



Lecture 05: 云中数据通信

SSE316: 云计算技术
Cloud Computing Technologies

陈壮彬

软件工程学院

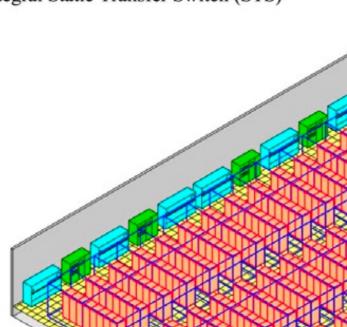
chenzhb36@mail.sysu.edu.cn

数据中心剖析

电力分配单元

Power Distribution Unit (PDU)

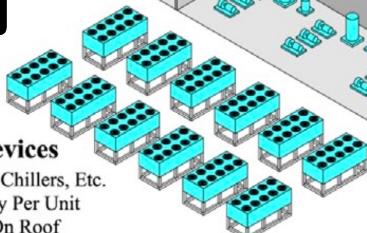
- Typical Capacities Up To 225 kVA Per Unit
- Redundancy Through Dual PDU's With Integral Static Transfer Switch (STS)



Colocation Suites

- Modular Configuration For Flexible Suite Sq.Ft. Areas.
- Suites Consist Of Multiple Cabinets With Secured Partitions (Cages, Walls, Etc.)

机架托管套间



Heat Rejection Devices

- Drycoolers, Air Cooled Chillers, Etc.
- Up To 400 Ton Capacity Per Unit
- Mounted At Grade Or On Roof
- N+1 Design

热量排放设备

计算机空气处理单元

Computer Air Handling Unit (CRAC)

- Up To 30 Ton Sensible Capacity Per Unit
- Air Discharge Can Be Upflow Or Downflow Configuration
- Downflow Configuration Used With Raised Floor To Create A Pressurized Supply Air Plenum With Floor Supply Diffusers

Individual Colocation Computer Cabinets

- Typ. Cabinet Footprint (28"W x 36"D x 84"H)
- Typical Capacities Of 1750 To 3750 Watts Per Cabinet

个体机柜

应急柴油发电机

Emergency Diesel Generators

- Total Generator Capacity = Total Electrical Load To Building
- Multiple Generators Can Be Electrically Combined With Paralleling Gear
- Can Be Located Indoors Or Outdoors At Grade Or On Roof.
- Outdoor Applications Require Sound Attenuating Enclosures

燃油储存罐

Fuel Oil Storage Tanks

- Tank Capacity Dependant On Length Of Generator Operation
- Can Be Located Underground Or At Grade Or Indoors

不间断电源系统

UPS System

- Uninterruptible Power Supply Modules
- Up To 1000 kVA Per Module
- Cabinets And Battery Strings Or Rotary Flywheels
- Multiple Redundancy Configurations Can Be Designed

Electrical Primary Switchgear

- Includes Incoming Service And Distribution
- Direct Distribution To Mechanical Equipment
- Distribution To Secondary Electrical Equipment Via UPS

电气主开关设备

泵房

Pump Room

- Used To Pump Condenser/Chilled Water Between Drycoolers And CRAC Units
- Additional Equipment Includes Expansion Tank, Glycol Feed System
- N+1 Design (Standby Pump)

集装箱数据中心节能技术

定义

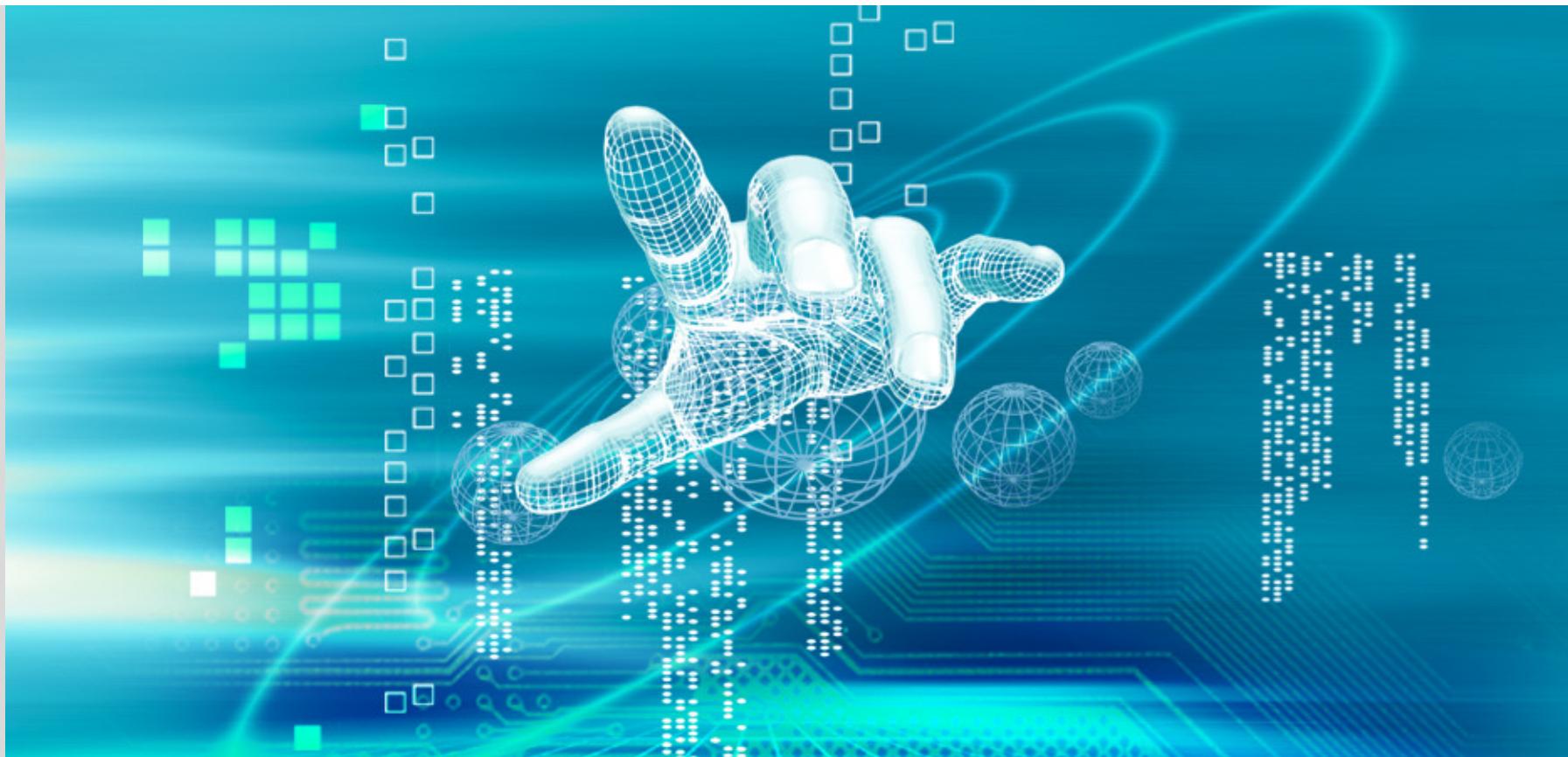
将数据中心的服务器设备、网络设备、空调设备、供电设备等**高密度地装入固定尺寸的集装箱中**，使其成为**数据中心的标准构建模块**，进而通过若干集装箱模块网络和电力的**互连互通构建完整的数据中心**。

1 高密度

2 模块化

3 按需快速部署

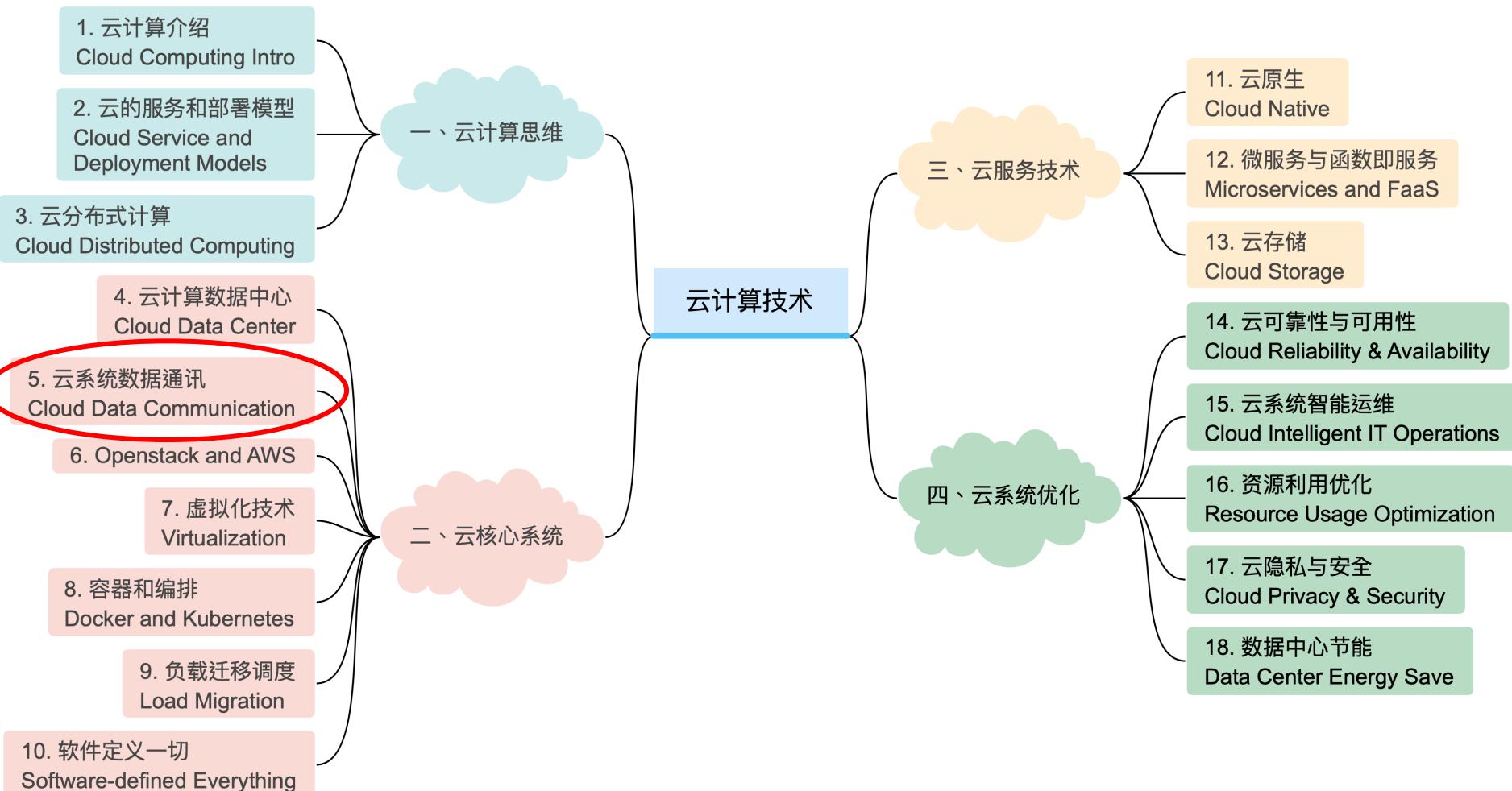
4 移动便携



自动化管理

云计算数据中心的自动化管理使得在规模较大的情况下，实现**较少工作人员**对数据中心的**高度智能管理**。





Today's topics

□ 云中数据通信的特点

□ 主机互联方式

□ 典型数据中心网络架构

- Clos 网络架构
- Fat-tree 网络架构

数据通信与存储在数据中心中的分量上升巨大



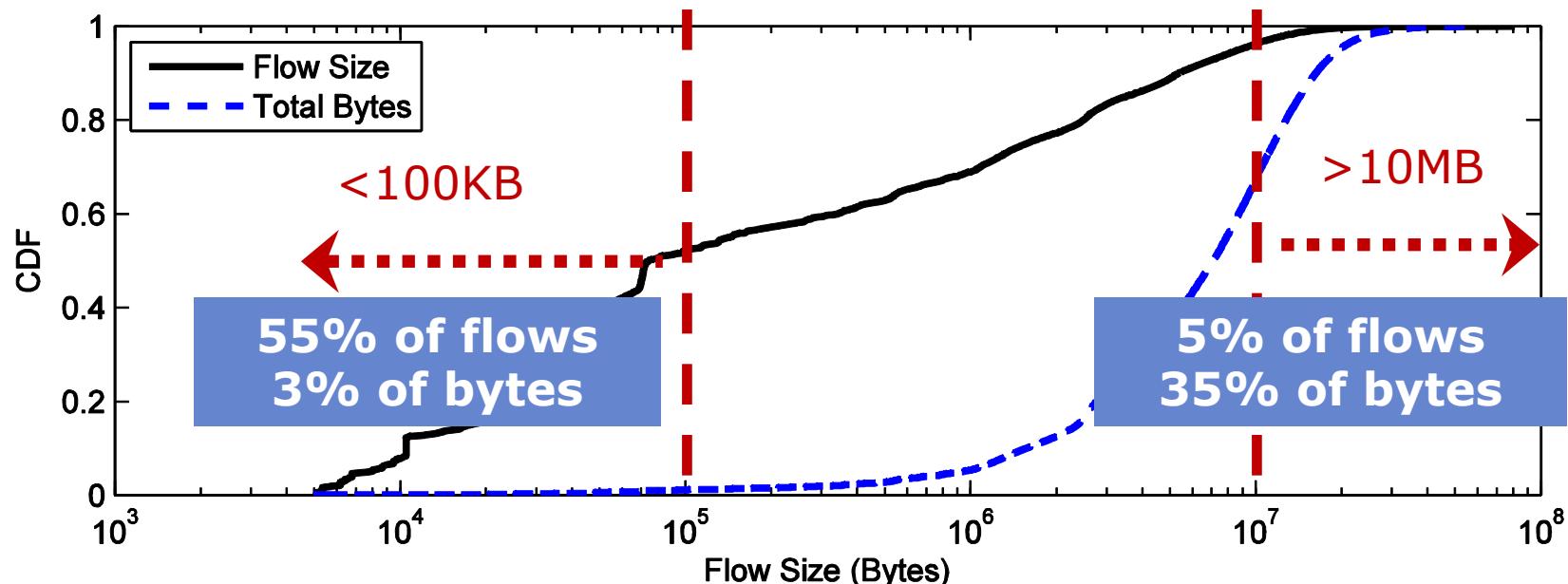
数据中心网络的流量特点

大部分流量由**大流**产生，小部分由**小流**产生

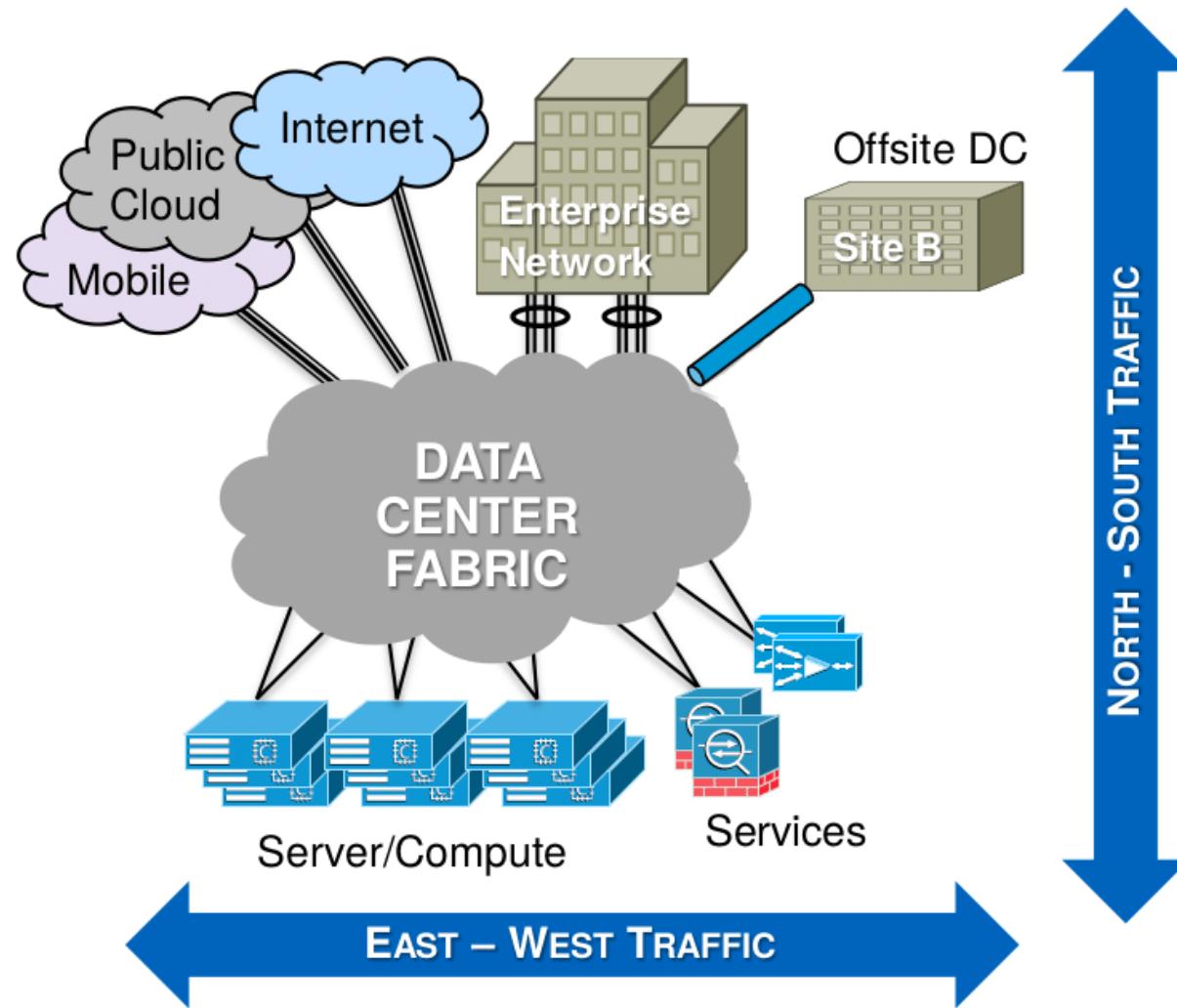
- 大流：宽密集型的应用，如数据备份、虚拟机迁移等
- 小流：一些交互式的应用，如web服务、数据库查询等

对数据中心网络的管理和优化很重要

- 大流：采取策略来保障其带宽，以防止网络拥塞或中断
- 小流：优化其传输延时，以提高应用的响应速度



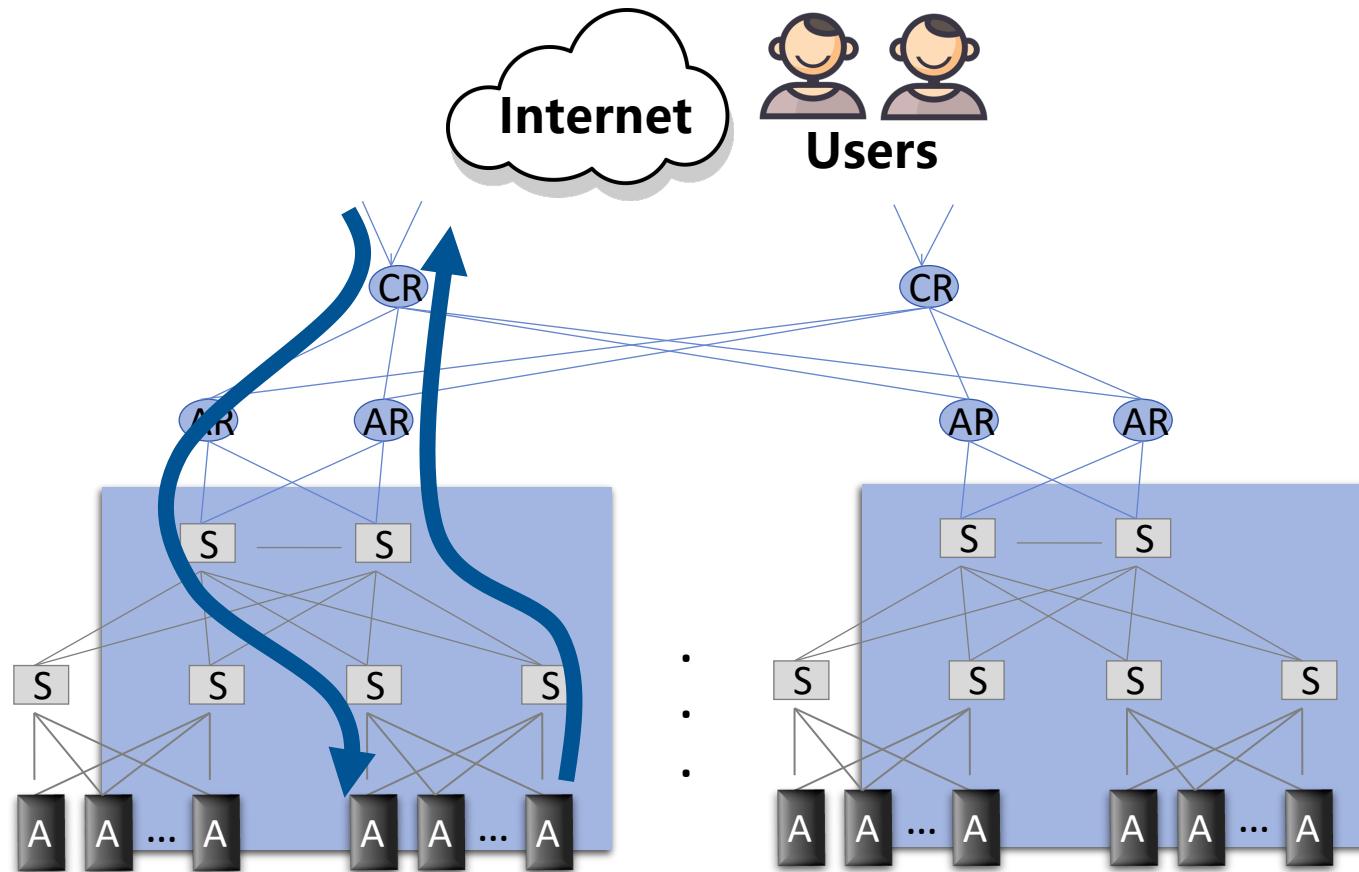
数据中心内的流量类型



南北向流量 (North-south Traffic)

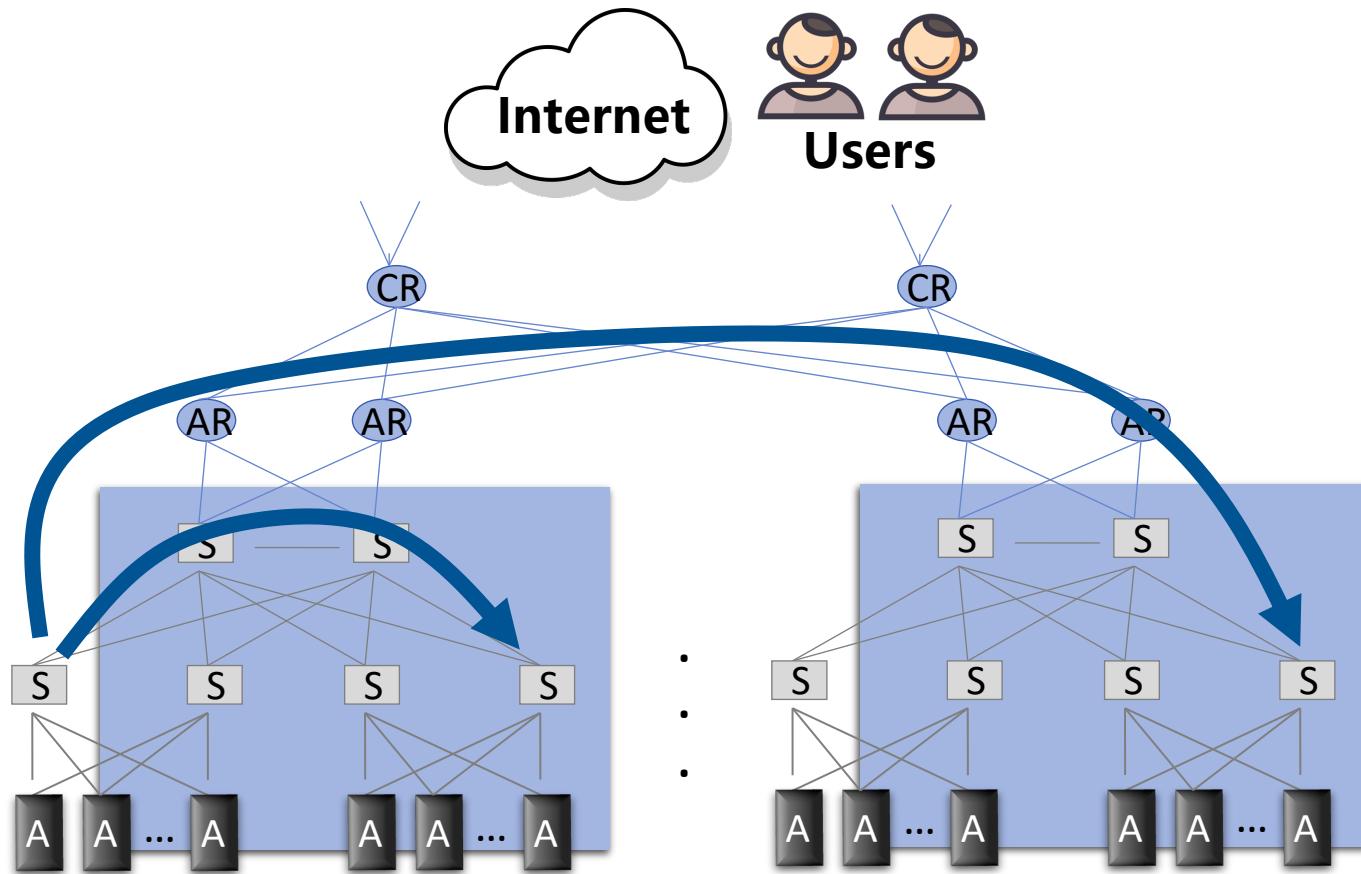
数据中心内部与外部客户端之间的流量

例如，用户通过互联网访问数据中心中的一个应用或服务



东西向流量 (East-west Traffic)

- 数据中心内部的流量，即服务器与服务器之间的通信
- 例如，“大数据”分布式计算、大模型训练中的流量



为什么要区分不同方向的流量？

具有不同的特征和需求，对数据中心网络设计和管理非常重要

南北向流量

- 需要更高的带宽和更强的安全性

东西向流量

- 更注重低延迟和高吞吐量

数据中心通信面临的挑战

- **Changing Traffic Patterns**

- Before: client-server traffic dominates (南北向流量)
- Now: machine-to-machine traffic dominates (东西向流量)

- **The rise of public/private cloud services**

- Want the agility to access applications, infrastructure, and other IT resources on demand (按需、灵活访问各种资源)

- **“Big data” also means more bandwidth**

- Hyper-scale data center networks maintaining any-to-any connectivity without going broke (高负载同时保证高可靠性)

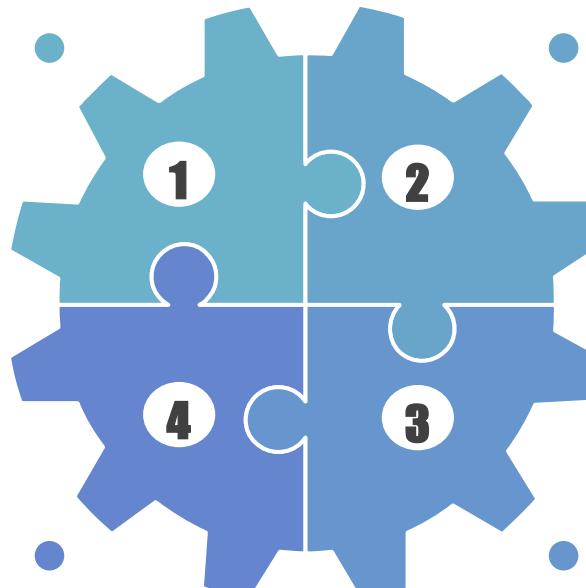
数据中心网络设计要求

自动化管理

云数据中心应是 24×7 小时无人值守并可远程管理的

高可用性

当网络故障或升级时，网络能够正常运行，对网络的性能影响不大



绿色节能

通过先进的供电和散热技术，降低数据中心的能耗

高设备利用率

采用良好的架构和虚拟化技术整合数据中心资源，提高利用率

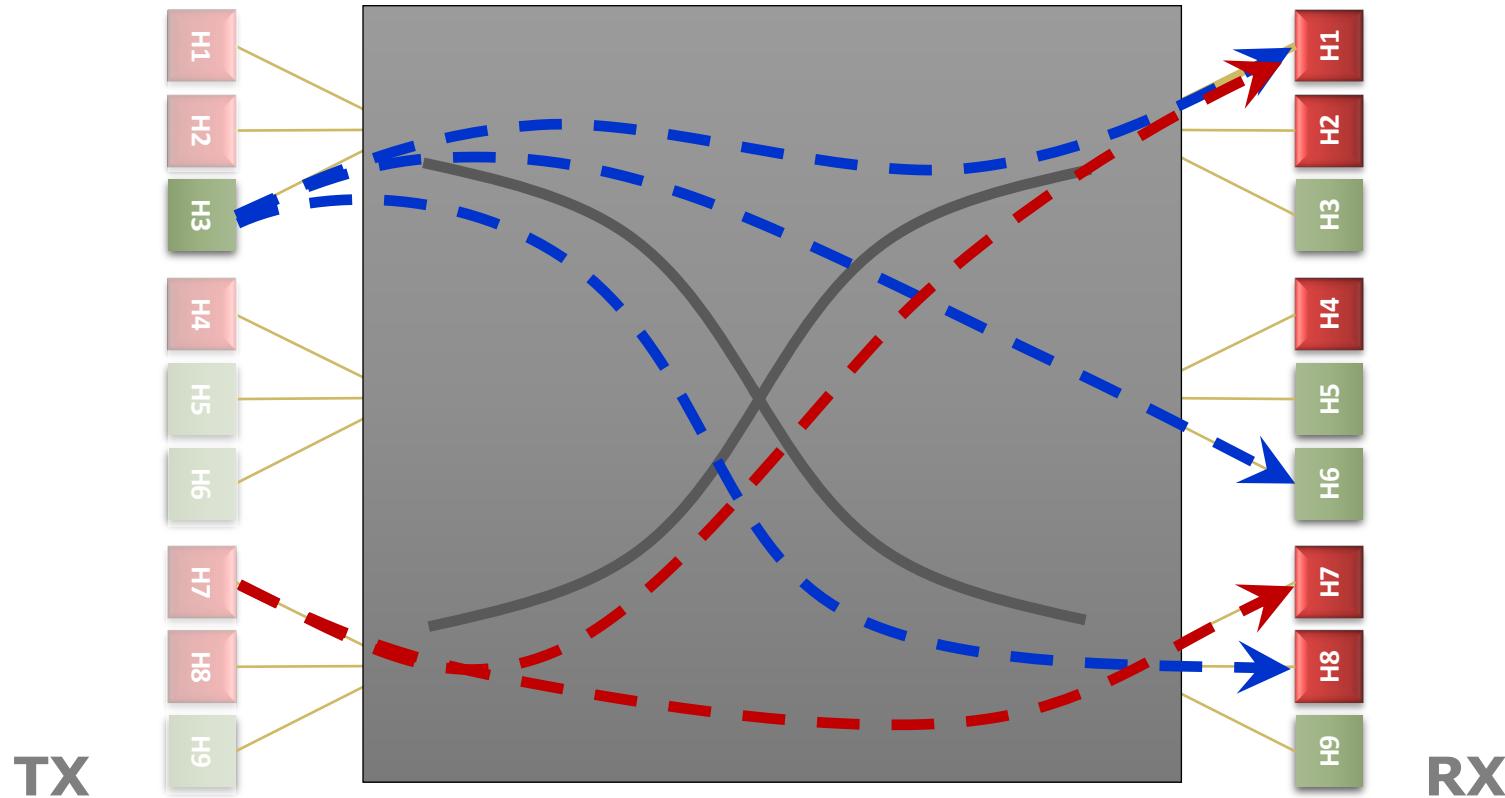
数据中心网络架构

数据中心网络架构是指用于组织和连接数据中心内部各种硬件设备（如服务器、存储设备、交换机等）的网络设计。

它是数据中心的基础架构之一，对数据中心的性能、可靠性和可扩展性都有很大的影响

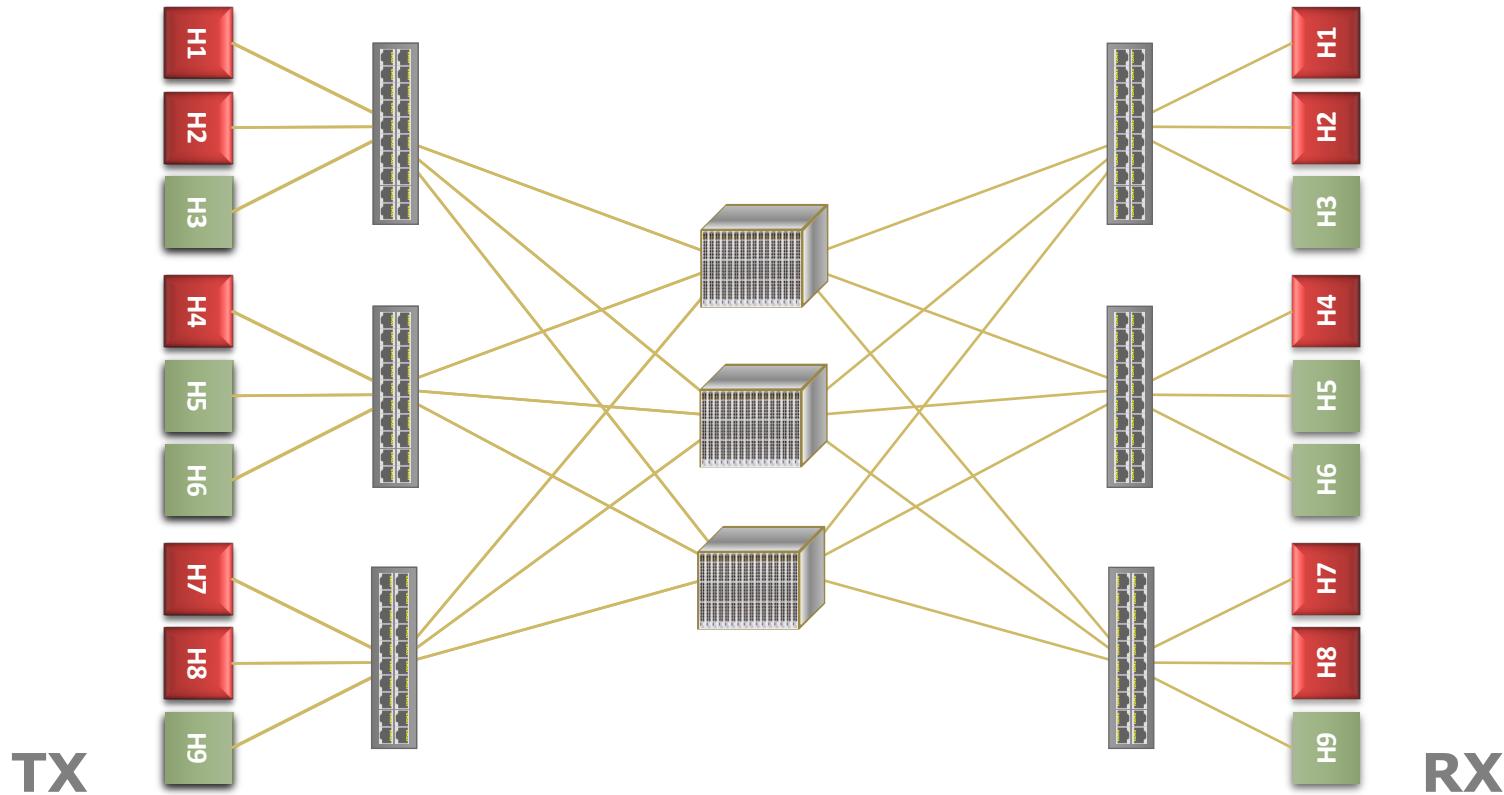
数据中心网络：一个大型交换机

口将数据从**发送端 (TX)** 传输至**接收端 (RX)**

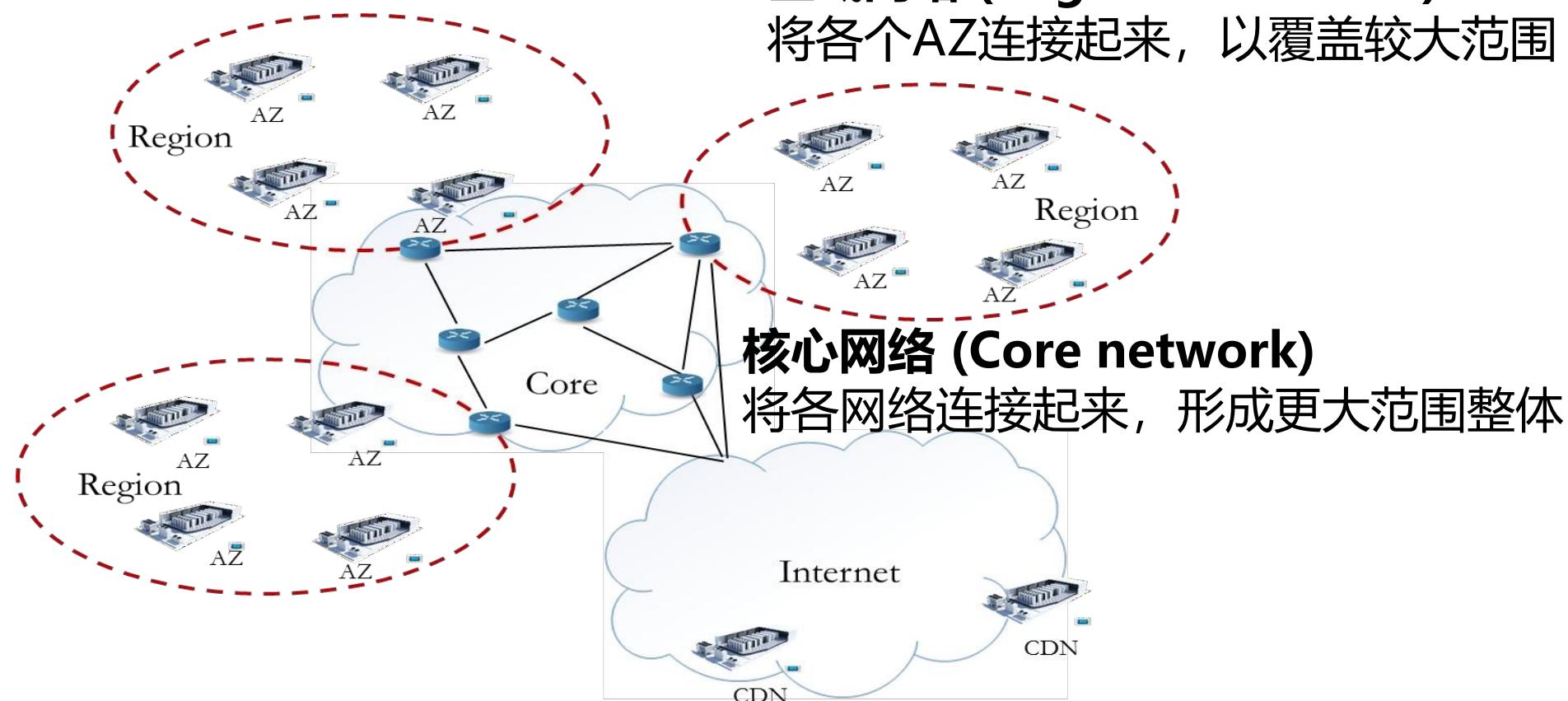


数据中心网络：一个大型交换机

内部需要复杂的网络架构来确保数据的**高效和准确传输**



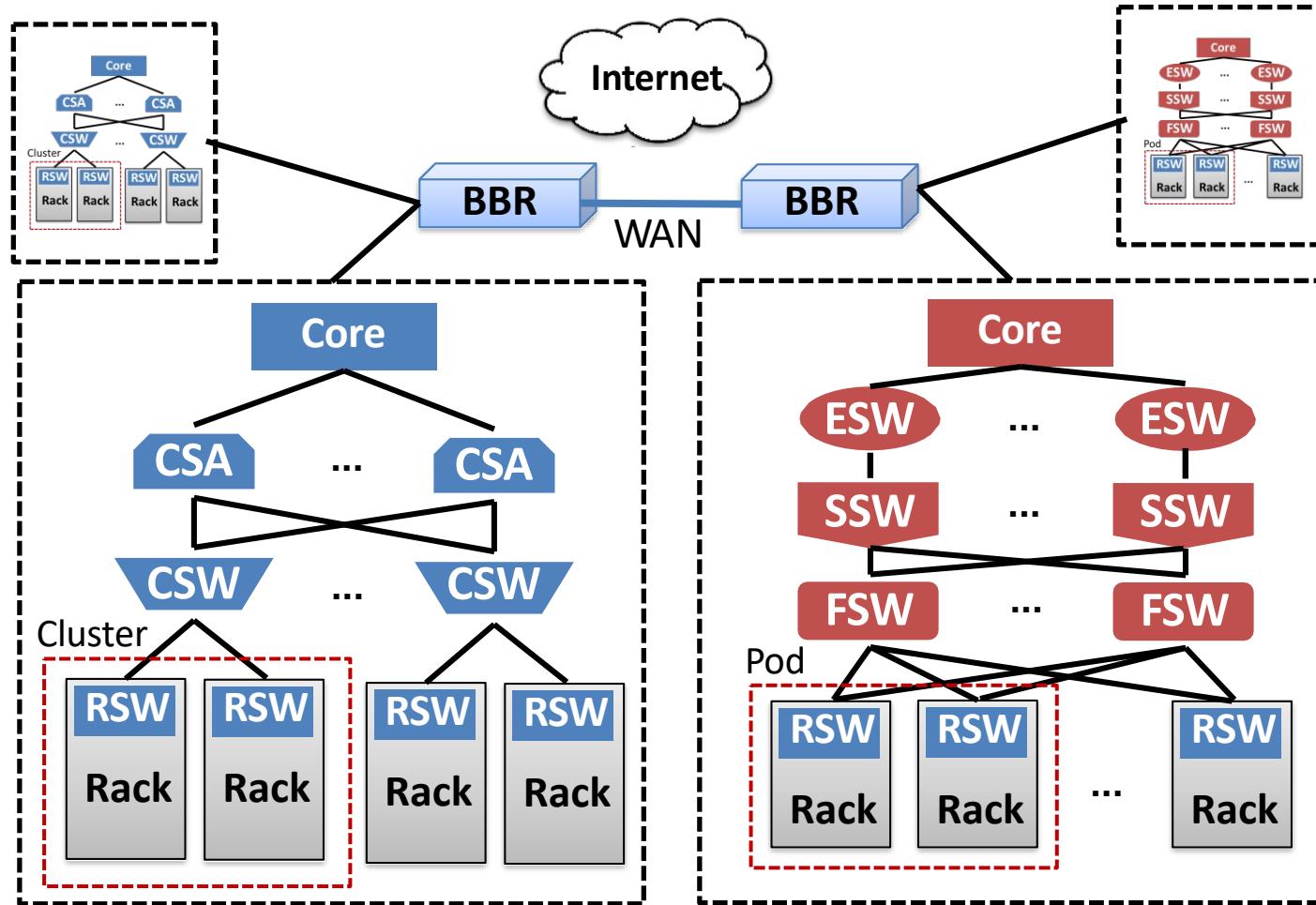
一朵云的主要网络组成部分



边缘或内容分发网络 (Edge/CDN)
连接了区域网络和公网或服务提供商

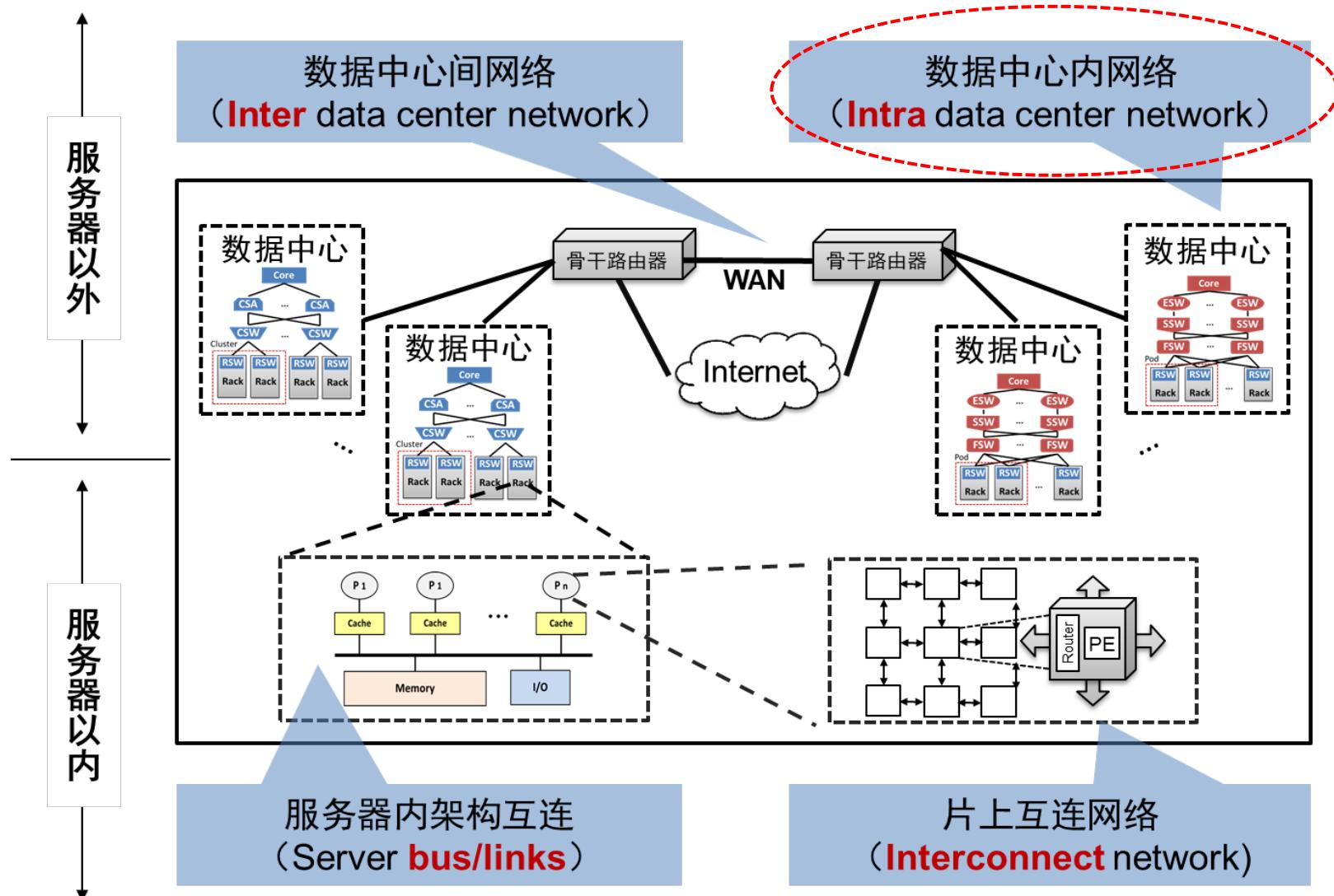
数据中心间网络 – 案例

Facebook 的数据中心网络

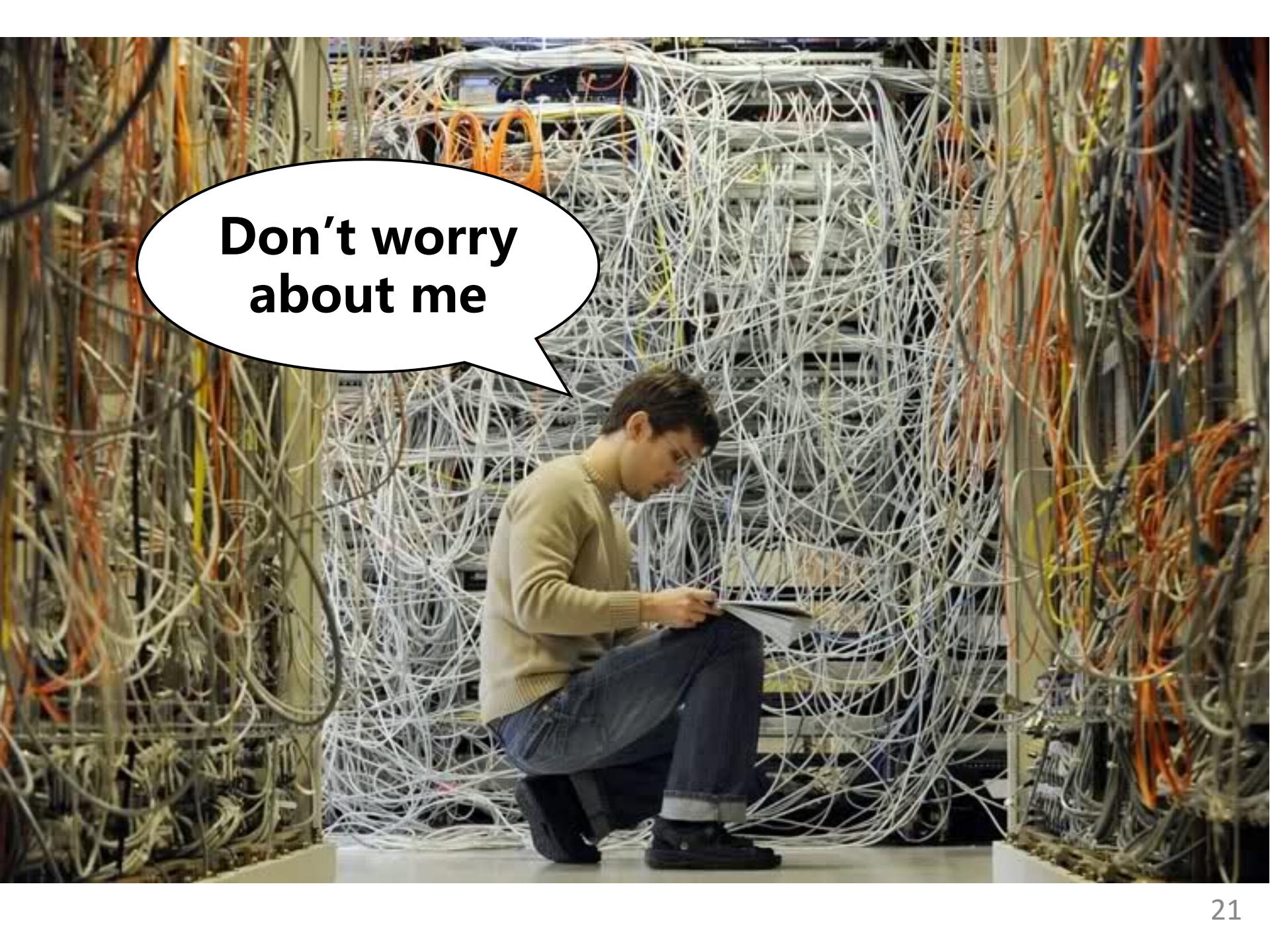


数据中心网络拓扑

两大四小层面看数据通信



如何高效地互联多个主机？

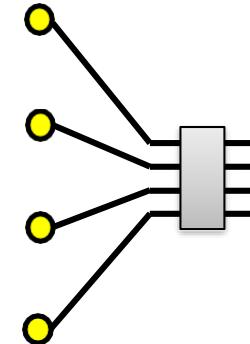


**Don't worry
about me**

互联架构

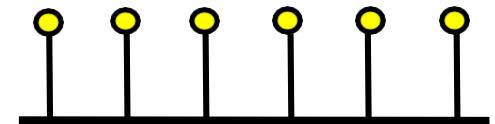
- **Connects all inputs to all outputs using a single switch?**

- Diameter is 1
- Degree is N
- No fault tolerance: single point of failure
- Low performance: bandwidth is $O(1)$



- **Bus: the interconnect is mostly just wires**

- Only one transaction in progress at a time
- Frequency affected by physical limitations



Diameter: the longest shortest path between any two nodes in the network.

互联架构

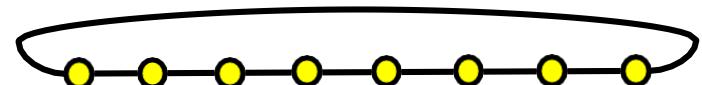
- **Array: connected by bidirectional links**

- Diameter is $N - 1$
 - Average distance is about $(N - 1)/2$
 - Bisection width is 1



- **Ring: connecting the two ends of an array**

- Assume **unidirectional** links:
 - Diameter is $N - 1$
 - Average distance is $N/2$
 - Bisection width is 1

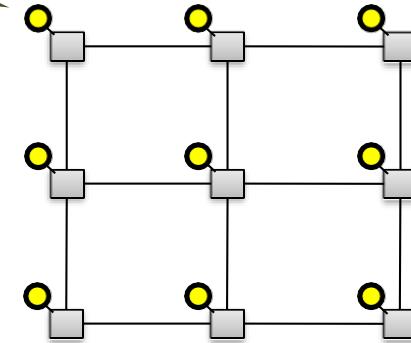


Distance : the number of hosts between two nodes in the network.

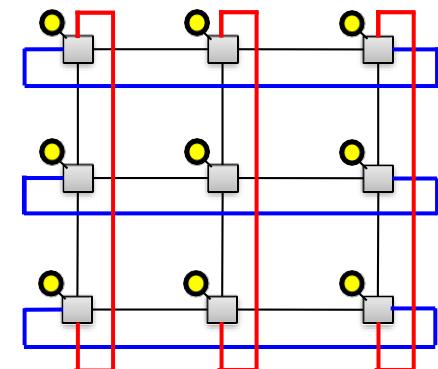
互联架构

Direct network have a host node connected to each switch

- **Mesh:** nodes are connected as a grid
 - Multiple routes between a pair of nodes
 - Non-uniform node-to-node latency
 - Cost of interconnect: $O(N)$

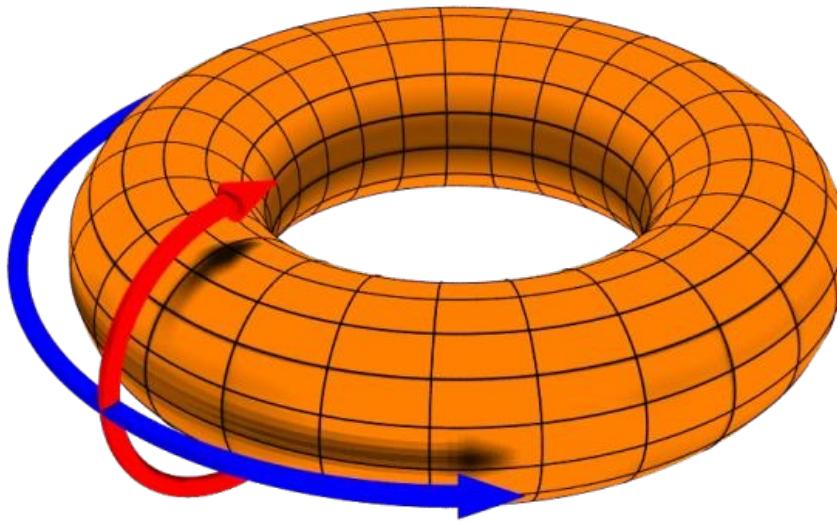


- **Torus:** adding end-round direct connections
 - An extension of the 2D mesh
 - A regular torus (环) has long warp-around links
 - Slightly lower latency while higher cost



互联架构

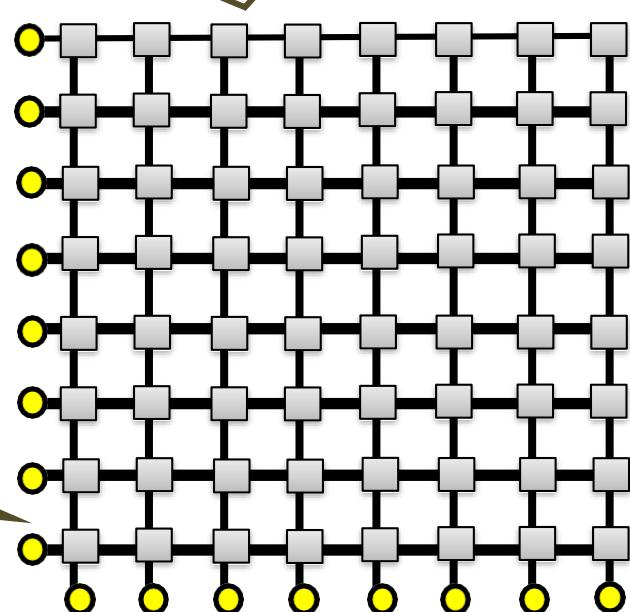
Torus: Reduces the diameter of a mesh network



互联架构

- **Crossbar switch (交叉开关矩阵)**
 - A type of fully-connected network
 - Every node connected to all others
 - $O(N)$ bandwidth
 - Cost of interconnect: $O(N^2)$
 - Good for small number of nodes

Crosspoint switch complexity increases quadratically with the number of ports

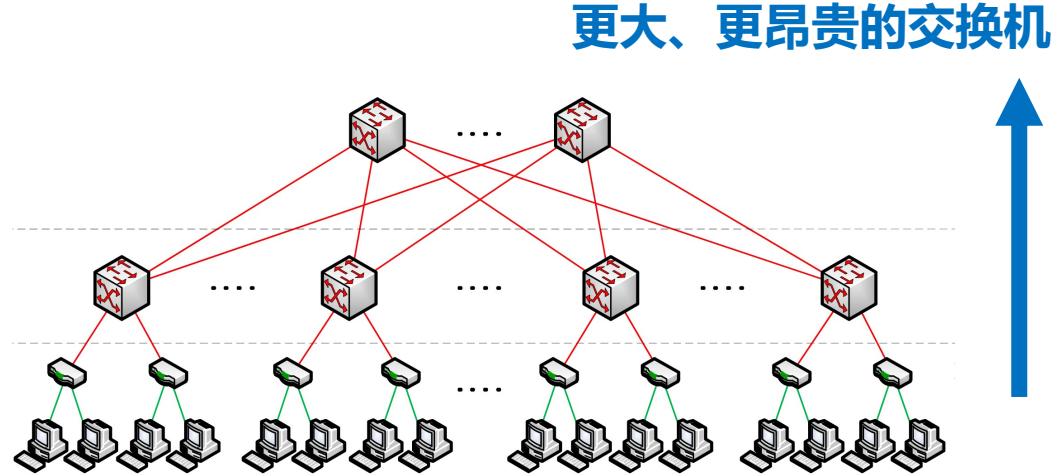
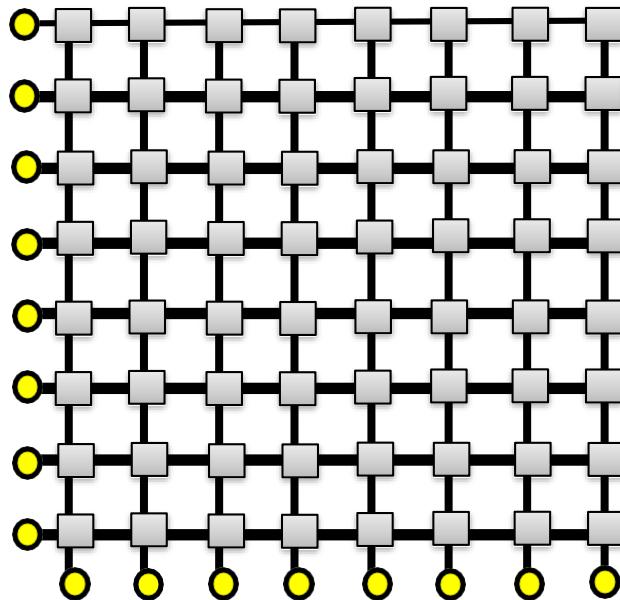


Indirect network have hosts connected only to certain switches

数据中心经典网络架构

Clos 网络架构

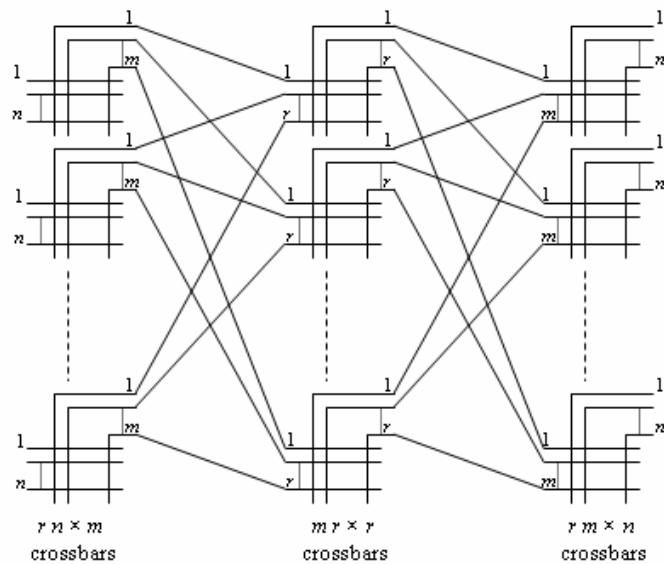
- Clos 网络是一种**非阻塞的多级交换网络** (non-blocking, multistage switching network)
- 在1950年代引入，以**提高电话交换网络的效率并降低其成本**（在此之前，交换机的数量须等于输入和输出数量的乘积，即**n的平方 n^2** ）
- 用多个**小规模、低成本**的交换机构建复杂、大规模的网络架构



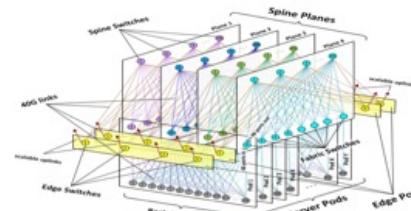
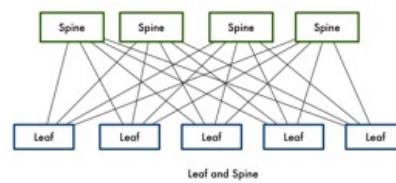
Clos 网络架构

□ 克洛斯利用数学理论证明，有可能使用更少的交换机实现非阻塞连接性，被称为 Fabric

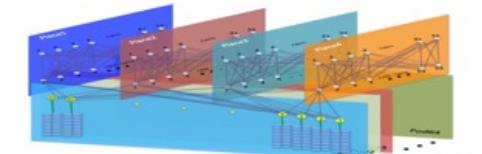
□ 至今，Clos 网络仍然是许多互联结构的基础，包括驱动云计算的大规模数据中心中的结构



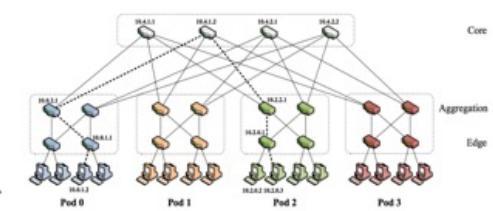
Three-stage Clos network



Reference: <https://code.fb.com/production-engineering/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/>



Reference: <https://engineering.linkedin.com/blog/2016/03/the-linkedin-data-center-100g-transformation>

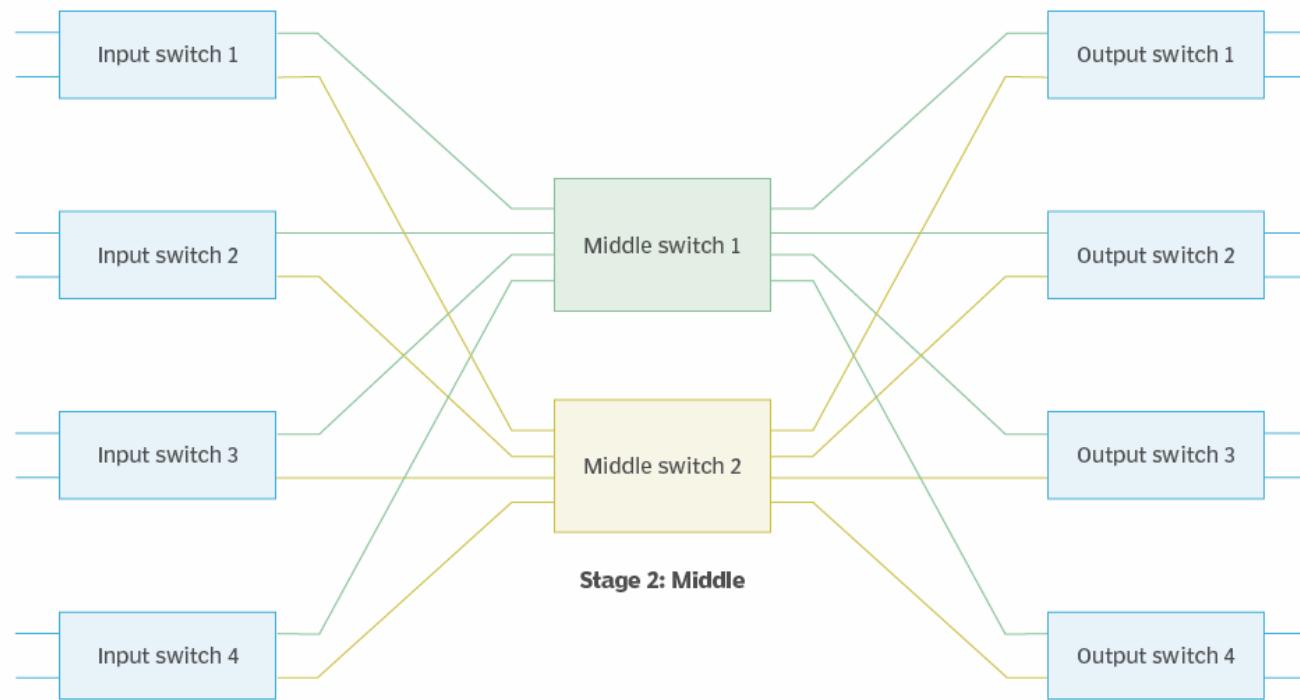


Reference: [A Scalable, Commodity Data Center Network Architecture](https://code.fb.com/production-engineering/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/)

Clos 网络架构

为了实现该连接性，交换机被组织成一个三阶段的架构：

- Ingress (输入) 阶段
- Egress (输出) 阶段
- Middle (中间) 阶段

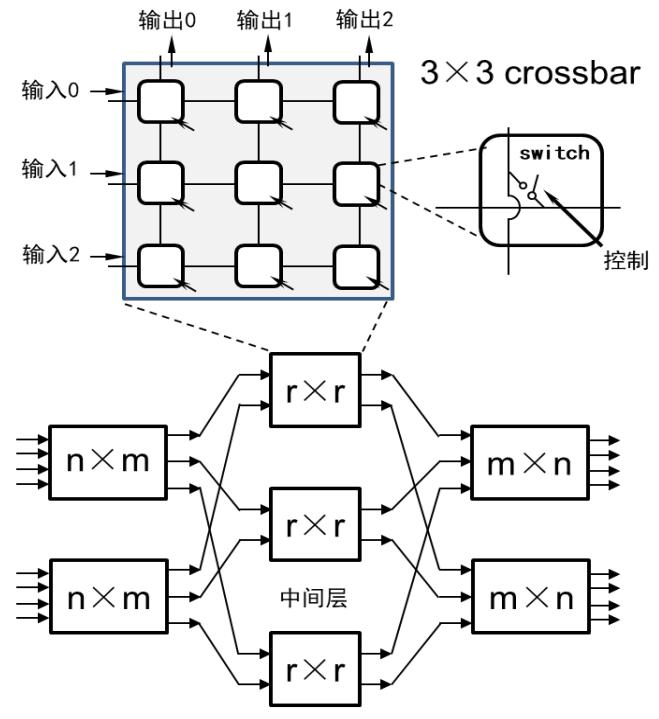
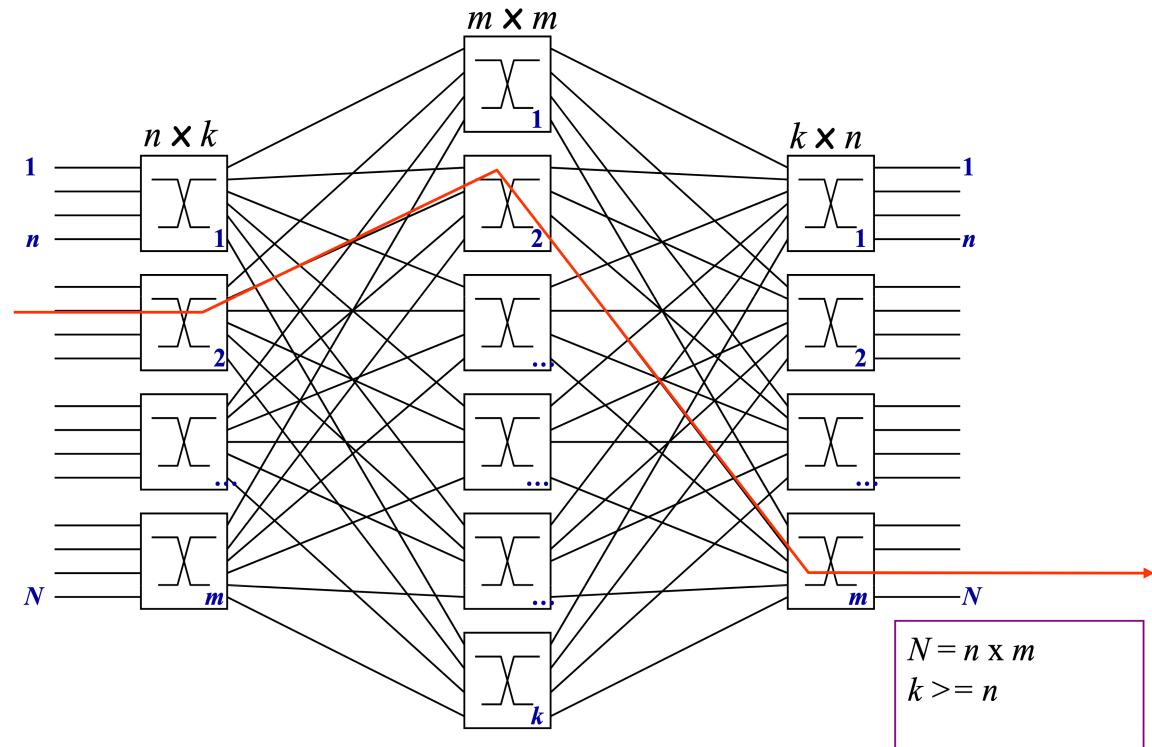


Three-stage Clos network

Clos 网络架构

□ 三阶段 Clos 网络由 3 个整数定义： n, m, k

- Ingress 阶段有 m 个交换机，每个包含 n 个输入和 k 个输出
- Middle 阶段有 k 个交换机，每个包含 m 个输入和 m 个输出
- Egress 阶段有 m 个交换机，每个包含 k 个输入和 n 个输出



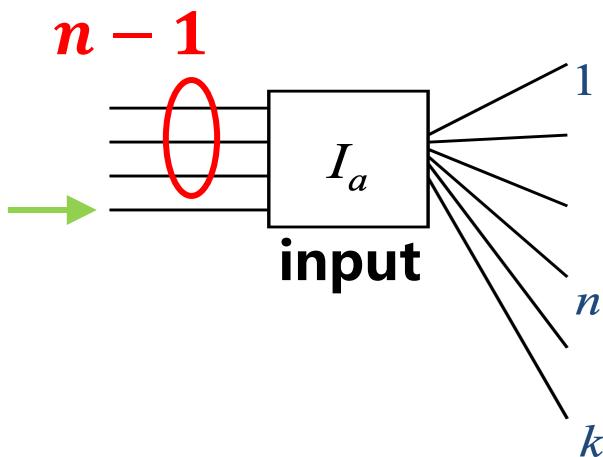
Clos(n, m, r)结构。上例中 $n=4, m=3, r=2$ 40

Clos 网络架构

Clos 定理

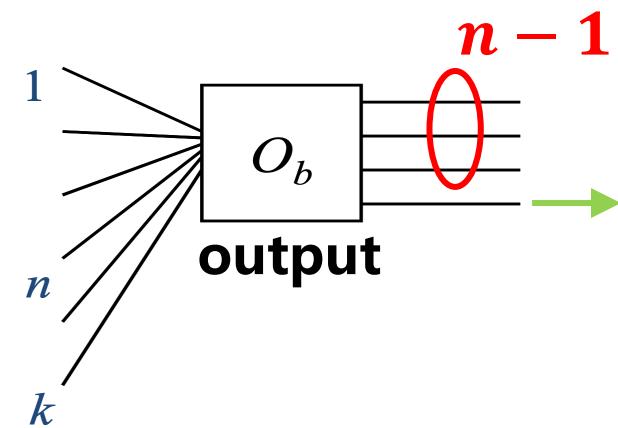
若 $m \geq 2n - 1$, 则 Clos 网络是严格
意义上的非阻塞

□ ==> 在 Ingress 交换机上一个未使用的输入总可以连接到 Egress 交换机上一个未使用的输出, 而无需重新排列



最差的情况是, 输入和输出的其他 $n - 1$ 条通道分别占用了不同的 Middle 交换机

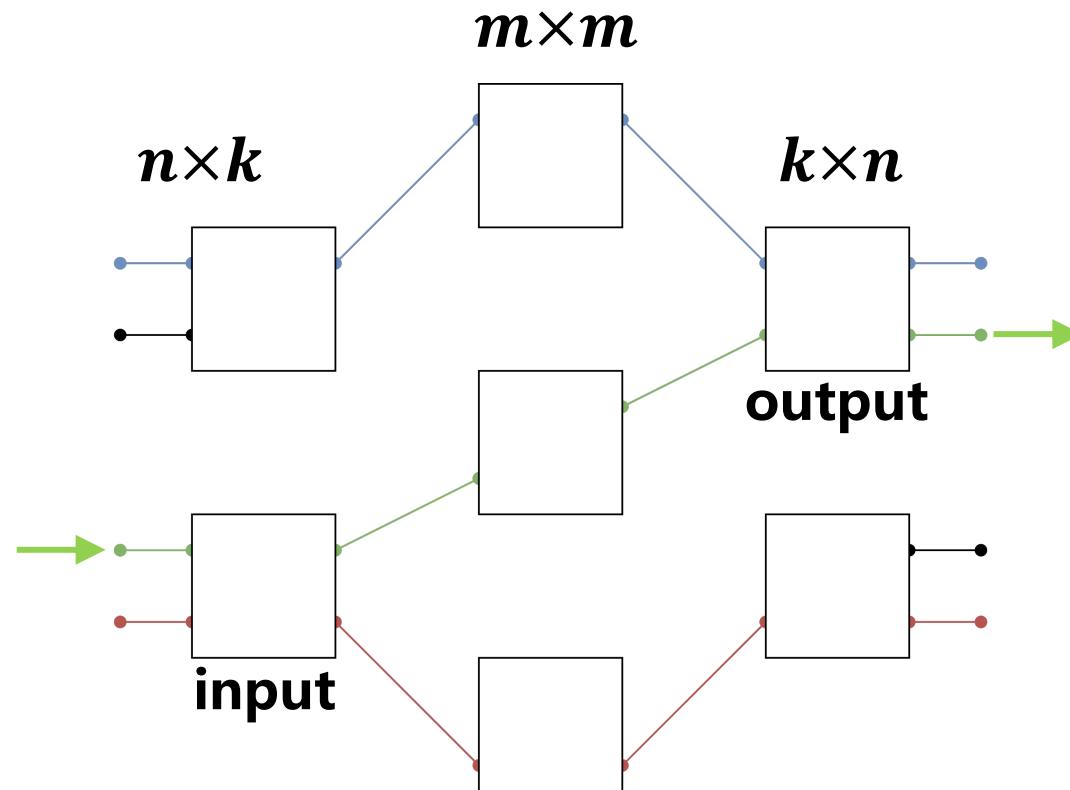
因此, 我们需要第 $(n - 1) + (n - 1) + 1$ 个 Middle 交换机, 即 $m \geq 2n - 1$



Clos 网络架构

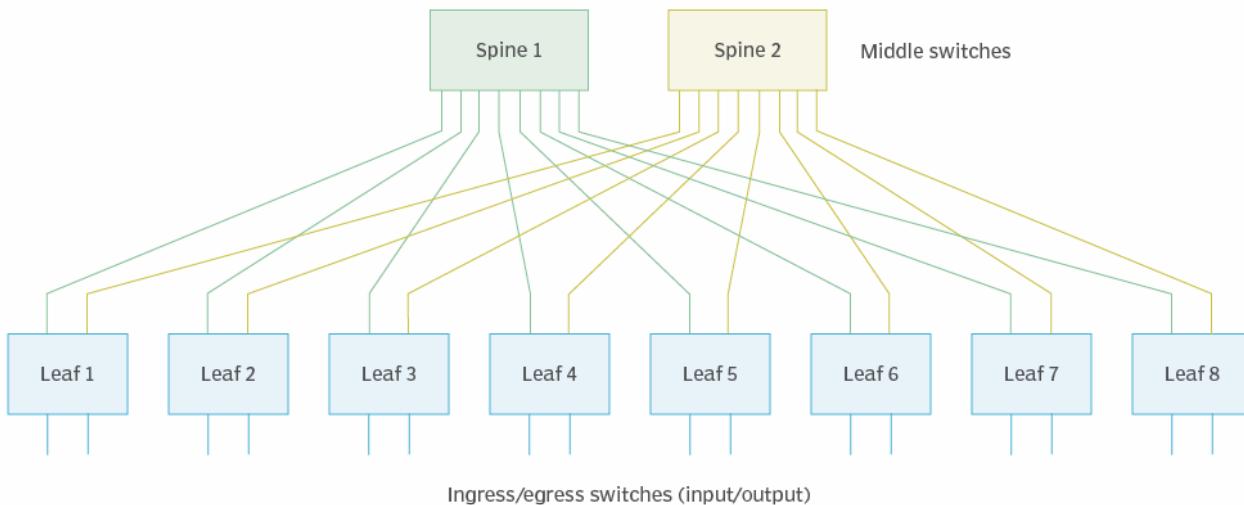
一个简单的例子

- $n = 2, k = 1, m = 1$
- 红色和蓝色的路径分别占用了不同的 Middle 交换机
- 因此，我们需要至少有 $m \geq 2n - 1 = 3$ 个 Middle 交换机

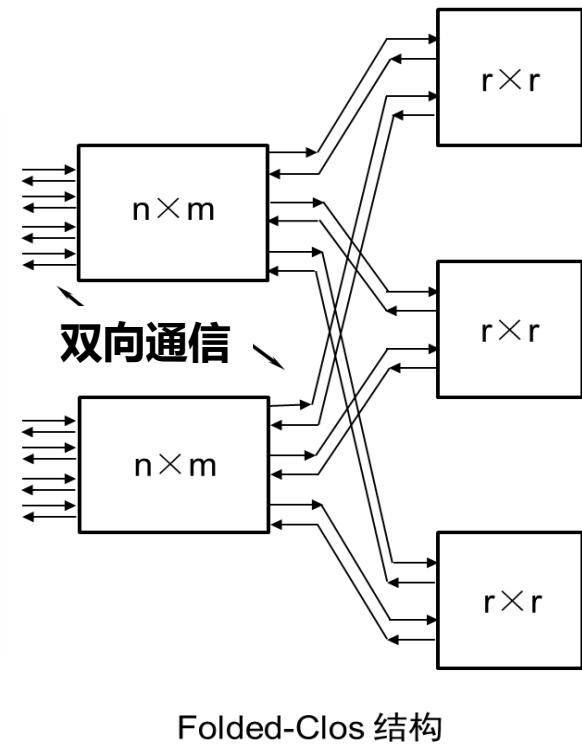


Clos 网络架构

- 如今的数据中心中，Clos多以叶脊 (leaf-spine) 形式布局
- 这种设计通常被称为折叠的 Clos 网络
- 更常被称为**胖树网络 (Fat-tree network)**



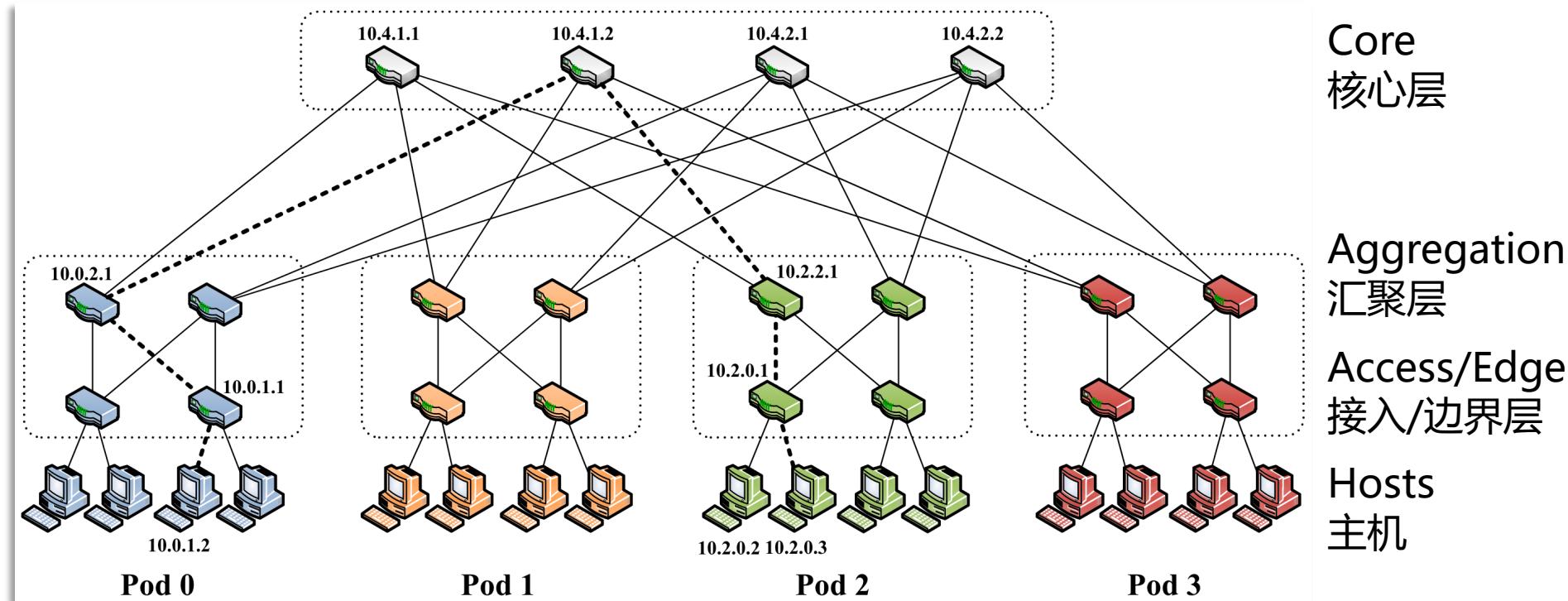
Clos leaf-spine network / Fat-tree network



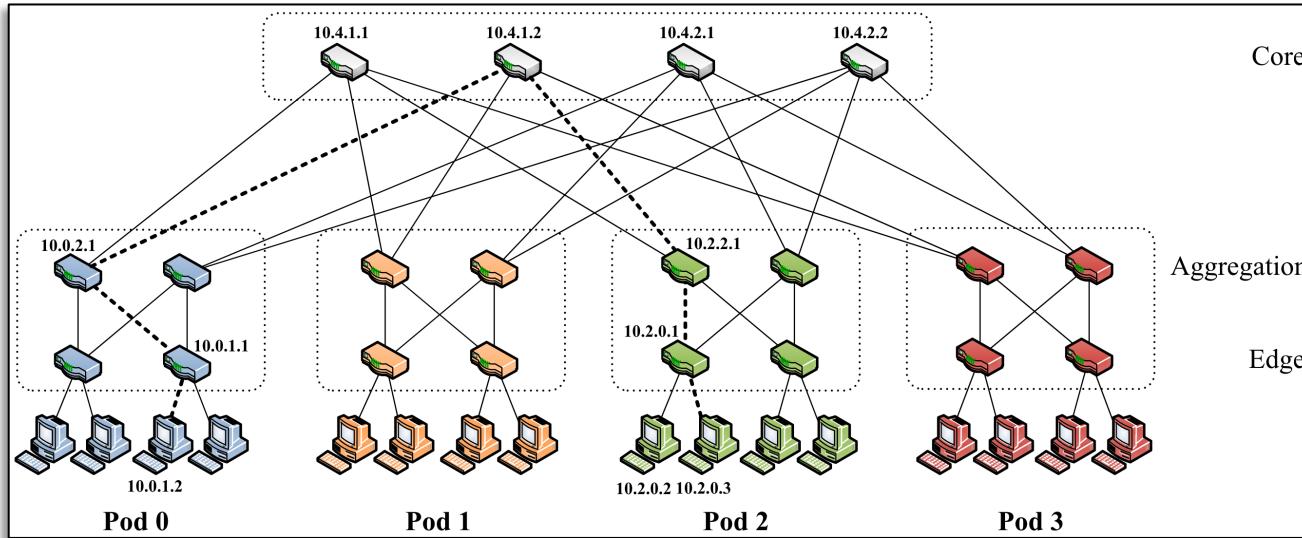
胖树网络架构

- 由三层网络交换机组成的多根树形网络拓扑结构

- 核心层连接汇聚层交换机和公网
- 汇聚层连接多个内部交换机
- 最底层服务器和接入交换机相连



Fat-tree 网络架构



An example Fat-tree architecture with $k=4$

□ k -ary Fat-tree:

- ✓ 每个交换机有 k 个端口（编号 0-3），网络包含 k 个 pod
- ✓ 每个 pod 包含两层交换机（汇聚和接入），每层有 $k/2$ 个交换机
- ✓ 每个接入交换机分别与 $k/2$ 个主机和 $k/2$ 个汇聚交换机相连
- ✓ 网络中共有 $(k/2)^2$ 个核心交换机，所有核心交换机的第 i 个端口连接第 i 个 pod
- ✓ 汇聚交换机的端口以 $k/2$ 的划窗连接核心交换机
- ✓ 能支持 $k^3/4$ 个主机，当 $k = 4$ 时， $k^3/4 = 4^3/4 = 16$

传统 Fat-tree 架构的优缺点

口传统 Fat-tree 架构的优点

- ✓ 所有交换机相同，可使用廉价的商用机降低成本，可随时替换
- ✓ 任意两个主机有 $(k/2)^2$ 条最短路径，主机获得完整带宽

口传统 Fat-tree 架构的两个问题

- ✓ 经典的 IP/Ethernet 网络只会构建**单路径路由协议**，会使得
Fat-tree 很快遇到**性能瓶颈**
- ✓ 大型数据中心网络中的**布线非常复杂**

改进 Fat-tree 架构

A Scalable, Commodity Data Center Network Architecture

Mohammad Al-Fares
malfares@cs.ucsd.edu

Alexander Loukissas
aloukiss@cs.ucsd.edu

Amin Vahdat
vahdat@cs.ucsd.edu

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

SIGCOMM ' 08

问题	解决思路
1. 性能瓶颈	升级版转发协议，提升扇出效率
2. 布线复杂	打包和放置技术（详见论文）

课堂问答

□与 10.110.12.29 mask 255.255.255.254 属于同一网段的主机 IP 地址是 ()

- A. 10.110.12.0
- B. 10.110.12.28
- C. 10.110.12.30
- D. 10.110.12.31

回顾子网掩码

- 将一个大型网络分割成许多小的子网
- 用于标识 IP 地址中的**网络和主机部分**，确定目标网络

IP地址	子网掩码
192.168.1.100	255.255.255.0



AND运算

192.168.1.0

- 任何以 192.168.1.0 开头的IP地址都在同一个子网中
- 也记作 192.168.1.100/24 (前面 24 个 1)

升级版转发协议 – 两级路由表

□ TCP/IP 路由表

- Destination: 目标主机 IP 地址
- Gateway: 到达目标主机的网关或下一跳地址

Routing tables						
Destination	Gateway	Flags	Refcnt	Use	Interface	
140.252.13.65	140.252.13.35	UGH	0	0	emd0	
127.0.0.1	127.0.0.1	UH	1	0	lo0	
default	140.252.13.33	UG	0	0	emd0	
140.252.13.32	140.252.13.34	U	4	25043	emd0	

□ 两级路由表

▪ 第一级是前缀查找

✓ 用于将拓扑向下路由到端主机

▪ 第二级是后缀查找

✓ 用于向核心路由

✓ 扩散和分散交通

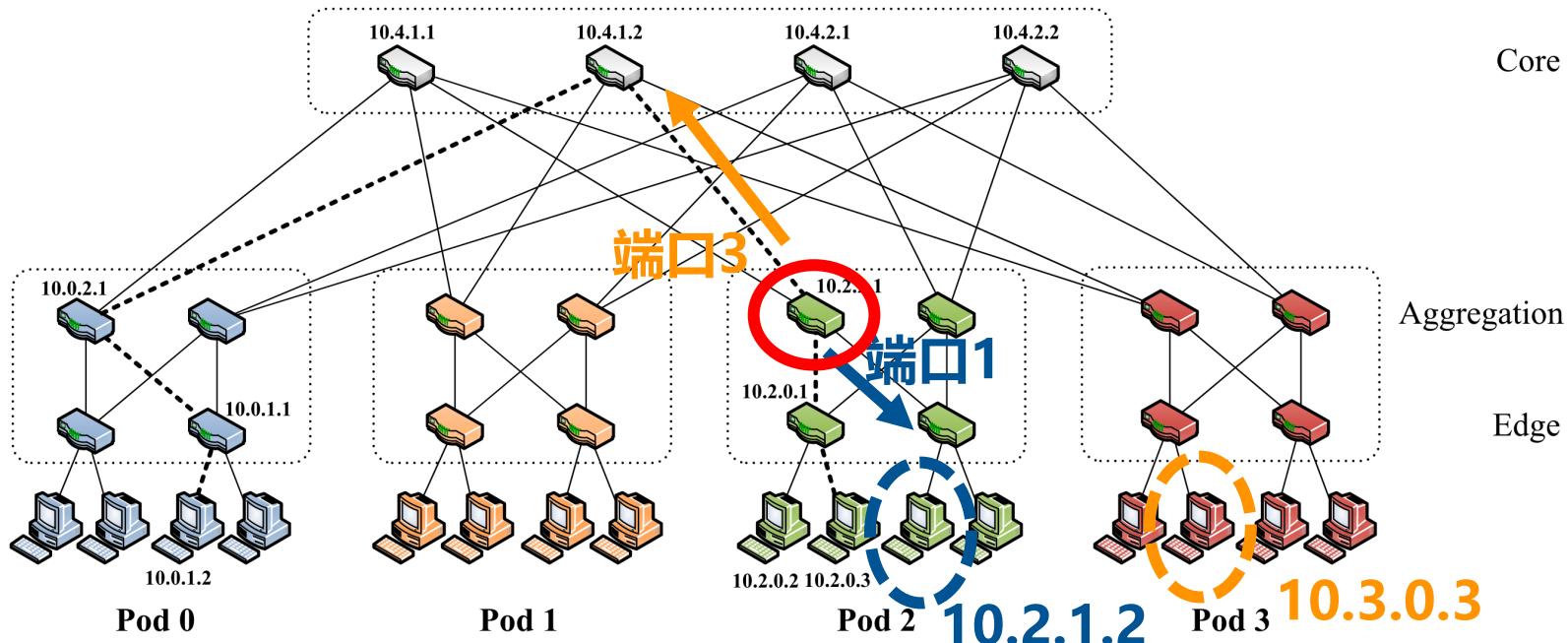
✓ 通过对相同的端主机使用相同的端口来维护数据包顺序

Prefix	Output port
10.2.0.0/24	0
10.2.1.0/24	1
0.0.0.0/0	

→

Suffix	Output port
0.0.0.2/8	2
0.0.0.3/8	3

两级路由表例子



Prefix	Output port
10.2.0.0/24	0
10.2.1.0/24	1
0.0.0.0/0	

前缀查找 → 后缀查找

Suffix	Output port
0.0.0.2/8	2
0.0.0.3/8	3

10.2.2.1 的路由表

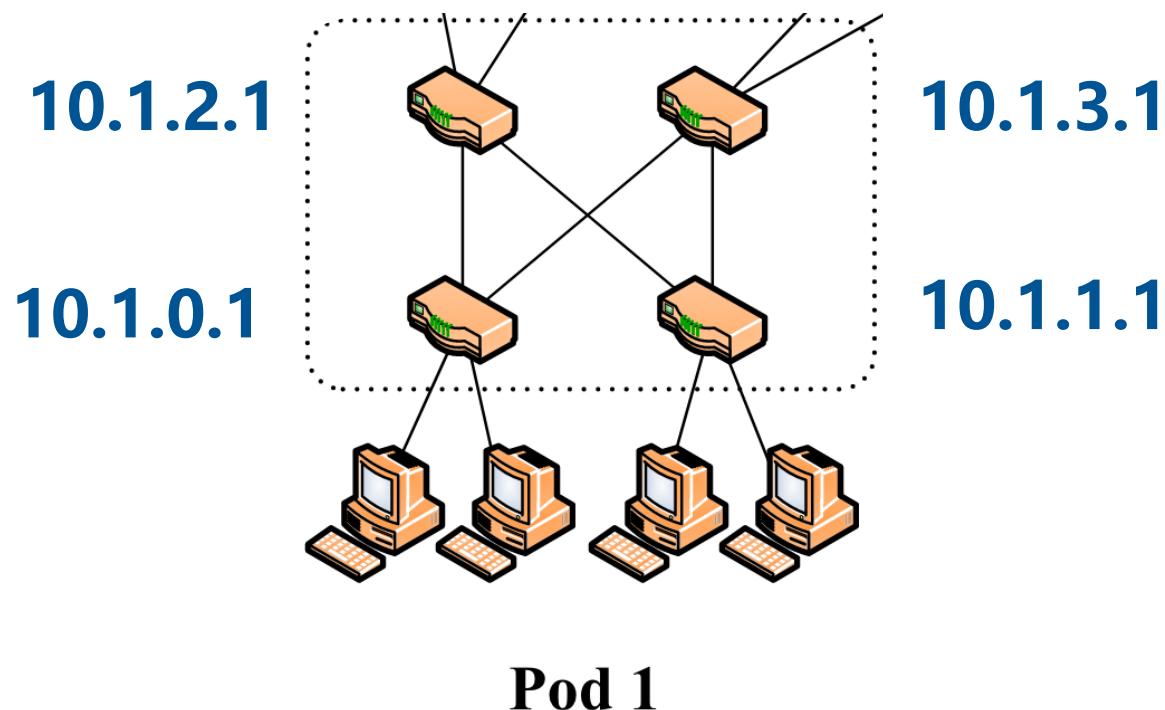
目的地IP地址为10.2.1.2的传入数据包在端口1上转发，而目的地IP名称为10.3.0.3的数据包则在端口3上转发

IP 地址编排

□ Pod 交换机 IP 地址: $10.pod.switch.1$

✓ pod 表示 pod ID ($[0, k - 1]$)

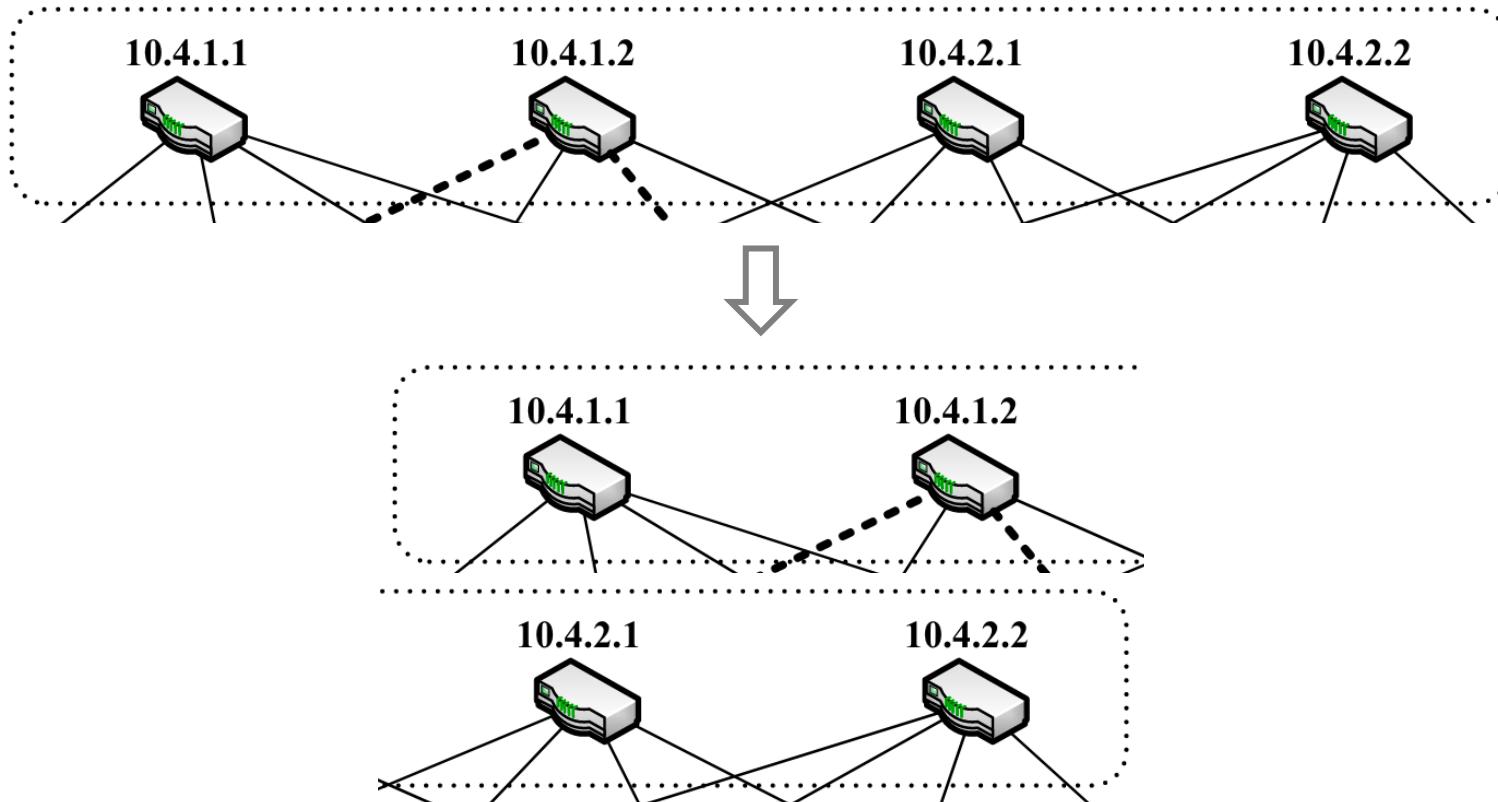
✓ $switch$ 表示 pod 内的 switch ID ($[0, k - 1]$) , 从左到右,
从下到上



IP 地址编排

□ Core 交换机的 IP 地址: $10.k.j.i$

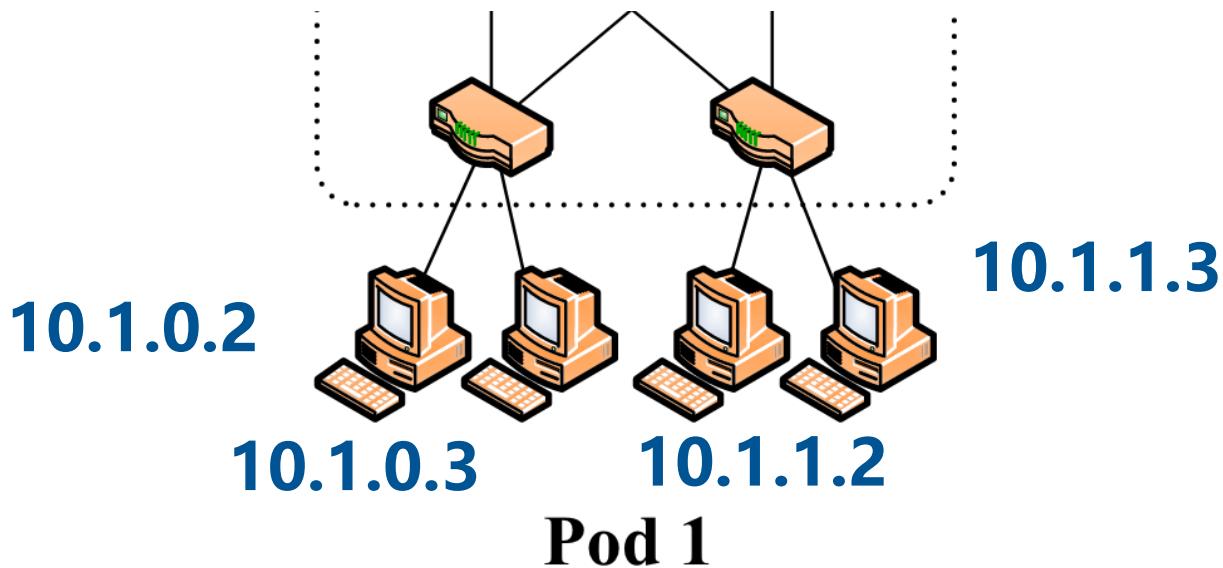
✓ j 和 i 表示交换机在 $(k/2)^2$ 个 Core 交换机网格中的坐标,
每个在区间 $[1, (k/2)]$



IP 地址编排

口 主机的 IP 地址: $10.pod.switch.ID$

- ✓ ID 表示主机在子网中的 ID ($[2, k/2 + 1]$)
- ✓ 每个 edge 交换机负责 $/24$ 子网中的 $k/2$ 个主机, $k < 256$



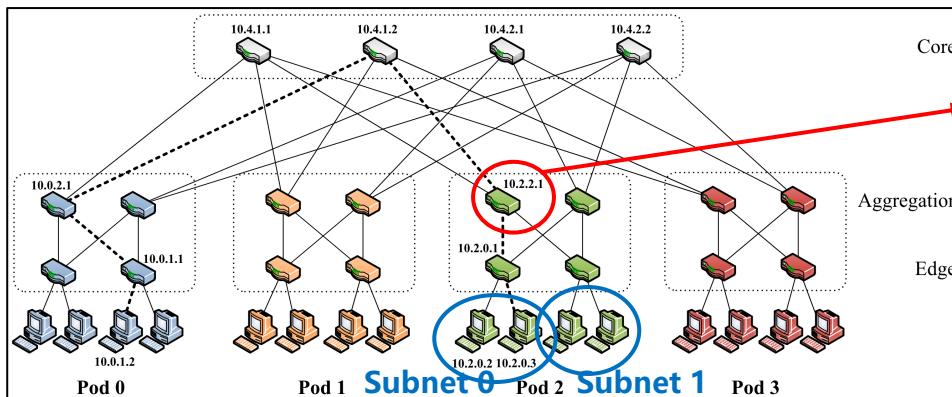
路由表生成

□ Aggregation 汇聚交换机路由表

```
1 foreach pod x in [0, k - 1] do
2   foreach switch z in [(k/2), k - 1] do
3     foreach subnet i in [0, (k/2) - 1] do
4       addPrefix(10.x.z.1, 10.x.i.0/24, i);
5     end
6     addPrefix(10.x.z.1, 0.0.0.0/0, 0);
7     foreach host ID i in [2, (k/2) + 1] do
8       addSuffix(10.x.z.1, 0.0.0.i/8,
9                 (i - 2 + z)mod(k/2) + (k/2));
10    end
11 end
```

Prefix	Output port
10.2.0.0/24	0
10.2.1.0/24	1
0.0.0.0/0	

Suffix	Output port
0.0.0.2/8	2
0.0.0.3/8	3



路由表生成

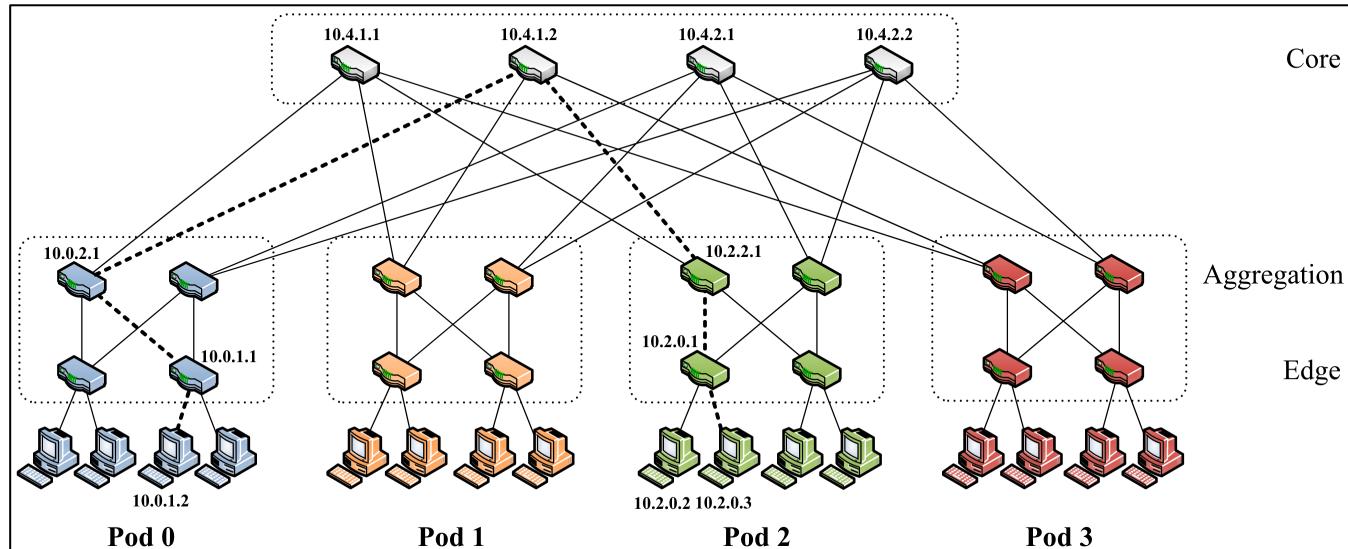
□ Access (接入) 交换机路由表

```
1 foreach pod  $x$  in  $[0, k - 1]$  do
2     foreach switch  $z$  in  $[(k/2), k - 1]$  do
3
4
5
6     addPrefix(10.x.z.1, 0.0.0.0/0, 0);
7     foreach host ID  $i$  in  $[2, (k/2) + 1]$  do
8         addSuffix(10.x.z.1, 0.0.0. $i$ /8,
9                     ( $i - 2 + z$ )mod( $k/2$ ) + ( $k/2$ ));
10    end
11 end
```

路由表生成

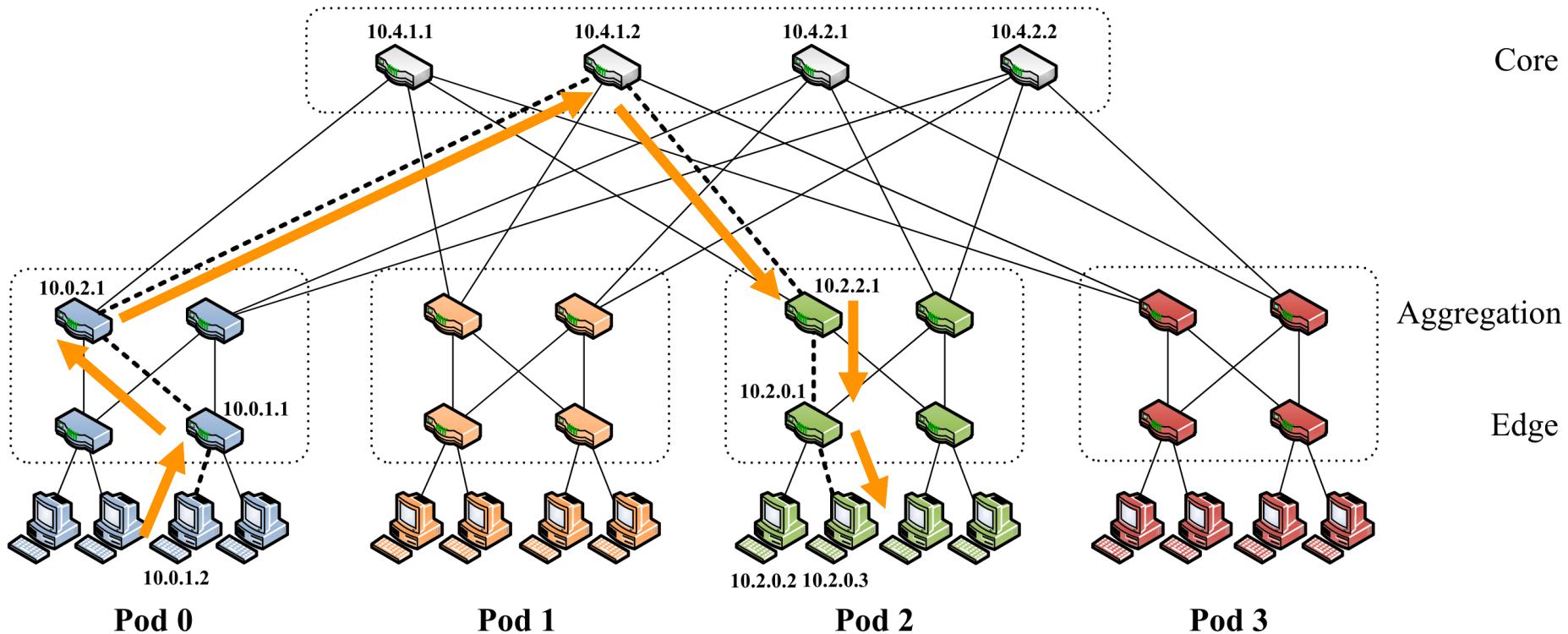
□Core (核心) 交换机路由表

```
1 foreach  $j$  in  $[1, (k/2)]$  do
2   foreach  $i$  in  $[1, (k/2)]$  do
3     foreach destination pod  $x$  in  $[0, (k/2) - 1]$  do
4       addPrefix( $10.k.j.i, 10.x.0.0/16$ , x);
5     end
6   end
7 end
```



网络容错

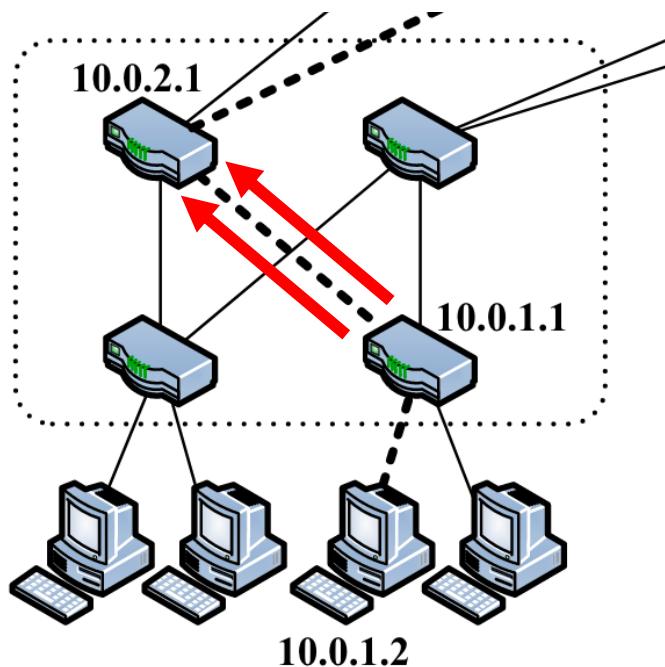
□一条网络流的路径



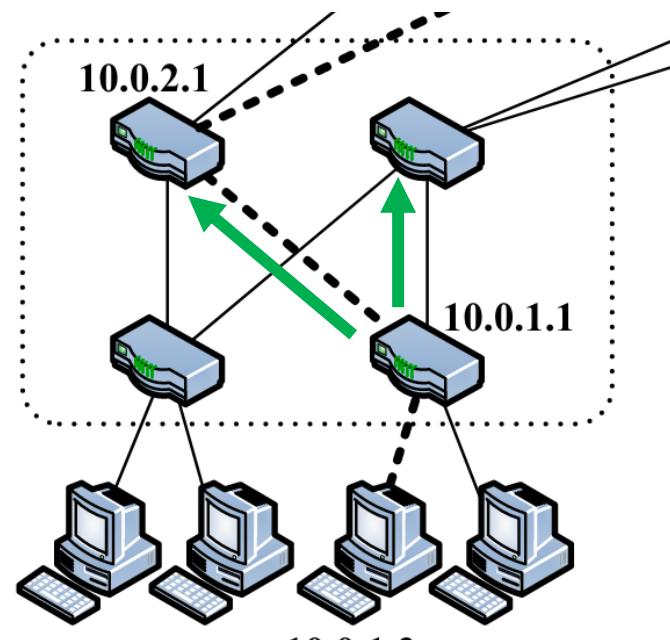
两级路由表分析

□ 能根据 IP 地址充分利用多条最短路径，但主机粒度的路由可能导致网络流冲突

□ 路由规则固定，不能根据网络负载情况动态调整



流量使用同一路径



流量使用不同路径

流粒度的路由改进

口流分类 (Flow classification)

- ✓ 消除本地阻塞
- ✓ 识别同一流的后续数据包，并在同一传出端口上转发它们
- ✓ 接入交换机定期重新分配流量的输出端口，以最大限度地减少不同端口的总流量之间的差异

口流调度 (Flow scheduling)

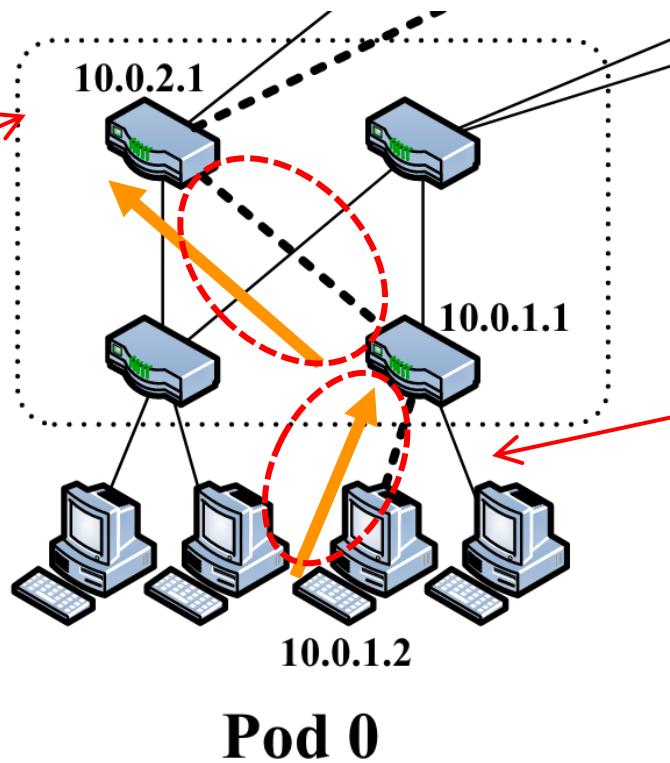
- 消除全局阻塞
- 大流才是决定网络对分带宽的关键
- Edge 交换机检测大流并发送信号给中央调度器
- 调度器分配大流路径，防止大流共享相同的链路

网络容错

口主机到 Pod 的链路故障

此链路故障则汇聚交换机通知接入交换机避免通过该路径发送数据

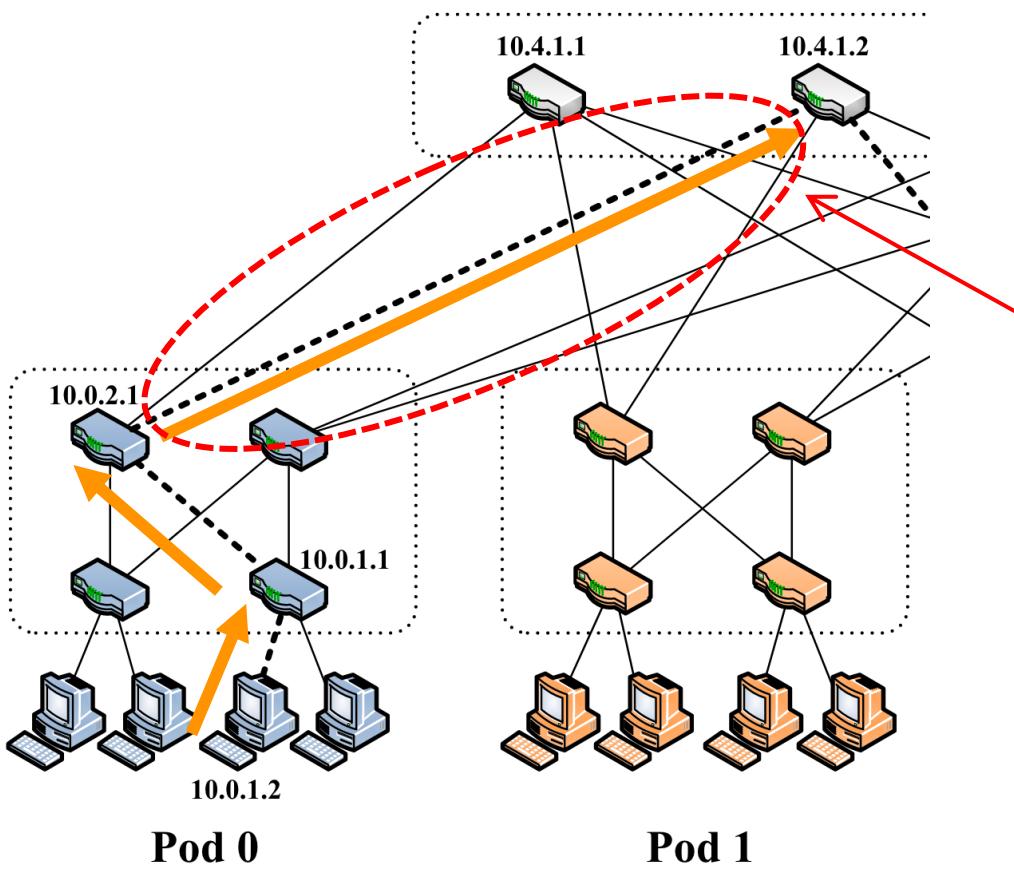
接入交换机标记此路径不可用



此链路故障则主机无法发送数据，需修复该链路

网络容错

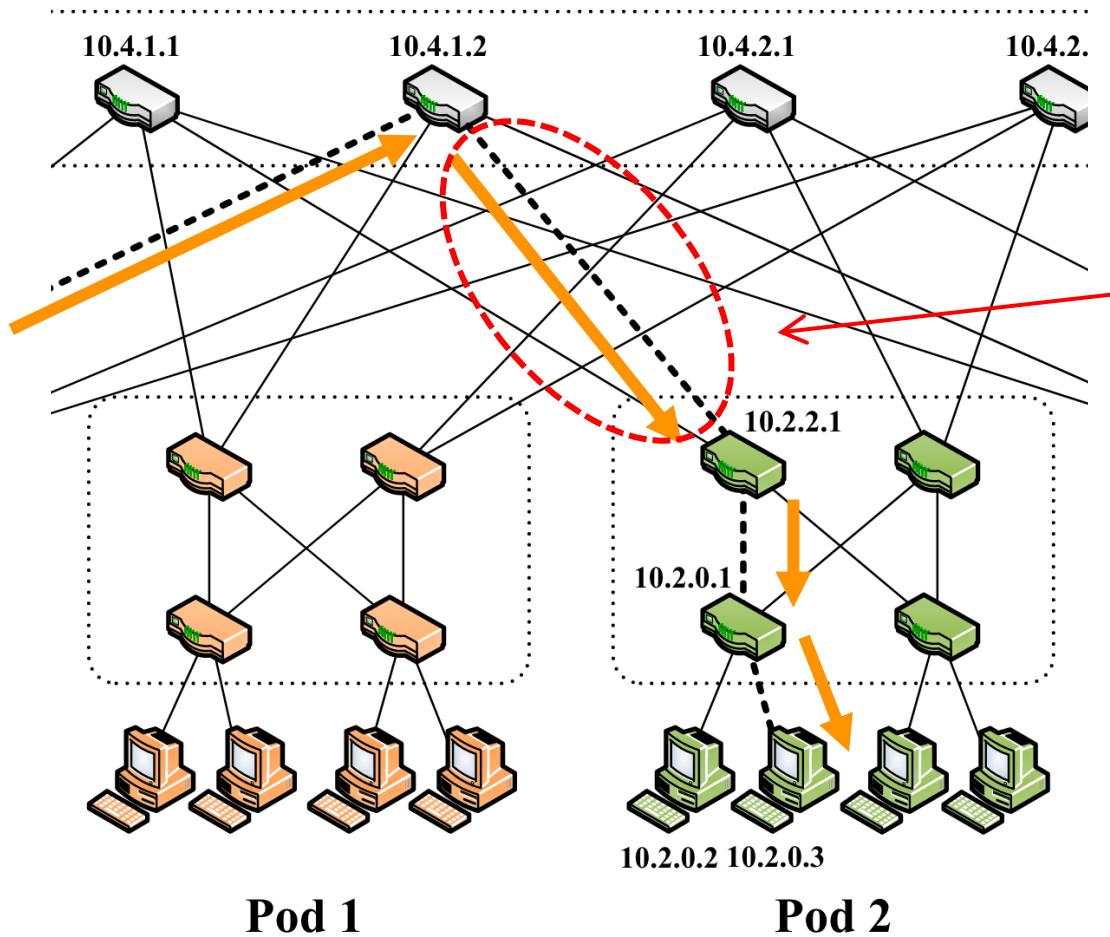
□ Pod 到核心交换机的链路故障



此链路故障则**核心交换机**通知**汇聚交换机**避免通过该路径发送数据

网络容错

核心交换机到 Pod 链路故障



此链路故障则核心交换机通知相连的汇聚交换机该路径不可用

汇聚交换机避免通过该核心交换机发送数据到目标Pod

Fat-tree 网络架构的缺点



扩展性问题



路由/布线复杂度高



成本较高

大量的新的网络架构被提出

网络拓扑	规模	带宽	容错性	扩展性	布线复杂性	成本	兼容性	配置开销	流量隔离	灵活性
FatTree	中	中	中	中	较高	较高	高	较高	无	低
VL2	大	大	中	中	较高	较高	中	较高	无	中
OSA	小	大	差	中	较低	较高	低	中	无	高
WDCN	小	大	较好	中	较低	中	中	中	无	高
DCell	大	较大	较好	较好	高	较高	中	较高	无	较高
FiConn	大	较大	较好	较好	较高	中	中	较高	无	较高
BCube	小	大	好	较好	高	较高	中	较高	无	较高
MDCube	大	大	较好	较好	高	高	中	较高	无	较高



中山大學 软件工程学院
SUN YAT-SEN UNIVERSITY SCHOOL OF SOFTWARE ENGINEERING

谢谢

陈壮彬
软件工程学院
chenzhb36@mail.sysu.edu.cn