基于 MapFile 的 HDFS 小文件存储效率问题①

洪旭升, 林世平

(福州大学 数学与计算机科学学院, 福州 350108)

摘 要: 针对 HDFS 最初是为流式访问大文件而开发的, 而对于大量小文件的存储效率不高问题, 采用 MapFile 设计一个 HDFS 中存储小文件的方案. 该方案的主要思想是在上传 HDFS 时增加一个文件类型判断模块, 建立一个小文件队列, 将小文件序列化存入一个 MapFile 容器,合并成大文件, 并建立相应的索引文件, 有效降低文件数目和提高访问效率. 通过和现有的 Hadoop Archives(HAR files)文件归档解决小文件问题的方案对比, 实验结果表明, 基于 MapFile 的存储小文件方案可以更为有效的提高小文件存储性能和减少 HDFS 文件系统的节点内存消耗.

关键词: HDFS; 小文件; MapFile; SequenceFile; 云存储

Efficiency of Storaging Small Files in HDFS Based on MapFile

HONG Xu-Sheng, LIN Shi-Ping

(School of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: The Hadoop distributes file system(HDFS) which can process large amounts of data effectively through large clusters. However, HDFS is designed to handle large files and suffers performance penalty while dealing with large number of small file. An approach based on MapFile is proposed to improve storage efficiency of small files in HDFS.The main idea is to add a file type judgment module while uploading a file, and create a small file queue, put the small file serialization in a MapFile container.and establishes the index file. Experimental results show that, the storage efficiency of small files is improved contrast to Hadoop Archives(HAR files).

Key words: HDFS; small file; MapFile; sequence file; cloud storage

1 引言

云计算[1]是网格计算, 并行计算和分布式计算的进一步发展, 是当前计算机研究与应用的热点问题. 云存储是云计算的底层服务, 对上层提供存储支持, 有效存储和管理海量数据, 所以更是成为业界关注的焦点.

开源项目 HADOOP[2]下的默认分布式文件系统 HDFS(Hadoop Distributed File System)既有高可靠性和伸缩性, 以流式数据访问模式存储大文件而设计的文件系统, 可以运行在低廉的硬件集群上. HDFS[3]采用 master/slaves 的主从架构, 一个 HDFS 集群由一个 Namenode 节点和多个 Datanode 节点组成的. Namenode 是一个中心服务器, 负责管理文件系统的元数据和客户端对文件的访问. 可以看出只有一个 Namenode 的架构设计简化了文件系统的整体结构, 但是也造成了小文件存储效率低的问题. 名称节点(Namenode)存储着文件的元数据, 因此 Namenode 的内存容量限制了文件的数量. 每个文件,block 以及索引目录在内存中均以对象的形式存储, 每个对象约占 150 字节. 举例来说, 如果有 1000000 个小文件, 每个文件占用一个 block, 则 Namenode 就至少需要 300M 的内存. 如果存储 1 亿甚至更多的文件时, Namenode 需要 20G 甚至更多的内存容量, 超出目前的硬件能力.

然而, 在实际互联网应用当中, 存在着海量的小文件. 尤其是随着博客, 微博、百科、空间等社交网站的兴起改变了互联网提高内容的方式, 基本上用户已

① 收稿时间:2012-03-28;收到修改稿时间:2012-05-01

经成为互联网的内容创造者, 其数据具有海量、多样、动态变化等特点, 由此产生了海量的小文件, 如日志文件, 资料介绍, 用户头像等.

由于以上问题的存在, 目前也有了一些相应的解决方案. Mackey 等[4]提出用 Hadoop Archive(HAR)技术实现小文件合并为大文件方法, 有效减小小文件数量. 但一旦创建 archices 就不可以改变, 不能增加或移除里面的文件, 必须重新创建归档文件. Liu 等[5]针对 WebGIS 系统特点, 利用 WebGIS 中数据相关性特点, 将保存相邻地理位置信息的小文件合并为一个大文件, 并建立相应的索引文件提高文件读取效率. Dong 等[6]针对 PPT 课件存储特点, 提出了将属于同一个课件的文件合并成为一个大文件并引入一种 two-level prefetching 机制以提高小文件读取效率, 即索引文件预取和数据文件预取.

以上的研究工作基本上是针对特定的应用场景提出的解决方案, 本文提出了一种基于 MapFile 技术在 HDFS 中存储小文件的方案: 在文件上传时增加一个辨别小文件模块, 建立一个小文件队列, 利用 MapFile 技术存储小文件, 合并成大文件, 建立相应的索引文件, 提高文件的读取效率.

2 相关研究:

对于小文件问题, 现在已经有几种基本的技术解决方案但都分别存在各自的问题, 分别为: Hadoop Archive, SequenceFile 和 CombineFileInputFormat.

2.1 **Hadoop Archive 文件归档**

Hadoop Archives (HAR files)文件归档[7]是为了缓解大量小文件消耗 namenode 内存的问题而设计的. HAR 文件是特殊的档案格式. 一个 HAR 对应一个文件系统目录. 它是在 HDFS 上构建一个层次化的文件系统来工作. 一个 HAR 文件是通过 hadoop 的 archive 命令来创建, 而这个命令是运行了一个 MapReduce 任务来将多个小文件打包成一个 HAR 文件. 对于客户端来说, 使用 HAR 文件没有任何影响, 可以对文件进行透明访问.

一个 HAR 文件包含元数据和数据文件. 元数据有两层索引文件(如图 1), 所以在通过 HAR 读取一个文件可能比直接从 HDFS 中读取效率低. 一个 HAR 文件一旦创建就不可在更改, 如要增加或删除文件, 必须重新创建归档文件.

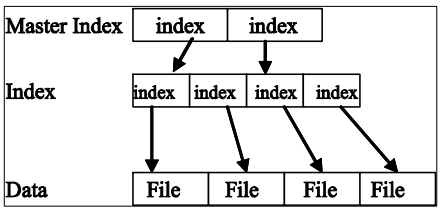

图 1 HAR 文件结构

2.2 **SequenceFile**

SequenceFile[8]是 Hadoop API 提供的一种二进制文件支持. 可以通过将多个小文件组织起来统一存储的容器. 它的数据结构是由一系列的二进制 key/value 组成, 在这种技术中, 将文件名存入 key, 文件内容存入 value, 则可以将大量小文件合并成一个大文件. 假设有 100000 个 10KB 的小文件, 可以写一个程序按照 key/value 结构把这些小文件存入一个 SequenceFile 容器中. 而且 SequenceFile 还支持压缩, 数据压缩有利于节省磁盘空间和加快网络传输速度.

由于 SequenceFile 结构(如图 2)没有建立相应的小文件到大文件的映射关系, 如没建立索引,则查询小文件就需遍历整个 SequenceFile, 降低文件读取效率.

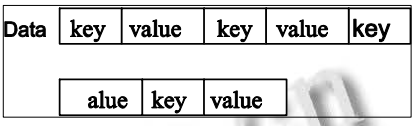

图 2 SequenceFile 文件结构

2.3 **CombineFileInputFormat**

Hadoop 不适合处理大量小文件有一个原因是 FileInputFormat 生成的 InputSplit 总是整个或一部分作为输入文件. 在处理大量小文件时, 每次 map 操作只处理很少的输入数据, 造成过多的 map 任务操作, 降低整体性能. CombineFileInputFormat 是一种新的 inputformat, 可以缓解这个问题. 它用于将多个小文件合并成一个单独的 split. 而且 Combine FileInput Format 会考虑数据的存储位置.

3 基于MapFile的小文件存储方案

3.1 **MapFile**

MapFile 可以说是带索引版的 SequenceFile, 是排序后的 SequenceFile, 通过观察其目录结构(如图 3)可以看到 MapFile 由两部分组成, 分别是 data 和 index.

data 存储数据, index 存储索引文件.

MapFile 也是序列化的将文件名存入 key, 文件内容存入 value , 合并成为大文件. 以此同时并建立从小文件到大文件之间的映射关系. 在索引文件中主要记录每个 Record 的 key 值, 以及该 Record 在文件中的偏移位置. 在通过 MapFile 访问文件时, index 索引文件会被加载到内存, 并从索引映射关系中可迅速定位到指定 Record 所在文件位置, 因此, 相对于 SequenceFile 而言, MapFile 的检索效率是明显提高很多.

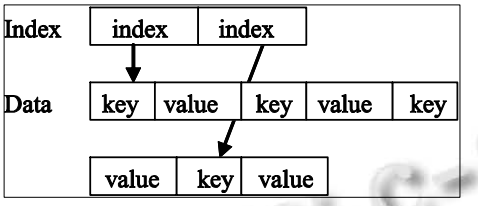

图 3 MapFile 文件结构

3.2 小文件存储方案

在上面介绍的几种小文件解决技术都存在着各自的问题, 而且均需要定期对 HDFS 中的小文件进行归档处理, 以便减少小文件数量,这样就对 Hadoop 的管理员带来了很多不便. 本文设计了在文件上传的时候增加一个文件类型判断模块, 当一个文件到达时, 判断该文件是否属于小文件, 如果是, 则交给小文件处理模块处理, 如果不是, 则直接上传 HDFS. 小文件处理模块设计的主要思想是, 先将多个小文件合并成一个大文件, 然后为这些小文件建立索引, 以便进行快速存取和访问.

小文件处理流程图如图 4 所示. 整个方案执行过程可以描述如下:

(1)用户上传文件到云存储系统, 在系统中有一个文件类型判断模块, 判断上传文件是大文件还是小文件, 具体可以设置一个阀值, 本系统设置阀值为 1M, 小于 1M 的为小文件, 其他为大文件.

(2)如果上传的文件为大文件, 则跳过小文件处理模块, 直接存储于 HDFS 中.

(3)如果上传的文件为小文件, 则把文件的索引放入小文件队列中, 当队列到达一定阀值时, 采用 MapFile 技术, 把队列中小文件合并为大文件, 最后完成小文件到大文件的映射.

(4)最后处理已经合并的小文件, 有效减少文件数目.

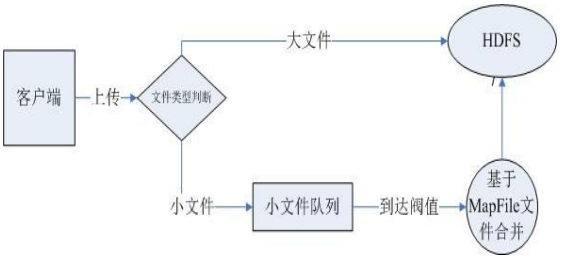

图 4 小文件处理流程图

4 实验和分析

本实验测试平台: 5 台机器组成的集群[9], 其中 1 台为 Namenode 名称节点, 其他 4 台为 Datanode 数据节点.

实验机器配置如下: CPU 为 AMD Athlon(tm) 64 X2 Dual Core Processor 4200+, 主频 2.21GHz, 2GB 内存, 160G 硬盘, 操作系统为 Windows XP Professional.

实验主要测试文件的上传和读取执行时间, 以及各个节点的平均内存占用情况.

(1)上传执行时间测试:

测试用的数据包含 277860 个文件, 总量 3.3G. 其中小于 100k 的占总数的 95.66%. 分别将这些文件直接上传到 HDFS 和经过本文改进方案上传到 HDFS, 如表 1 所示:

表 1 直接上传和改进后上传的时间消耗

	直接上传	改进后上传
时间(s)	10029	10203

(2)文件读取时间测试:

首先, 对上传测试中的未经改进上传的小文件用 Hadoop Archive(HAR)技术实现小文件合并为大文件. 现在分别测试 3 种环境下的文件读取效率: 1.未经改进合并的 HDFS 文件系统; 2.经过 HAR 合并的 HDFS 文件系统; 3.经过本文改进的 MapFile 合并的 HDFS 文件系统.

为了更加明确对比读取文件效率, 分别读取 1.10000 组随机文件, 每组一个文件, 不重复. 2.1000 组随机文件, 每组 10 个连续文件. 3.100 组随机文件, 每组 100 个连续文件. 测试结果如图 5 所示.

(3)内存平均占用情况

分别在上述 3 种环境下, 文件上传存储过程中以及上传完成后 HDFS 处于空闲状态时, 节点的平均内存占用情况. 测试结果如图 6 所示.

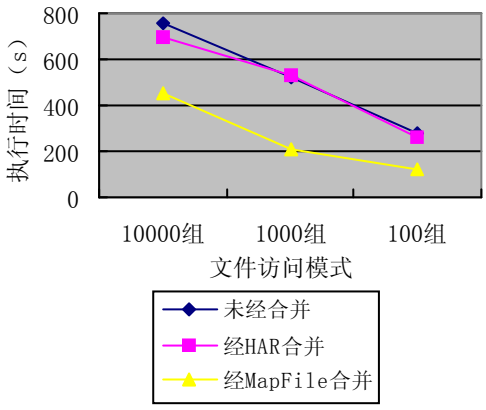

图 5　访问文件读取执行时间对比

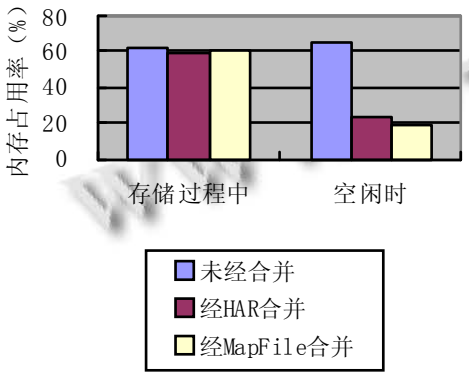

图 6　内存占用率对比

通过以上实验，可以看出本文的改进小文件存储策略在文件上传时的效率与未经改进没什么差别，但是在经过基于 MapFile 序列化的将小文件合并成大文件后，在文件读取方面，比未经改进和经 HAR 合并的环境下效率都高，而且在 HDFS 空闲时，合并过后的内存占用率明显下降，这就减轻了 Namenode 名称节点的负担，提高内存使用率.

5　结语

本文针对 HADOOP 的 HDFS 对于大量小文件的存储效率不高问题，采用基于 MapFile 设计一个在 HDFS 存储小文件的方案. 在上传文件时首先对文件类型进行判断，如为大文件则直接存储，如为小文件则进入小文件处理模块，应用 MapFile 序列化合并小文件为大文件，有效减少小文件数目，节省 Namenode 名称节点内存占用率.

参考文献

1 刘鹏.云计算.第 2 版.北京:电子工业出版社,2011.1–15.
2 White T.周敏奇,王晓玲,金澈清,钱卫宁译.Hadoop 权威指南.第 2 版.北京:清华大学出版社,2011.41–73.
3 HDFS.http://Hadoop.apache.org/hdfs/.
4 Mackey G, Sehrish S, Wang J. Improving metadata management for small files in HDFS. Proc. of 2009 IEEE International Conference on Cluster Computing and Workshops,2009:1–4.
5 Liu XH, Han JZ, Zhong YQ, Han CD, He XB. Implementing WebGIS on Hadoop: A case study of improving small file I/O performance on HDFS. Proc.of the 2009 IEEE Conf.on Cluster Computing and Workshops, 2009:1–8.
6 Dong B, Qiu J, Zheng QH, et al. A nivel approach to improving the efficiency of storing and accessing small files on hadoop: a case study by PowerPoint files. Proc. of the 7th Int. Conf. on Services Computing. Piscataway, NJ, USA: IEEE, 2010: 65–72.
7 Hadoop Archives. http://hadoop.apache.o-rg/common/docs/ r0.20.2/hadoop_archive.
8 Sequence File. http://wiki.apache.org/hadoop/SequenceFile.
9 刘鹏.实战 Hadoop.北京:电子工业出版社,2011.11–34.

(上接第 101 页)

参考文献

1 沈虹.UPQC 并联侧 PSO-Fuzzy 检测算法研究.电力电子技术,2011,45(3):54–56.
2 基于滑窗迭代 DFT 的电力谐波检测.华北电力大学学报,2006,33(3):27–30.
3 曹立威.SPWM 谐波分析的一般方法.电力电子技术,2002,36(4):62–65.
4 汪玉凤,覃荆伟,章振海.基于滑窗迭代和 SVPWM 的检测控制算法的研究.电力电子技术,2011,45(9).
5 陈明凯,段小华,李敏,余虹.扇合矢量法在谐波与无功电流检测中的应用.中国电机工程学报,2008,28(7).
6 朱军卫,龚春英.逆变器单极性电流 SPWM 控制与滞环控制比较.电力电子技术,2004,38(1):26–29.
7 洪峰,单任仲,王慧贞,严仰光.一种变环宽准恒频电流滞环控制方法.电工技术学报,2009,24(1):115–119.
8 Kolhatkar YY. Experimental Investigation of a Single Phase UPQC With Minimum VA Loading. IEEE Trans. on Power Delivery, 2007,22(1):373–380.