

大数据存储与处理

韩冀中

Email: hanjizhong@iie.ac.cn

中国科学院信息工程研究所

“大数据”

近两年信息技术领域最时髦的词汇

“Why big data is
a big deal”

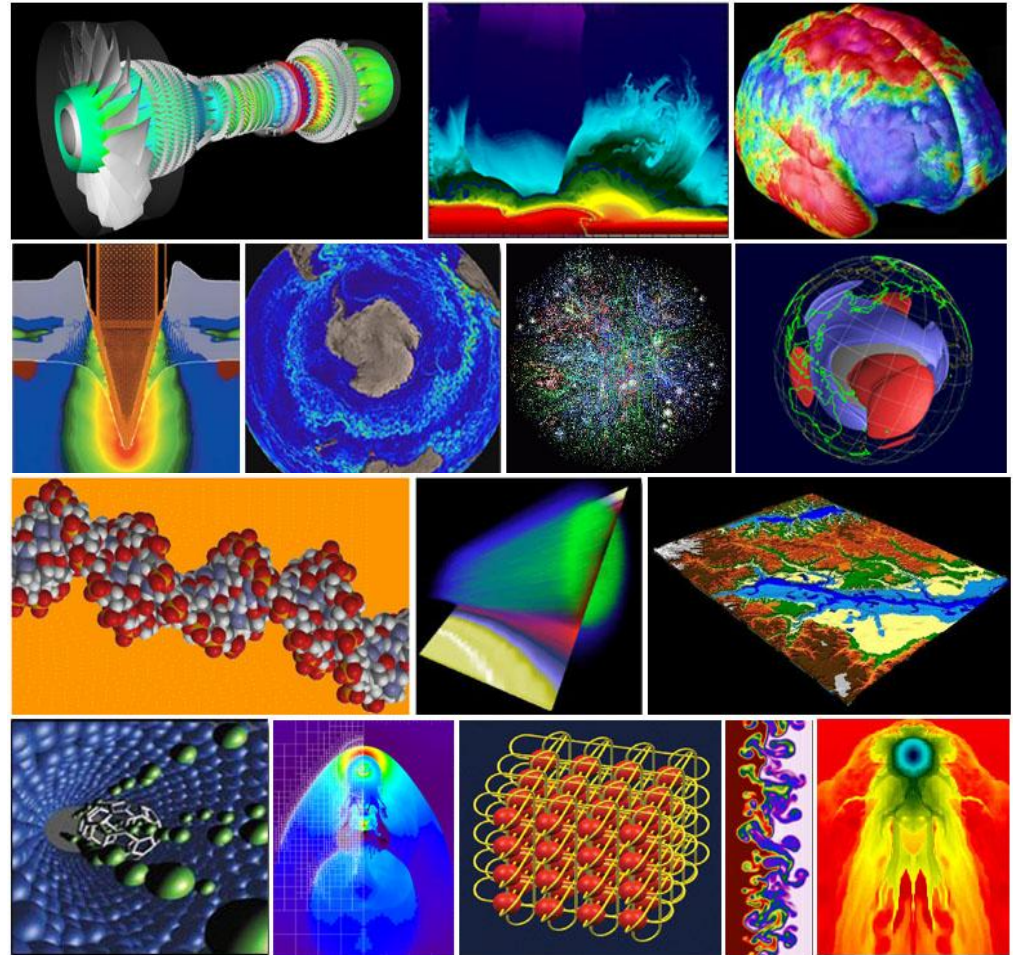
“The challenge—
and opportunity—
of big data”

“Ten reasons why Big
Data will change the
industry”

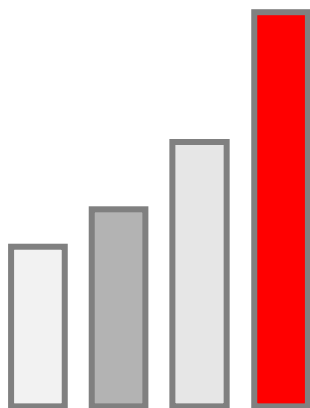
麦肯锡全球研究所去年的调查报告显示，美国需要**14万至19万名**以上具备“深度分析”专长的人员，而对具备数据知识的经理的需求超过**150万**。

大数据无处不在

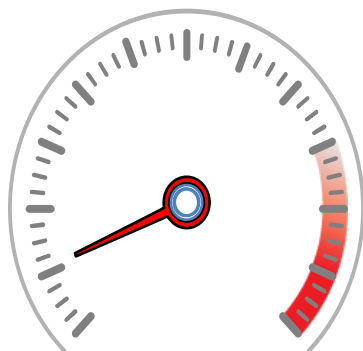
- 科学研究
 - 基因组
 - LHC 加速器
 - 地球与空间探测
- 企业应用
 - Email、文档、文件
 - 应用日志
 - 交易记录
- Web 1.0数据
 - 文本
 - 图像
 - 视频
- Web 2.0数据
 - 查询日志/点击流
 - Twitter/ Blog / SNS
 - Wiki



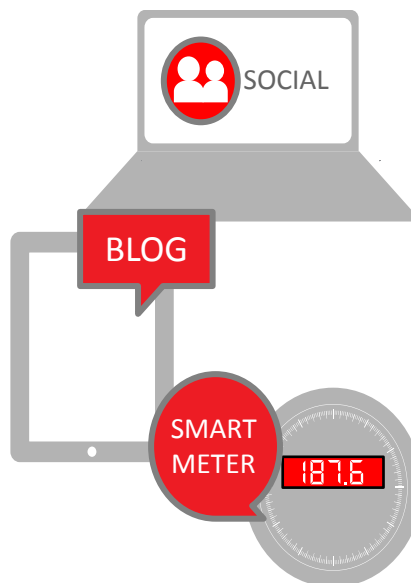
大数据的四个特征



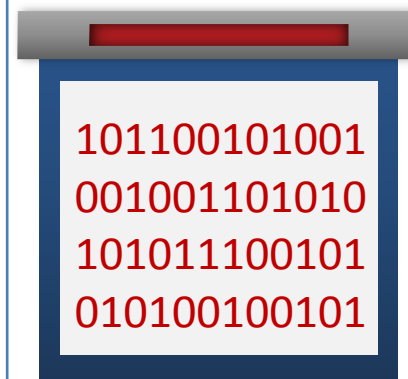
VOLUME
大量化



VELOCITY
快速化



VARIETY
多样化

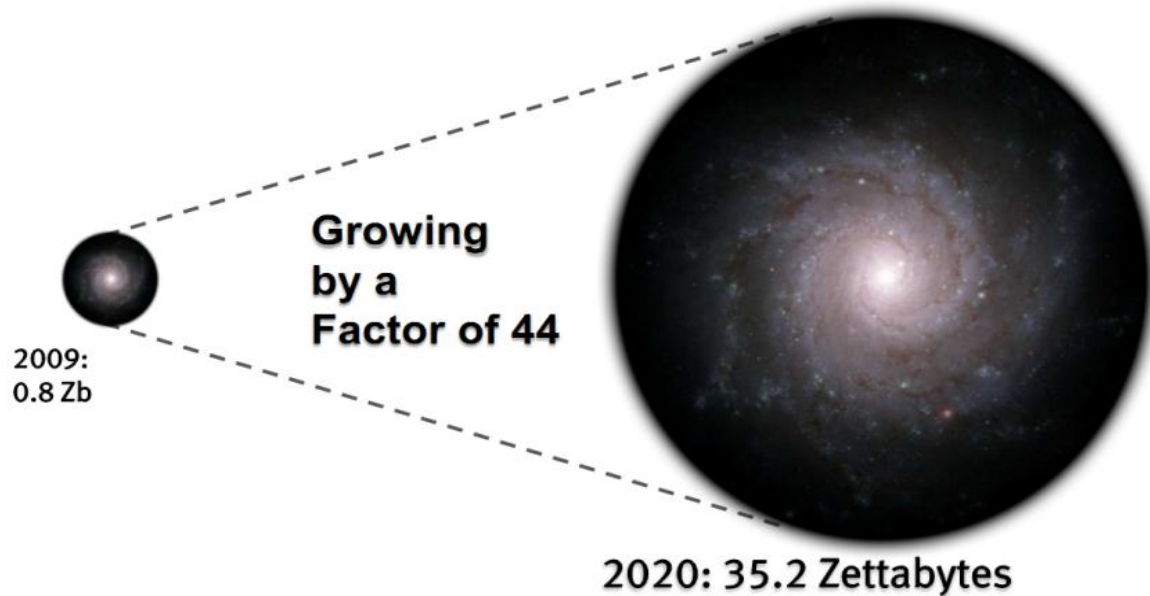


VALUE

大数据不仅仅是数据的“大量化”，而是包含“快速化”、“多样化”和“价值化”等多重属性。

Volume——“大量化”

- 2010年，全球数据量已达1.2ZB，到2020年将暴增30倍达35ZB



数据的量度

TERABYTE	10 的 12 次方	一块 1TB 硬盘		200,000 照片或 mp3 歌曲
PETABYTE	10 的 15 次方	两个数据中心机柜		16 个 Blackblaze pod 存储单元
EXABYTE	10 的 18 次方	2,000 个机柜		占据一个街区的 4 层数据中心
ZETTABYTE	10 的 21 次方	1000 个数据中心		纽约曼哈顿的 1/5 区域
YOTTABYTE	10 的 24 次方	一百万个数据中心		特拉华州和罗德岛州

进入大数据的时代

- 2011年，中国互联网行业持有数据总量达到**1.9EB**（1EB字节相当于10亿GB）
- 2011年，全球被创建和复制的数据总量为**1.8ZB**（1.8万亿GB）
- 2013年，我们生成这样规模的信息量只需**10分钟**
- 2015年，全球被创建和复制的数据总量将增长到**8.2EB**以上
- 2020年，全球电子设备存储的数据将暴增**30倍**，达到**35ZB**

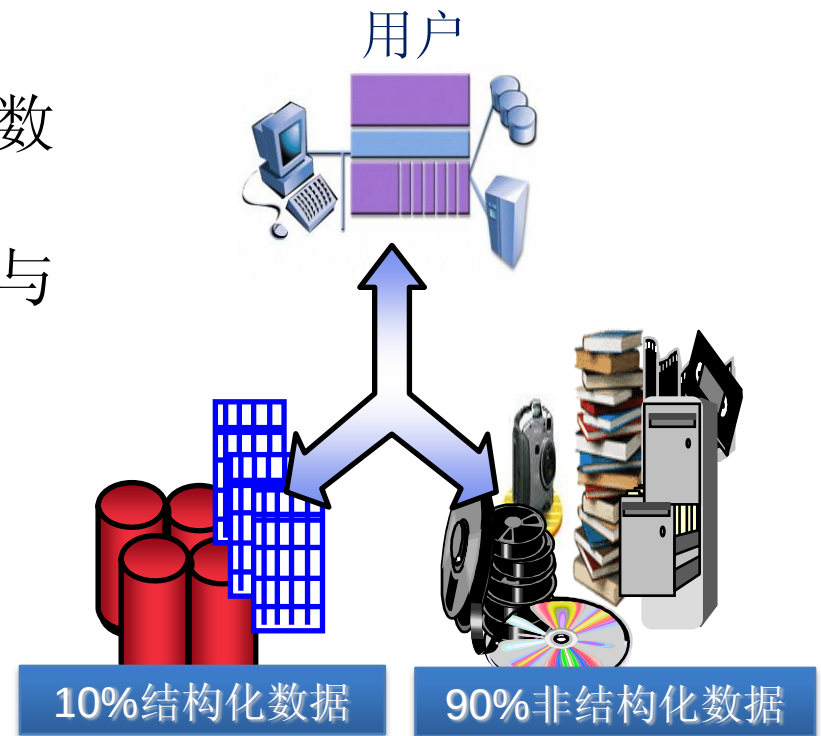
Velocity——快速化

从数据的生成到消耗，时间窗口非常小，可用于生成决策的时间非常少。

- 每秒钟，人们发送**290万封**电子邮件
- 每分钟，人们向Youtube上传**60个小时**的视频
- 每一天，人们在Twitter上发消息**1.9亿条**微博
- 每一天，人们在Twitter上发出**3.44亿条**消息
- 每一天，人们在Facebook发出**40亿条**信息

Variety——多样化

- 大数据是由结构化和非结构化数据组成的
 - 10%的结构化数据，存储在数据库中
 - 90%的非结构化数据，它们与人类信息密切相关
- 非结构化数据类型多样
 - 邮件、视频、微博
 - 位置信息、链接信息
 - 手机呼叫、网页点击
 - “长微博”



第4个V——Value

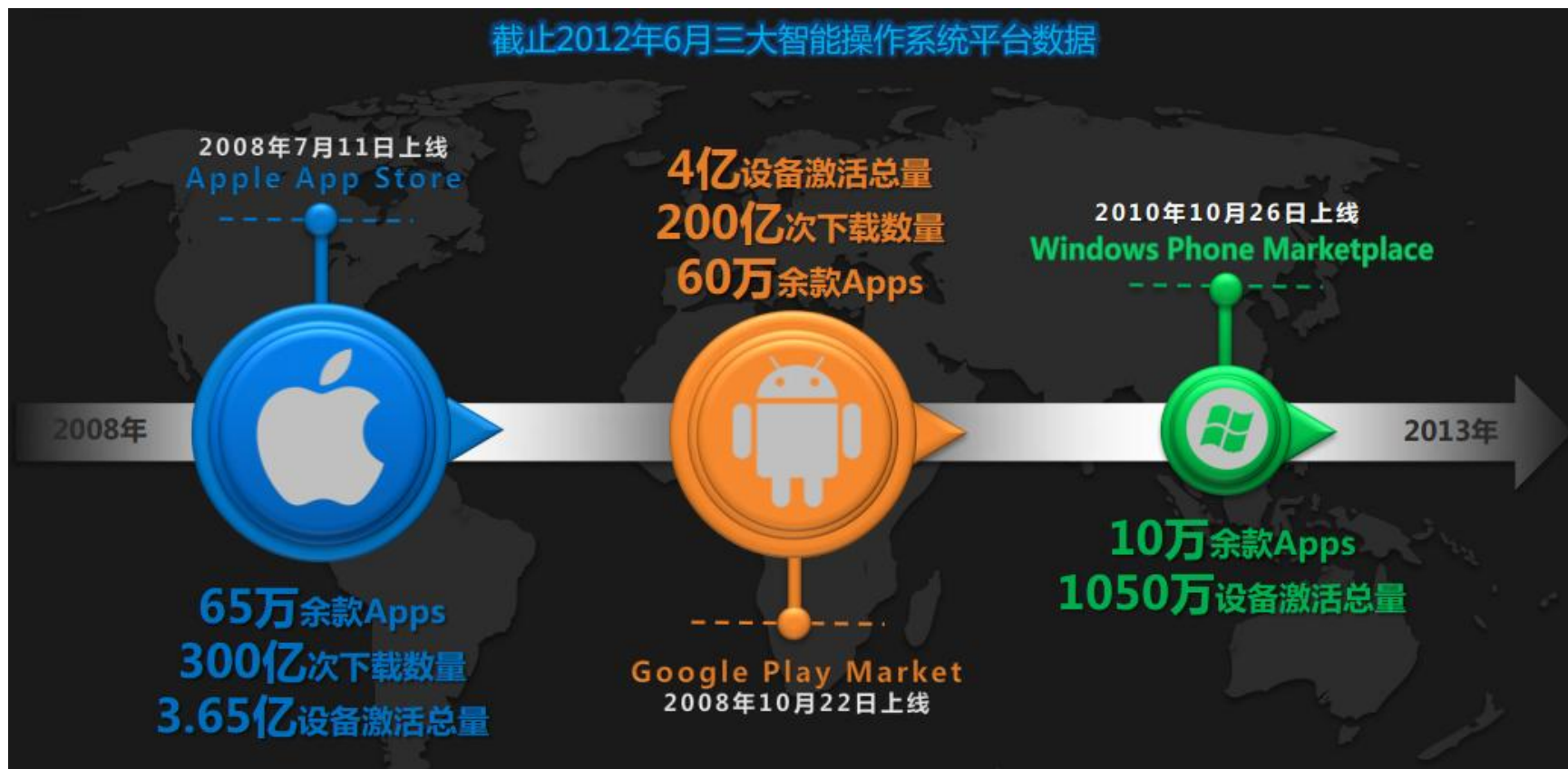
- Value——价值化
 - 科学研究
 - 企业应用
 - 社会网络



© FAS.research 2002

互联网生态结构转变的影响

截止2012年6月，中国有网民5.38亿，其中手机网民达到3.88亿。



移动互联网促进了大数据的出现

- 移动互联网使得人类社会可以“测量”
 - 与4世纪前发明的显微镜一样具有重大的意义
 - 移动互联网产生更多用户活动与行为的数据

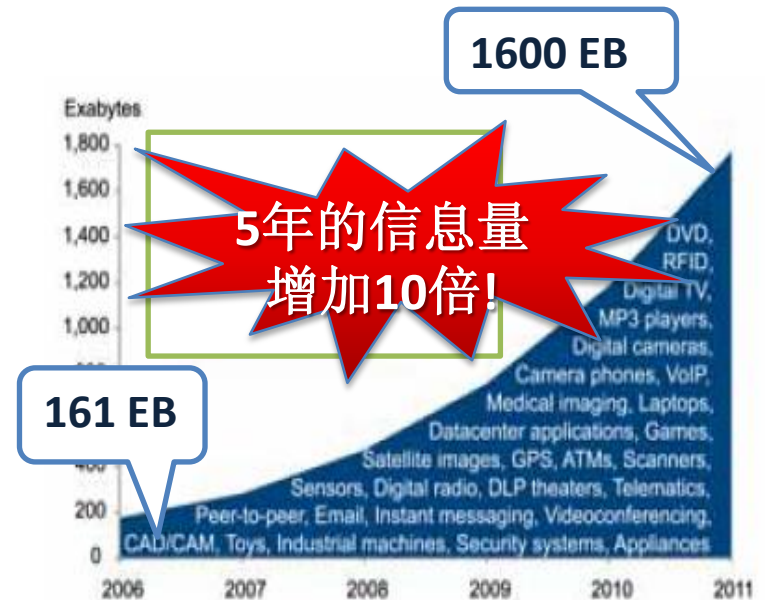


一些有趣的移动互联网应用



数据的膨胀及其应对

- 移动互联网时代的数据飞速增长
 - 过去3年产生的数据量比过去4万年还要多
- 掌握大数据之后该怎么做？



Big Data Landscape

Log Data Apps

splunk>

loggly

sumologic

Vertical Apps



PREDICTIVE
POLICING



bloomreach
GET FOUND.

Business Intelligence

ORACLE | Hyperion



Business Objects

Microsoft | Business Intelligence



COGNOS



MicroStrategy

GoodData

Analytics and Visualization



tableau



METAMARKETS

TERADATA ASTER



visual.ly



GREENPLUM



platfora



Data Providers

GNIP

DATASIFT



SPACE
CURVE

INRIX

Analytics Infrastructure



Hortonworks

cloudera

MAPR
TECHNOLOGIES



Operational Infrastructure

COUCHBASE

TERADATA

10gen | the MongoDB company

HADAPT

Infrastructure As A Service



amazon
web services



Windows Azure

Structured Databases

ORACLE



MySQL



PostgreSQL



Technologies



APACHE
HBASE



Cassandra

大数据存储与处理面临的挑战

系统问题

如何弹性地应对
数据和负载的动
态变化

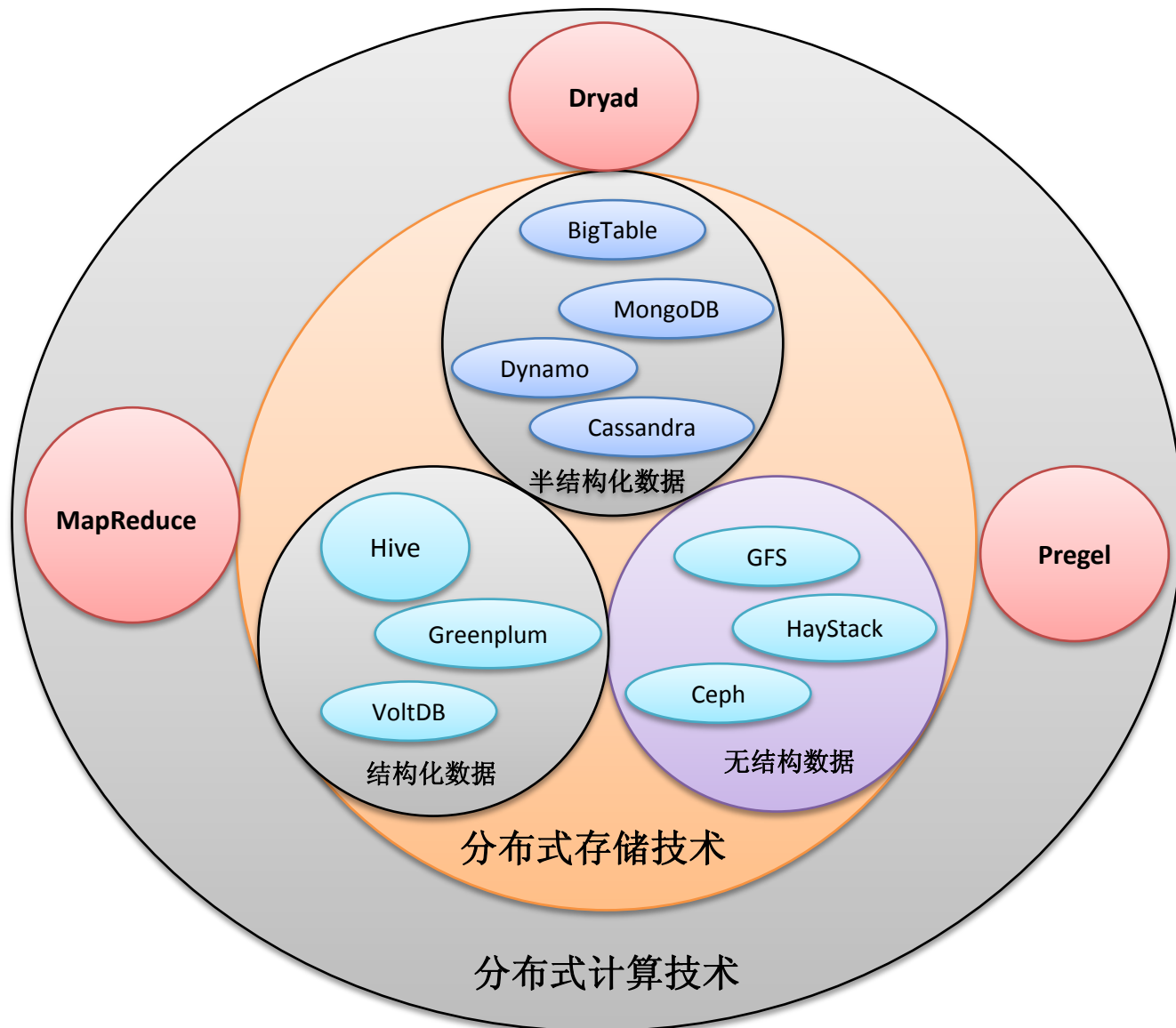
管理问题

如何有效存储和
处理海量的数据

应用问题

如何支持大数据
上的智能分析和
事务处理

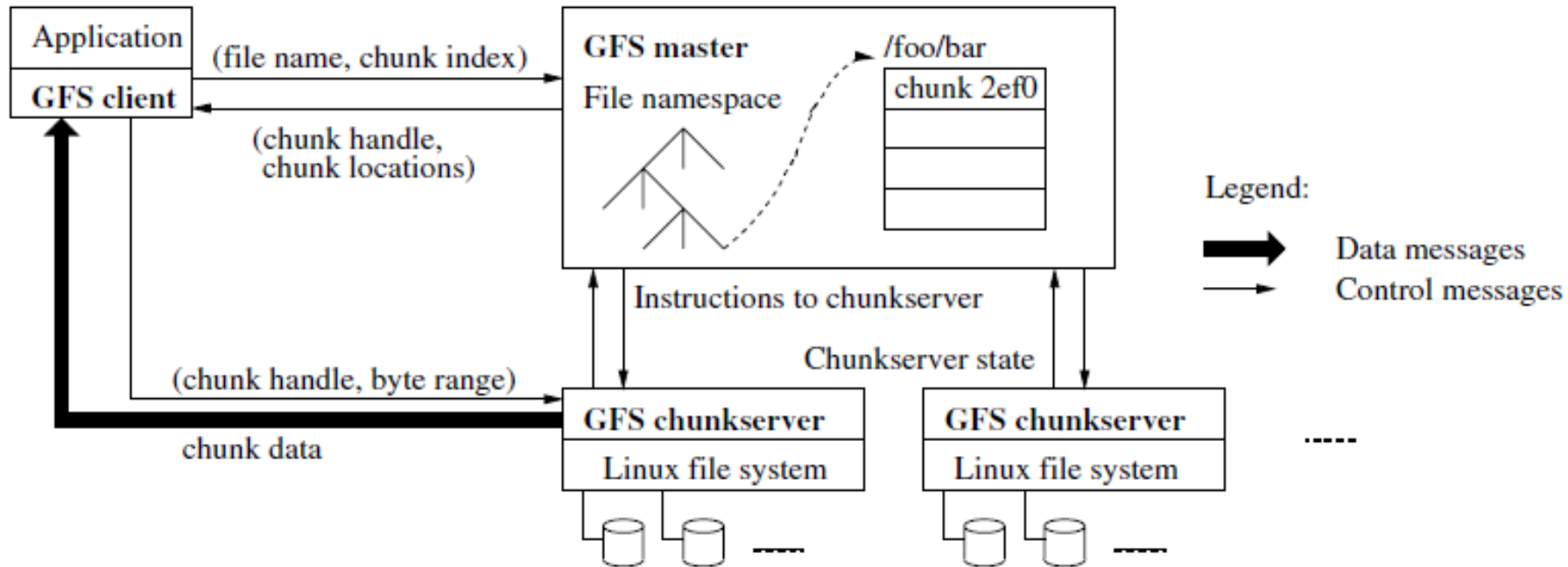
流行的分布式存储与计算技术



1、“大数据”的存储与管理

- 解决的问题
 - 海量文件的存储与管理
 - 海量小文件的存储、索引和管理
 - 海量大文件的分块与存储
 - 系统扩展性与可靠性
- 代表性系统
 - GFS（Hadoop DFS）
 - Facebook HayStack
 - BigTable

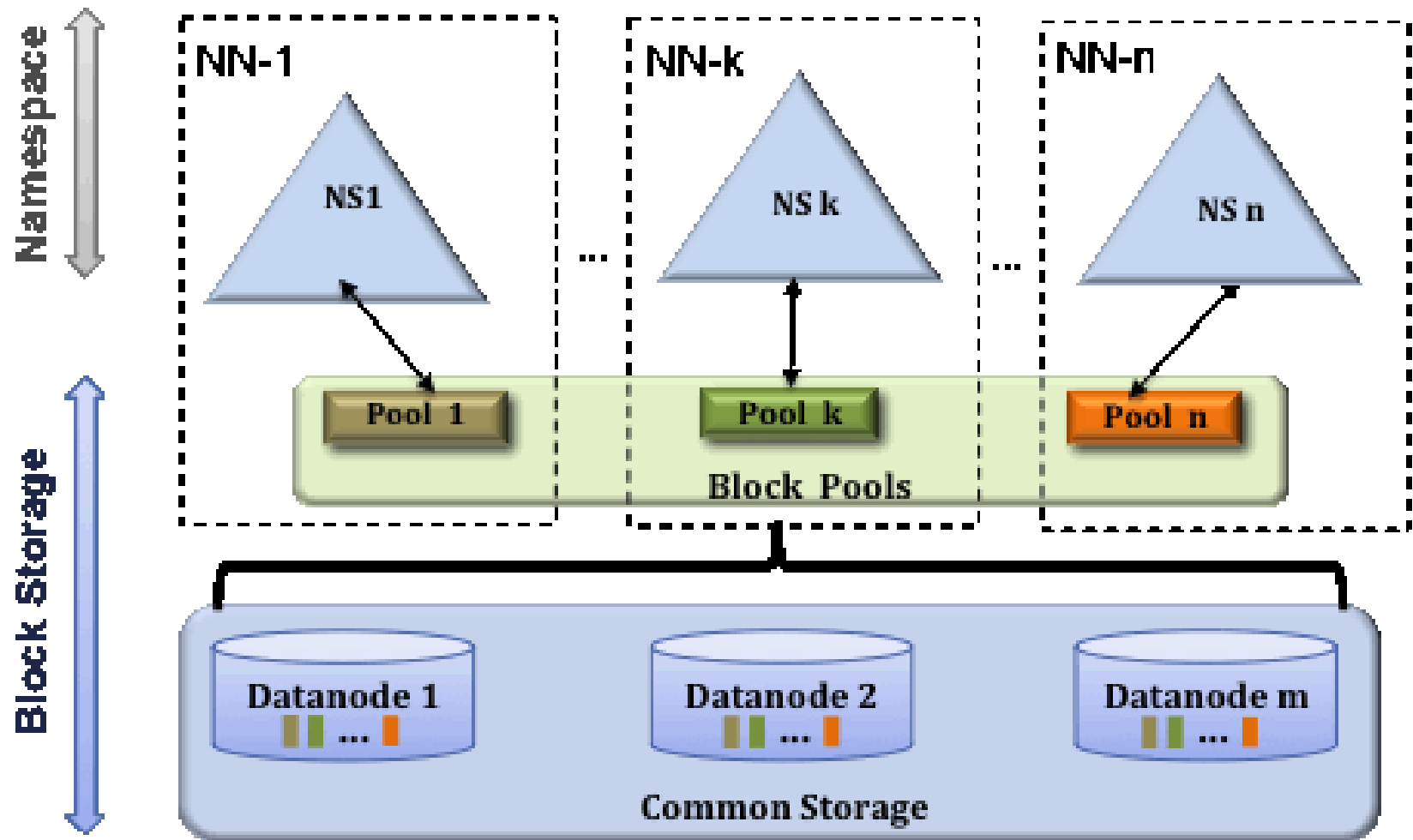
Google File System



- 单元数据服务器，多数据服务器
- 大数据块，顺序读写优化
- 数据多副本容错

“大数据”使得元数据成为问题

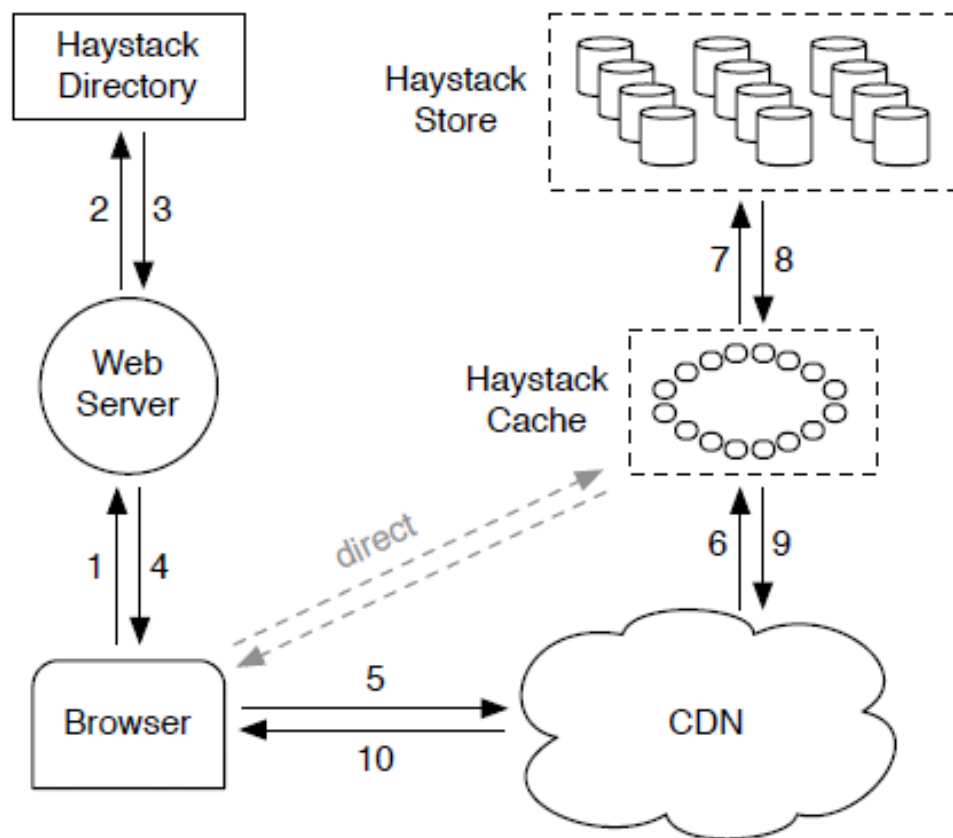
HDFS Federation



方法一：多元数据服务器

Facebook HayStack File System

- 目录服务器
 - 负责将文件映射到物理设备
 - 完成URL到存储服务器的映射
- 缓存服务器
 - 减少磁盘I/O次数
 - 降低访问延迟
- 存储服务器
 - 包括存储文件和内存索引



方法二：根据应用特征简化元数据

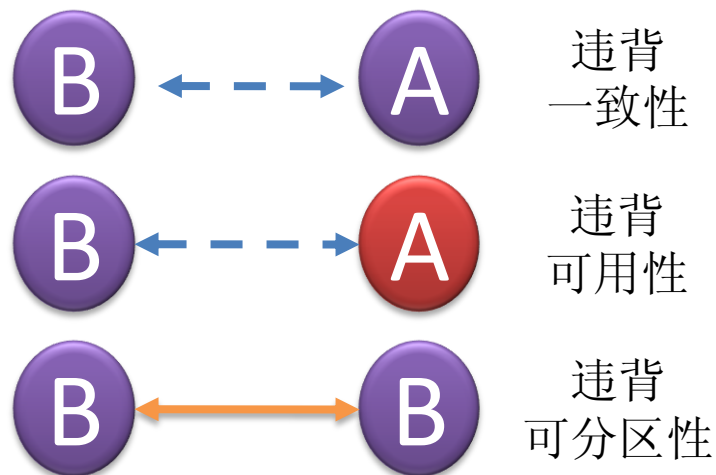
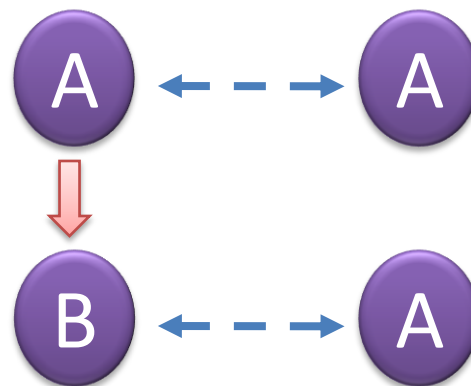
大数据与NOSQL

- 出现的原因
 - 社会网络带来的数据膨胀（Facebook，Twitter）
 - 大量非文件形式的互联网数据（Google、Yahoo！）
 - 云计算所需的云存储（Amazon S3）
 - 开源社区的发展（Hadoop）
- 分布式键值对数据存储系统
 - 数据以键值对的形式存储（按键访问）
 - Google BigTable,
 - Yahoo! Sherpa/PNUTS
 - Amazon Dynamo
 - ...

不是No SQL，而是Not Only SQL

NOSQL 与 CAP Theorem

- 系统的三个特性
 - Consistency 一致性（所有副本的值相同）
 - high Availability 可用性（部分失效时系统可运行）
 - tolerance to network Partitions 可分区性（网络故障时，每个网络分区均可独立工作）
- **Brewer的CAP “Theorem”**：任何基于网络的数据共享系统最多只能满足这三个属性中的两个（PODC 2000）
- 网络故障难以避免，一致性与可用性的权衡
 - 传统数据库 → 一致性
 - 大多数网络应用 → 可用性



12年之后看CAP Theorem

- CAP Theorem
 - 2000年的PODC Keynote
 - 12年来NOSQL设计者的依据
- “三取二”的误区
 - 分区很少发生，为此牺牲C或A不划算
 - C和A之间无法“一次选择，终生受益”
 - C、A、P之间有许多权衡，并非绝对
- 两种模式
 - “运行态”：C、A、P
 - “维护态”：C或A之间权衡



ACID与CAP

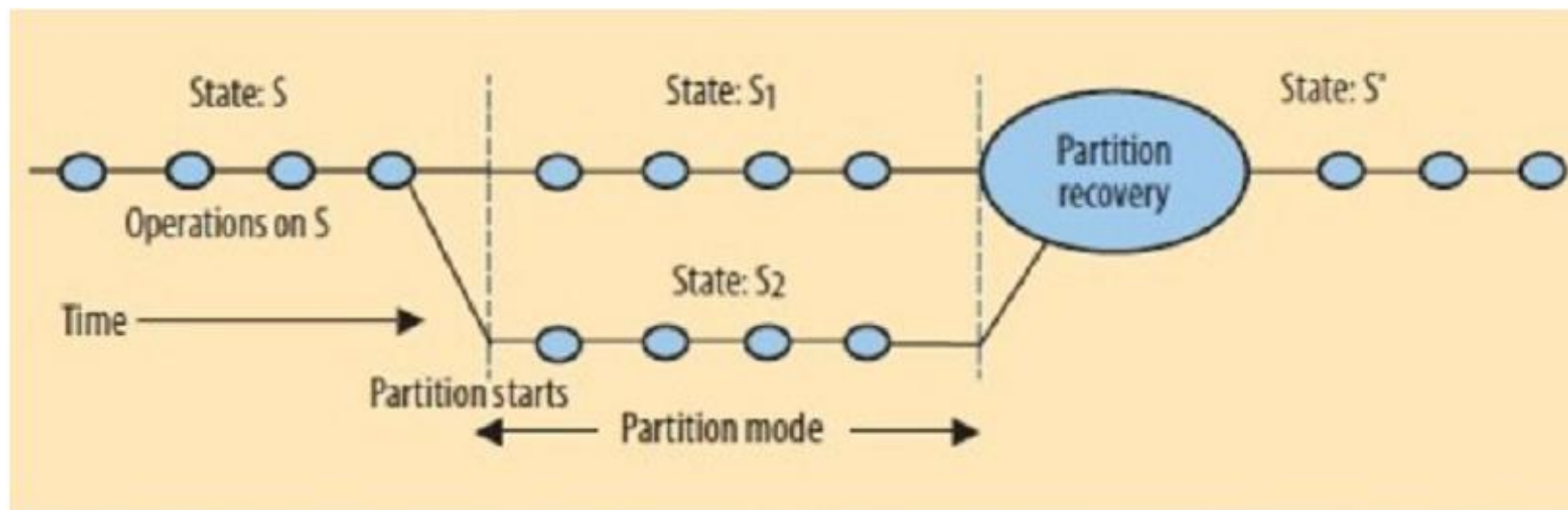
ACID

- **Atomicity (A)** 原子性
- **Consistency (C)** 一致性
 - 涉及DB内部所有规则
 - 网络分区时，ACID的C实际上无法维护（分区时，有些操作应禁止）
- **Isolation (I)** 隔离性
- **Durability(D)**持久性

CAP

- **Consistency** 一致性
 - 仅指复制一致性
- **high Availability** 可用性
- **tolerance to network Partitions**可分区性

CAP之间的一个新平衡点

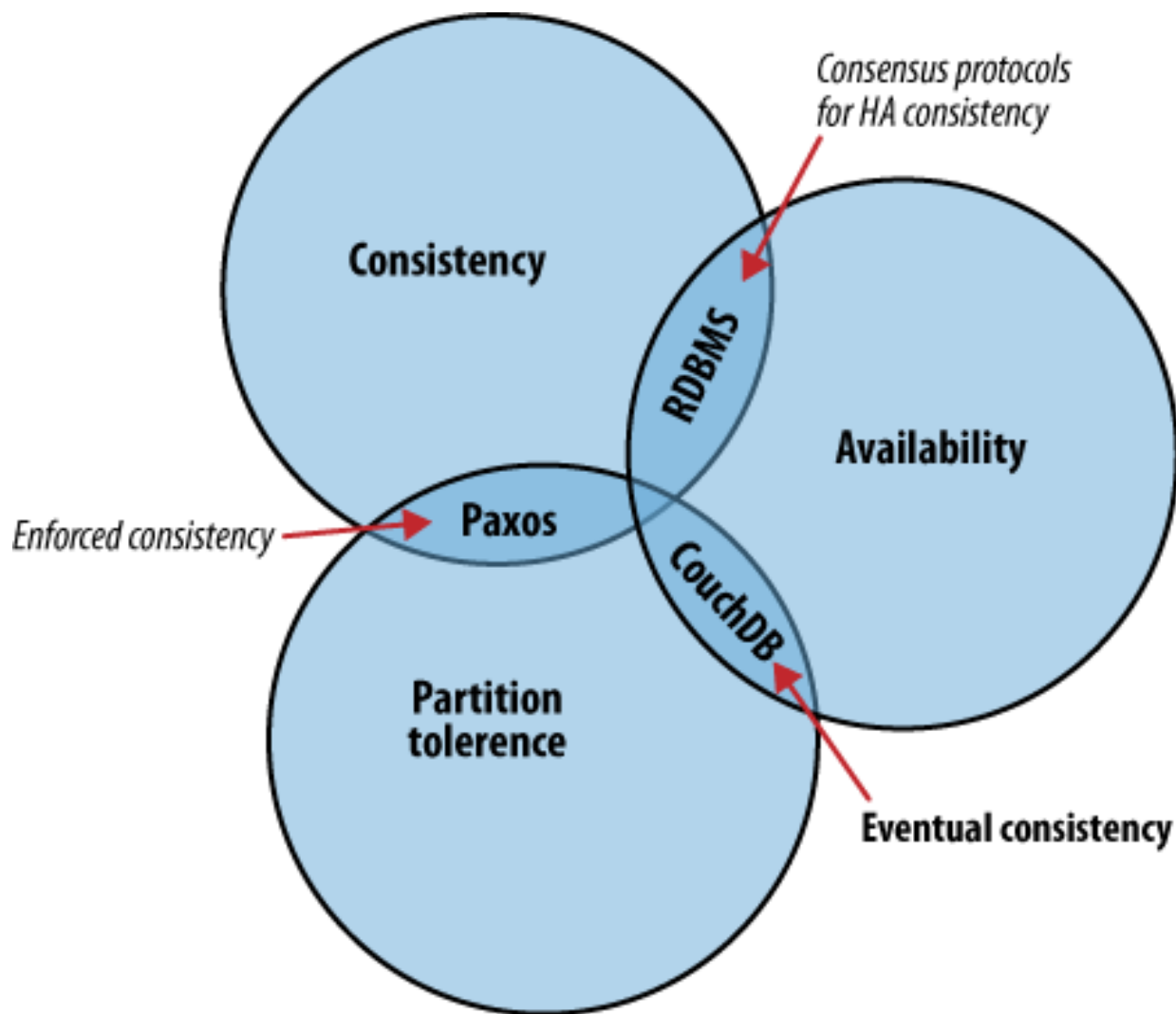


- 网络侦测，及时掌控网络分区状态
- 进入“**Partition Mode**”，限制部分操作
- 通信恢复时，进行“**partition recovery**”
 - 错误的特殊处理（做一个类比：**ATM**）

BASE

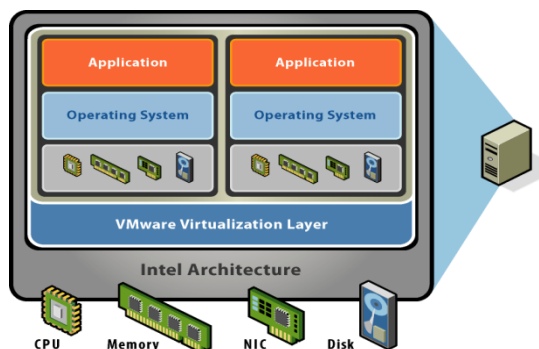
- BASE
 - **B**asically **A**vailable
 - **S**oft state
 - **E**ventual consistency
 - 所有节点在更新后保持一致，但在此之前可能不一致。
 - 对于指定节点，最终均为进行更新，或者被移除。

“大数据”存储管理的小结



2、“大数据”系统管理

弹性资源管理离不开虚拟化技术



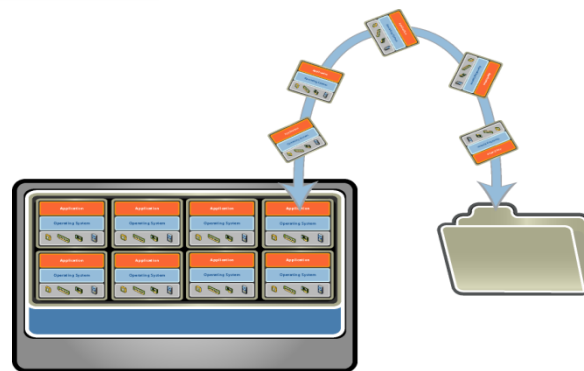
分区

- ❖ 在一个物理机上运行多个OS
- ❖ 更充分利用服务器资源
- ❖ 支持高可用——分区之间可以组建集群（负载均衡、双机容错）



隔离

- ❖ 从硬件层面隔离系统故障和安全威胁
- ❖ 在虚拟机之间动态的分配CPU、内存等系统资源
- ❖ 保证服务可用



封装

- ❖ 将虚拟机封装成与硬件配置无关的文件
- ❖ 随时对虚拟机进行快照
- ❖ 通过简单的文件拷贝对虚拟机进行迁移

虚拟化之一——虚拟机

- 完全虚拟化

这种技术在虚拟机和底层硬件之间建立的抽象层：Hypervisor来管理各个虚拟机。完全虚拟化技术几乎能让兼容所有的操作系统，但中间层的代码转换会给系统带来开销。完全虚拟化技术目前的项目主要有VMware，KVM等。

- 部分虚拟化（半虚拟化）

完全虚拟化要求Hypervisor管理各个虚拟服务器，并提供代码级别的翻译转换，因此负担沉重。部分虚拟化就是为了减轻这种负担：改动客户操作系统，集成与Hypervisor协同工作的界面，从而提升了性能。如Xen等。

- 操作系统级别的虚拟化

操作系统层虚拟化没有独立的Hypervisor层，主机操作系统负责在多个虚拟服务器之间分配硬件资源，这种虚拟化方式速度最快，不过操作系统的类型受到了严格的限制。如Linux Container等。

虚拟机（VM）介绍

- 以VMWare、XEN为代表
- 是一种系统级别的虚拟方式，可以将按照用户需求构造虚拟的计算单位、存储、网络等设备资源，提供了一种透明而自然的虚拟计算环境，实现隔离级别最高的资源管理方案，适宜于具有不同操作系统要求的作业间进行资源分配。

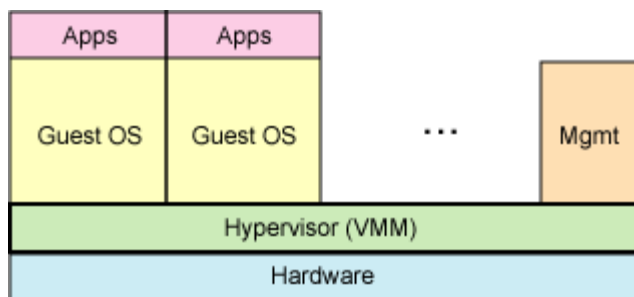


图1. 完全虚拟化 Vmware等

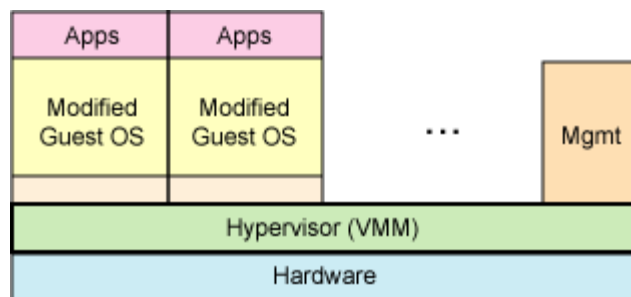


图2. 半虚拟化 Xen 等

资源容器介绍

- 以LXC(linux container)、OpenVZ为代表。
- OS级别的虚拟方案，可以在同一操作系统中提供多个互相隔离的用户空间实例，适宜于对同构操作系统和运行环境中的作业进行资源分配和隔离。

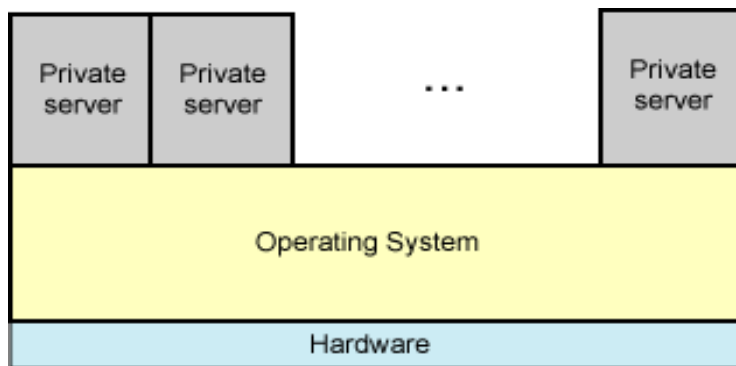


图3. 操作系统级别的虚拟化 LXC 、OpenVZ等

腾讯案例

业务	机器数量	平均负载
Qzone	3053	0.2974
Qzone第三方应用	2626	0.265
QQ秀	906	0.4415
QQ音乐	392	0.3293
拍拍网	311	0.5958
QQ校友	292	0.4564
QQ相册	291	0.368
QQ会员	267	0.3237
QQLive	261	0.5719
数据中心	209	0.2456
数据安全	186	0.3091
运营分析	110	0.6286
QQVideo	93	0.3162
P2P	69	0.3594
业务公共类	50	0.2417
支付营销	39	0.413
白领社区	18	0.0723



- 业务突发性强

快速占领市场，占领用户，业务需要快速的资源请求回应

问题：各个部门维护自己的集群服务器。

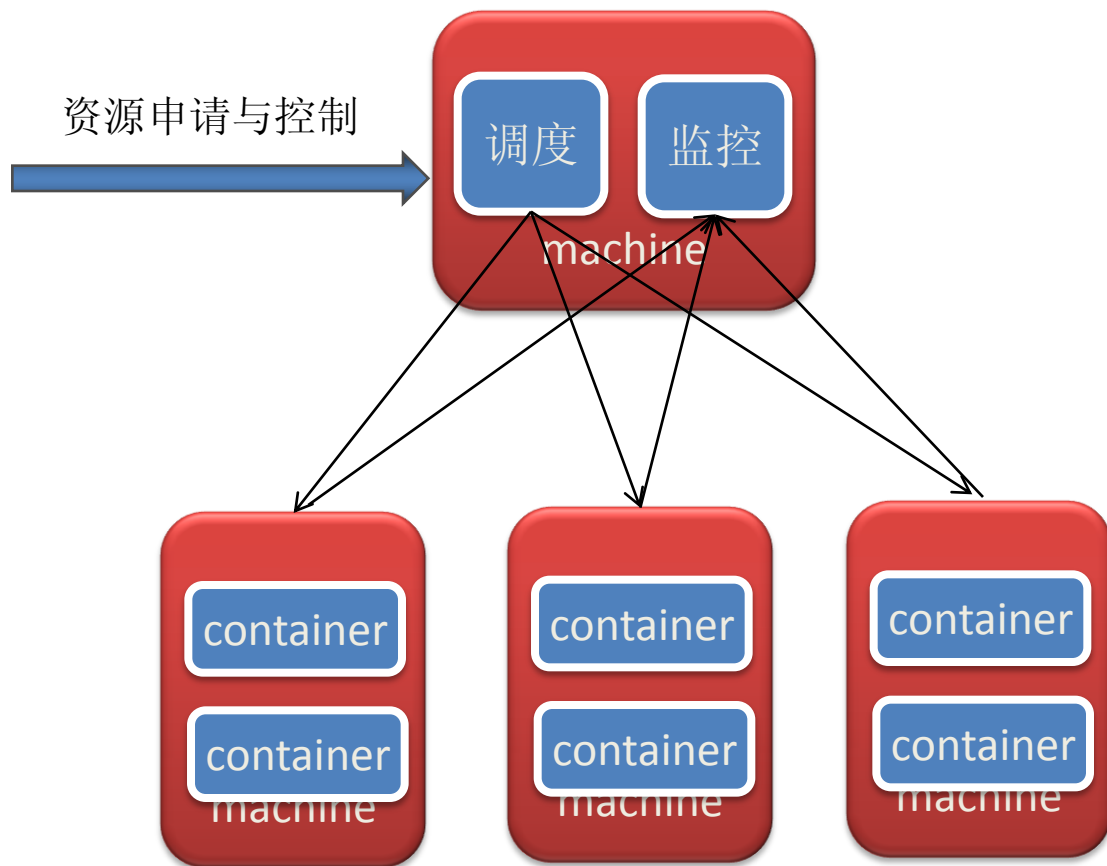
各部门的集群服务器负载不高，资源浪费严重

集群资源的弹性管理

- 基于Linux Container的集群资源管理

- 资源调度与分配
- 资源的实时监控

- ◆ 实时上报container的资源情况进行监控
- ◆ 根据机器的资源使用状况动态分配任务资源



3、“大数据”的计算

- 高性能集群计算
 - MPI
 - PVM
- 多核并行计算
 - OpenMP
 - pThread
- 云计算
 - MapReduce等
- “大数据”的计算
 - Stream Computing
 - Graph Computing等

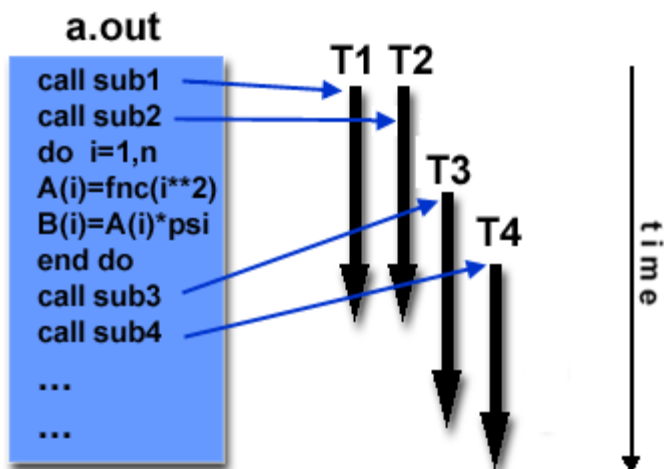


分布式与并行计算

分布式并行计算的几种常见模型

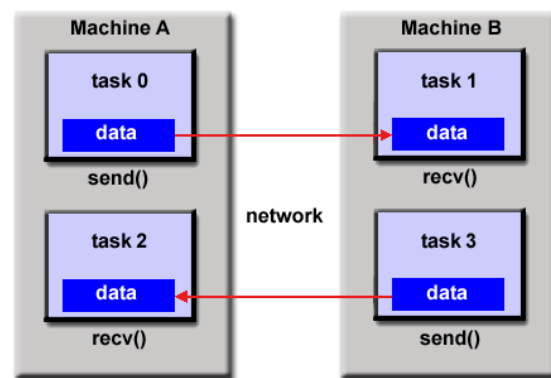
基于**线程**的共享内存模型

- Posix threads
- OpenMP



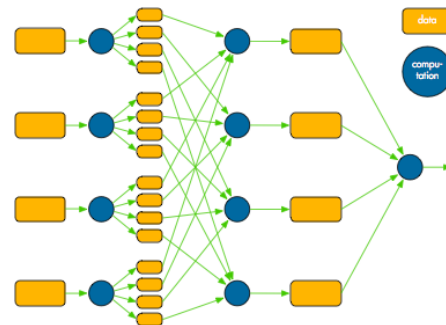
基于**消息**的分布式内存模型

- MPI



基于**文件**的并行编程模型

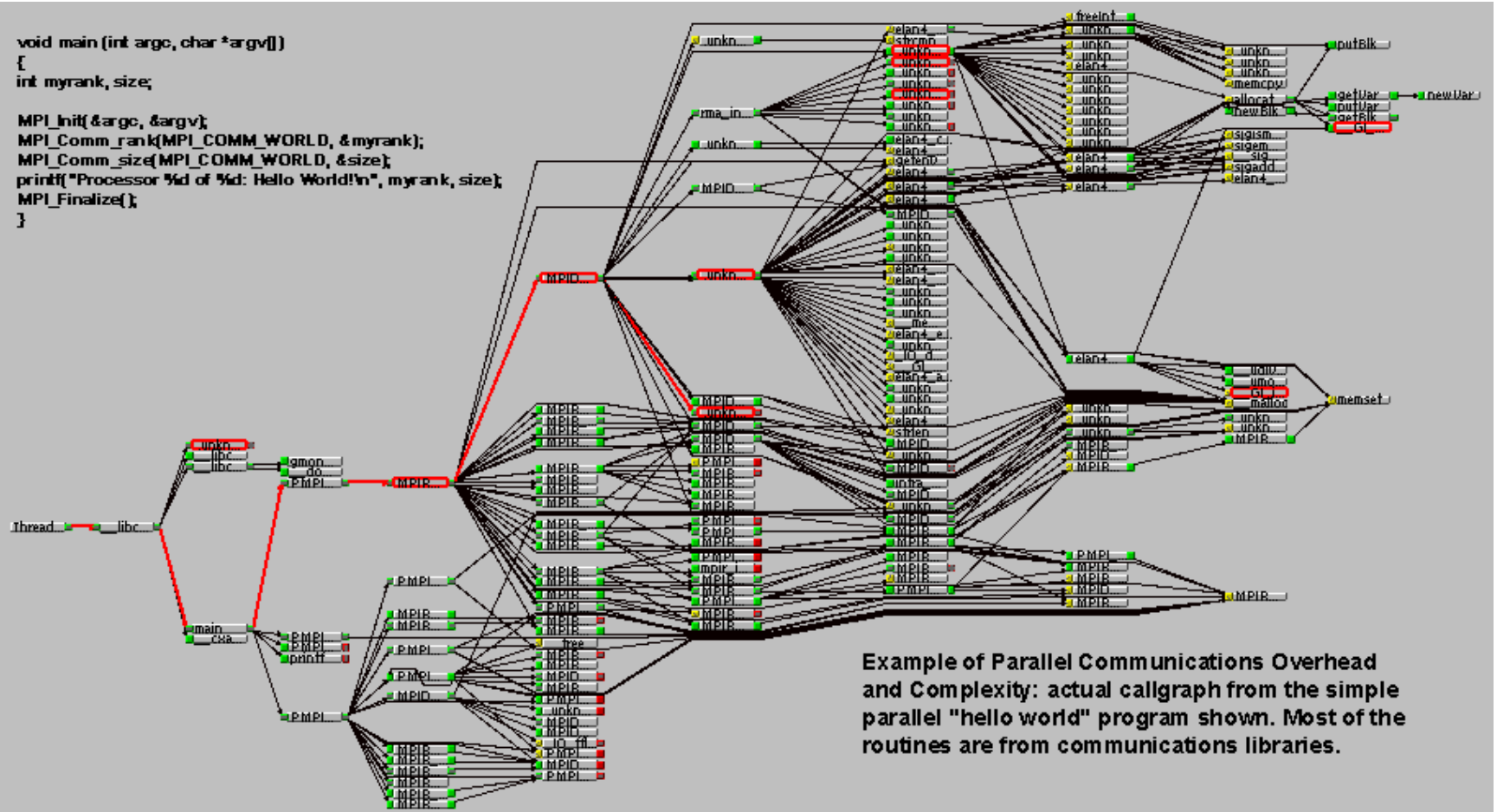
- MapReduce



通信常常是并行计算的瓶颈

```
void main(int argc, char *argv[])
{
    int myrank, size;

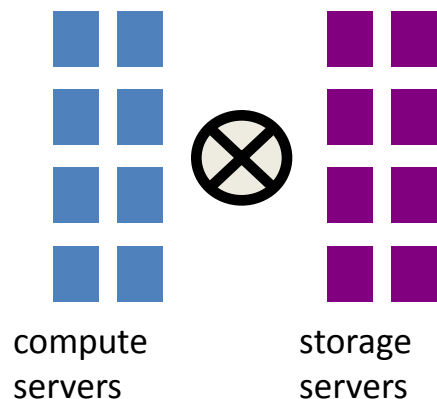
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    printf("Processor %d of %d: Hello World!\n", myrank, size);
    MPI_Finalize();
}
```



Example of Parallel Communications Overhead and Complexity: actual callgraph from the simple parallel "hello world" program shown. Most of the routines are from communications libraries.

计算与存储的关系

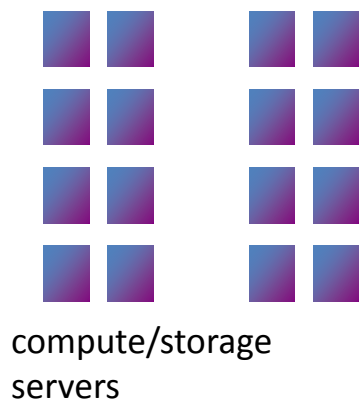
Model 1: Separate Compute/Storage



Compute and storage can scale independently

Many opportunities for reliability

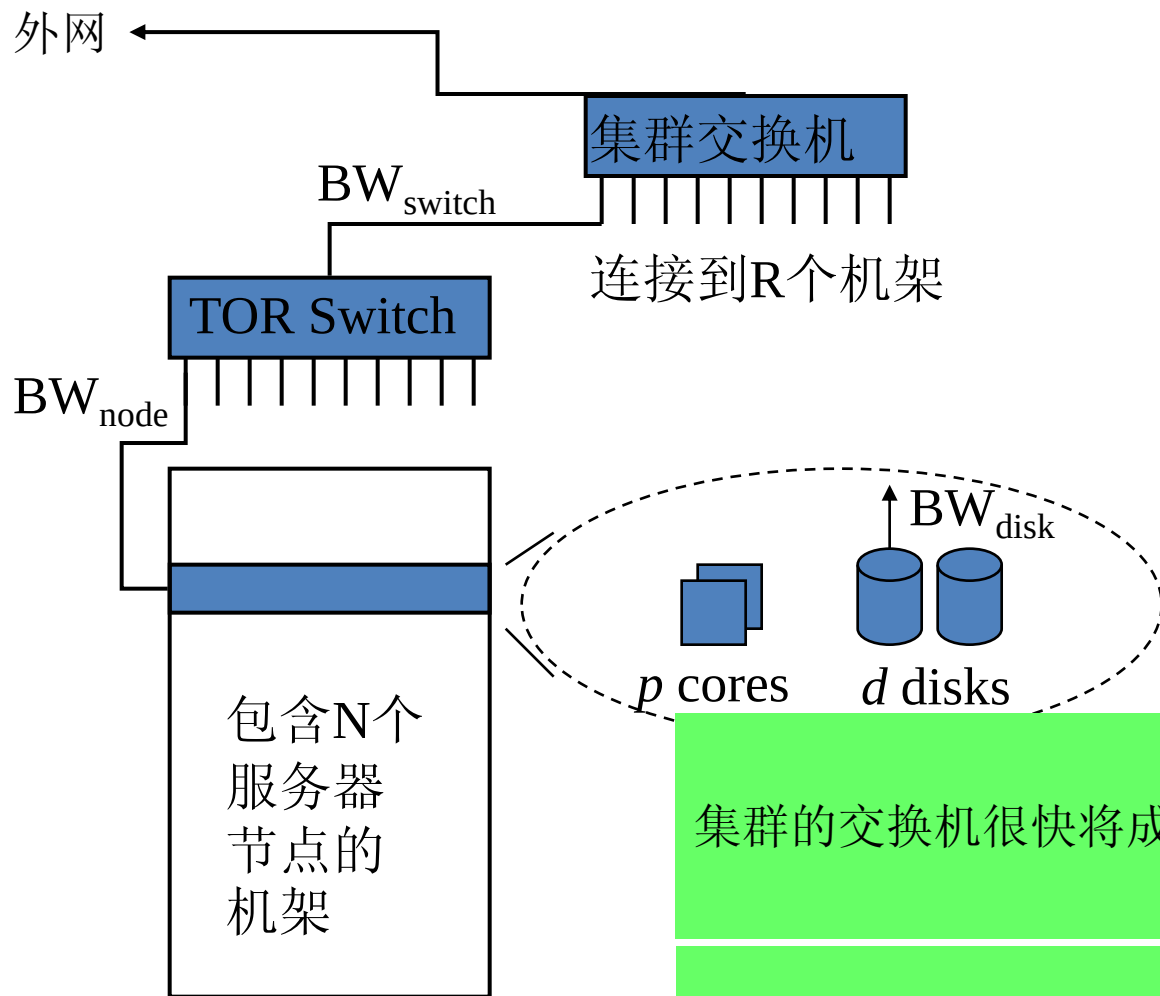
Model 2: Co-located Compute/Storage



No compute resources are under-utilized

Potential for higher throughput

集群的典型结构



集群的交换机很快将成为瓶颈

本地化成为关键

编程模型与通信机制

	Posix Threads	OpenMP	MPI	MapReduce
媒介	线程	线程	消息	文件
同步	读/写同步	读/写同步	在同步点通信	对用户透明
粒度	最细	细	较粗	粗粒度

- 通信机制是编程模型的映射，大体可分为基于线程、消息和文件三种
- 基于线程和基于消息的编程模型均需要用户编程来实现数据通信
- 基于文件的编程模型并行粒度较粗，不适合逻辑复杂的细粒度并行任务

“大数据”的计算

	Volume	Velocity	Variety
高性能计算	***	***	**
多核并行计算	*	****	*
云计算	*****	**	***
大数据计算	*****	*****	*****

传统的分布式并行计算无法满足大数据的计算需求

大数据的计算

- 流式处理
 - Twitter's Storm
 - Yahoo's S4
- 图计算
 - 数据库 HyperGraphDB
 - 分布式系统 Pregel (Google)
 - 综合系统 Trinity (Microsoft)
 - 开源系统 Hama (Apache)



结束语

- “大数据”不仅仅是大的数据
 - Volume、Velocity、Veriaty、Value
- “大数据”将带来新的技术发展
 - 数据存储
 - 数据管理
 - 数据计算



感谢聆听！