* **第1章 云计算概述**

1. 定义

云计算是分布式计算的一种特殊形式，它引入效用模型来远程供给可扩展和可测量的资源。

云计算是一种信息技术（IT）范例，它可以无所不在地访问系统资源和高级服务的共享池，这些资源通常可以通过Internet以最少的管理工作来快速进行配置。

1. 基本属性、概念和术语、交付模式、经济合理性、基本机制（虚拟机，监控器，文件系统）、云架构（安全，性能，可得性）

* **基本属性：**

按需随选（on-demand）: 资源使用者可以根据需要单方面获取计算服务

资源池(Resource pooling)：云提供商的计算资源已合并

无处不在的网络访问（Ubiquitous network access）：云服务和资源可通过异构网络访问

位置独立（Location independence）：资源的位置不必为资源的使用者所关注。

快速弹性（Rapid elasticity）： 可以快速而灵活地提供功能。

现收现付（Pay-as-you-go）： 服务的消费者仅按其使用付费。

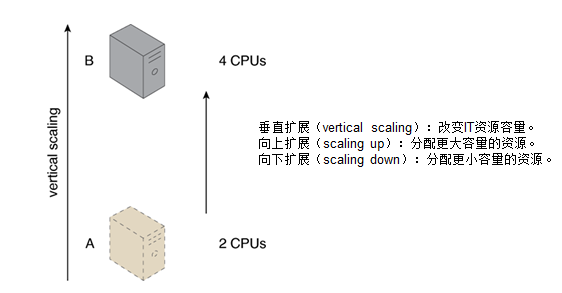
多租户（Multi-tenancy）： 可以在彼此不知道的多个使用者之间共享应用程序和资源。

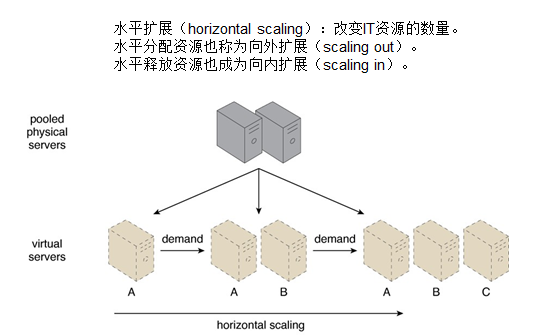
* **基本概念和术语——可扩展性**

从IT资源的角度来看，可扩展是指IT资源可以处理增加或减少的使用需求的能力。

水平扩展（horizontal scaling）： 水平分配资源也称为向外扩展（scaling out），水平释放资源也成为向内扩展（scaling in）。

垂直扩展（vertical scaling）：当一个现有IT资源被具有更大或更小容量的资源所替代，则称为垂直扩展。被具有更大容量的IT资源替代，称为向上扩展（scaling up），被具有更小容量的IT资源替代，称为向下扩展（scaling down）。





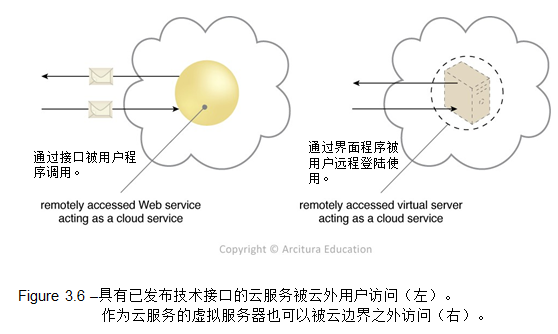
|  |  |
| --- | --- |
| **水平扩展** | **垂直扩展** |
| 更便宜（使用商品化的硬件组件） | 更昂贵（专用服务器） |
| IT资源立即可用 | IT资源通常为立即可用 |
| 资源复制和自动扩展 | 通常需要额外设置 |
| 需要额外IT资源 | 不需要额外IT资源 |
| 不受硬件容量限制 | 受限于硬件最大容量 |

* **基本概念和术语——云服务**

云服务（cloud service）是指任何可以通过云远程访问的IT资源。

云服务中的“服务”与其他IT领域中的服务技术（比如面向服务的架构，SOA）的“服务”含义更为宽泛。

并非云中所有的IT资源都可以被远程访问，其中有公开发布的API的软件程序可以专门部署为允许远程客户访问。

****

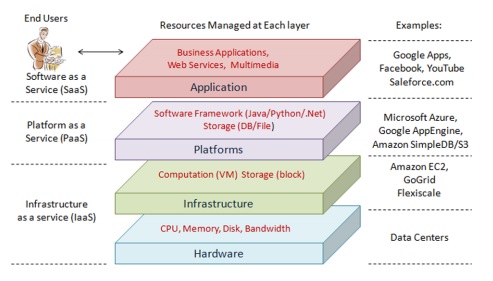
* **基本概念和术语——云服务用户**

云服务用户（cloud service consumer）是一个临时的运行时角色，由访问云服务的软件程序担任。

云服务用户常见类型：

1. 能够通过已发布的服务合同远程访问云服务的软件程序和服务
2. 运行某些软件的工作站、便携电脑和移动设备。

* **云服务交付模式 Service Model**

****

Software as a Service，软件即服务，简称SaaS，这层的作用是将应用作为服务提供给客户。E.g. e-mail services

Platform as a Service，平台即服务，简称PaaS，这层的作用是将一个开发平台作为服务提供给用户。E.g., Google App Engine, Microsoft Azure,

Infrastructure as a Service， 基础设施即服务，简称IaaS，这层的作用是提供虚拟机或者其他资源作为服务提供给用户。E.g., Amazon EC2

* **经济合理性 Economic Justification**

**Economies of scale（规模经济）**

大型数据中心的运行（按单位度量）比小型数据中心便宜。

在这种情况下，大型意味着100,000多个服务器

在这种情况下，小的表示少于10,000台服务器。

Reasons for Economies of Scale：

电力成本***Cost of power****.*

硬件成本***Hardware costs.***

基础设施人工成本***Infrastructure labor costs****.*

安全性和可靠性***Security and reliability***

**Utilization of equipment（设备利用）**

如何提高数据中心的利用率？

* 虚拟化允许将不同的应用程序置于同一位置

利用工作负载的变化来提高利用率

1. 随机访问(Random access):

最终用户随机访问应用程序。 更多的用户更有可能施加均匀的负载

1. 一天中的时间(Time of day):

将与工作场所相关的那些服务与与消费者相关的那些服务并置在一起。

不同地理位置之间的时差。

1. 资源使用模式(Resource usage patterns):

将较重的CPU服务与较重的I / O服务并置在一起

1. 不确定(Uncertainty) :考虑用量高峰

新闻事件，营销事件，体育事件

利用公共云来维持足够的容量以支持使用高峰

关键技术是分析负载变化模式，并利用该模式在资源上分配负载

**Multi-tenancy（多租户）**

多租户是一种软件体系结构，其中一个软件实例在服务器上运行并为多个租户提供服务,在多实例体系结构中，单独的软件实例代表不同的租户运行。

* **云机制** （虚拟机，监控器，文件系统）**Cloud Mechanisms**

**虚拟机Virtual Machine**

* 虚拟机具有与任何其他虚拟机隔离的地址空间。
* 从应用程序角度看，它看起来像一台裸机。
* 分配了IP地址并具有网络功能。
* 可以加载可以在主机处理器上执行的任何操作系统或应用程序。

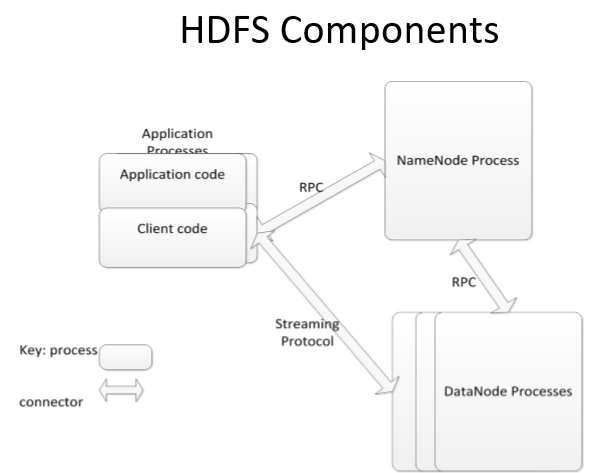
**监控器Hypervisor**

监控器是用于创建和管理虚拟机的操作系统E.g., VMWare, Xen, KVM

**文件系统File system**

文件系统：每个虚拟机都可以访问文件系统。

HDFS（Hadoop分布式文件系统）–一种广泛使用的开源云文件系统。



* **云架构（安全，性能，可得性） Cloud Architecture**

安全Security

•多租户带来了对非云环境的更多关注。

–疏忽的信息共享。 可能由于共享资源使用而共享信息。 例如。 如果重新分配磁盘，则磁盘上的信息可能会保留。

–虚拟机逃逸是指脱离虚拟机（管理程序）并与主机操作系统进行交互的过程

–拒绝服务攻击。 一个用户可以消耗主机服务器的资源，而拒绝其他用户使用。

性能Performance

•两种保证性能的方法

–负载平衡是在多个计算资源之间分配工作负载，以避免单个资源的过载

–自动扩展是一种方法，通过该方法通常可以根据活动服务器的数量来衡量计算资源的数量， 根据负载自动缩放

可用性Availability

•故障是云中的常见现象

–预计会有数千台服务器出现故障。

•云提供商确保云本身将保持可用，但有一些明显的例外。

•应用程序开发人员必须假设实例将失败，并在失败的情况下建立检测和更正机制。

* **第2章 云计算与大数据处理**
  + **并发控制（锁、基于时间戳协议）**

常用的并发控制方法

– 基于锁的并发控制

• 两阶段锁协议

两段锁协议规定所有的事务应遵守的规则：

　　① 在对任何数据进行读、写操作之前，首先要申请并获得对该数据的封锁。

　　② 在释放一个封锁之后，事务不再申请和获得其它任何封锁。

　　即事务的执行分为两个阶段：

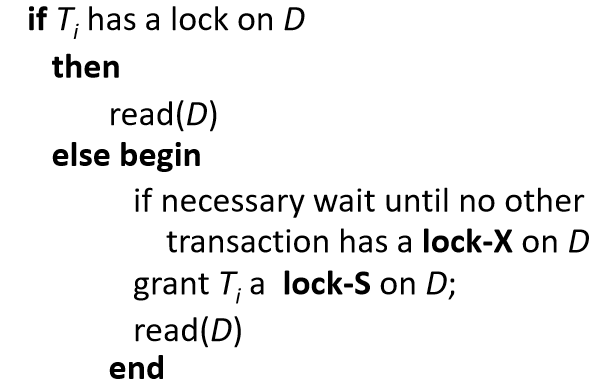
　　第一阶段是获得封锁的阶段，称为扩展阶段。

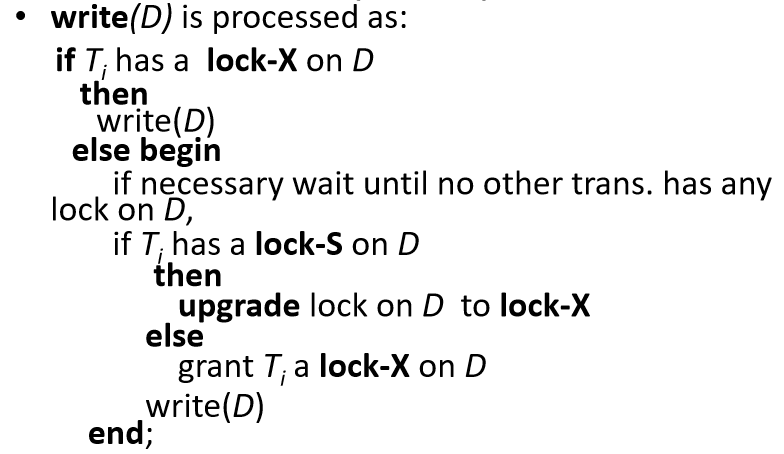
　　第二阶段是释放封锁的阶段，称为收缩阶段。

自动获取锁

•事务Ti发出标准的读/写指令，而没有显式的锁调用。

•操作read（D）的处理方式为：





•提交或中止后释放所有锁

– 基于时间戳的并发控制

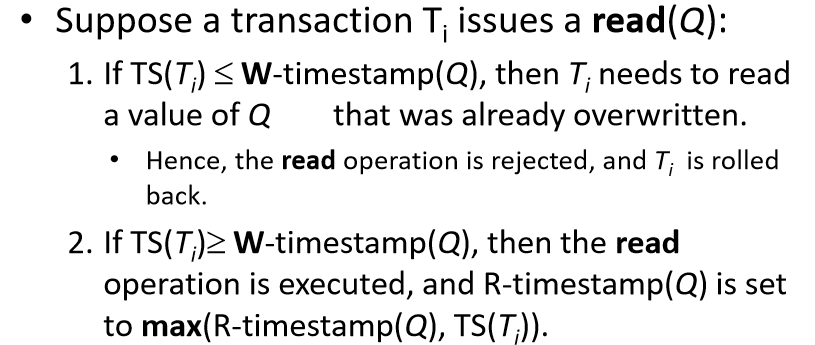
•每笔交易进入系统时都会发出一个时间戳。

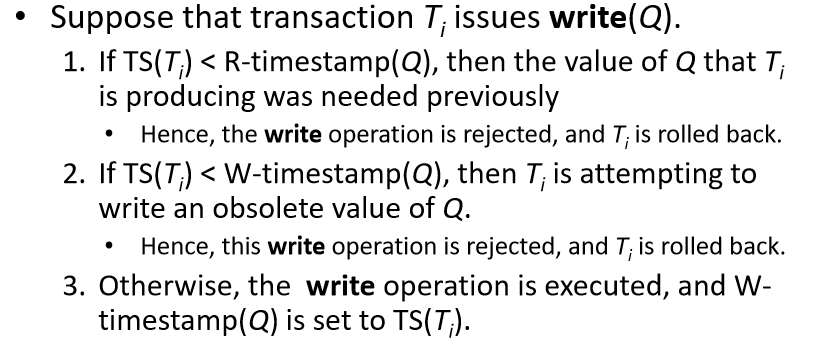
•基于时间戳的协议可确保按时间戳顺序执行任何冲突的读写操作。

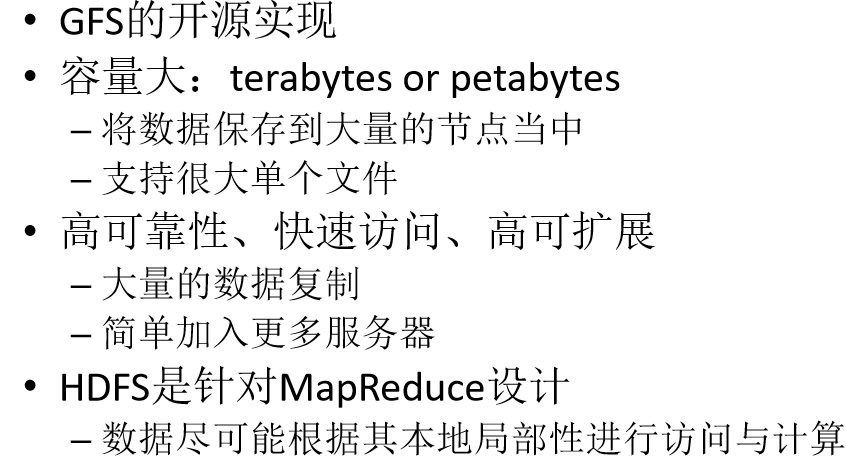
•为了确保这种行为，协议为每个数据Q维护两个时间戳值：

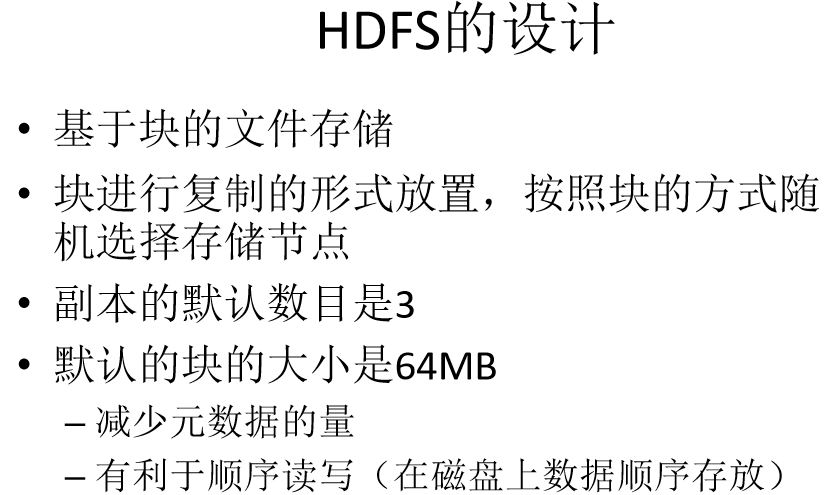
– W-timestamp（Q）是成功执行write（Q）的所有事务中的最大时间戳。

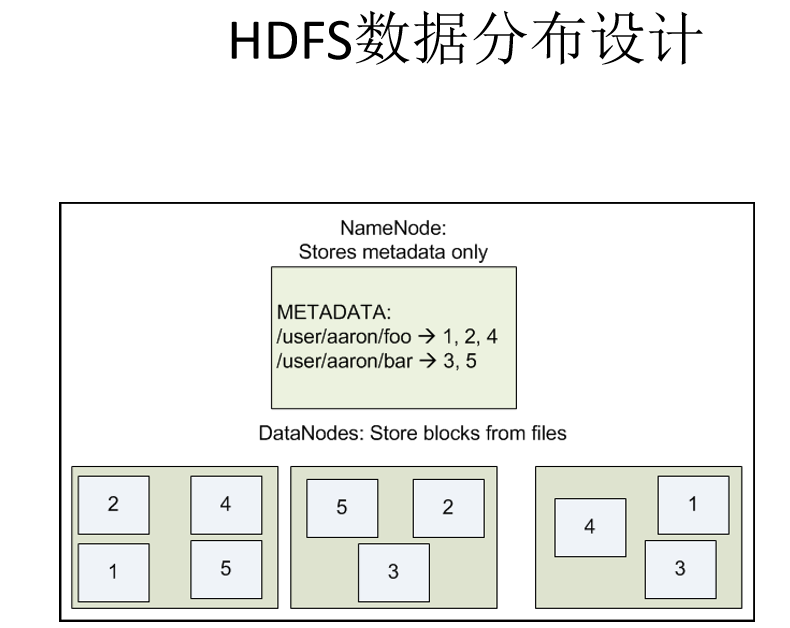
– R-timestamp（Q）是成功执行read（Q）的所有事务的最大时间戳。

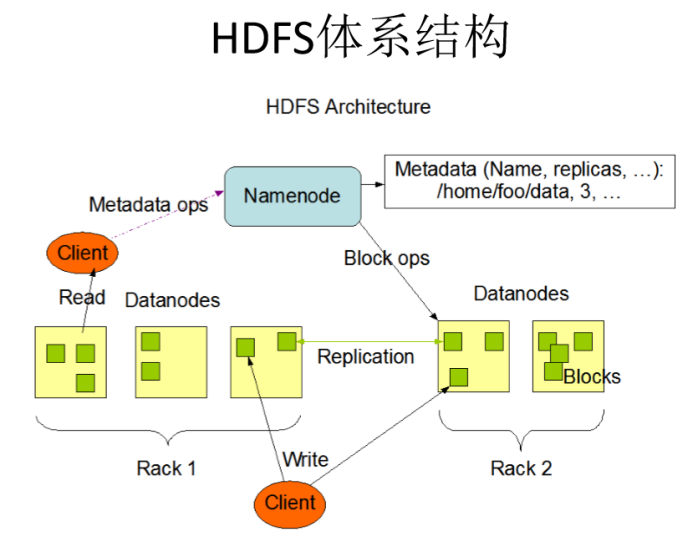




* + **分布式文件系统（HDFS）**
  + 

****

****

****

* + **NoSQL系统**

关键功能（优点）：

–非关系，不需要架构

–数据被复制到多个节点并可以分区：

•易于替换的下行节点•无单点故障

–水平可扩展

–便宜，易于实现

–庞大 写入性能

–快速键值访问

缺点：

–不完全支持关系功能

•没有联接，分组，按操作排序（分区内除外）

•没有分区间的参照完整性约束

–没有声明性查询语言（例如SQL）→更多编程–宽松的ACID（请参见 CAP定理）→更少的保证

–无法轻松与支持SQL的其他应用程序集成



NoSQL类别

1.键值•示例：DynamoDB，Voldermort，Scalaris

2.基于文档•示例：MongoDB，CouchDB

3.基于列•示例：BigTable，Cassandra，HBase

4.基于图•示例：Neo4J，InfoGrid

•“无模式”是大多数NoSQL存储系统的共同特征

•提供“灵活”的数据类型

* + **MapReduce**

•来自Google，[OSDI’04]•简单的编程模型

•功能模型

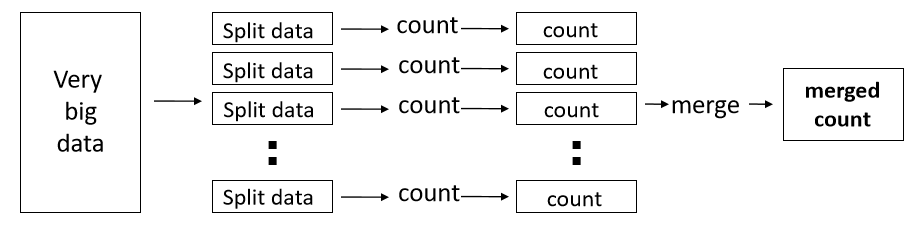
•用于大规模数据处理

–利用大量的商用计算机

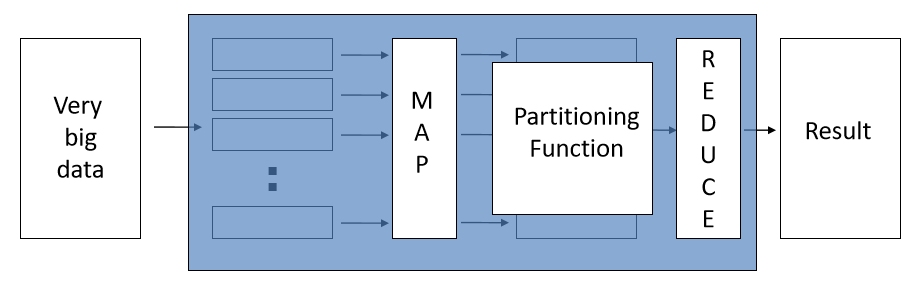
–以分布式方式执行过程

–提供高可用性

Distributed Word Count



Map+Reduce

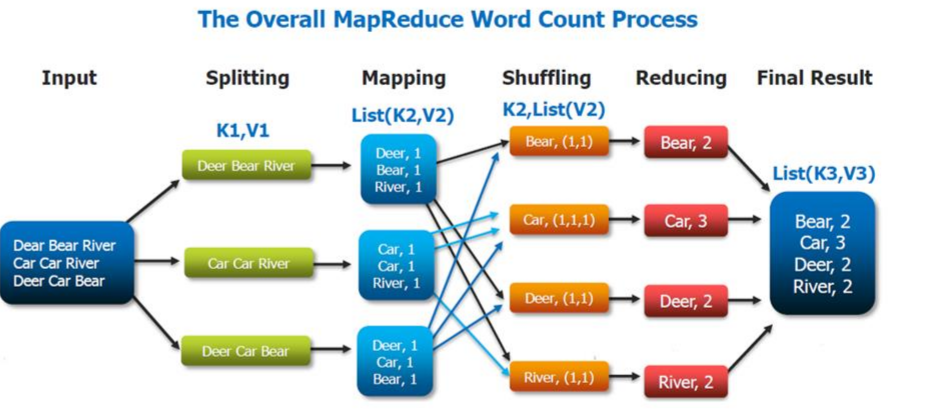


• Map: – Accepts input key/value pair – Emits intermediate key/value pair

•映射：–接受输入键/值对–发出中间键/值对

•Reduce : – Accepts intermediate key/value\* pair – Emits output key/value pair

•减少：–接受中间键/值\*对–发出输出键/值对



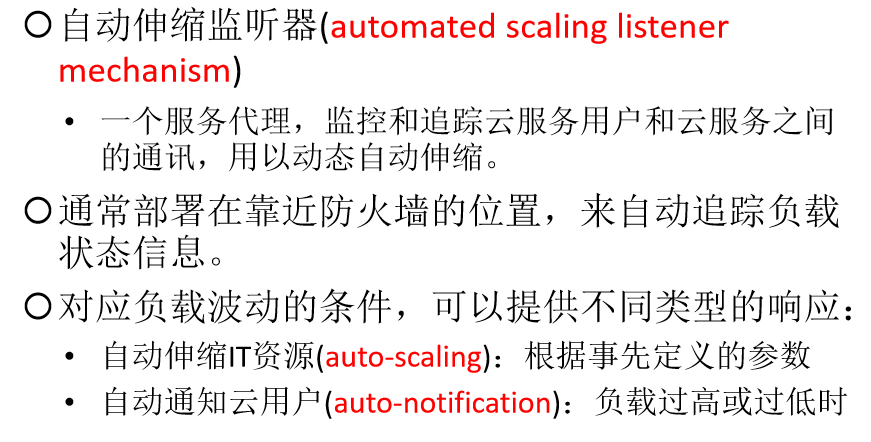
* **第3章 特殊云机制**
  + **云使用监控（三种实现方式）：（基于代理）**

监控代理（monitoring agent）是一个中间的时间驱动程序，对数据流进行透明的监控和分析。 

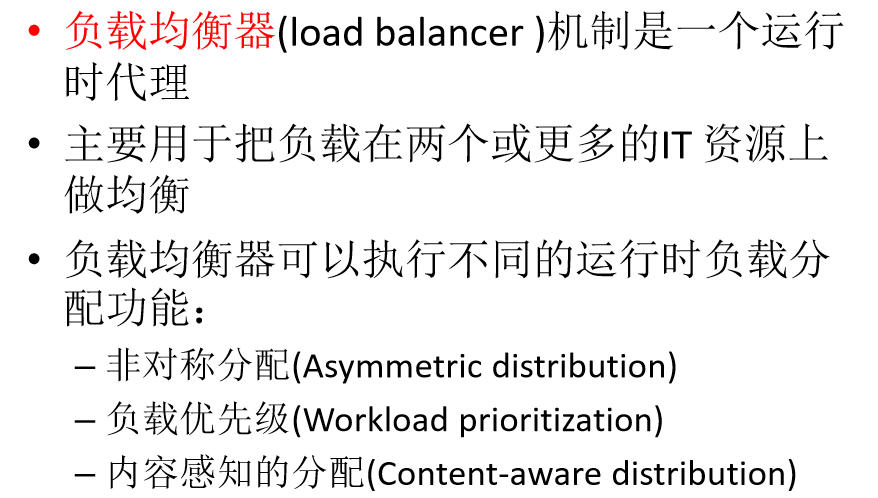
资源代理（resource agent）是一种处理模块，在资源软件级别监控预定义的且可观测事件的使用指标，比如：启动、暂停、恢复和垂直扩展。 

轮询代理（polling agent）是一种处理模块，通过轮询IT资源来周期 性地监控IT资源状态（比如正常运行时间和停机时间）。

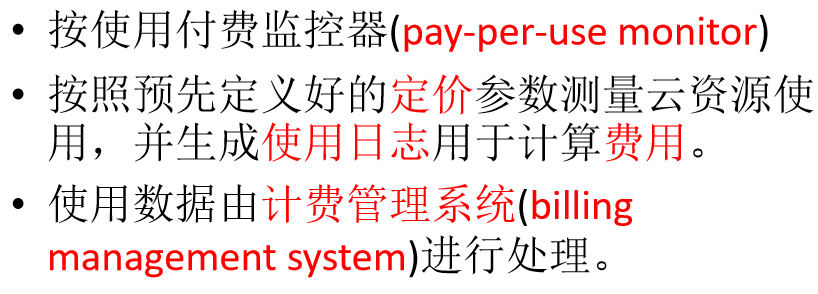
* + **自动伸缩监听器**

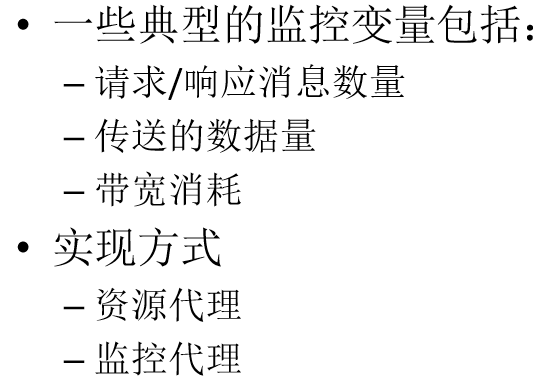
****

* + **负载均衡器**

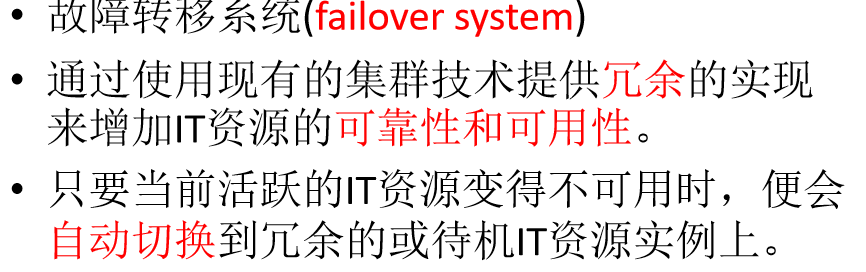
****

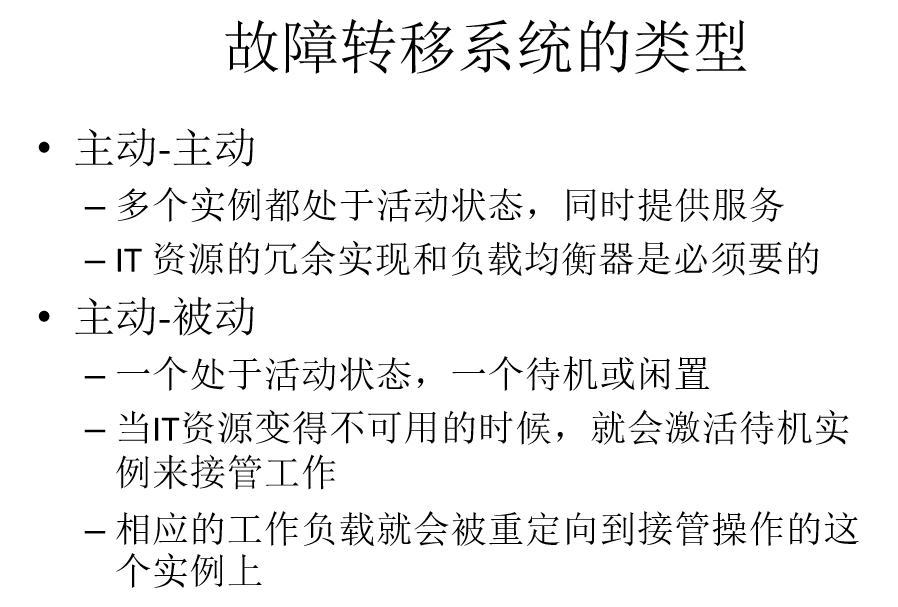
* + **按使用付费监控器**

****

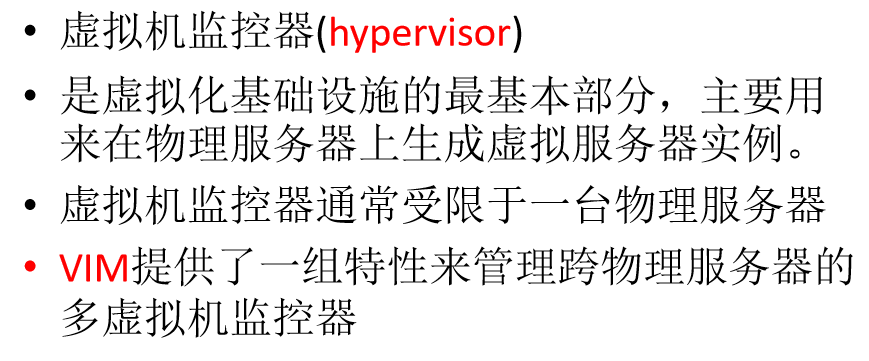
****

* + **故障转移系统**

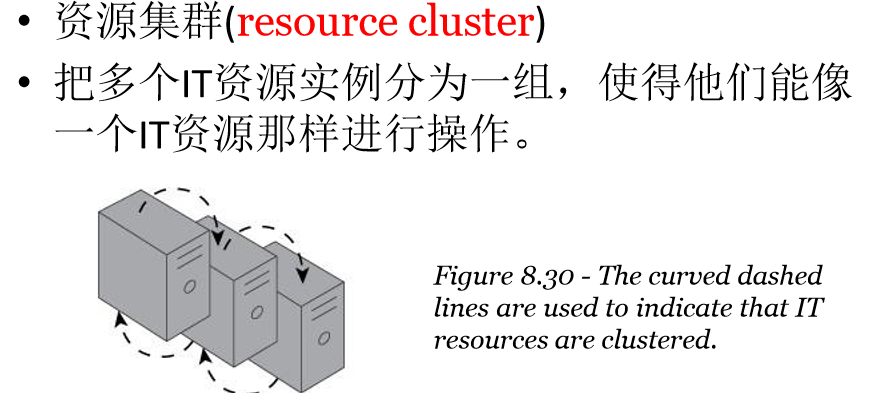
****

****

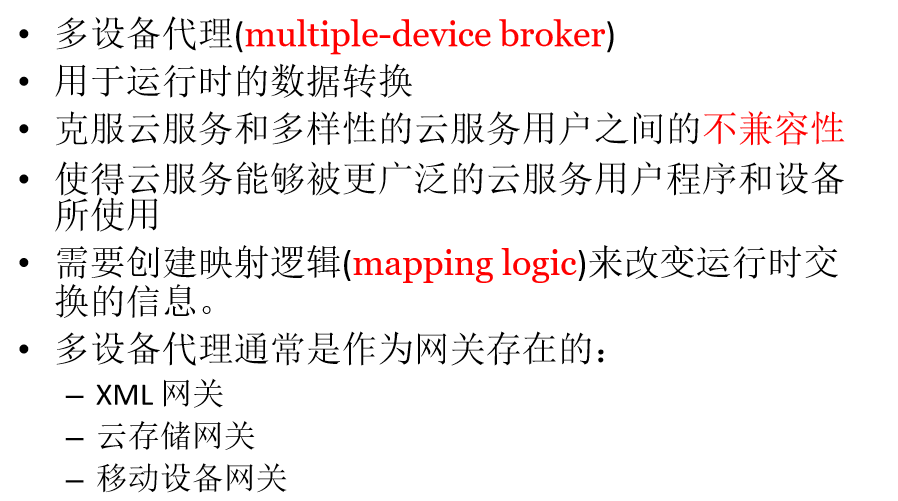
* + **虚拟机监控器**

****

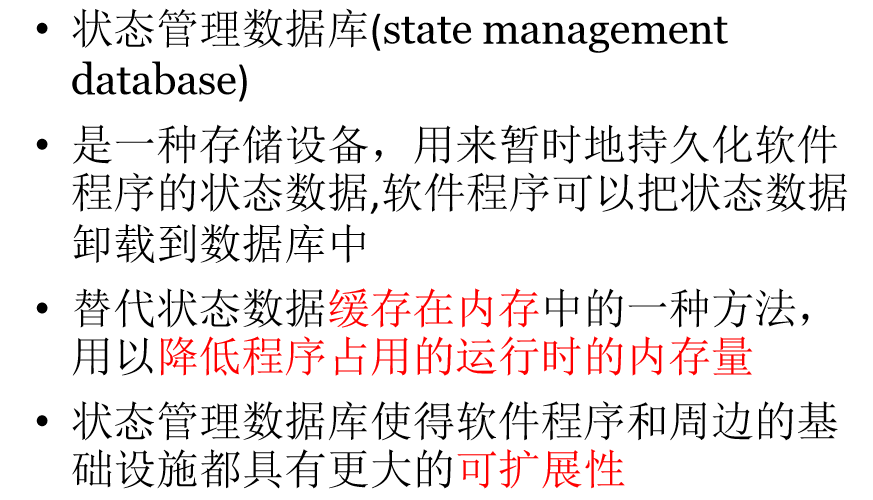
* + **资源集群**

****

* + **多设备代理**

****

* + **状态管理数据库**

****

* **第4章 调度问题与机制**

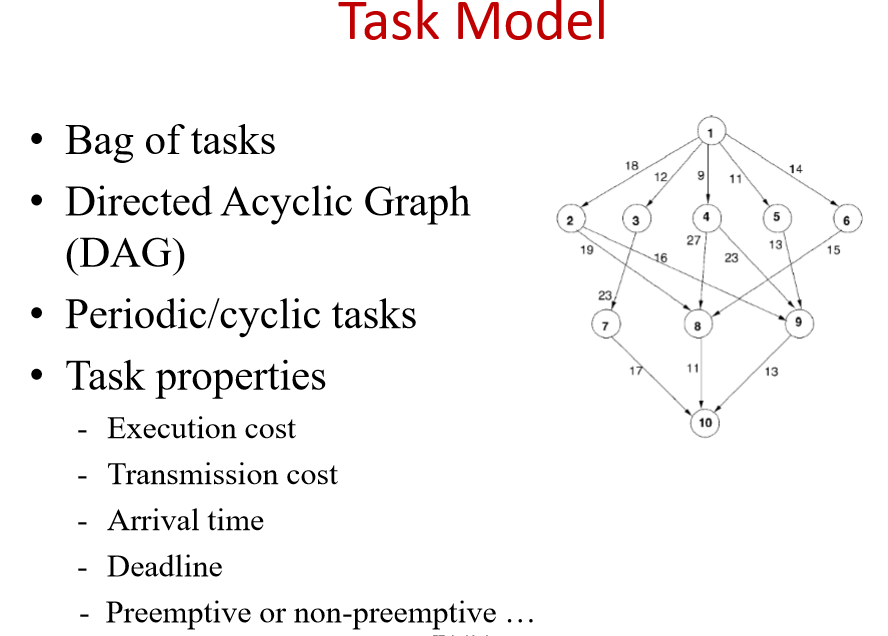
当有资源争用时，必须调度资源。

–处理器需要调度–缓冲区需要调度–网络需要调度

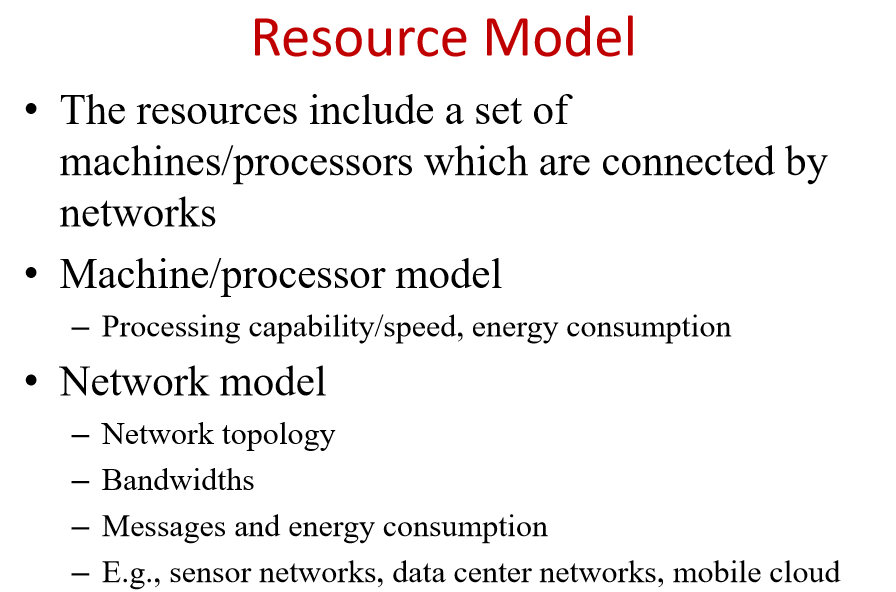
* + **常见调度问题**

**调度问题的三维框架**

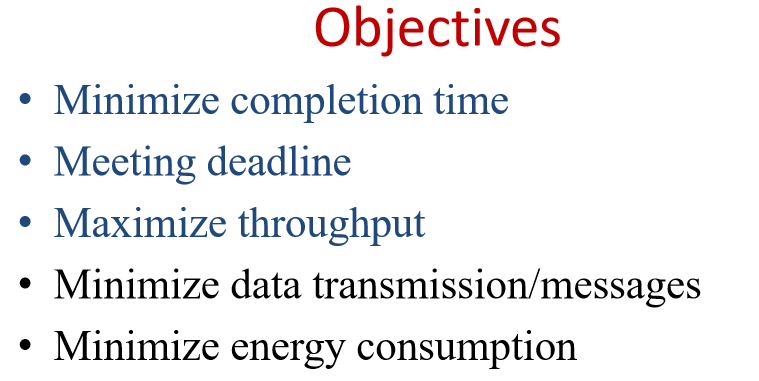
1. **任务**

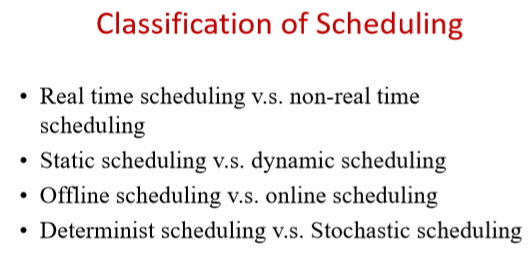
****

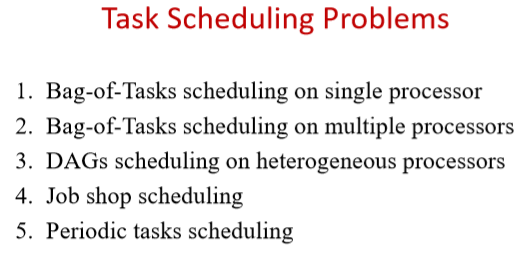
1. **资源**

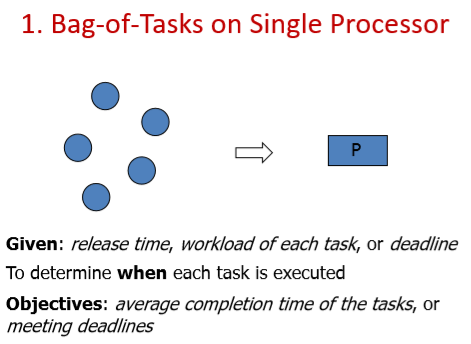
****

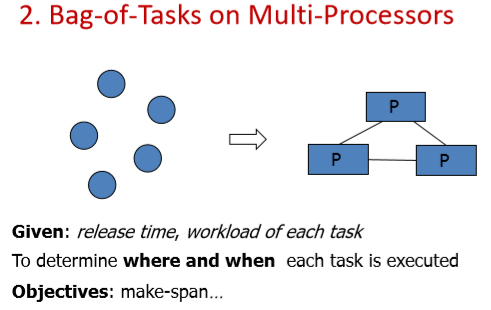
**3.目标**

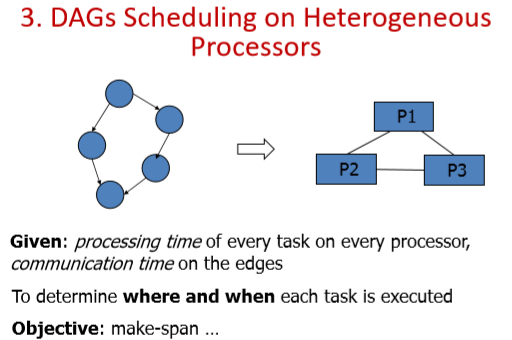
****

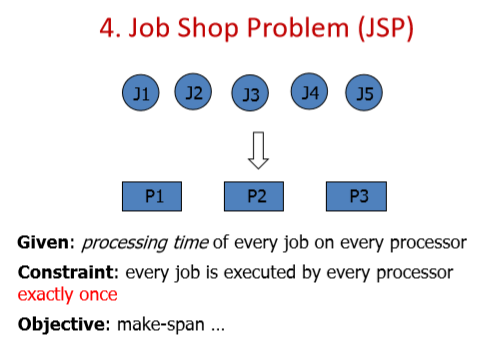
****

****

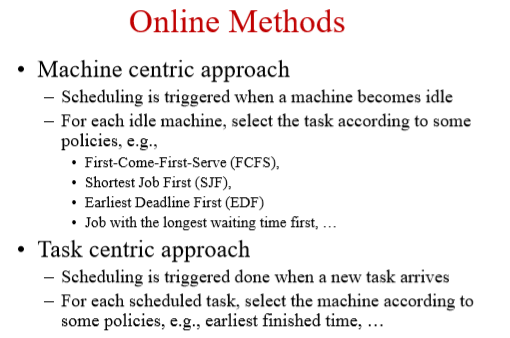








* **调度方法**



* **列调度（ List Scheduling ）算法**

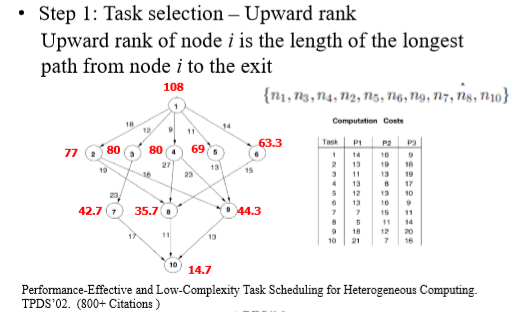
•步骤1：任务选择

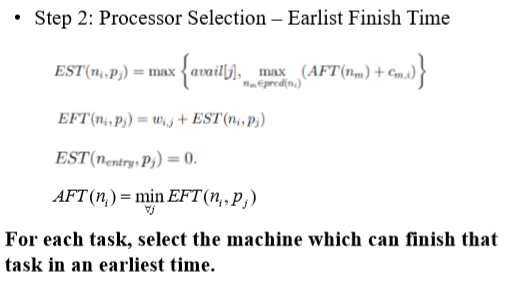
通过为每个任务分配优先级来构造任务的有序列表，然后按优先级顺序选择任务。

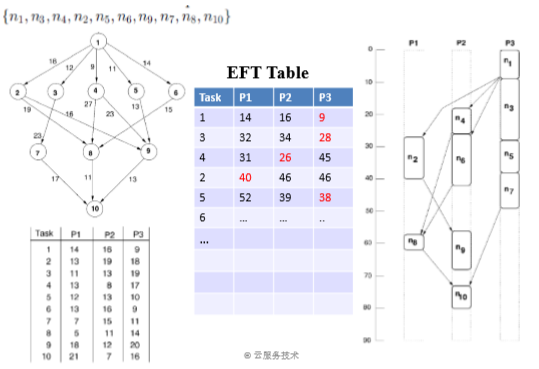
•步骤2：处理器选择

每个选择的任务都安排给处理器，以最大程度地减少预定义的成本函数。

•重复步骤1和步骤2，直到安排了所有任务







* **第5章 数据中心网络**
  + **设计目标与需求**

云：下一代大规模计算

–基础架构即服务（IaaS）

–平台即服务（PaaS）

–软件即服务（SaaS）

•云需要大型弹性数据中心的支持

–大量服务器

–大量存储

–与数据中心网络一起编排

–虚拟机支持

•示例：Google及其服务

需求：

•统一的高容量

•性能隔离

•易于管理：即插即用（第2层语义）

•对可伸缩，易于管理，容错和高效的数据中心网络（DCN）的要求：

–R1：任何VM都可以迁移到任何物理计算机而无需更改其IP地址

–R2：管理员无需在配置交换机之前进行任何配置 部署

–R3：任何终端主机都应通过任何可用路径与任何其他终端主机进行有效通信

–R4：没有转发环路

–R5：故障检测应该快速而有效

•网络协议的含义：

–整个数据中心的单一layer2结构（R1＆R2）

–带有成千上万个条目的Mac转发表（R3）

–高效的路由协议，可将拓扑更改快速传播到所有点（R5）

* + **成本**

•总成本各不相同

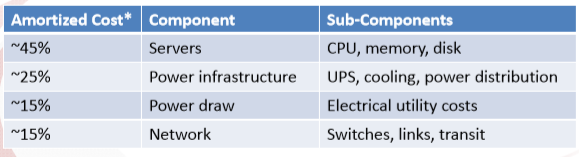
–大型数据中心的成本高达$ 1/4 B

–服务器成本占主导地位

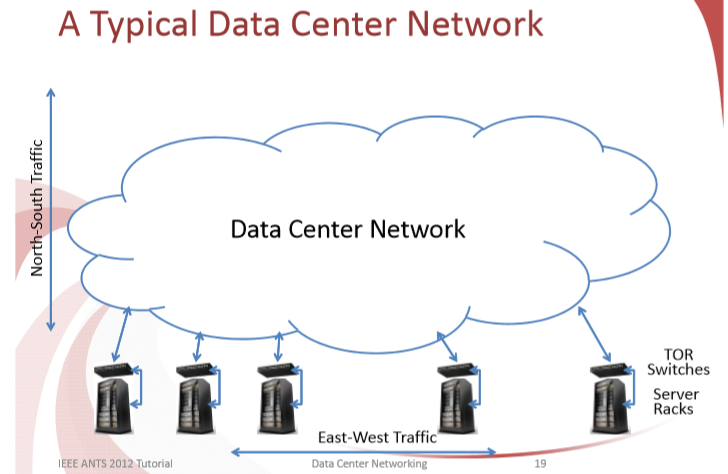
–网络成本很高

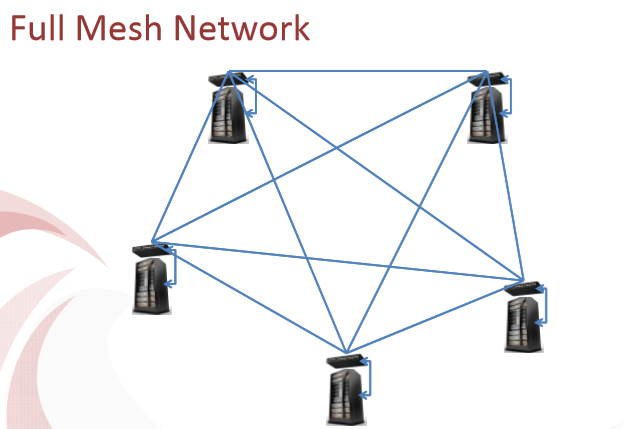
•较长的供应时间范围：

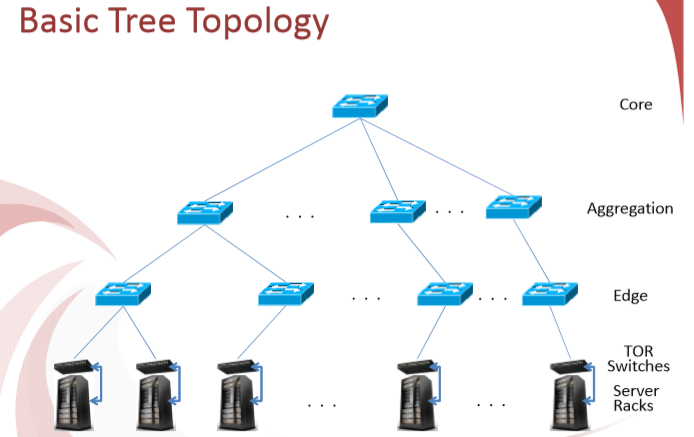
–最好每季度购买新服务器



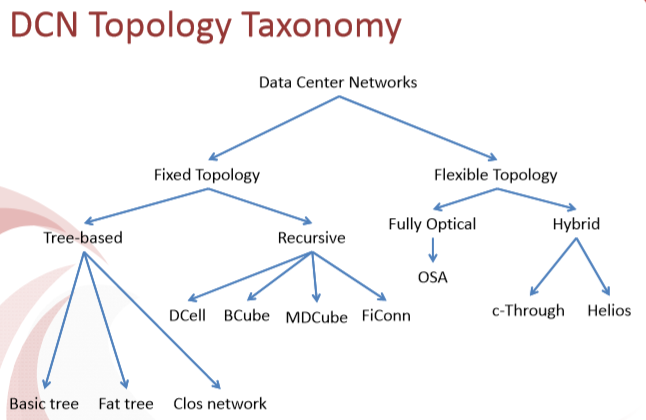
* + **数据中心拓扑结构（分类）常见拓扑结构**



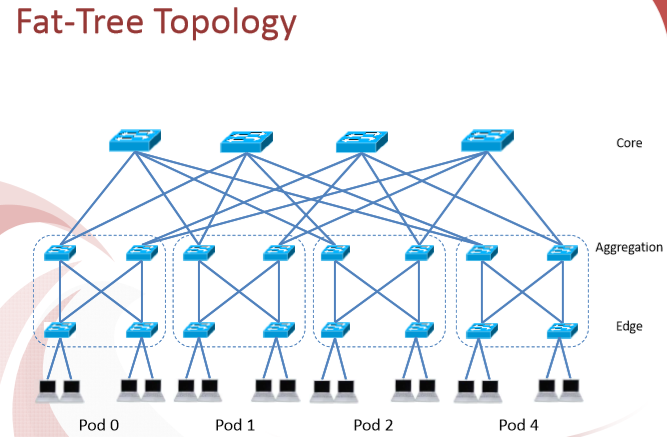




**分类：**



* + **Fat-Tree 架构及属性（重点掌握）**



•Fat-Tree：一种特殊的Clos网络

–K-aryfat树：三层拓扑结构（边缘，聚合和核心）

–将胖树拆分为k个Pod

–每个Pod由（k / 2）2个服务器和2个层组成 k / 2个k端口交换机

–每个边缘交换机连接到k / 2个服务器和k / 2 aggr 开关

–每个aggr。 交换机连接到k / 2个边缘和k / 2个核心交换机

–（k / 2）2个核心交换机：每个都连接到k个容器

–每个容器支持（k / 2）2个主机之间的无阻塞操作

–每个源和目标都有（k / 2）2条路径

•Fat-Tree属性

–任何二等分带宽相同

–每个层具有相同的聚合带宽

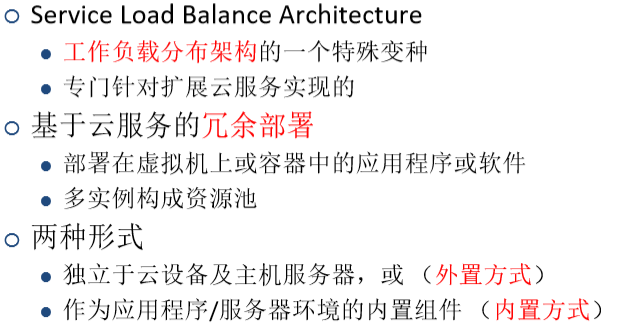
•可以使用具有统一容量的廉价设备来构建

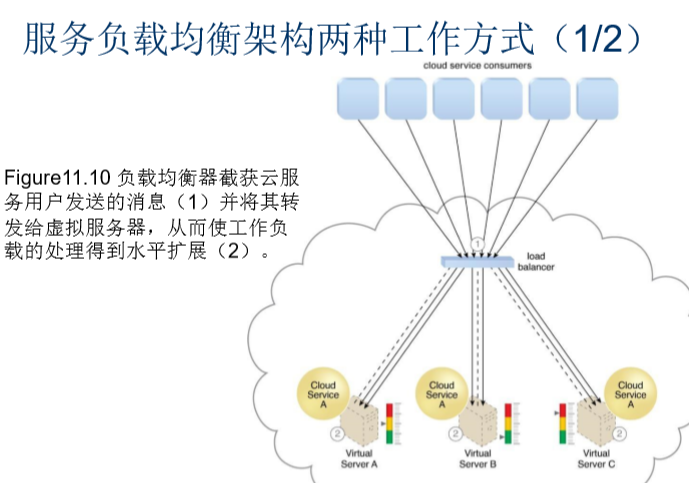
–每个端口与终端主机支持的速度相同

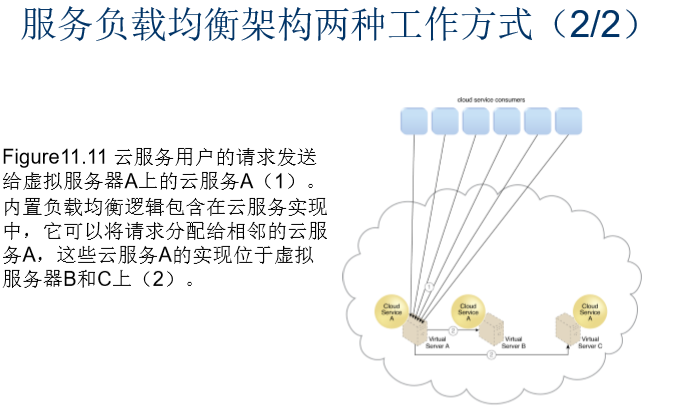
–如果数据包被分发，则所有设备都可以沿可用路径线速统一传输

•扩展性强：k端口交换机支持k2 / 4服务器

* **第6章 云架构**
* **负载均衡架构**



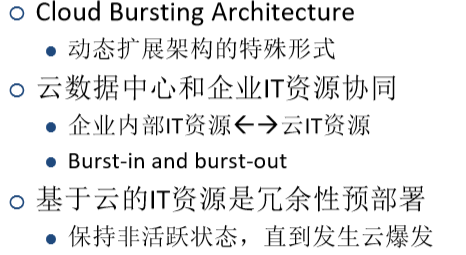


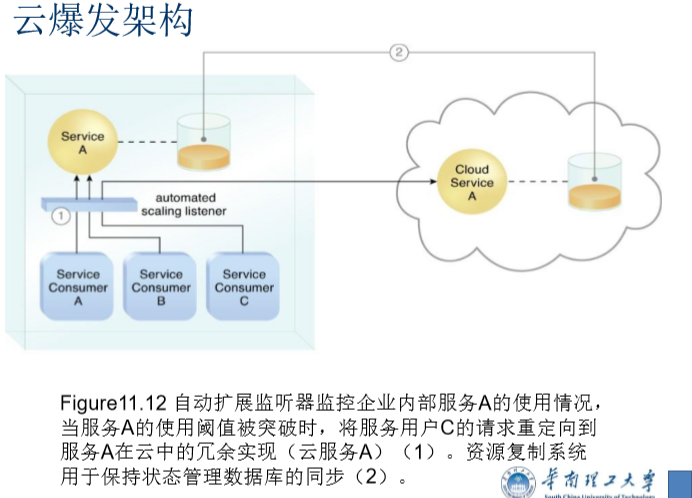


* **动态可扩展架构（水平、垂直、重定位）**



* **云爆发架构**

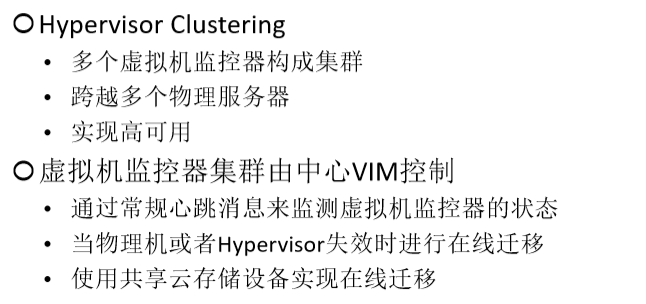


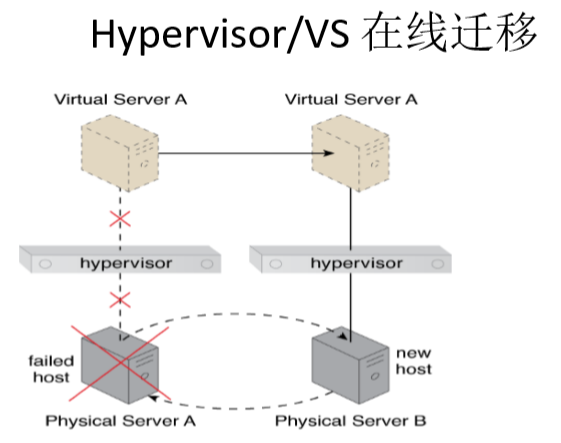


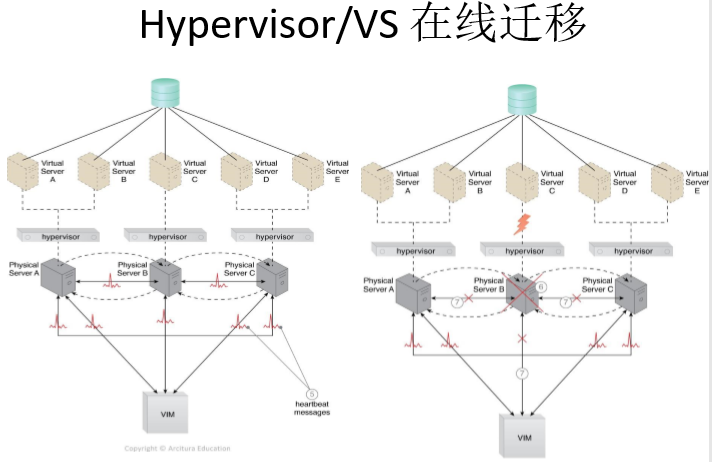
* **弹性磁盘供给架构**

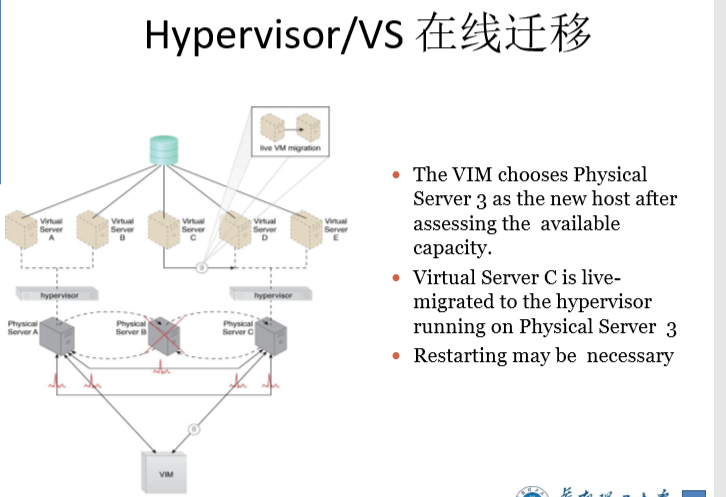
弹性磁盘供给架构（Elastic Disk Provisioning Architecture）按云用户实际所需进行存储分配和计费。

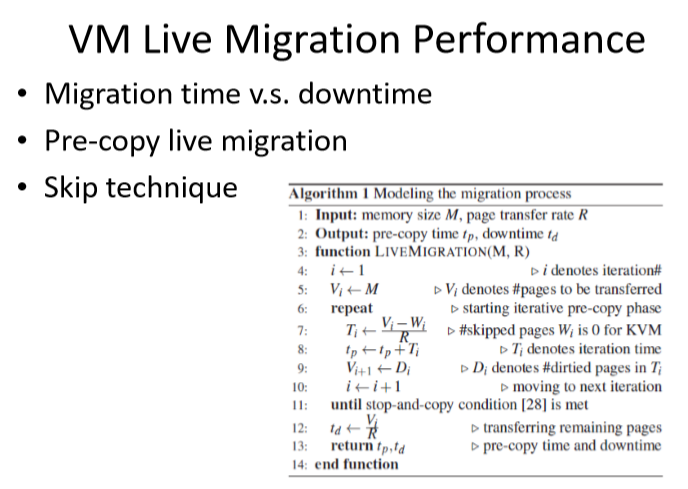
* **虚拟机（架构，在线迁移机制，性能指标及函数）**











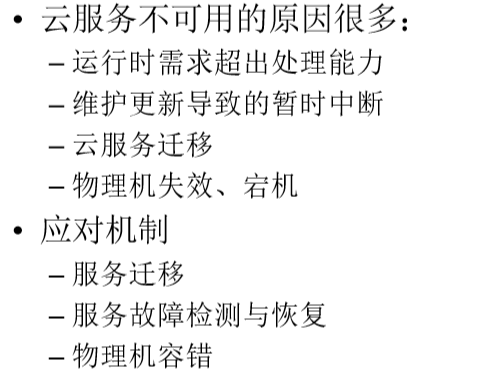
VM实时迁移性能

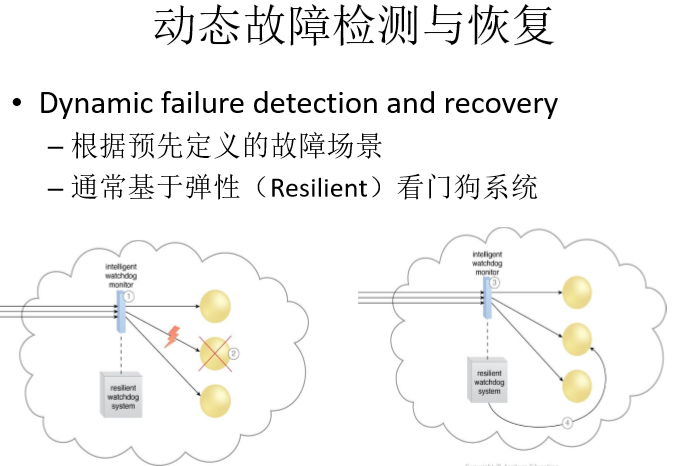
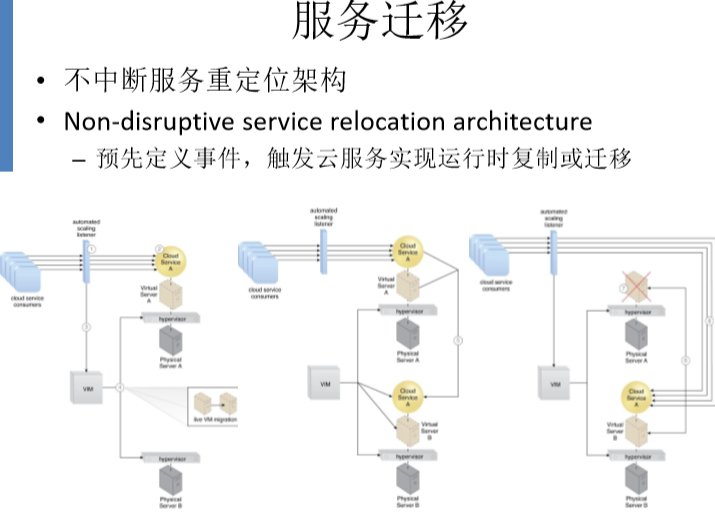
•迁移时间 v.s.停机

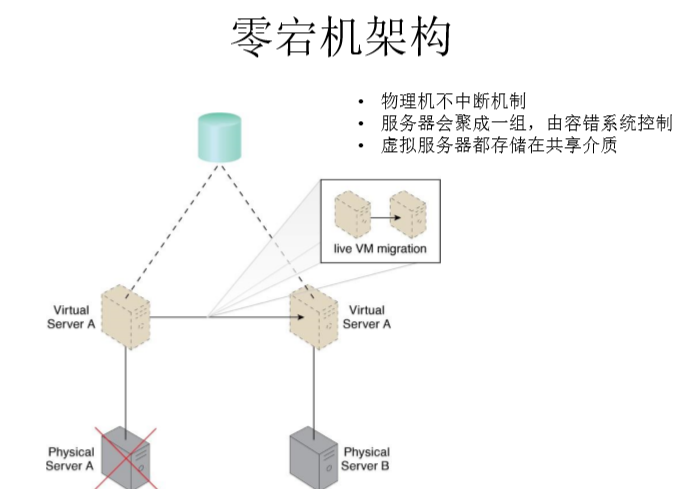
•复制前的实时迁移

•跳过技术

* **云服务容错架构以及容错机制**







* **第7章 移动云计算**
* **定义**

基于云计算的定义，移动云计算是指通过移动网络以按需、易扩展的方式获得所需的基础设施、平台、软件(或应用)等的一种IT资源或(信息)服务的交付与使用模式。移动云计算是云计算技术在移动互联网中的应用。

* **三个视角**

**移动云计算–AI视角**

* 移动云–AI应用的重要计算基础设施
* 智能交通系统
* 犯罪追捕Face++

**移动云计算–云计算视角**

* Cloud Computing 云计算

通过网络为客户提供和交付动态可配置资源的共享池，作为对客户的服务

* Mobile Cloud Computing 移动云计算

通过移动设备向用户提供云计算服务，例如 智能手机和汽车

克服的挑战：

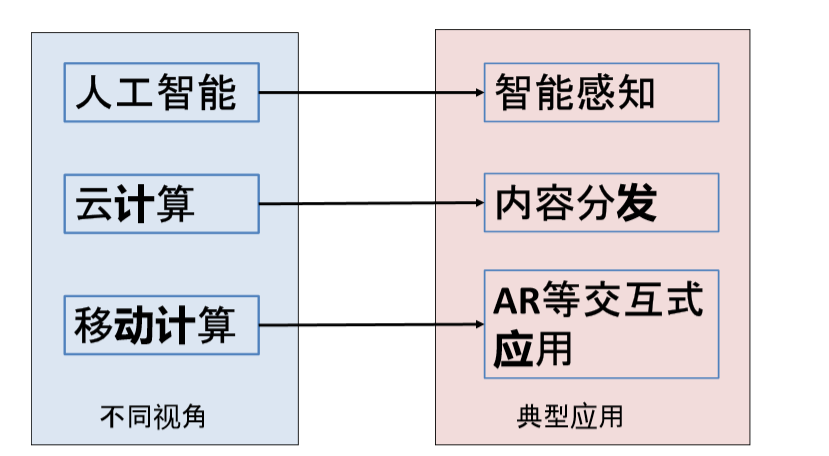
* 移动设备约束: 电池、显示屏等
* 移动网络掉线、链接不稳定
* 用户移动性
* 视频分发服务、云游戏服务

新的机会：

* 终端感知能力：如Siri、翻译等
* 位置信息：基于位置的服务

**移动云计算–移动计算视角**

* 移动设备计算资源、电池受限
* 利用云计算资源节约移动设备资源受限问题
* Computation Offloading -将Mobile App.的复杂计算迁移到云



* **性能建模与优化**

用户响应时间：

* 本地计算时间
* 据传输（等待）时间
* 云上计算（等待）时间

影响因素 ：

* 本地设备计算能力
* 网络带宽、延迟、拥塞
* 云处理能力、负载
* 计算模块部署

移动云性能优化技术：

* Computation Partitioning 计算划分
* Data (Service) Placement and Load Dispatching 数据存放和负载分配
* **计算划分**

计算划分将应用程序的工作量分解为较小的工作量，并确定哪些部分在本地执行，哪些部分卸载到云中。

、

A simple example:

