

Since 2012.11.27:

Visitors

 CN 214,009	 AU 1,051
 US 10,994	 CA 907
 HK 6,317	 GB 682
 TW 3,578	 DE 478
 JP 2,732	 FR 420
 SG 1,331	 KR 339

Pageviews: 311,619



HDFS基本知识整理

设计理念:

- 1、超大文件
- 2、流式数据访问
- 3、商用普通硬件

不适合场景:

- 1、低时间延迟的数据访问
- 2、大量的小文件
- 3、多用户写入, 任意修改文件

一、HDFS的基本概念

1.1、数据块(block)

HDFS([Hadoop](#) Distributed File System)默认的最基本的存储单位是64M的数据块。

和普通文件系统相同的是, HDFS中的文件是被分成64M一块的数据块存储的。

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。

目的: 最小化寻址, 加快数据传输速度

列出文件系统中各个文件有哪些块构成:

```
hadoop fsck / -files -blocks
```

1.2、元数据节点(Namenode)和数据节点(datanode)

Namenode用来管理文件系统的命名空间, 维护着文件系统树即整棵树内所有的文件和目录。

永久保存即保存在硬盘上的信息: 命名空间镜像(namespace image)及修改日志(edit log)

其还保存了一个文件包括哪些数据块, 分布在哪些数据节点上。然而这些信息并不存储在硬盘上, 而是在系统启动的时候从数据节点收集而成的, 存放在内存中的, 所以namenode对内存要求很大。

数据节点是文件系统中真正存储数据的地方。

客户端(client)或者元数据节点(namenode)可以向数据节点请求写入或者读出数据块。

数据节点(datanode)周期性的向元数据节点回报其存储的数据块信息。

其主要功能就是周期性将元数据节点的命名空间镜像文件和修改日志合并，以防日志文件过大。这点在下面会详细叙述。

合并过后的命名空间镜像文件也在从元数据节点保存了一份

(`${dfs.name.dir}/image/fsimage`)，以防元数据节点失败的时候，可以恢复。

1.2.1、元数据节点文件夹结构

```
${dfs.name.dir}/current/VERSION
                        /edits
                        /fsimage
                        /fstime
```

VERSION文件是java properties文件，保存了HDFS的版本号。


layoutVersion是一个负整数，保存了HDFS的持续化在硬盘上的数据结构的格式版本号。

namespaceID是文件系统的唯一标识符，是在文件系统初次格式化时生成的。

cTime此处为0

storageType表示此文件夹中保存的是元数据节点的数据结构。

```
namespaceID=1232737062
cTime=0
storageType=NAME_NODE
layoutVersion=-5
```



1.2.2、文件系统命名空间映像文件及修改日志

当文件系统客户端(client)进行写操作时，首先把它记录在修改日志中(edit log)

元数据节点在内存中保存了文件系统的元数据信息。在记录了修改日志后，元数据节点则修改内存中的数据结构。

每次的写操作成功之前，修改日志都会同步(sync)到文件系统

(`${dfs.name.dir}/current/edits`)。

fsimage文件，也即命名空间映像文件，是内存中的元数据（namenode所维护的目录树的结构信息，不包括blocks的位置信息）在硬盘上的checkpoint，它是一种序列化的格式，并不能够在硬盘上直接修改。

同数据的机制相似，当元数据节点失败时，则最新checkpoint的元数据信息从fsimage（存放在secondary namenode中，即`${dfs.name.dir}/image/fsimage`文件）加载到内存中，然后逐

昵称: [beanmoon](#)

园龄: 8年

粉丝: 89

关注: 16

[+加关注](#)

< 2012年11月 >						
日	一	二	三	四	五	六
28	29	30	31	1	2	3

18	19	20	21	22	23	24
25	26	27	28	29	30	1
2	3	4	5	6	7	8

搜索

 谷歌搜索

最新随笔

- 1.如何克隆kvm虚拟机
- 2.如何在mac os中安装gdb及为gdb进行代码签名
- 3.android 中targetSdkVersion和与target属性的区别
- 4.在android中如何通过点击edittext之外的部分使软键盘隐藏
- 5.如何交换两个等长整形数组使其数组和的差最小 (C和java实现)
- 6.如何对excel进行列查重
- 7.eclipse快捷键大全
- 8.unix/linux中图形界面那些事
- 9.vmware虚拟机工具vmware tools介绍及安装排错
- 10.debian软件源source.list文件格式说明

积分与排名

积分 - 145083

排名 - 4113

随笔分类

- Android(3)
- C语言(8)
- Database(6)
- Debian环境(6)
- hadoop&大数据(16)
- Hadoop编程笔记(4)
- Hadoop学习笔记(10)
- hbase学习(2)
- Java(23)

checkpoint的过程如下:

从元数据节点通知元数据节点生成新的日志文件, 以后的日志都写到新的日志文件中。

从元数据节点用http get从元数据节点获得fsimage文件及旧的日志文件。

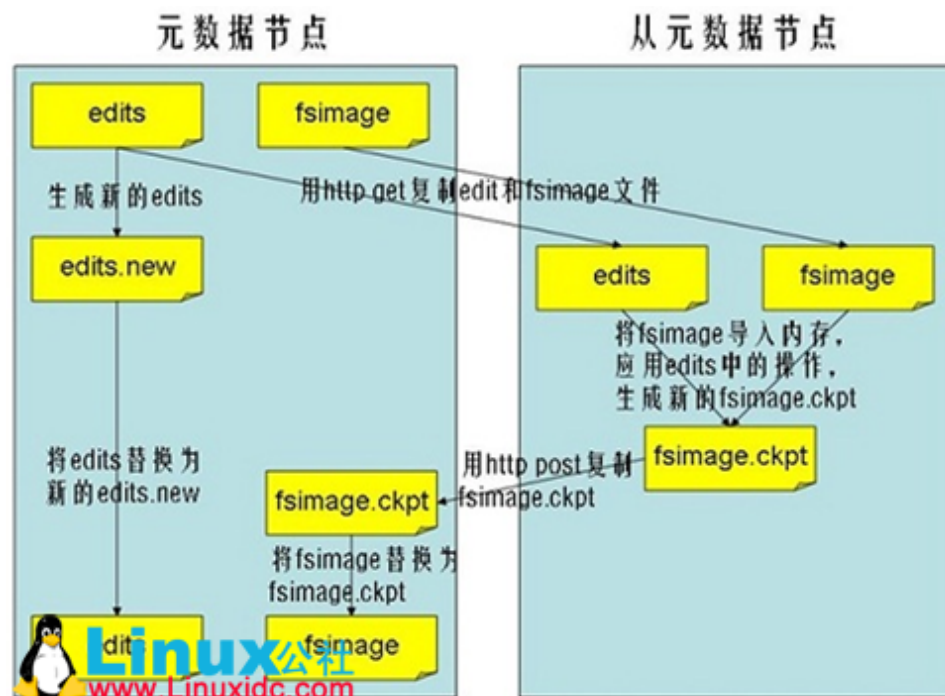
从元数据节点将fsimage文件加载到内存中, 并执行日志文件中的操作, 然后生成新的fsimage文件。

从元数据节点将新的fsimage文件用http post传回元数据节点

元数据节点可以将旧的fsimage文件及旧的日志文件, 换为新的fsimage文件和新的日志文件(第一步生成的), 然后更新fstime文件, 写入此次checkpoint的时间。

这样元数据节点中的fsimage文件保存了最新的checkpoint的元数据信息, 日志文件也重新开始, 不会变的很大了(细心的读者可以发现系统启动的时候, 日志文件

`dfs.name.dir}/current/edits`的修改时间会发生变化, 而且size会变小, 这就是secondary namenode合并了fsimage和edits的缘故)。



1.2.3、从元数据节点的目录结构

[php\(2\)](#)
[Python\(1\)](#)
[Web综合\(2\)](#)
[分布式计算与集群\(2\)](#)
[感悟&杂谈\(8\)](#)
[计算机杂学\(13\)](#)
[嵌入式\(2\)](#)
[算法与数据结构\(3\)](#)
[心理学\(3\)](#)
[云计算&虚拟化\(5\)](#)

最新评论

1. Re:详细解析Java中抽象类和接口的区别

@ xingmengshenhua package com.xxg;
interface A{ void aa(); } class Test9 implements A{ @Override pub...

--凉月缘

2. Re:java中static作用详解

static成员是可以被其所在class创建的实例访问的，只不过被该class创建的实例访问还不如被该class直接调用.....

--java浪里小白龙

3. Re:java中static作用详解

static成员是不能被其所在class创建的实例访问的。倒数第三句话对？

--钱明浪

4. Re:java中static作用详解

a ni a se you

--黄聖宇

5. Re:详细解析Java中抽象类和接口的区别

哥们,写的很好,不过字体也太小了吧,眼睛都要看瞎了..

--10号球员

阅读排行榜



1.2.4、数据节点的目录结构

```
/${dfs.data.dir}/current/VERSION
/blk_<id_1>
/blk_<id_1>.meta
/blk_<id_2>
/blk_<id_2>.meta
/...
/blk_<id_64>
/blk_<id_64>.meta
/subdir0/
/subdir1/
/...
/subdir63/
```



数据节点的VERSION文件格式如下：

```
namespaceID=1232737062
storageID=DS-1640411682-127.0.1.1-50010-1254997319480
cTime=0
storageType=DATA_NODE
```



blk_<id>保存的是HDFS的数据块，其中保存了具体的二进制数据。

blk_<id>.meta保存的是数据块的属性信息：版本信息，类型信息，和checksum

当一个目录中的数据块到达一定数量的时候，则创建子文件夹来保存数据块及数据块属性信息。

[2. MySQL数据库无法远程连接的解决办法\(42715\)](#)

[3. java中static作用详解\(26479\)](#)

[4. 详细解析Java中抽象类和接口的区别\(20213\)](#)

[5. Hadoop学习笔记（七）：使用distcp并行拷贝大数据文件\(15773\)](#)

评论排行榜

[1. 如何统计博客园的个人博客访问量\(17\)](#)

[2. Hadoop学习笔记（九）：如何在windows上使用eclipse远程连接hadoop进行程序开发\(6\)](#)

[3. Hadoop学习笔记（二）：从map到reduce的数据流\(6\)](#)

[4. Hadoop学习笔记（五）：一些关于HDFS的基本知识\(5\)](#)

[5. 详细解析Java中抽象类和接口的区别\(4\)](#)

推荐排行榜

[1. 如何统计博客园的个人博客访问量\(14\)](#)

[2. Hadoop学习笔记（五）：一些关于HDFS的基本知识\(5\)](#)

[3. Hadoop学习笔记（九）：如何在windows上使用eclipse远程连接hadoop进行程序开发\(3\)](#)

[4. Hadoop学习笔记（一）之示例程序：计算每年的最高温度MaxTemperature\(3\)](#)

[5. Hadoop编程笔记（二）：Hadoop新旧编程API的区别\(2\)](#)

FileSystem.open()

客户端(client)用FileSystem的open()函数打开文件

DistributedFileSystem用RPC调用元数据节点，得到文件的数据块信息。

对于每一个数据块，元数据节点返回保存数据块的数据节点的地址。

DistributedFileSystem返回FSDataInputStream给客户端，用来读取数据。

客户端调用stream的read()函数开始读取数据。

DFSInputStream连接保存此文件第一个数据块的最近的数据节点。

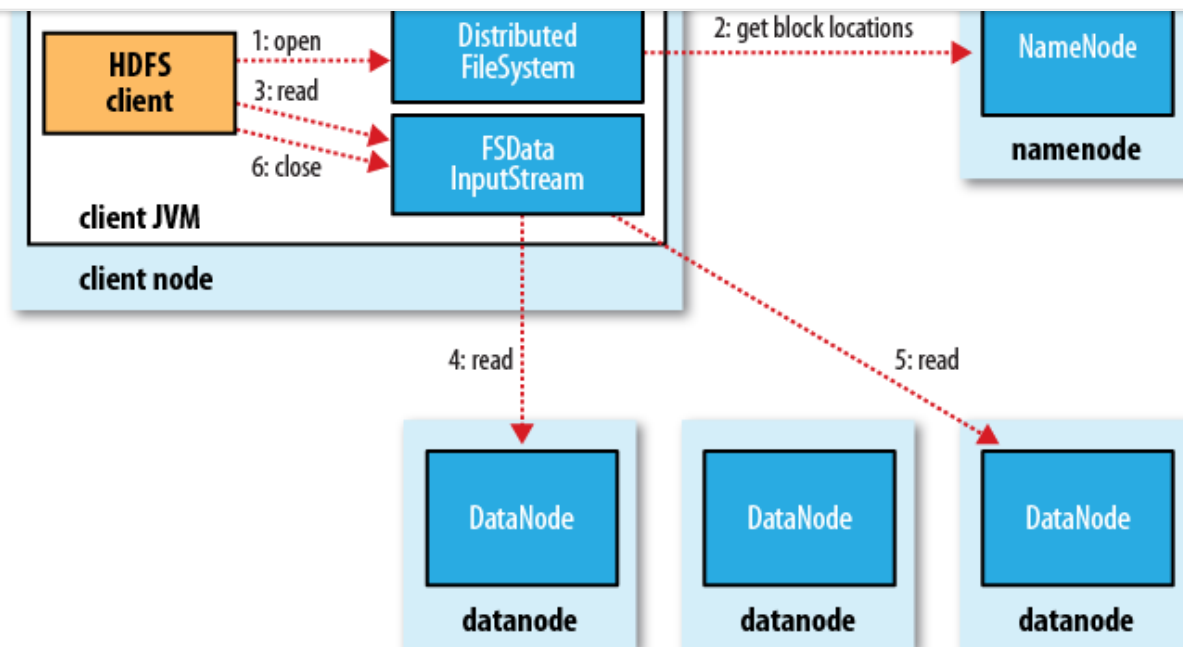
Data从数据节点读到客户端(client)

当此数据块读取完毕时，DFSInputStream关闭和此数据节点的连接，然后连接此文件下一个数据块的最近的数据节点（可能有人会问为什么同一个文件的同一个备份的不同数据块还可能保存在不同的datanode上吗，是这样的，因为可能存在datanode当机，或者某个datanode空间不足的情况，这样写数据时的pipeline就会导向到其他的datanode从而数据块保存到其他位置上去）。

当客户端读取完毕数据的时候，调用FSDataInputStream的close函数。

在读取数据的过程中，如果客户端在与数据节点通信出现错误，则尝试连接包含此数据块的下一个数据节点。

失败的数据节点将被记录，以后不再连接。



2.2、写文件的过程

客户端调用create()来创建文件

DistributedFileSystem用RPC调用元数据节点，在文件系统的命名空间中创建一个新的文件。

元数据节点首先确定文件原来不存在，并且客户端有创建文件的权限，然后创建新文件。

DistributedFileSystem返回DFSOutputStream，客户端用于写数据。

客户端开始写入数据，DFSOutputStream将数据分成块，写入data queue。

Data queue由Data Streamer读取，并通知元数据节点分配数据节点，用来存储数据块(每块默认复制3块)。分配的数据节点放在一个pipeline里。

Data Streamer将数据块写入pipeline中的第一个数据节点。第一个数据节点将数据块发送给第二个数据节点。第二个数据节点将数据发送给第三个数据节点。

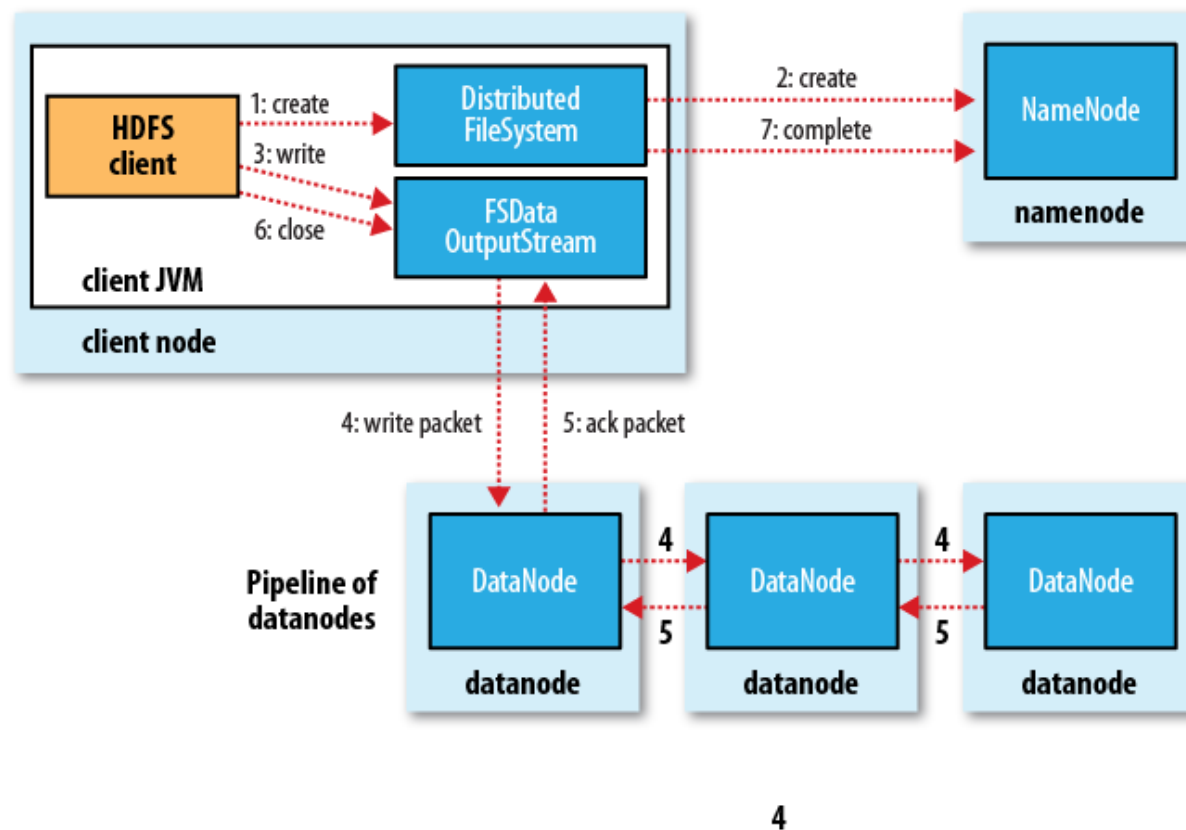
DFSOutputStream为发出去的数据块保存了ack queue，等待pipeline中的数据节点告知数据已经写入成功。

如果数据节点在写入的过程中失败：

关闭pipeline，将ack queue中的数据块放入data queue的开始（即重新写入未被确认的数据块）。

当前的数据块在已经写入的数据节点中被元数据节点赋予新的标示，则错误节点重启后能够察觉其

pipeline中只有两个数据节点，但并不影响数据块的写入，因为hdfs默认只要有一个备份写入成功，就认为此数据写入成功，将来其他两个备份可以从这一个备份中重新创建）。元数据节点则被通知此数据块是复制块数不足，将来会再创建第三份备份。当客户端结束写入数据，则调用stream的close函数。此操作将所有数据块写入pipeline中的数据节点，并等待ack queue返回成功。最后通知元数据节点写入完毕。



参考链接: <http://www.linuxidc.com/Linux/2012-06/62885p2.htm>

作者: [beanmoon](#)

出处: <http://www.cnblogs.com/beanmoon/>

本文版权归作者和博客园共有，欢迎转载，但未经作者同意必须保留此段声明，且在文章页面明显位置给出原文

分类: [hadoop&大数据](#)

好文要顶 关注我 收藏该文



 [beanmoon](#)
[关注 - 16](#)
[粉丝 - 89](#)
[+加关注](#)

0

0

« 上一篇: [java主要集合类的数据结构学习](#)
» 下一篇: [工厂方法模式 \(Factory method\)](#)

posted @ 2012-11-23 10:46 beanmoon 阅读(1745) 评论(0) 编辑 收藏

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问](#) 网站首页。

- 【推荐】超50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库
- 【推荐】阿里毕玄16篇文章，深度讲解Java开发、系统设计、职业发展
- 【推荐】阿里专家五年方法论总结！技术人如何实现职业突破？

- [Hadoop 学习总结之一：HDFS简介](#)
- [HadoopHDFS](#)
- [HDFS简介](#)
- [HDFS基本概念](#)
- » [更多推荐...](#)

[精品问答：微服务架构 Spring 核心知识 50 问](#)

最新 IT 新闻:

- [Zoom再出问题 故障中断干扰了英国政府的COVID-19通报会](#)
- [覆盖100多城市去年巨亏近8亿 丰巢坚持收费背后的资本暗战早已打响](#)
- [瑞幸门店调整 宣判前的挣扎?](#)
- [Linux平板PineTab即将开启预订 承诺可运行多数Linux发行版](#)
- [realme TV定档5月25日 智能手表有望同场亮相](#)
- » [更多新闻...](#)

Copyright © 2020 beanmoon
Powered by .NET Core on Kubernetes