + k* (k/2)^2 = (k^3)*3/4

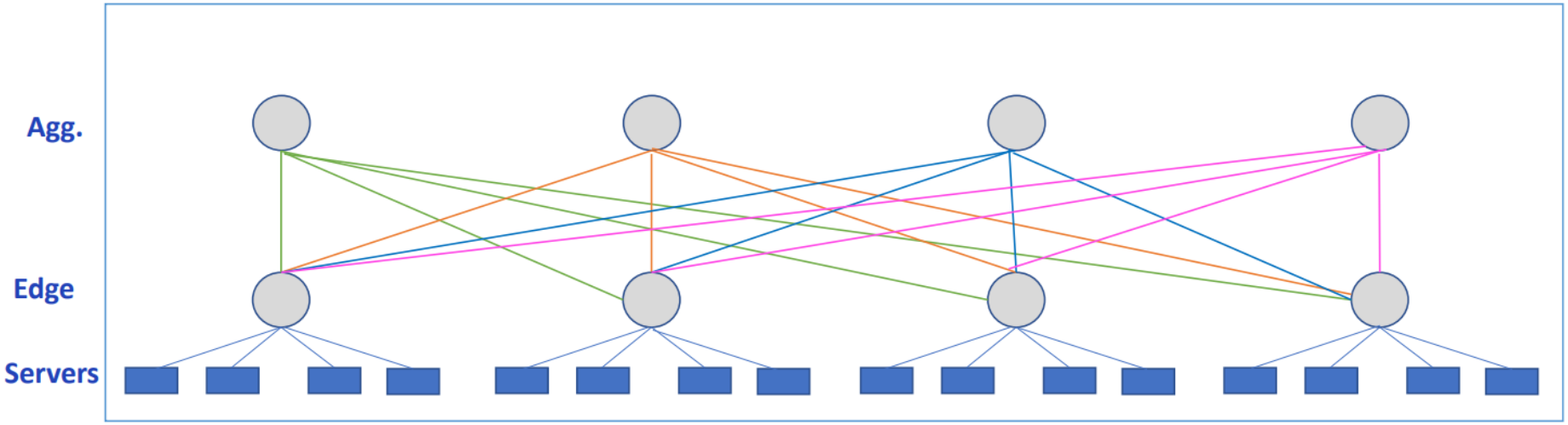
Fat-Tree Based Networks

K	Racks / Hosts / Servers	Switches	Links
4	16	20	48
8	128	80	384
16	1024	320	3072
24	3456	720	10368
32	8192	1280	24576
48	27648	2880	82944
…			
256	4194304	81920	12582912

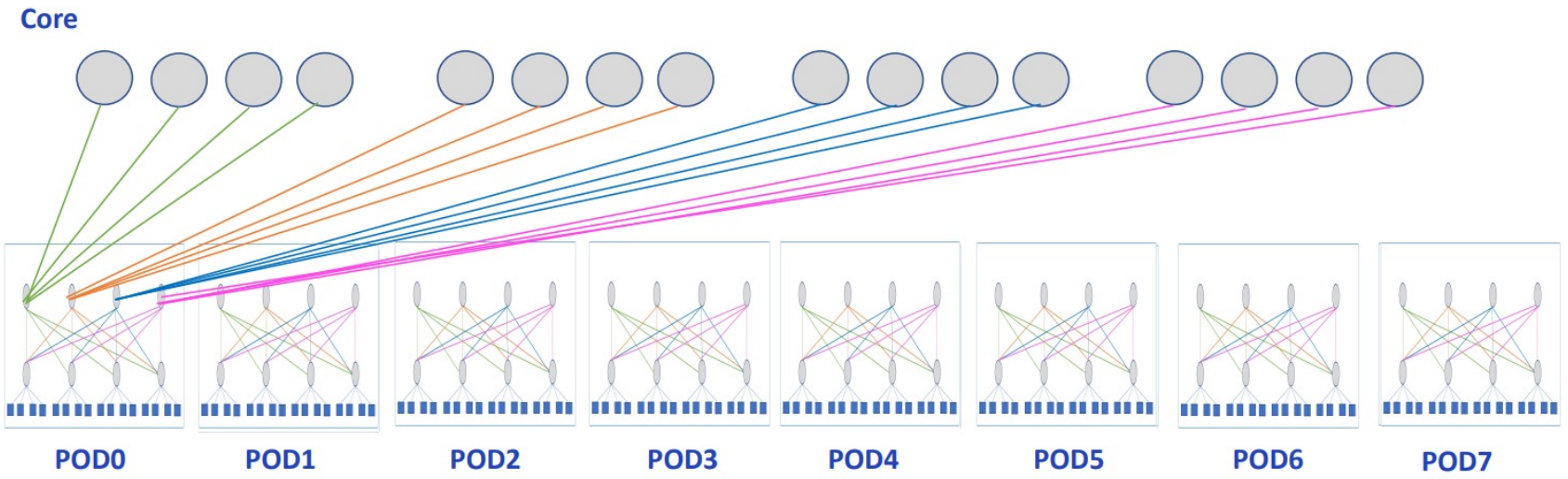
Design an 8-ary Fat tree data center

#PODs	8
#Edge switches	32
#Aggregation Switches	32
#Core Switches	16
#Servers	128

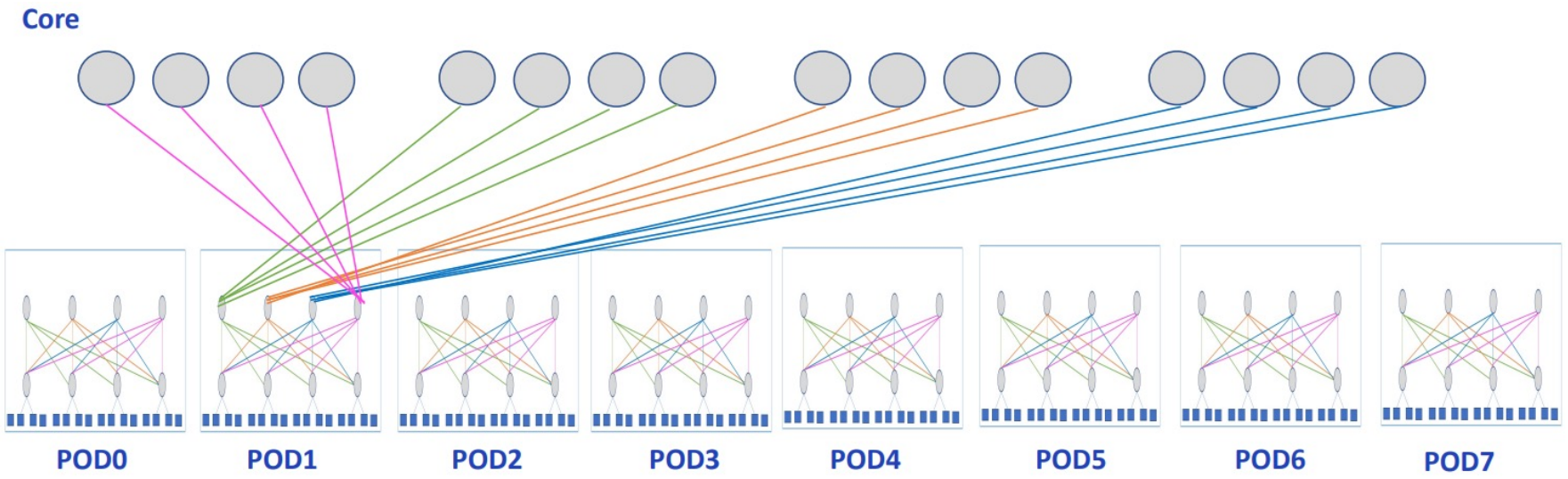
Design an 8-ary Fat tree data center



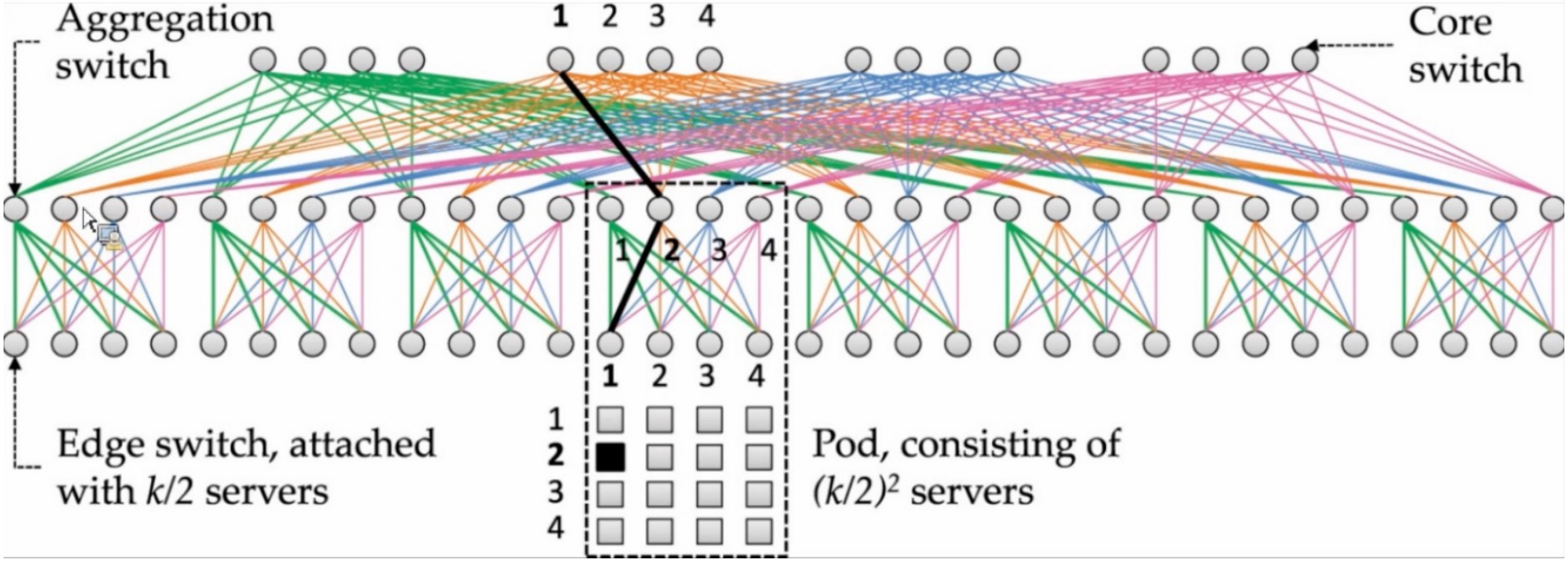
Design an 8-ary Fat tree data center



Design an 8-ary Fat tree data center



Non-conventional Design

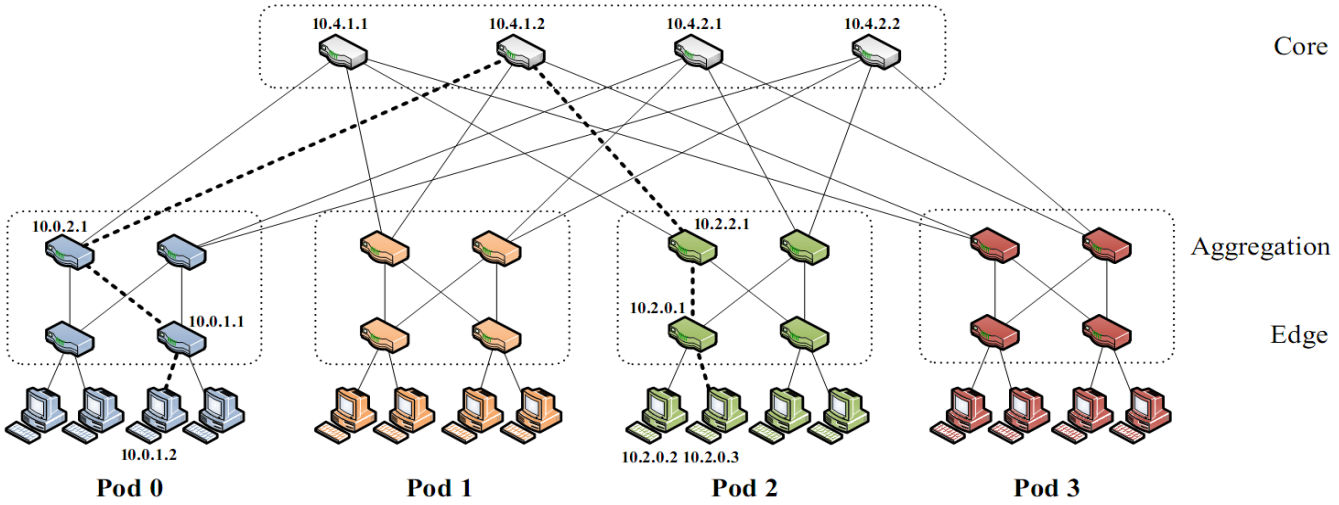


Fat-Tree Based Networks

- Advantages
  - 1:1 Oversubscription ratio
  - Low Cost: Commodity switches
    - Commodity switches refer to off-the-shelf network switches that are mass-produced and widely available in the market at relatively low cost.
    - These switches are typically designed for general-purpose networking applications and are not specialized or customized for specific industries or use cases.

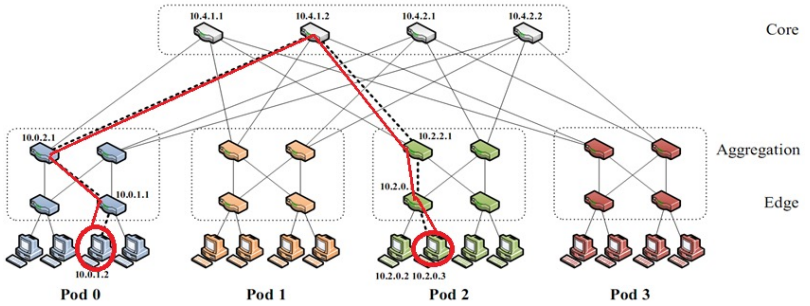
What is the length of shortest path in fat-tree?

- The length of the shortest path in a Fat-tree network is typically 2 hops when the source and destination are on different edge switches.
- The path length can be longer in more complex scenarios, but it generally remains $\boldsymbol{O(Log\ N)}$in terms of hops, where $\boldsymbol{N}$ is the number of nodes in the network.

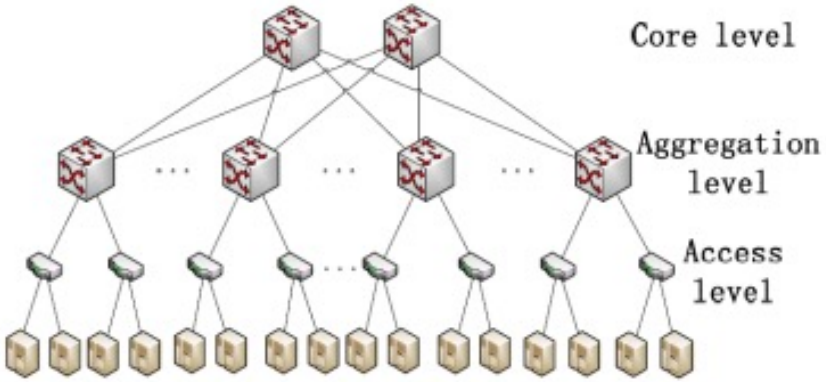


Fat-Tree Based Networks

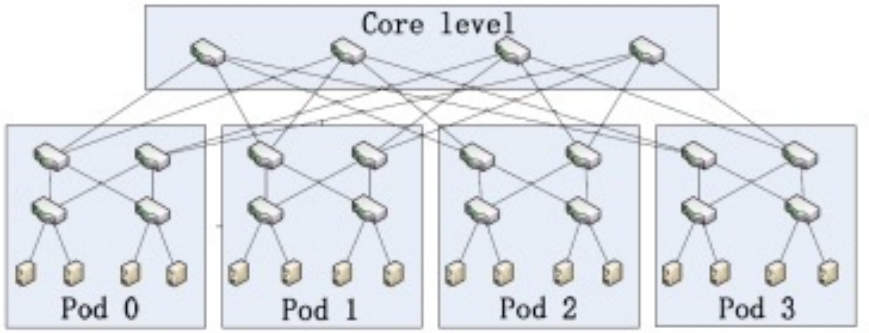
- Motivation (*the need for a slight modification in the routing table structure*)
  - Routing protocols such as OSPF2 usually take the hop-count as their metric of "shortest path"
  - In the k-ary fat-tree topology, there are $(k/2)^2$ such shortest-paths between any two hosts on different pods, but only one is chosen.
  - **Switches**, therefore, concentrate traffic going to a given subnet to a single port even though other choices exit that give the same cost.



Conventional DC VS Fat-Tree Based Networks



A multi-tiered network.



Fat tree network.

Conventional DC VS Fat Tree-based DC

- The design of fat tree network is motivated by the fact that the price differential between *high-end switch* (switches with higher link bandwidth or higher number of ports) and *low-end switches* is considerably large.

- The main idea behind the construction of fat tree topology is to replace **the high-end switches in multi-tiered topology** by **interconnecting several low-end switches.**

- The main difference in fat tree is that all of **the aggregation level** and **core level switches** are replaced with **interconnections of a set of low end switches.**

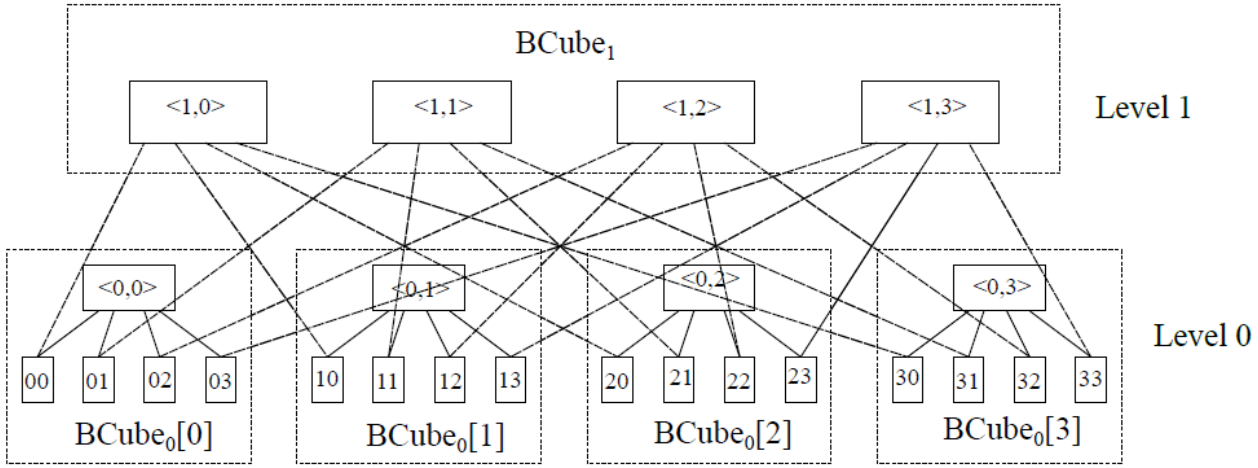
Conventional DC VS Fat Tree-based DC

- Each subset is called a pod. As the number of **uplinks** and **downlinks** for *each pod are equal*, fat tree has **full bisection bandwidth**.
- **Bisection bandwidth** is the **maximum bandwidth** that can be transferred across the midpoint of the system.
- Moreover, since all the switches in the fat tree topology are inexpensive low-end access level switches, this network topology is supposed to be **highly scalable and economical.**

BCube Topology (BCube Architecture)

- **BCube** is a recursively defined structure specially designed for modular data center.

- **Regarding its construction**, it contains nodes with multiple ports and switches connecting a constant number of nodes.

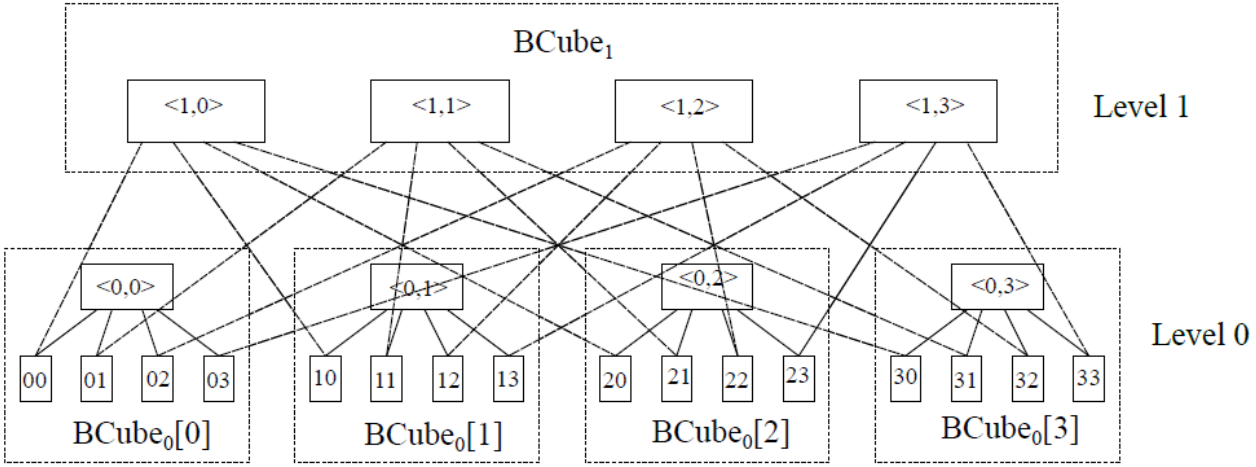


Nodes in a $BCube_1$ topology with $n = 4$.

BCube Topology (BCube Architecture)

- To start with, a $\mathbf{BCube_0}$ contains just $\boldsymbol{n}$ **nodes** connected via an $\boldsymbol{n}$-**port switch**.

- Furthermore, a $\mathbf{BCube_1}$ is built up from $\boldsymbol{n}$ $\mathbf{BCube_0}$ with $\boldsymbol{n}$ $\boldsymbol{n}$-**port switches** and so on.

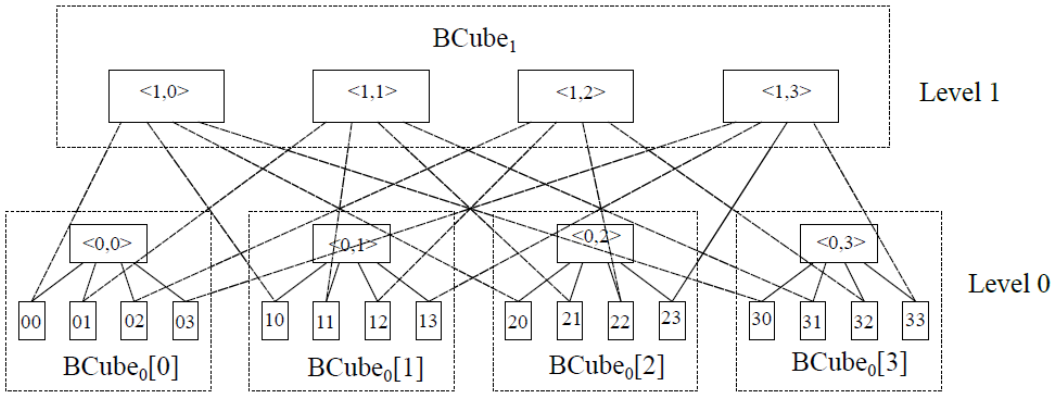


Nodes in a BCube_1 topology with $n = 4$.

BCube Topology (BCube Architecture)

- The figure describes the structure of $BCube_k$ network topology, specifically focusing on $BCube_1$. Here's a breakdown of the key points:

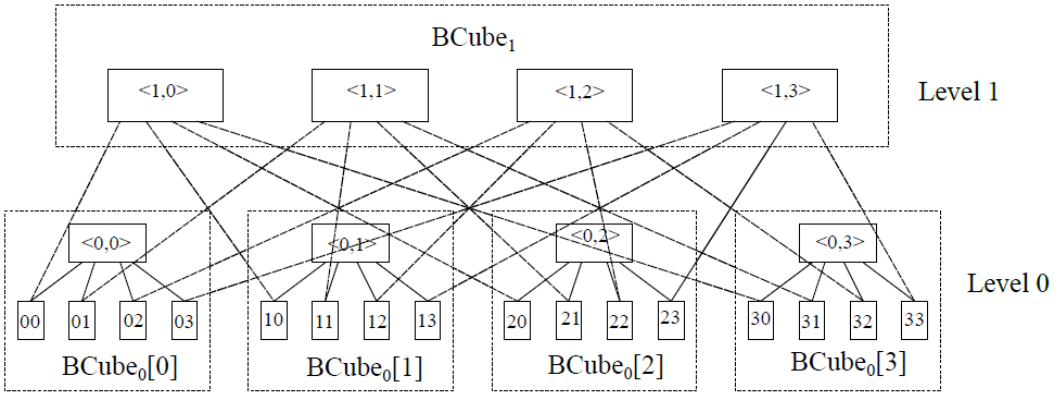
  - **$\mathbf{BCube_k}$ Structure:** A $\mathrm{BCube_k}$ network is constructed using n instances of $\mathrm{BCube_{k-1}}$ and $\mathrm{n^k}$ $\mathrm{n}-$ port switches. Each node in the network has $\mathrm{k}+1$ ports.



Nodes in a $\mathrm{BCube_1}$ topology with $n = 4$.

BCube Topology (BCube Architecture)

- **$\mathbf{BCube_1}$ Example:** The figure provides an example of a $BCube_1$ topology with $n = 4$. This means it consists of 4 $BCube_0$ instances (which are essentially individual nodes) and 4 switches.
- **Nodes and Switches:** In the $BCube_1$ topology, each node is connected to a switch, and the switches facilitate communication between the nodes.
- **Levels:** The nodes are organized into levels, with Level 1 shown in the example. Each level represents a different layer of connectivity within the network.



Nodes in a $BCube_1$ topology with $n = 4$.

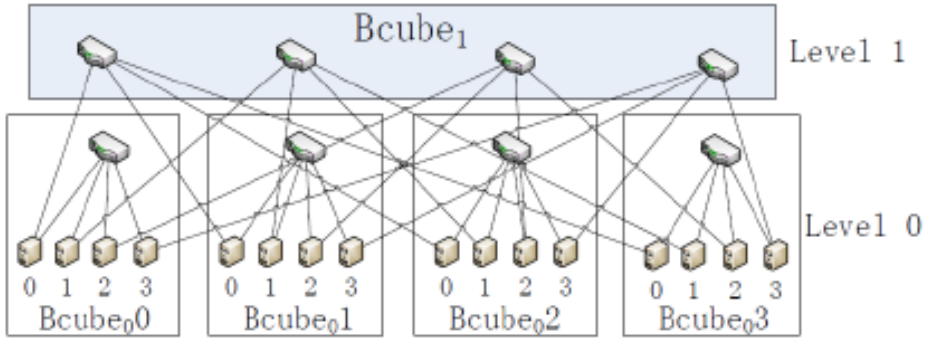
BCube Topology (BCube Architecture)

- level 1 is constructed from level 0 means higher level layer is formed using lower level switches and ports.

- $\mathbf{BCube_0}$ is designed using ***n*** servers (nodes) connected with ***n*** port switches and $\mathbf{BCube_1}$ is constructed using the connection between ***n*** port core switches and ***n*** servers.

BCube Architecture

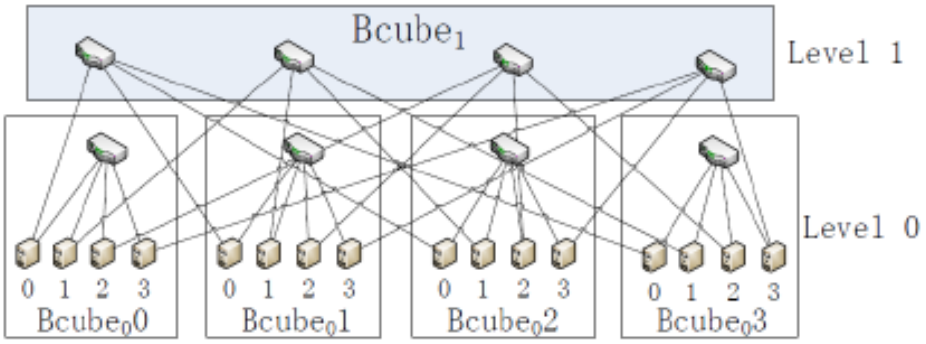
- The **BCube** architecture uses both switches and servers for routing traffic.
- $\mathrm{BCube_k}$ is recursively defined from $\mathrm{BCube_0}$, which contains *n* servers and an *n*- port switch, each server connecting to a switch port.
- There are no direct connections between any two servers.



B-Cube Architecture

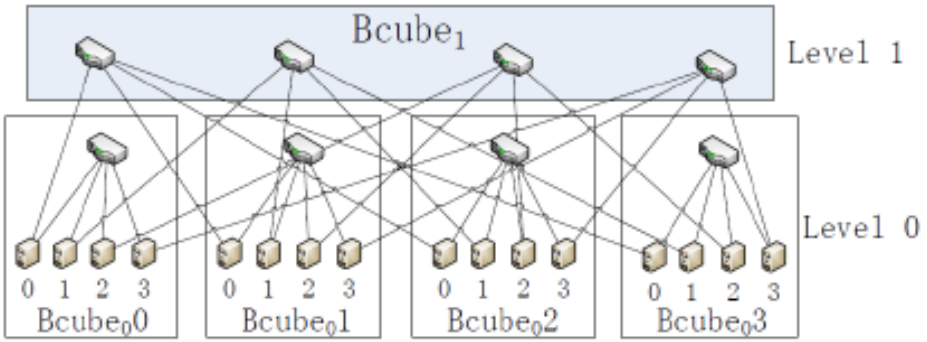
- A $\mathrm{BCube_k}$ network can be constructed using n $\mathrm{BCube_{k-1}}$ topologies and n^k n-port switches.

- In $\mathrm{BCube_k}$, there are k+1 levels, and $N = n^{k+1}$ servers, each server connecting a switch at each level.



B-Cube Architecture

- A BCube_k network supports n^{k+1} servers:
  - *n* is the number of servers in a BCube_0
  - *k* is the level of that BCube



$\mathrm{BCube}_1(n = 4)$

B-Cube Architecture

- **Number of layers :** k+1
- **Number of servers :** N= n ^ (k+1)
- **Number of switches in each layer**: #servers/#switch ports = n ^ (k+1) /n = n^k
- **Number of BCube0s :** n^k
- **Number of switches :** (n^k)*(k+1)
- **Number of links :** (n^k)*(k+1)*n

In-class Activity

Design a data center using BCube topology for 16 servers. How the number of links for n=4, and n=2 are different?

	n = 2	n = 4
# Servers		
# Layers		
# Switches		
# Links		

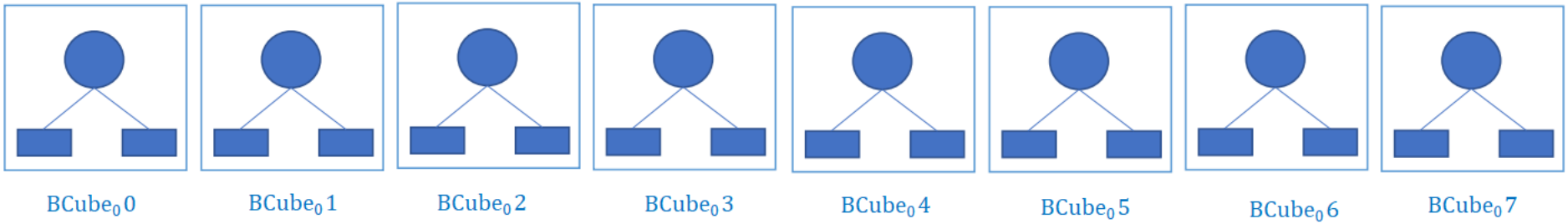
In-class Activity

Design a data center using BCube topology for 16 servers. How the number of links for n=4, and n=2 are different?

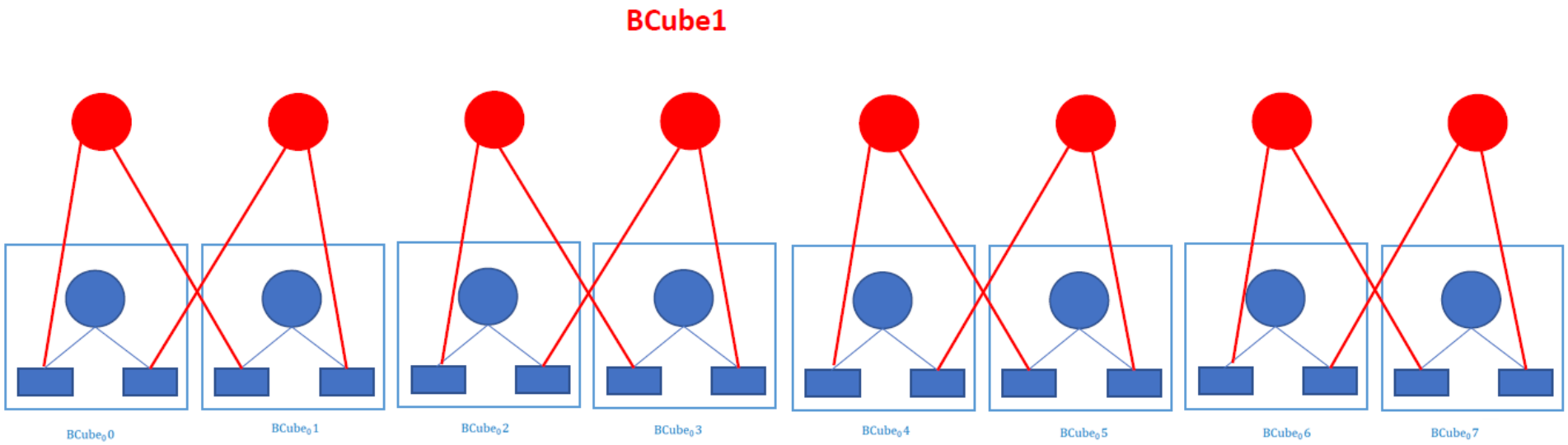
	n = 2	n = 4
# Servers	16	16
# Layers	4	2
# Switches	32	8
# Links	64	32

Design a BCube-based DC using 16 servers and 2-port switches

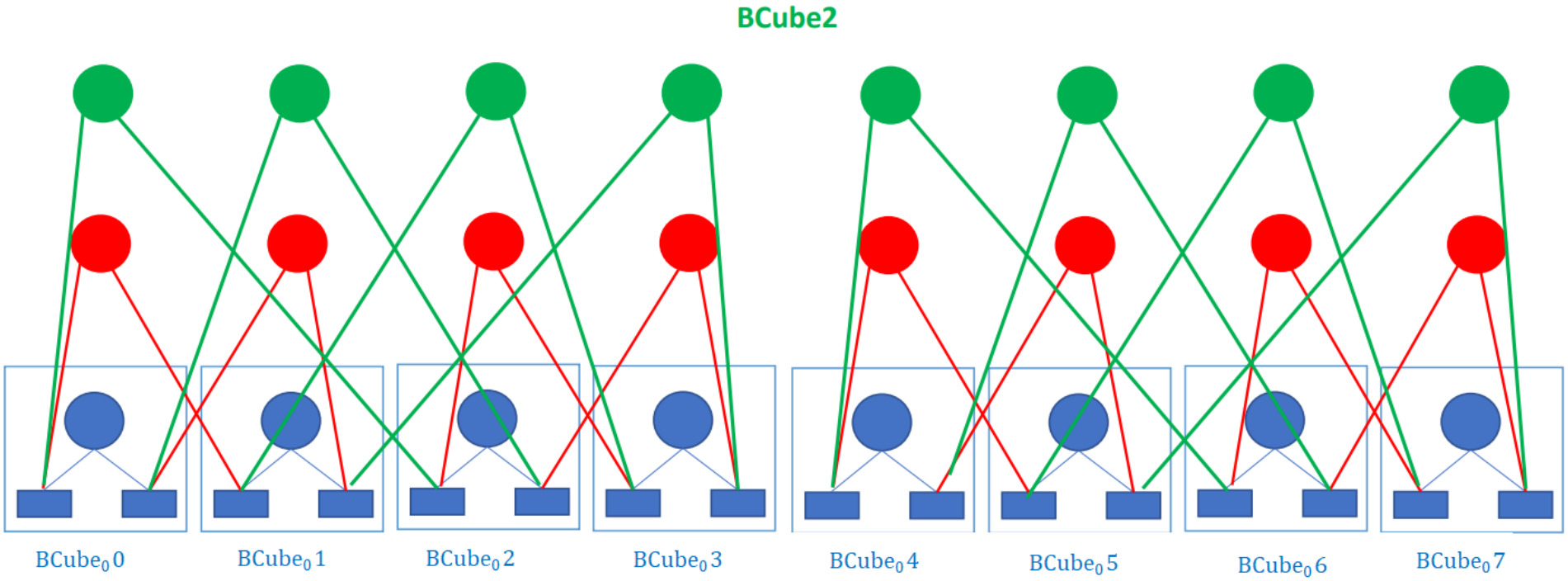
- N= n ^ (k+1) → 16 = 2^(k+1) → k=3
- Number of BCube_0s = n^k= 8



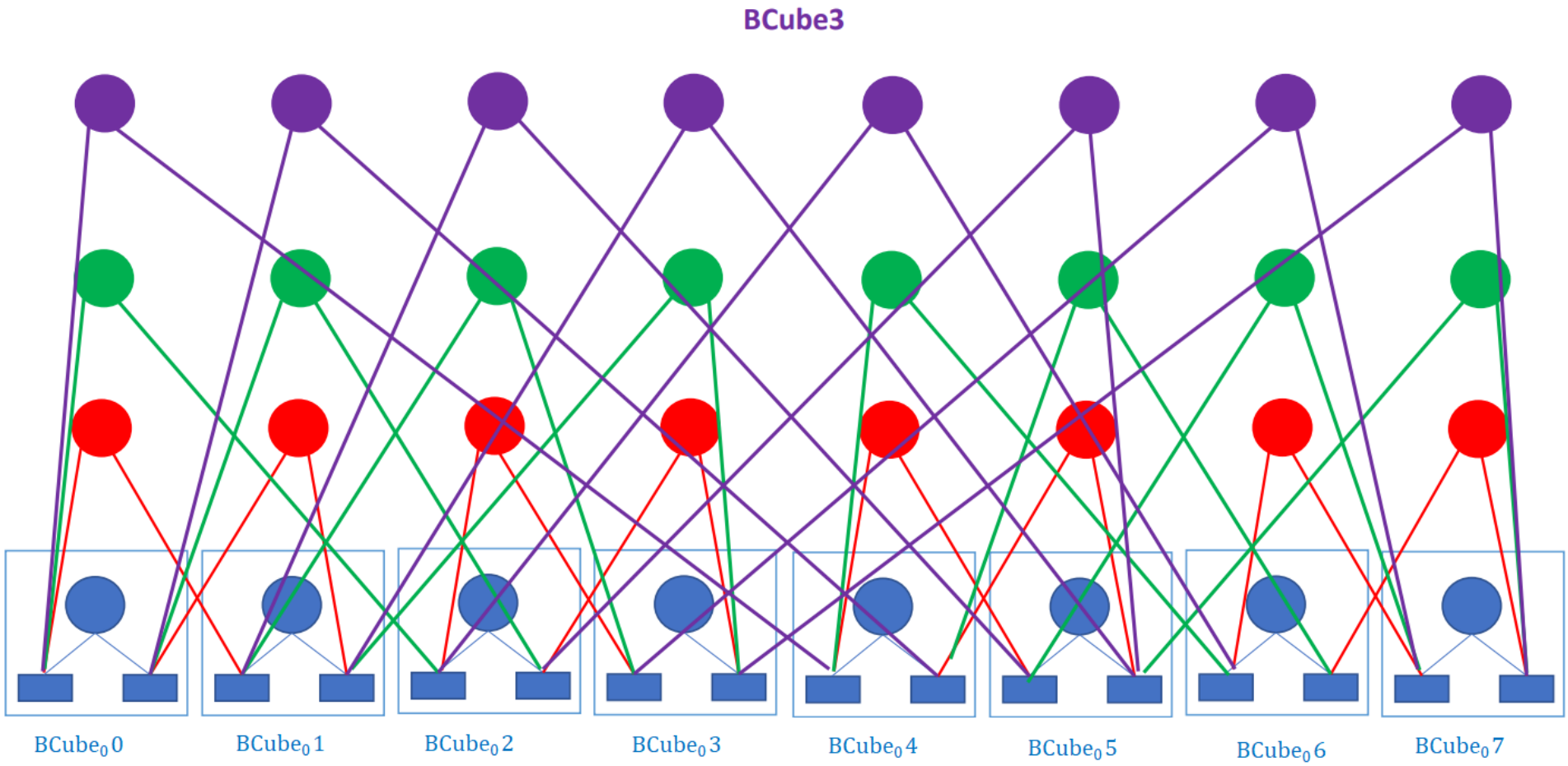
Design a BCube-based DC using 16 servers and 2-port switches



Design a BCube-based DC using 16 servers and 2-port switches

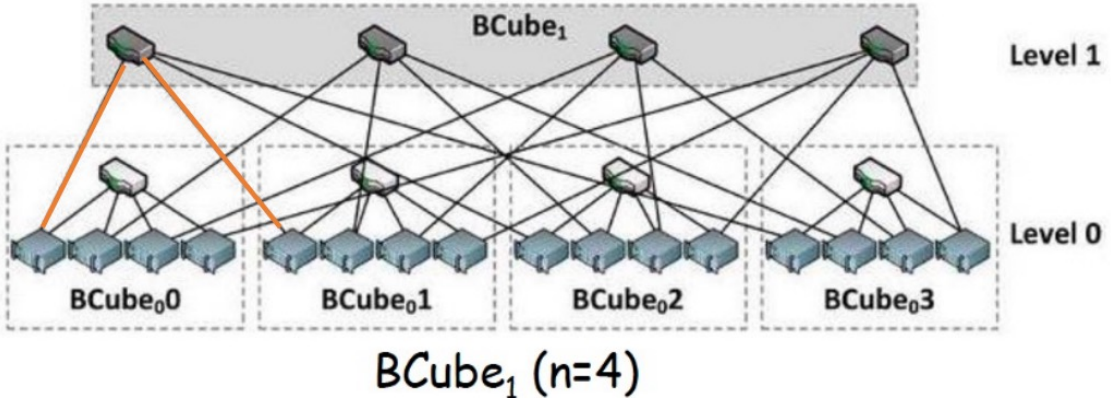


Design a BCube-based DC using 16 servers and 2-port switches

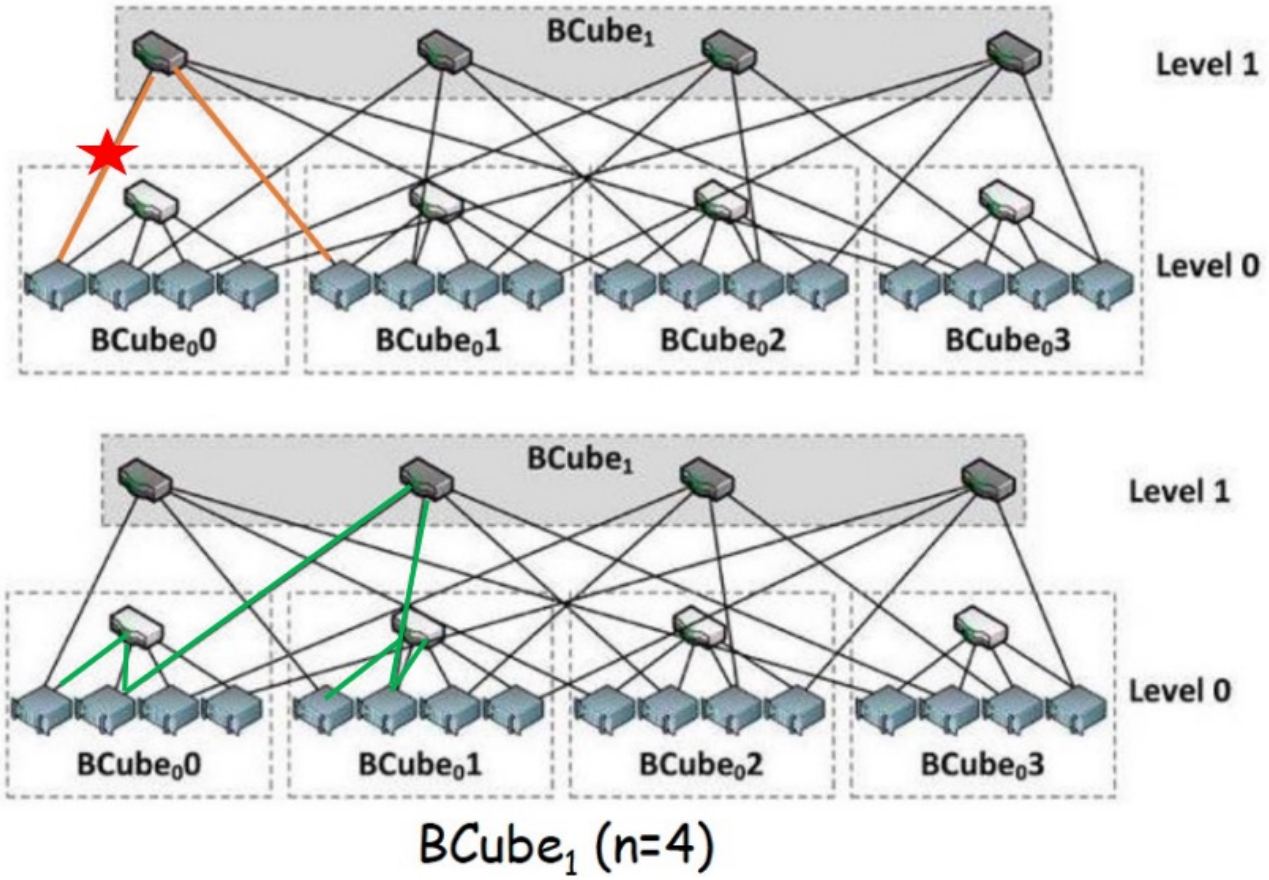


BCube Architecture – Shortest path

- The shortest path in BCube is determined by its multi-level structure, enabling efficient routing.
- Hierarchical forwarding is used, with lower levels handling local communication and higher levels managing long-distance traffic. This ensures minimal hops and optimal performance between servers.



BCube Architecture – Shortest path

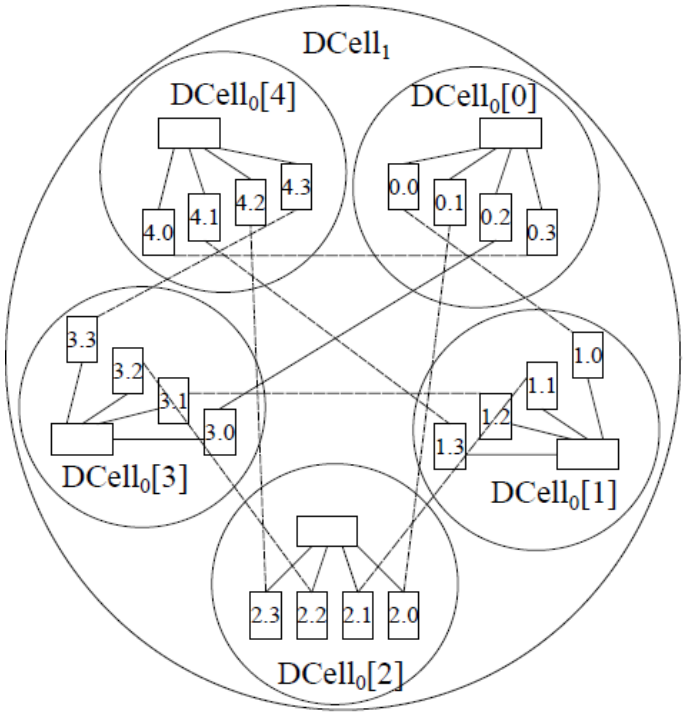


B-Cube Vs. Fat-Tree

- BCube uses servers and switches for packer forwarding. It leads to higher resilience against switch failures.
- BCube uses less number of switches that Fat-Tree. It leads to lower cost.
- Shortest path is always fixed in Fat-Tree even if failure happens. It guarantees shortest path/latency

DCell Architecture

- DCell is a recursively defined structure specifically designed for modular data centers.
- The building block for its construction is $\mathbf{DCell_0}$, which has $\boldsymbol{n}$ nodes and a **mini switch**.
- Further, $\mathbf{DCell_1}$ is made of $\boldsymbol{n+1}$ $\mathbf{DCell_0}$ blocks, where each of those is connected to the other $\mathbf{DCell_0}$ blocks with just one link.



Nodes in a DCell_1 topology with $n = 4$.

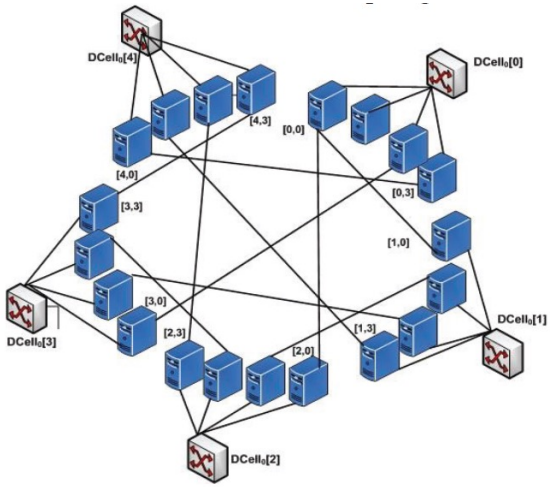
DCell Architecture

- DCell topology is recursively constructed with small DCells starting with DCell_0 which contains ***n*** servers connected to a n-port mini-switch and considered as lowest level DCell.

- Main idea behind DCell structure not only depend on switches but also to take the advantage of Network Interface Card (NIC) deployed within host servers to design the topology.

DCell Architecture

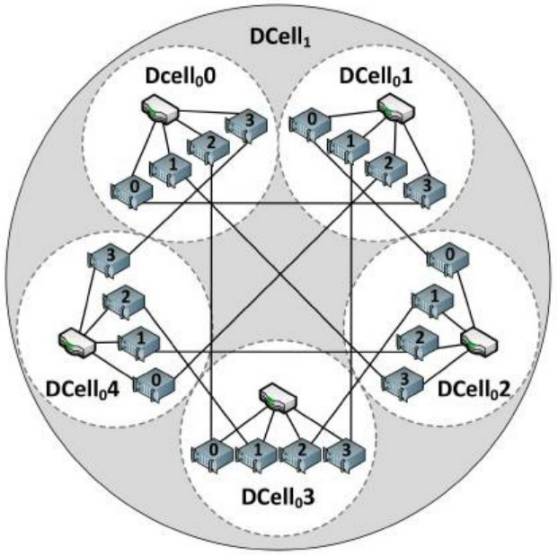
- DCell topology provides an alternative to
  - Scale up to several millions servers without any *expensive core* and *aggregation switches* and also,
  - It overcomes the constraint of single point of failure as in tree-based topologies.



DCell Architecture

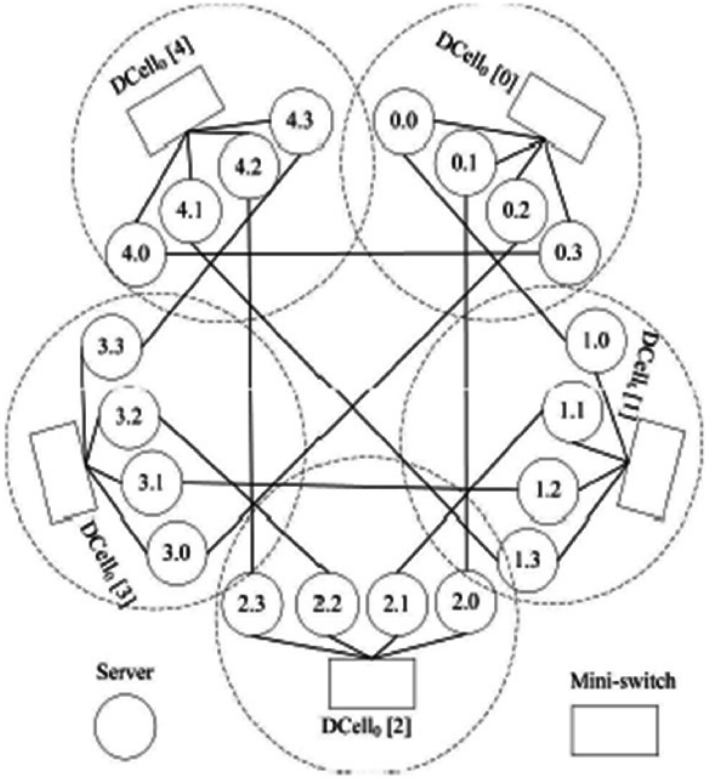
- Similar to BCube, DCell is defined recursively and uses servers and mini-switches for *packet forwarding*.

- The main module is $\mathbf{DCell_0}$ which is composed of a switch connected to *n* servers.



DCell Architecture

- Use mini-switches to scale out.
- Leverage servers be part of the routing infrastructure
  - *Servers have multiple ports and need to forward packets*
- Use recursion to *scale and build complete graph to increase capacity.*

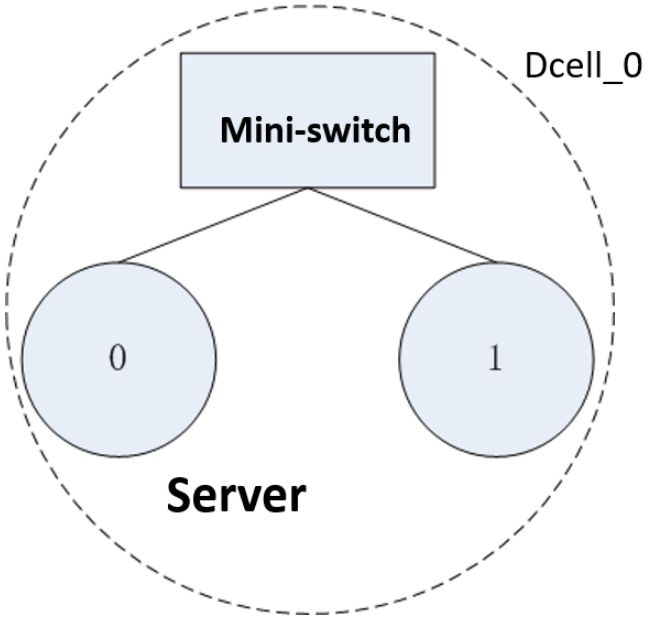


Summery – DCell Ideas

1. Use mini-switches to scale out
2. Leverage servers be part of the routing infrastructure
   - Servers have multiple ports and need to forward packets
3. Use recursion to scale and build complete graph to increase capacity

Summery – DCell Ideas

- **The Construction**

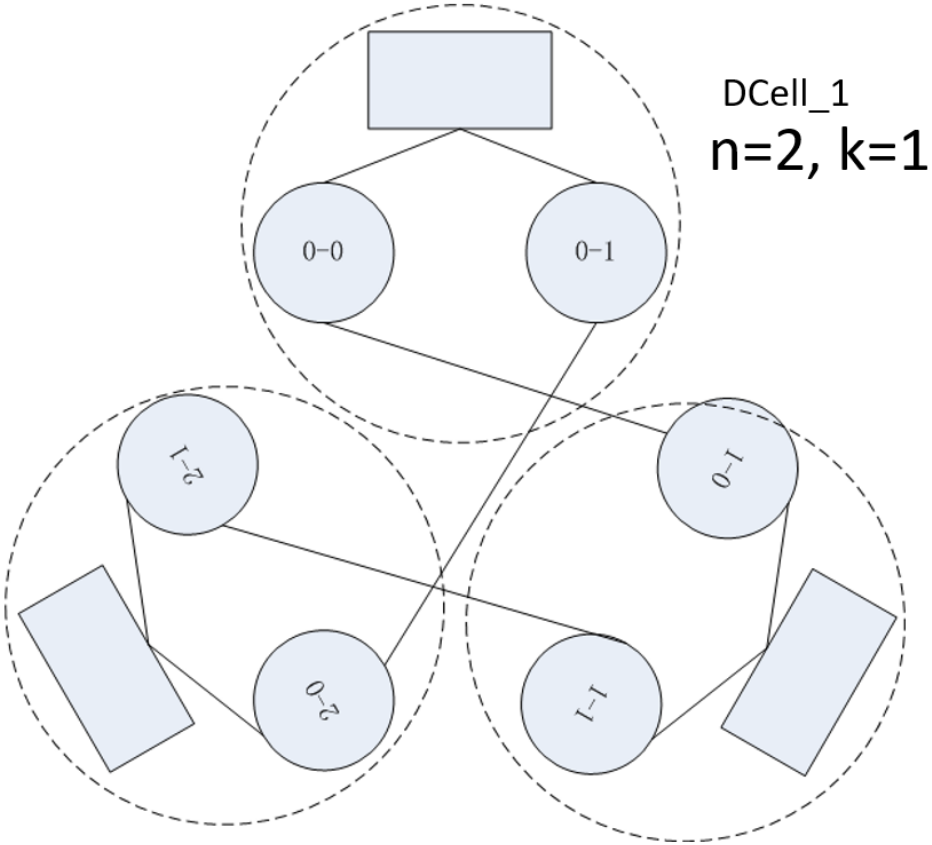


n servers in a DCell_0

n=2, k=0

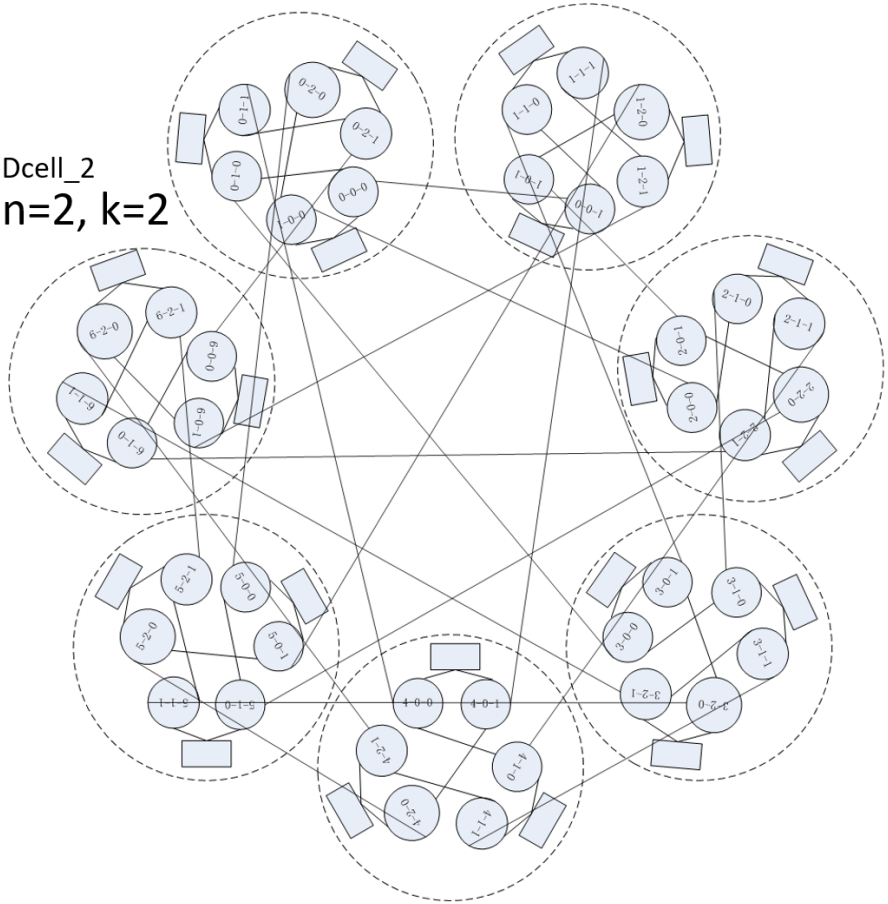
Summery – DCell Ideas

- **The Construction**



Summery – DCell Ideas

- **The Construction**



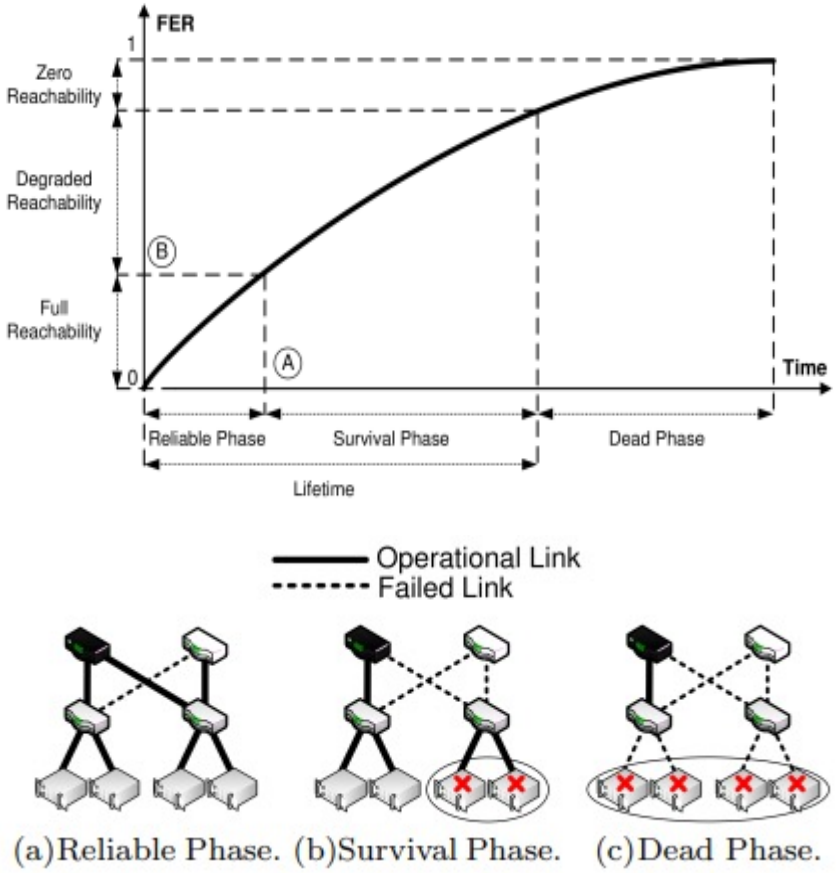
Comparing Data Center Topologies

- Failure Type:
  - Link
  - Switch
  - Server
- Criterion
  - Reliable Phase
    - MTTF (Mean Time to Failure)
    - FER (Failure Element Ratio)
  - Survival Phase
    - Reachability
      - Accessible Server Ratio
      - Server Connectivity
    - Path Quality
      - Average Shortest Path Length

Comparing Data Center Topologies

- **MTTF (Mean Time to Failure):**
  - **MTTF is a measure of the expected time a system or component will operate before it fails**
  - In this work, the authors evaluate **the mean value of this metric, called MTTF (Mean Time To Failure)**, which is the expected value of the TTF in a network (i.e., **mean time elapsed until the first server disconnection**).
- **FER (Failure Element Ratio)**
  - **FER is a measure that indicates the proportion of elements (or components) within a system that fail within a specific period. It helps understand the distribution of failures among different components.**
  - In this work, the authors evaluate this **metric as a mean value**, called the Critical FER.
    - For example, a network with 100 switches that disconnects a server, on average, after the removal of 2 random switches, has a critical FER of 2/100 = 0.02. The mean time to have a Critical FER is thus equal to the MTTF.

Comparing Data Center Topologies



- Figure illustrates a hypothetical situation of the Failed Elements Ratio (FER) evolution according to the time.
- Starting the lifetime by the moment where a full maintenance was completed, a DC passes through a first phase in which **failures do not cause server disconnection**, defined here as the **Reliable Phase**, and
- a second phase where at least one server is **disconnected**, that we define a the **Survival Phase**.
- The lifetime period ends when the DC has no connected servers. After that, the DC enters the **Dead Phase**.

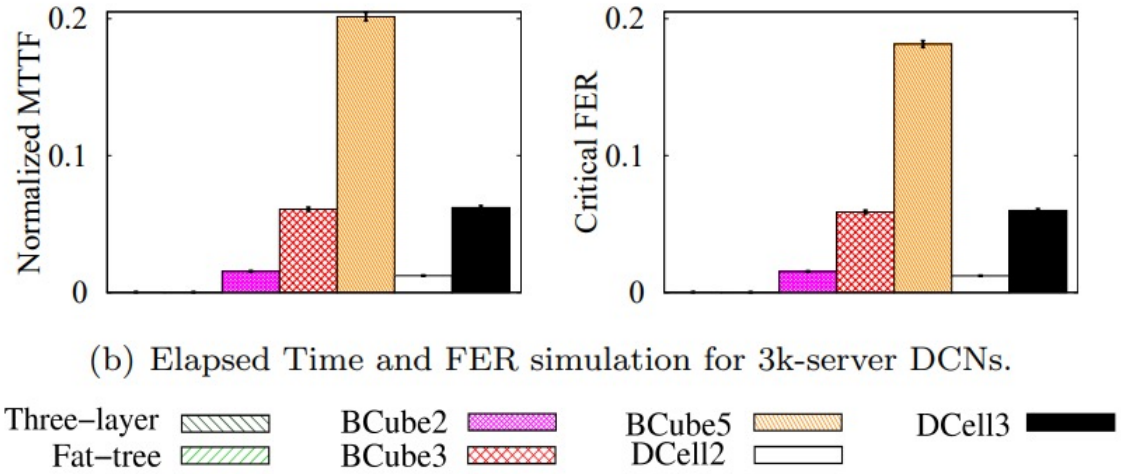
Comparing Data Center Topologies

- DCN topology configuration used in the analysis

Size	Name	Switch Ports	Server Ports	Links	Switches	Servers
3K	Three-Layer	12(core)	1	3630	86	3456
	Fat-tree	24	1	10368	720	3456
	BCube2	58	2	6728	116	3364
	BCube3	15	3	10125	670	3375
	BCube5	5	5	15625	3125	3125
	DCell2	58	2	5133	59	3422
	DCell3	7	3	6384	456	3192

Comparing Data Center Topologies

- Reliable Phase analysis for link failures

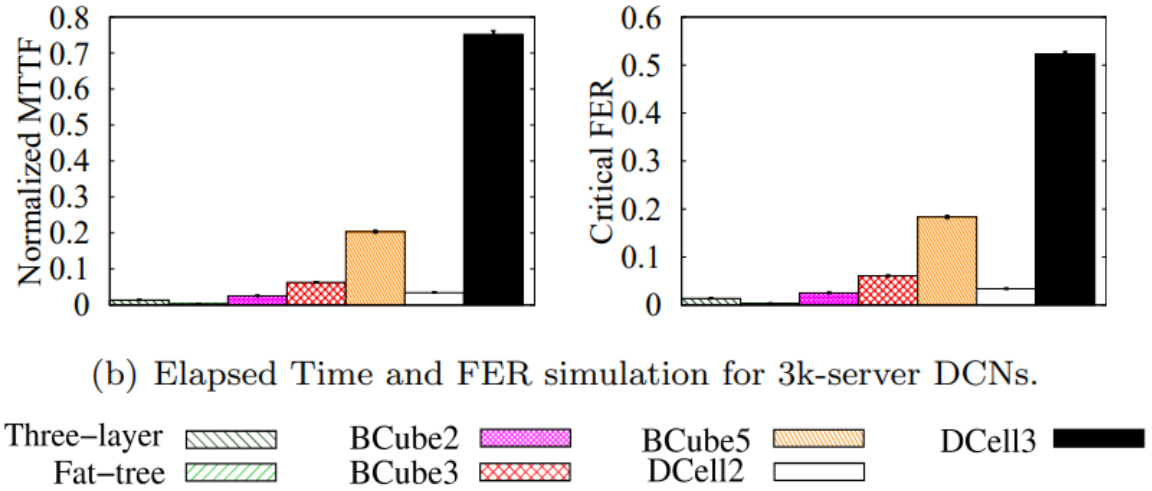


(b) Elapsed Time and FER simulation for 3k-server DCNs.

Figure shows the simulation of the Normalized MTTF and Critical FER for 3k-server topologies as an example. Note that the reliability of Three-layer and Fat-tree is substantially lower than that of the other topologies, and as a consequence their corresponding boxes cannot be seen in Figure.

Comparing Data Center Topologies

- Reliable Phase analysis for switch failures

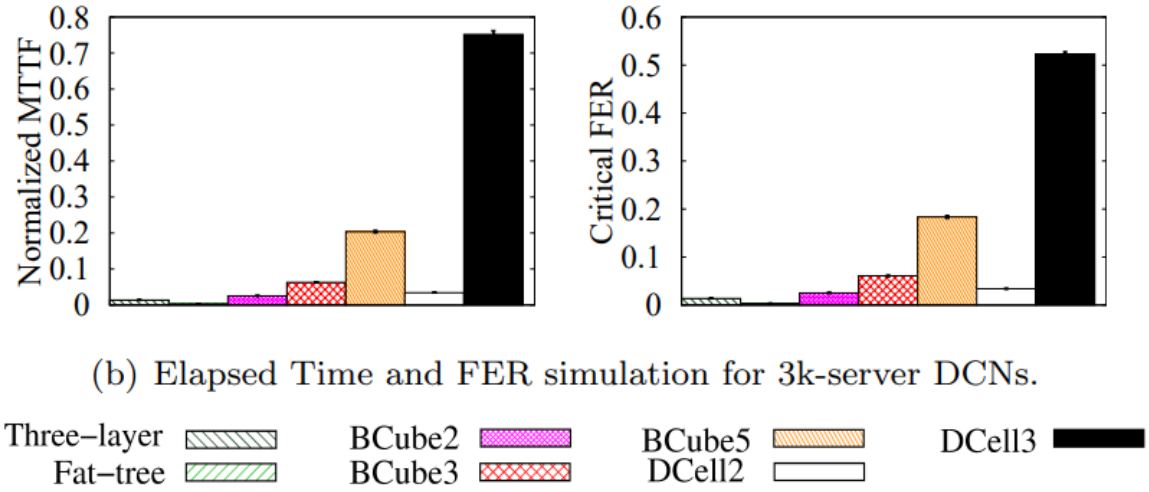


(b) Elapsed Time and FER simulation for 3k-server DCNs.

Three-layer and Fat-tree Performance. Three-layer and Fat-tree have very low reliability compared with other topologies, because a single failure in an edge switch disconnects the network.

Comparing Data Center Topologies

- Reliable Phase analysis for switch failures



(b) Elapsed Time and FER simulation for 3k-server DCNs.

DCell has a higher reliability than BCube. This is due to less dependence on switches in DCell, as in DCell, each server is connected to 1 switch and ***L*** servers while on BCube, only switches are attached to the servers.