基于 Hadoop 的海量车牌图像处理优化技术

侯向宁
(成都理工大学 工程技术学院,四川 乐山 614007)

摘 要:Hadoop 集群下每个小文件均占据一个 Block,一方面存储海量元数据信息消耗了大量的 NameNode 内存,另一方面,Hadoop 为每个小文件单独启动一个 Map 任务,大量的时间花费在启动和关闭 Map 任务上,从而严重降低了 MapReduce 的执行速率。对此,在详细分析已有解决方案的基础上,采用 CFIF 将多个小文件分片打包到大分片中,给每个大分片只启动一个 Map 任务来执行,通过减少启动 Map 任务的数量,提高了处理海量小文件时的效率。通过设计 Hadoop 图像接口类,继承并实现 CFIF 抽象类,最终完成了对海量图像小文件的处理。与常规 HDFS、HAR 和 MapFile 方案在 NameNode 内存空间和运行效率方面进行了对比,结果表明,CFIF 在 NameNode 内存占用率和运行效率方面,都有很好的表现。

关键词:海量小文件;Hadoop 分布式文件系统;分片;打包
中图分类号:TP391 **文献标识码**:A **文章编号**:1673-629X(2018)10-0135-04
doi:10.3969/j.issn.1673-629X.2018.10.028

A Processing Optimization Technique for Massive License Plate Images Based on Hadoop

HOU Xiang-ning
(Engineering & Technical College of Chengdu University of Technology, Leshan 614007, China)

Abstract: Under the Hadoop cluster, each small file occupies a block. On the one hand, to store massive metadata information consumes a lot of NameNode memory; on the other hand, Hadoop starts a Map task for each small file, spending a lot of time on startup and shutdown Map tasks, which severely reduces the execution speed of the MapReduce. In view of this, on the basis of analysis of several existing solutions, we use CFIF abstract class to package multiple small files into a big split, for each big split only start a Map task to perform. By reducing the number of Map tasks, we improve the efficiency when dealing with massive small files. Through designing the Hadoop image interface class, we inherit and implement CFIF abstract class for final completion of the processing of large image small files. The comparison between CFIF and conventional HDFS, HAR and MapFile solutions in the NameNode memory usage rate and operating efficiency shows that the CFIF performs well.

Key words: massive small files; HDFS; split; package

0 引 言

面对海量的车牌数据,传统的基于终端的车牌识别系统受到很大的挑战。Hadoop 云计算平台具有强大的海量数据分发、存储以及对大数据进行并行运算的能力,是大数据处理领域的首选。HDFS 和 MapReduce 是 Hadoop 框架的核心,然而,HDFS 设计的初衷是为了能够存储和分析大文件,并且在实际应用中取得好的效果,但却不适合处理大小在 10 KB ~ 1 MB 的海量车牌图像小文件。这是因为 HDFS 集群下每一个小文件均占据一个 Block,海量的小文件会消耗大量 NameNode 内存,另外,Hadoop 会为每个文件启动一个 Map 任务来处理,大量的时间花费在启动和关闭 Map 任务上,从而严重降低了执行速率。因此,Hadoop 在存储和处理海量小文件时,存储效率和性能会大幅下降。

文中在分析现有解决方案的基础上,利用 CFIF 将小文件打包分片,以解决 HDFS 在存储海量小文件时的瓶颈问题,以及 MapReduce 的执行效率问题。

1 HDFS 小文件问题

HDFS(Hadoop distributed file system)[1-3]是用 Java 开发的能够运行在通用机器上的具有高容错性、高

收稿日期:2017-09-01 **修回日期**:2018-01-11 **网络出版时间**:2018-05-16
基金项目:四川省教育自然科学重点项目(12ZA200);成都理工大学工程技术学院青年科学基金(C122016006)
作者简介:侯向宁(1972-),男,硕士,讲师,CCF 会员(77730M),研究方向为图形图像处理、机器学习与云计算。

吞吐量的分布式文件系统。HDFS 基于 M/S 模式,由一个 NameNode 节点和若干 DataNode 节点构成。NameNode 是主控服务器,其职责是维护 HDFS 命名空间,协调客户端对文件的访问和操作,管理元数据并记录文件数据在每个 DataNode 节点上的位置和副本信息[4-6]。DataNode 是数据存储节点,在 NameNode 的调度下,负责客户端的读写请求以及本节点的存储管理[7-8]。

1.1 问题描述

(1)NameNode 内存瓶颈。

HDFS 中所有元数据信息都存储在 NameNode 内存中。HDFS 在设计时,每一个文件、文件夹和 Block 大约占 150 Byte。假设 HDFS 集群中有 100 万个小文件,每一个小文件均占据一个 Block,就至少需要 3G 内存[9]。所以,海量的小文件会占用大量 NameNode 内存,造成 NameNode 内存的严重浪费,极大限制了集群中文件数量的规模,成为 HDFS 文件系统的一大瓶颈。

(2)数据访问效率低。

在 M/S 模式下,所有的数据读写请求大都要经过 NameNode,当存储海量小文件时,NameNode 会频繁地接受大量地址请求和处理数据块的分配。此外,海量小文件一般分储于不同的 DataNode 上,访问时要不断地从一个 DataNode 切换到另一个 DataNode,严重影响了访问效率和性能[10-11]。此外,客户端在读取海量小文件时,需要与 NameNode 节点频繁通信,极大降低了元数据节点的 I/O 性能。最后,读取海量小文件时,由于小文件存储空间连续性不足,HDFS 顺序式文件访问的优势难以发挥。

(3)MapReduce 运行效率低。

Hadoop 处理海量小文件时,会为每一个小文件启动一个 Map 任务,因此,大量时间浪费在 Map 任务的启动和关闭上,严重降低了执行速率。

1.2 现有解决方案

处理小文件一直是 Hadoop 的一大难题。目前,解决问题的常见方案大致有:Hadoop Archives 方案、Sequence File 方案、MapFile 方案和 HBase 方案[11-13]。

Hadoop Archives 即 HAR,是 Hadoop 系统自带的一种解决方案[14]。其原理是先把文件打包,然后做上标记以便查询。HAR 的缺点是两级索引增加了系统搜索和处理文件的时间,HAR 没有对文件进行合并,文件的数量没有明显减少。这一方面会耗费 NameNode 内存,另一方面 MapReduce 要给每个小文件各启动一个 Map 任务,总的 Map 任务数并没有减少,因此在 MapReduce 上运行时效率很低。

Sequence File 基于文件合并,它把多个小文件归并成一个大文件,通过减少文件数来减轻系统的内存消耗和性能。但是它没有设置索引方式,导致每次查找合并序列中的小文件都要从整个系统的磁盘中去查找,严重影响了系统的效率。

MapFile 是排序后的 Sequence File,MapFile 由索引文件(Index)和数据文件(Data)两大部分构成。Index 作为文件的数据索引,主要记录了每个小文件的键值,以及该小文件在大文件中的偏移位置。其缺点是当访问 MapFile 时,索引文件会被加载到内存,因此会消耗一部分内存来存储 Index 数据。

HBase 以 MapFile 方式存储数据,通过文件合并与分解提高文件的存储效率[15]。其缺点主要是:HBase 规定数据的最大长度是 64 KB,因此不能存储大于 64 KB 的小文件。另外,HBase 只支持字符串类型,要存储图像、音频、视频等类型还需用户做相关的处理。还有,随着文件数的增多,HBase 需要进行大量的合并与分解操作,这样既占用系统资源又影响系统性能[13]。

总之,通过对以上常用方案的分析,发现都不适合解决海量车牌图像的存储与执行效率问题。因此,下节将采用另外一种方案来解决当前所面临的困境。

2 基于 CFIF 的海量车牌处理

Hadoop 在新版本中引入了 CombineFileInputFormat(简称 CFIF)抽象类[16],其原理是利用 CFIF 将来自多个小文件的分片打包到一个大分片中,这种打包只是逻辑上的组合,让 Map 以为这些小文件来自同一分片,每个大分片只启动一个 Map 任务来执行,通过减少 Map 任务的启动数量,节省了频繁启动和关闭大量 Map 任务所带来的性能损失,提高了处理海量小文件的效率。另外,CFIF 在将多个小文件分片打包到一个大分片时会充分考虑数据本地性,因此节省了数据在节点间传输所带来的时间开销。

CFIF 只是一个抽象类,并没有具体的实现,需要自定义。文中利用 CFIF 处理海量车牌小文件,具体流程设计如图 1 所示。

由图 1 可知,要利用 CFIF 来实现海量车牌图像小文件的处理,需要做如下工作:

(1)设计 Hadoop 图像接口类,因为 Hadoop 没有提供图像处理接口,不能处理图像文件数据,因此必须自己定义。

(2)继承 CFIF 类,定义 CombineImageInputFormat 类将海量图像小文件打包成分片,作为 MapReduce 的输入格式。

(3)实现 CombineImageRecordReader,它是 CombineFileSplit 的通用 RecordReader,就是为来自不同文

件的分片创建相对应的记录读取器,负责分片中文件的处理。因为 Hadoop 中已经提供了 CombineFileSplit 的实现,因此 CombineFileSplit 无须再设计。继承 CombineImageRecordReader 类,定义 ImageRecordReader 实现对 CombineFileSplit 分片中单个小文件记录的读取。

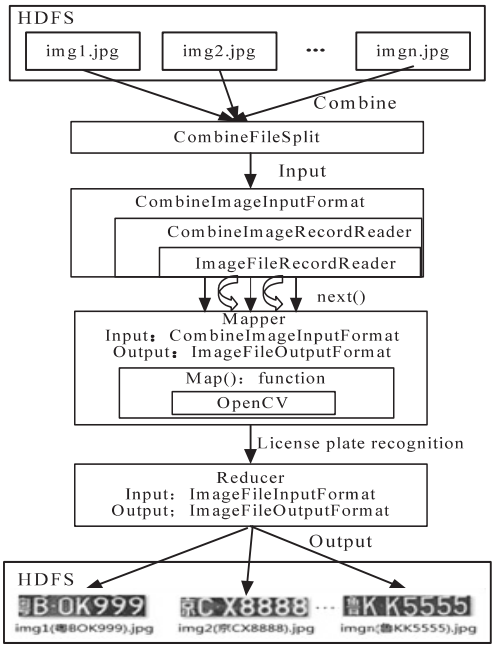

图 1　海量车牌图像小文件处理流程

(4)配置 MapReduce 以实现对海量车牌图像的处理。

2.1　Hadoop 图像接口类

Hadoop 没有提供专用的图像接口,因此不能直接处理车牌图像数据。Hadoop 的 Writable 接口定义了一种二进制输入流方法和一种二进制输出流方法,这两种方法可以实现数据的序列化和反序列化。因此,要处理图像数据,需要继承 Writable 接口,定义一个图像类接口 Image 类,并在 Image 类中重写 Writable 类的 readFields 和 write 两种方法,分别用于写入和读出图像的相关数据信息。经过设计的 Image 类图如图 2 所示。

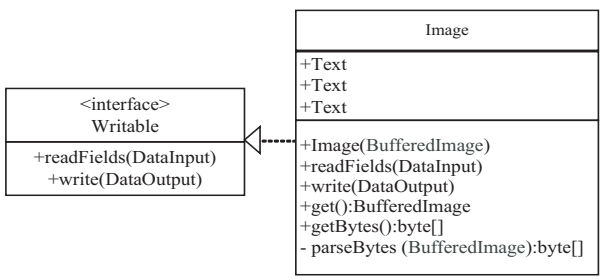

图 2　Image 类图

2.2　主要类的实现

CFIF 作为 Mapper 的输入格式,将来自多个小文件的分片打包到一个大的输入分片中,因此需要从 CFIF 类继承一个 CombineImageInputFormat 类,并在 CombineImageInputFormat 类中创建 CombineImageRecordReader。CombineFileRecordReader 是针对 CombineFileSplit 的通用 RecordReader,是为来自不同文件的分片创建相对应的记录读取器,负责分片中文件的处理,即读取 CombineFileSplit 中的文件 Block 并转化为<Text,Image>键值对。

(1)CombineImageRecordReader 类。

从 RecordReader 继承一个 CombineImageRecordReader 类,把 CombineFileSplit 中的每一个小文件分片转化为<Text,Image>键值对,其中 Text 是图像文件的路径,Image 是图像数据,这样就把输入流转化成图像格式数据,Map 就可以对图像数据做相关的运算了,具体伪代码如下:

```
CombineImageRecordReader extends RecordReader<Text,Image> {
CombineImageRecordReader( CombineFileSplit, split, TaskAttemptContext context,Integer index) {CombineFileSplit ThisSplit=split; //获取文件分片
//对缓冲区中的图像解码,img 用作键值对的值
img=new Image(BufferedImage)};
//获取当前输入文件分片的路径,用作键值对的键
filePath=ThisSplit. getPath(index);}}
```

(2)CombineImageInputFormat 类。

继承 CFIF 类,定义 CombineImageInputFormat 作为 Mapper 的图像输入格式,具体代码如下:

```
CombineImageInputFormat extends CFIF<Text,Image>{
//设置 RecordReader 为 CombineImageRecordReader
RecordReader<Text,Image> createRecordReader(){
return newCombineImageRecordReader<Text,Image>();}
//不对单个图像文件分片
isSplitable(JobContext context,Path file){return false;}}
```

2.3　配置 MapReduce

在 Mapper 阶段,从 CombineImageRecordReader 那里得到<key,image>键值对,对输入的 Image 提取车牌特征,生成新的中间<key,image>键值对,其中 key 仍为图片路径值,image 为特征图,其中包括提取出来的车牌图像以及识别出的车牌号。

在 Reducer 阶段,新 key 表示输出路径,其中图像文件以“原主文件名(车牌号).jpg”的形式命名,并把 image 特征图以新 key 为路径保存。

3　实　验

3.1　实验环境与实验方案

采用伪分布式搭建 Hadoop 云计算平台,该云计算平台由一个 NameNode 节点和三个 DataNode 节点组成。各节点的配置如表 1 所示,所需软件配置如表

2 所示。

表 1 Hadoop 集群节点配置

主机名	节点类型	IP	CPU 核心数	内存
Master	NameNode	192.168.80.100	2	4
Slave1	DataNode	192.168.80.101	2	2
Slave2	DataNode	192.168.80.102	2	2
Slave3	DataNode	192.168.80.103	2	2

表 2 软件配置

序号	软硬件	版本描述
1	VMware	VMware workstation10.0.0
2	OS	CentOS 6.5
3	JDK	jdk 1.7.0_75
4	Hadoop	Hadoop 2.2.0
5	OpenCV	OpenCV 3.0.0

为了测试 CFIF 方式在内存消耗与运行时间方面的性能,特意设计两组对比实验,准备 6 组(1 000,2 000,3 000,4 000,5 000,6 000)车牌图像文件,在不同方案下,分别通过分布式环境运行每组等量的车牌识别任务。

(1)测试 HDFS(即传统的单文件单 Map 任务)、Hadoop Archives(HAR)、MapFile(简称 MF)及 CFIF 方案下,处理 6 组车牌图像文件的完成时间。

(2)测试 HDFS、HAR、MF 和 CFIF 四种方案下 NameNode 节点所占用的内存百分比。

3.2 实验结果与分析

两组实验的测试结果分别如图 3、图 4 所示。

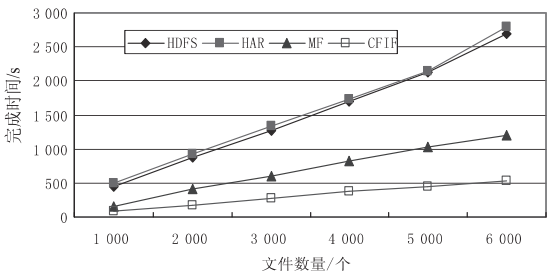

图 3 完成时间对比

由图 3 可见,在运行效率方面,CFIF 在四种方案中表现最好,原因在于 CFIF 方案把小文件分片打包成了大分片,减少了 Map 任务的数量,从而避免了频繁开启和关闭 Map 所带来的时间损耗。MapFile 的处理效率稍逊于 CFIF,原因是 CFIF 在将多个小文件的分片打包到一个大分片时,充分考虑了数据的本地性原则,即尽量将同一节点的小文件打包,因此节省了数据在节点间传输所带来的时间开销,而 MF 方案在合并文件时没有考虑这点。HAR 与 HDFS 方案在运行效率上相当,虽然 HAR 将多个图像小文件打包作为 MapReduce 的输入,但 HAR 没有对文件进行合并,MapReduce 还是将每个图像小文件各启动了一个 Map 任务,这并没有减少总的 Map 任务数,因此 HAR 并不比 HDFS 在运行效率上有效。

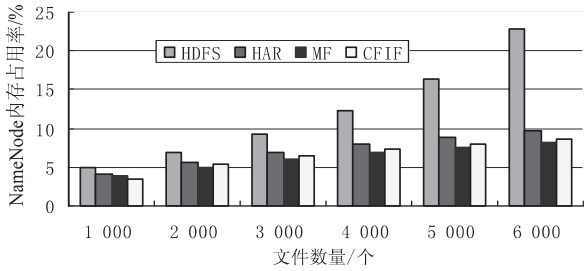

图 4 NameNode 内存占用率对比

由图 4 可见,在 NameNode 内存消耗方面,HAR、MF 和 CFIF 远低于 HDFS,这是因为 HDFS 集群下每一个小文件均占据一个 Block,海量的小文件会消耗大量 NameNode 内存。从图中还可以看出,HAR、MF 和 CFIF 的表现不相上下,原因是三者通过对文件打包及合并,减少了元数据所占用的 NameNode 内存空间,因此 NameNode 内存消耗不明显。

4 结束语

文中深入分析了 Hadoop 存储及处理海量小文件时所引发的性能问题,并对现有的几种解决方案的缺点进行了详细的分析。在此基础上,采用 CFIF 抽象类将多个小文件分片打包到大分片中,以减少 Map 任务的启动数量,从而提高处理海量小文件的效率。对 CFIF 抽象类给出了具体实现,并通过实验与常规 HDFS、HAR 和 MF 方案在 NameNode 内存空间和运行效率方面进行了对比。实验结果表明,CFIF 在 NameNode 内存占用率和运行效率方面都有很好的表现。

参考文献:

[1] WANG Youwei, MA Can, WANG Weiping, et al. An approach of fast data manipulation in HDFS with supplementary mechanisms[J]. Journal of Supercomputing, 2015, 71(5):1736-1753.

[2] CHEN Jilan, WANG Dan, FU Lihua, et al. An improved small file processing method for HDFS[J]. International Journal of Digital Content Technology & Its Application, 2012, 6(20):296-304.

[3] BOK K, OH H, LIM J, et al. An efficient distributed caching for accessing small files in HDFS[J]. Cluster Computing, 2017(20):1-14.

[4] 吴夫丹. 基于云平台的服务器监控系统设计[D]. 西安:西安工业大学,2014.

[5] HAN Yongqi, ZHANG Yun, YU Shui. Research of cloud storage based on Hadoop distributed file system[J]. Applied Mechanics and Materials, 2014, 513-517:2472-2475.

[6] 罗 军,陈仕强. 基于支持向量机的 HDFS 副本放置改进

(下转第 144 页)

信息处理和对句子语义信息表示不足的问题;同时使用 bootstrapping 算法通过种子模板抽取关系模式,不断迭代学习,最终达到需要的数据信息规模,解决了人工干预和语料标注的问题。

下一步将研究跨文档中隐式关系的抽取,以及基于 Web 的企业关系抽取,从而挖掘出更多的实体关系,自动建立全方位的企业生态关系图谱。

参考文献:

[1] 刘克彬,李 芳,刘 磊,等. 基于核函数中文关系自动抽取系统的实现[J]. 计算机研究与发展,2007,44(8):1406-1411.

[2] ZHOU Guodong, QIAN Longhua, FAN Jianxi. Tree kernel-based semantic relation extraction with rich syntactic and semantic information[J]. Information Sciences, 2010, 18(8): 1313-1325.

[3] 陈 鹏,郭剑毅,余正涛,等. 融合领域知识短语树核函数的中文领域实体关系抽取[J]. 南京大学学报:自然科学版,2015,51(1):181-186.

[4] 高俊平,张 晖,赵旭剑,等. 面向维基百科的领域知识演化关系抽取[J]. 计算机学报,2016,39(10):2088-2101.

[5] RINK B, HARABAGIU S. A generative model for unsupervised discovery of relations and argument classes from clinical texts[C]//Proceedings of the conference on empirical methods in natural language processing. Edinburgh, United Kingdom: Association for Computational Linguistics, 2011: 519-528.

[6] 陈 宇,郑德权,赵铁军. 基于 Deep Belief Nets 的中文名实体关系抽取[J]. 软件学报,2012,23(10):2572-2585.

[7] WANG Wei, BESANÇON R, FERRET O, et al. Semantic clustering of relations between named entities[C]//International conference on natural language processing. [s. l.]: Springer International Publishing, 2014: 358-370.

[8] YE Feiyue, SHI Hao, WU Shanpeng. Research on pattern representation method in semi-supervised semantic relation extraction based on bootstrapping[C]//Seventh international symposium on computational intelligence and design. Hangzhou, China: IEEE, 2014: 568-572.

[9] BATISTA D S, MARTINS B, SILVA J M. Semi-supervised bootstrapping of relationship extractors with distributional semantics[C]//Proceedings of the 2015 conference on empirical methods in natural language processing. Lisbon, Portugal: Association for Computational Linguistics, 2015: 499-504.

[10] ZHANG Chunyun, XU Weiran, MA Zhanyu, et al. Construction of semantic bootstrapping models for relation extraction[J]. Knowledge-Based Systems, 2015, 83: 128-137.

[11] 李明耀,杨 静. 基于依存分析的开放式中文实体关系抽取方法[J]. 计算机工程,2016,42(6):201-207.

[12] 郭喜跃,何婷婷,胡小华,等. 基于句法语义特征的中文实体关系抽取[J]. 中文信息学报,2014,28(6):183-189.

[13] 甘丽新,万常选,刘德喜,等. 基于句法语义特征的中文实体关系抽取[J]. 计算机研究与发展,2016,53(2):284-302.

[14] 李培峰,周国栋,朱巧明. 基于语义的中文事件触发词抽取联合模型[J]. 软件学报,2016,27(2):280-294.

[15] 刘绍毓,席耀一,李弼程,等. 无监督实体关系触发词词典自动构建[J]. 计算机应用与软件,2016,33(5):72-76.

[16] LU Guangming, XIA Yule, WANG Jiamei, et al. Research on text classification based on TextRank[C]//International conference on communications, information management and network security. [s. l.]: [s. n.], 2016.

[17] CARLSON A, BETTERIDGE J, WANG R C, et al. Coupled semi-supervised learning for information extraction[C]//Proceedings of the third ACM international conference on web search and data mining. New York, NY, USA: ACM, 2010: 101-110.

[18] MILAJEVS D, SADRZADEH M, ROELLEKE T. IR meets NLP: on the semantic similarity between subject-verb-object phrases[C]//Proceedings of the 2015 international conference on the theory of information retrieval. Northampton, Massachusetts, USA: ACM, 2015: 231-240.

(上接第 138 页)

策略[J]. 计算机工程,2015,41(11):114-119.

[7] 王 来,瞿健宏. 基于 HDFS 的分布式存储策略分析[J]. 智能计算机与应用,2016,6(1):5-8.

[8] 朱刘江. 基于 Hadoop 的海量城市交通流数据分布式存储与分析研究[D]. 扬州:扬州大学,2015.

[9] 洪旭升,林世平. 基于 MapFile 的 HDFS 小文件存储效率问题[J]. 计算机系统应用,2012,21(11):179-182.

[10] 刘晓霞. Hadoop 中大量小文件性能优化方法研究[J]. 计算机光盘软件与应用,2013,16(18):78-80.

[11] 漆 铖. 云环境下海量小文件存储技术的研究与应用[D]. 广州:广东工业大学,2015.

[12] 余 思,桂小林,黄汝维,等. 一种提高云存储中小文件存储效率的方案[J]. 西安交通大学学报,2011,45(6):59-63.

[13] TIAN Fengping, YANG Ke, CHEN Langnan. Realized volatility forecasting of agricultural commodity futures using the HAR model with time-varying sparsity[J]. International Journal of Forecasting, 2017, 33(1): 132-152.

[14] 张宇翔,赵建民,朱信忠,等. 基于 HDFS 的海量指纹数据云存储优化研究[J]. 浙江师范大学学报:自然科学版,2015,38(2):179-184.

[15] 张 海,马建红. 基于 HDFS 的小文件存储与读取优化策略[J]. 计算机系统应用,2014,23(5):167-171.

[16] HE Hui, DU Zhonghui, ZHANG Weizhe, et al. Optimization strategy of Hadoop small file storage for big data in healthcare[J]. Journal of Supercomputing, 2016, 72(10): 3696-3707.