第三讲　分布式文件系统HDFS

徐 辰
cxu@dase.ecnu.edu.cn
華東師範大學

大纲

2

- 文件系统(FS)
  - 文件系统概述
  - 文件与目录
  - 文件的物理结构
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)

文件系统概述

3

- 文件系统出现的原因
  - 用户直接操作和管理辅助存储器上信息（01二进制序列），繁琐复杂、易于出错、可靠性差
- 文件系统是操作系统中负责**管理**和**存取**信息的模块
  - 统一管理用户和系统信息的存储、检索、更新、共享和保护
  - 为用户提供一整套方便有效的文件使用和操作方法

文件系统概述

4

- 文件系统的功能：
  - 文件的按名存取（基本功能）
  - 文件目录的建立和维护（用于实现上述基本功能）
  - 实现逻辑文件到物理文件的转换（核心内容）
  - 文件存储空间的分配和管理
  - 数据保密、保护和共享
  - 提供一组用户使用的操作

大纲

5

- 文件系统(FS)
  - 文件系统概述
  - 文件与目录
  - 文件的物理结构
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)

文件

6

- 文件
  - 文件是由文件名字标识的一组信息的集合
  - 各操作系统的文件命名规则略有不同
- 实现按名存取的文件系统的优点
  - 将用户从复杂的物理存储地址管理中解放出来
  - 可方便地对文件提供各种安全、保密和保护措施
  - 实现文件的共享（同名共享、异名共享）

文件目录

7

- 如何实现“按名存取”?
  - 当用户要求存取某个文件时，系统查找**目录文件**，获得对应的**文件目录**
  - 在文件目录中，根据用户给定的文件名寻找到对应该文件的**文件控制块**（文件目录项）
  - 通过文件控制块所记录的该文件的相关信息（如文件信息存放的相对位置或文件信息首块的物理位置）依次存取该文件的内容。

文件目录

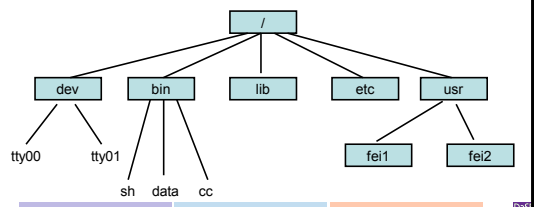
8

- 概念：
  - 文件目录：建立和维护的关于系统的所有文件的清单
  - 文件控制块：每个目录项对应一个文件的信息描述
  - 目录文件：目录信息也以文件的形式存放
- 文件控制块的基本内容
  - 文件存取控制信息：如文件名、用户名、文件主存取权限等
  - 文件结构信息：文件逻辑结构、文件的物理结构等
  - 文件使用信息：已打开该文件的进程数、文件的修改情况等
  - 文件管理信息：文件建立日期、文件访问日期等

文件目录

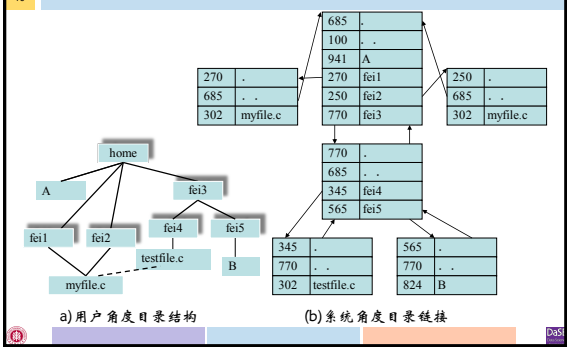
9

- 树形目录结构
  - 文件的全名，应该从根目录开始，到该文件名为止，目录路径+文件名
    - 例如，/user/include/testfile.c



不同角度的目录结构

10



大纲

11

- 文件系统(FS)
  - 文件系统概述
  - 文件与目录
  - 文件的物理结构
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)

文件的物理结构

12

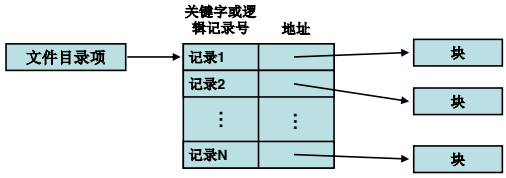
- 文件在物理存储中的存放方法和组织关系
  - 块（物理记录）的划分
  - 记录的排列
  - 索引的组织
  - 信息的搜索
- 常见的文件物理结构：
  - 顺序文件，连续存储
  - 链接文件
  - 索引文件

} 非连续存储

举例：索引文件

13

- 系统为每个文件建立一张索引表，每个表目包含一个记录键（或逻辑记录号）及其对应的存储地址



大纲

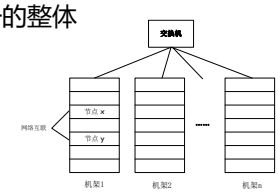
14

- 文件系统(FS)
- 分布式文件系统(DFS)
  - 体系架构
  - 文件访问
  - 备份与一致性
  - 容错管理
- Hadoop分布式文件系统(HDFS)

DFS实现的思路

15

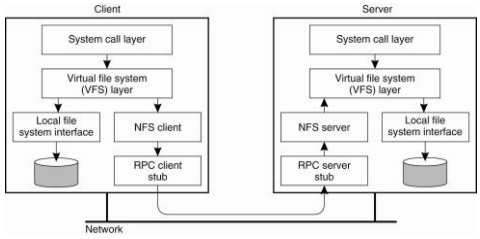
- 思路1：保证每台机器均可透明地访问其它机器上的文件
- 思路2：将所有机器的文件系统关联起来，形成一个对外统一的整体



Client-Server Architectures

16

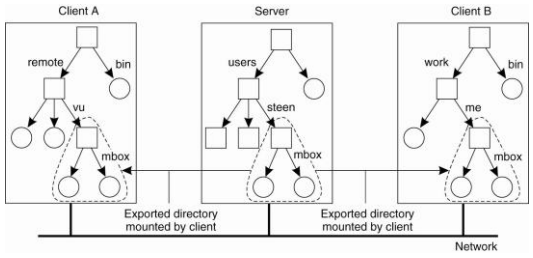
- 通过RPC调用访问其它机器上的文件，对用户而言透明



Client-Server Architectures

17

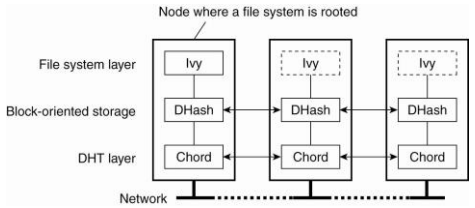
- 文件系统的挂载



Symmetric Architectures

18

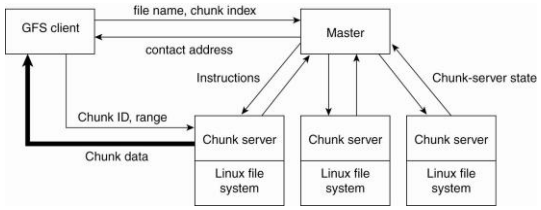
- 通过特殊的hash算法将文件划分到各台机器上，需要访问文件时可根据hash算法进行定位



Cluster-Based Distributed File Systems (1)

19

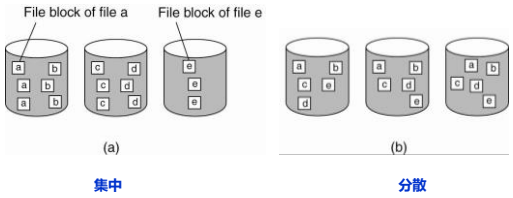
- 主节点进行管理，从节点存储数据



Cluster-Based Distributed File Systems (2)

20

- 文件切分成块，分散存储在从节点上



大纲

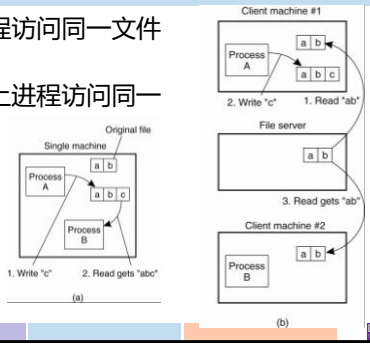
21

- 文件系统(FS)
- 分布式文件系统(DFS)
  - 体系架构
  - 文件访问
  - 备份与一致性
  - 容错管理
- Hadoop分布式文件系统(HDFS)

文件访问

22

- 单机多进程访问同一文件
  - 读写锁
- 不同机器上进程访问同一文件



大纲

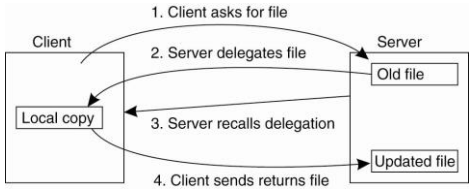
23

- 文件系统(FS)
- 分布式文件系统(DFS)
  - 体系架构
  - 文件访问
  - 备份与一致性
  - 容错管理
- Hadoop分布式文件系统(HDFS)

备份与一致性

24

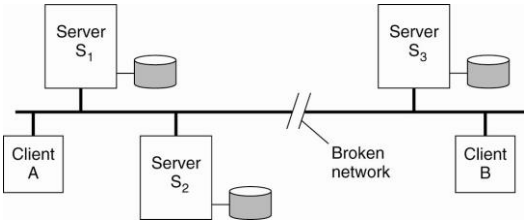
- 客户端备份：Client-Server DFS



备份与一致性

25

- 服务器端备份： Cluster-Based DFS



大纲

26

- 文件系统(FS)
- 分布式文件系统(DFS)
  - 体系架构
  - 文件访问
  - 备份与一致性
  - 容错管理
- Hadoop分布式文件系统(HDFS)

故障类型

27

- 发生故障即停机：Fail stop
- 拜占庭将军问题
  - 拜占庭失效指一方向另一方发送消息，另一方没有收到，或者收到了错误的信息的情形

大纲

28

- 文件系统(FS)
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)
  - 设计思想
  - 体系架构
  - 工作原理
  - 容错机制
  - 编程使用

Hadoop发展简史

29

- Hadoop最初是由Apache Lucene项目的创始人Doug Cutting开发的文本搜索库。Hadoop源自始于2002年的Apache Nutch项目——一个开源的网络搜索引擎并且也是Lucene项目的一部分
- 2004年，Nutch项目也模仿GFS开发了自己的分布式文件系统NDFS（Nutch Distributed File System），也就是HDFS的前身
- 2004年，谷歌公司又发表了另一篇具有深远影响的论文，阐述了MapReduce分布式编程思想
- 2005年，Nutch开源实现了谷歌的MapReduce
- 2006年2月，Nutch中的NDFS和MapReduce开始独立出来，成为Lucene项目的一个子项目，称为Hadoop，同时，Doug Cutting加盟雅虎
- 2008年1月，Hadoop正式成为Apache顶级项目，Hadoop也逐渐开始被雅虎之外的其他公司使用
- 2008年4月，Hadoop打破世界纪录，成为最快排序1TB数据的系统，它采用一个由910个节点构成的集群进行运算，排序时间只用了209秒
- 在2009年5月，Hadoop更是把1TB数据排序时间缩短到62秒。Hadoop从此名声大震，迅速发展成为大数据时代最具影响力的开源分布式开发平台，并成为事实上的大数据处理标准

Hadoop核心项目

30

HDFS

HDFS: Hadoop Distributed File System 分布式文件系统

MapReduce

MapReduce：并行计算框架

需要解决的问题

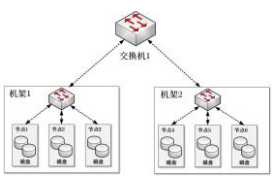
31

- 如何存储上百GB/TB级别大文件?
  - 例如，某一主题的网页构成的数据集存成一个大文件
- 如何保证文件系统的容错?
  - 集群由低廉的普通服务器甚至个人PC组成，节点发生故障是普遍的现象
- 如何进行大文件的并发读写控制?
  - 文件的并发读写往往需要加锁等一系列复杂的措施来避免读写冲突

集群网络拓扑

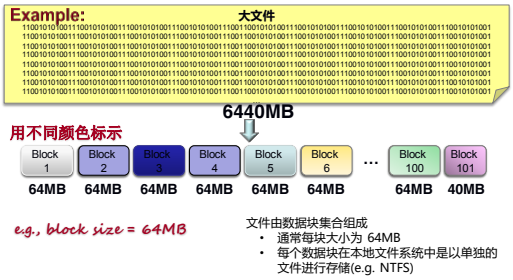
32

- 机架（Rack）间通信
- 机架内通信



文件分块存储

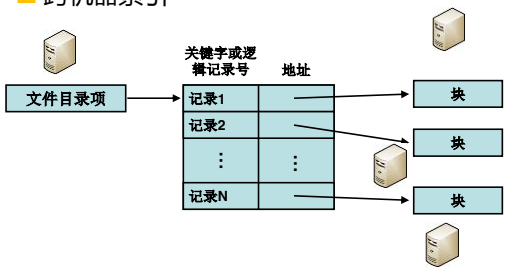
33



回顾：索引文件

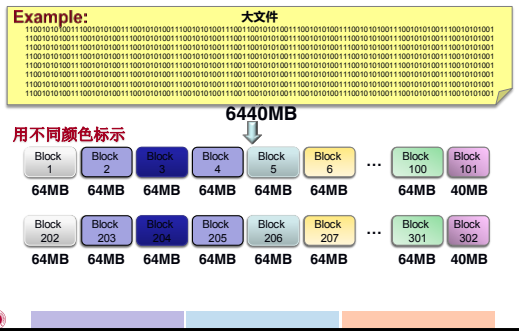
34

- 跨机器索引



分块冗余存储

35



简化文件读写

36

- 避免读写冲突
  - 文件一次写入后不再修改而仅容许多次读取
- 避免随机写
  - 仅支持顺序写入而不容许随机写入

HDFS设计假设、目标

37

硬件失效	流式数据访问	存储数据较大	简化的数据一致性模型	多硬件平台支持	移动计算能力比移动数据更划算
• 硬件的异常比软件的异常更加常见。 • 对于有上百台服务器的数据中心来说，认为总有服务器异常，硬件异常是常态。 • HDFS需要监测这些异常，并自动恢复数据。	• 基于HDFS的应用仅采用流式方式读取数据。 • 运行在HDFS上的应用并非以通用业务为目的的应用程序 • 应用程序关注的是吞吐量，而非响应时间。 • 非POSIX标准接口的数据访问。	• 运行在HDFS的应用程序有较大的数据需要处理。 • 典型的文件大小为GB到TB级别。	• 应用程序采用WORM（Write Once Read Many）的数据读写模型。 • 文件仅支持追加，而不允许修改。	• HDFS易于运行不同的平台上。	• 计算和存储采用就近原则，计算离数据最近。 • 就近原则将有效减少网络的负载，降低网络拥塞。

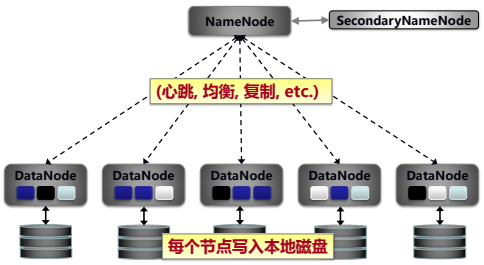
大纲

38

- 文件系统(FS)
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)
  - 设计思想
  - 体系架构
  - 工作原理
  - 容错机制
  - 编程使用

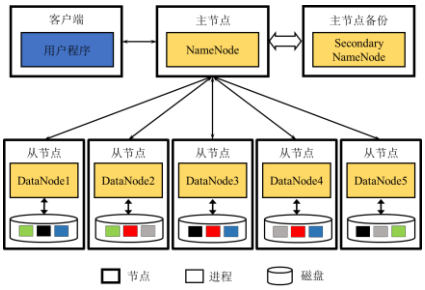
HDFS架构图

39



HDFS架构图

40



HDFS角色

41

- **NameNode** – 通常每个集群一个NameNode
  - 负责文件系统元数据操作、数据块的副本及其定位

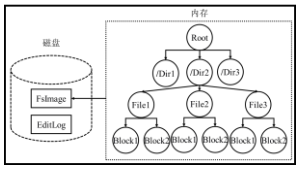
- **SecondaryNameNode** – NameNode的备份

- **DataNodes** – 通常集群中每个节点一个DataNode
  - 负责数据块的存储
  - 为客户端提供实际文件数据

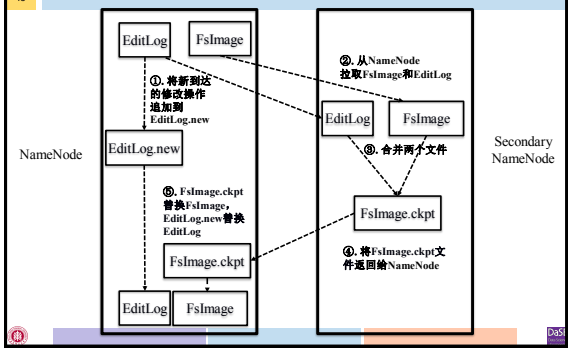
NameNode

42

- FsImage：内存在文件目录结构及其元信息在磁盘上的快照
- EditLog：两次快照之间，针对目录及文件修改的操作

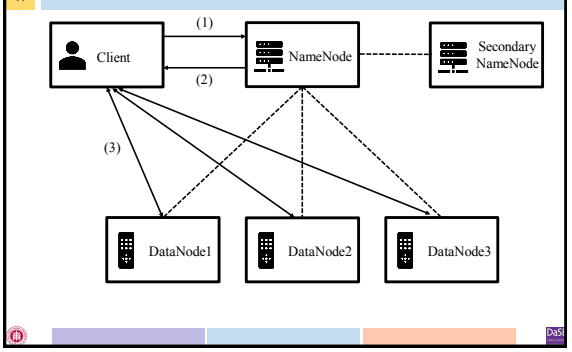


NameNode与SecondaryNameNode



并非实时备份

应用程序执行流程



应用程序执行流程

45

- 客户端向NameNode发起文件操作请求
- NameNode反馈
  - 如果是读写文件操作，则NamNode告知Client文件块存储的位置信息。
  - 如果是创建、删除、重命名目录或文件等操作，NameNode修改文件目录结构成功后结束。
  - 对于删除操作，HDFS并不会立即去删除DataNode上的数据块，而是等到特定时间才会真正删除。
- 对于读写文件操作，客户端获知具体位置信息后再与DataNode进行读写交互

大纲

46

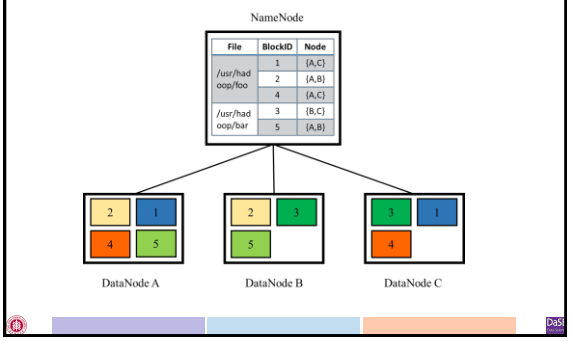
- 文件系统(FS)
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)
  - 设计思想
  - 体系架构
  - 工作原理
  - 容错机制
  - 编程使用

文件分块与备份

47

- DataNode提供文件的存储　位置：hdfs-site.xml的dfs.data.dir
- 文件块（block）：最基本的存储单位，一个Linux文件
  - HDFS默认Block大小是64MB，
  - 以一个256MB文件，共有256/64=4个Block
- 不同于普通文件系统的是，HDFS中如果一个文件小于一个数据块的大小，并不占用整个数据块存储空间
- 数据备份：默认是三个　hdfs-site.xml的dfs.replication属性

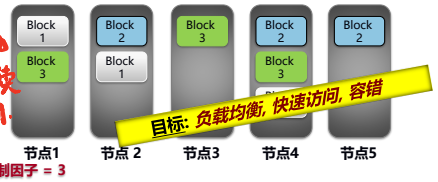
文件分块与备份



文件块存放策略（启发式）

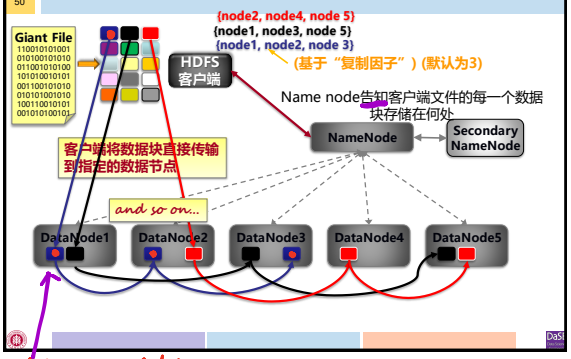
49

- 第一个副本：放置在上传文件的数据节点；如果是集群外提交，则随机挑选一台磁盘不太满、CPU不太忙的节点 (**快速写入**)
- 第二个副本：放置在与第一个副本不同的机架rack的节点上 (**减少跨rack的网络流量**)
- 第三个副本：与第一个副本相同机架的其他节点上(**应对交换机故障**)
- 更多副本：随机节点



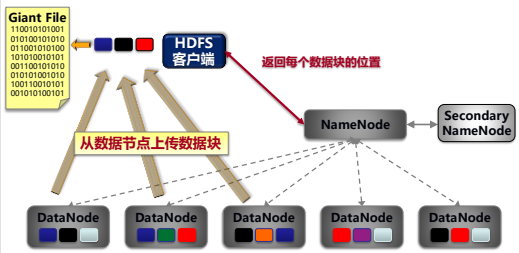
文件写入HDFS

50



从HDFS读取文件

51



数据读取策略

52

- 当客户端读取数据时，从NameNode获得数据块不同副本的存放位置列表，列表中包含了副本所在的数据节点
- 可以调用API来确定客户端和这些数据节点所属的机架ID
- 最近者优先原则：当发现某个数据块副本对应的机架ID和客户端对应的机架ID相同时，就优先选择该副本读取数据，如果没有发现，就随机选择一个副本读取数据

文件读写与一致性

53

- “一次写入多次读取”
  - 一个文件经过创建、写入和关闭后就不得改变文件中的内容
  - 已经写入到HDFS文件，仅容许在文件末尾追加数据，即append操作
  - 当对一个文件进行写入操作时，包括文件的追加操作，NameNode将拒绝其它针对该文件的读、写请求
  - 当对一个文件进行读取操作时，NameNode容许其它针对该文件的读请求。

简化的一致性模型

54

- 简化的好处
  - 避免读写冲突、用户编程无需考虑文件锁
- 问题
  - 假如用户的确需要修改已有文件中的内容，怎么办？
  - 如果HDFS容许修改文件中的已有内容，会带来哪些问题？

OLAP (分析)

OLTP

	OLTP	OLAP
用户	操作人员,低层管理人员	决策人员,高级管理人员
功能	日常操作处理	分析决策
DB 设计	面向应用	面向主题
数据	当前的, 最新的细节的, 二维的分立的	历史的, 聚集的, 多维的集成的, 统一的
存取	读/写数十条记录	读上百万条记录
工作单位	简单的事务	复杂的查询
用户数	上千个	上百个
DB 大小	100MB-GB	100GB-TB

大纲

55

- 文件系统(FS)
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)
  - 设计思想
  - 体系架构
  - 工作原理
  - 容错机制
  - 编程使用

容错机制

56

- HDFS在设计时就考虑到故障(**硬件和软件**)会经常出现

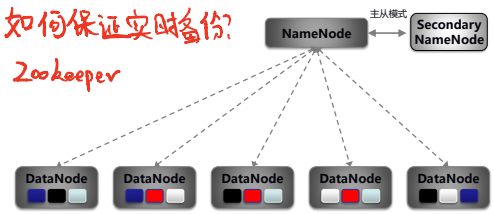
故障类型:

- 磁盘错误和故障
- DataNode故障
- 交换机/机架故障
- NameNode故障
- 数据中心故障

NameNode故障

57

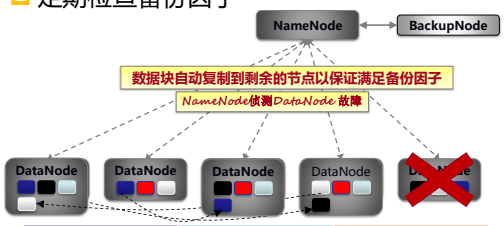
- 根据SecondaryNameNode中的FsImage和Editlog数据进行恢复



DataNode故障

58

- “宕机”，节点上面的所有数据都会被标记为“不可读”
- 定期检查备份因子



其它故障

59

- 磁盘错误或故障：数据校验
- 交换机/机架故障
- 数据中心故障

大纲

60

- 文件系统(FS)
- 分布式文件系统(DFS)
- Hadoop分布式文件系统(HDFS)
  - 设计思想
  - 体系架构
  - 工作原理
  - 容错机制
  - 编程使用

HDFS Shell

61

- 新建目录
  - ./bin/hdfs dfs –mkdir input
- 上传文件
  - ./bin/hdfs dfs -put ./README.txt ./input
- 查看文件
  - ./bin/hdfs dfs –cat ./input/README.txt
  - ./bin/hdfs dfs -ls /
- 拷贝
  - ./bin/hdfs dfs -cp ./input/README.txt /

Java程序

62

- 设置环境
  - Configuration conf = new Configuration();
  - conf.set("fs.defaultFS", "hdfs://localhost:9000");
  - conf.set("fs.hdfs.impl", "org.apache.hadoop.hdfs.DistributedFileSystem");
- 写文件
  - FSDataOutputStream
- 读文件
  - FSDataInputStream

实践：单机伪分布式

HDFS功能

63

- HDFS适合做什么
  - 大文件存储
  - 流式数据访问
- HDFS不适合做什么
  - 大量小文件
  - 随机读取
  - 低延迟读取
- 谁在使用HDFS？地球人都在用

YAHOO! facebook amazon.com Bai百度 IBM 中国移动通信 CHINA MOBILE 淘宝网 Taobao.com

课后阅读

64

- 分布式系统概念与设计，George Coulouris等著，金蓓弘等译
  - 第12章 12.1、12.2
  - 第21章 21.5.1
- 论文
  - Ghemawat, S., Gobioff, H., & Leung, S.-T. (2003). The Google File System. In SOSP (pp. 29–43).

本讲小结

65

- FS
- DFS
- HDFS
  - 设计思想
  - 体系架构
  - 工作原理
  - 容错机制
  - 编程使用

谢谢！Q&A