**基于hadoop的大数据清洗框架设计与应用**

靳丹 张磊 王洪军 王宝会

（国网甘肃省电力公司信息通信公司 甘肃730050

北京航空航天大学软件学院 北京 100191）

摘 要：构建和运行数据仓库的关键步骤是ETL，而ETL中的最关键步骤是数据的清洗和转换。在当今数据爆炸式增长的背景下，数据清洗与转换的挑战主要来自于源数据的复杂性和数据量的庞大，针对数据庞大的问题，目前Hadoop体系的mapreduce框架已经成为海量数据处理领域的事实标准。本文主要分析在大数据环境下数据清洗工作中数据来源的复杂性问题，并针对这些问题提出了基于Hadoop的简单的可扩展的数据清洗框架，可以让本框架的使用者只需要用少量代码完成基于mapreduce框架的海量复杂数据的清洗工作，mapreduce框架的复杂性对开发者透明，并以某互联网公司使用此框架收集的用户行为日志数据作为示例，在示例中，此框架相比之前的解决方案极大地提高了海量数据清洗的准确性和效率。另外，本框架还可以应用于海量的非结构化数据的清洗。

关键词：数据清洗、hadoop、mapreduce、大数据

Design and Application of Hadoop Based Data Cleaning Framework

JIN Dan, ZHANG Lei, WANG Hongjun, WANG BaoHui

(Information and communication company in Gansu province of State Grid Corporation of China730050;Software College of Beihang University,BeiJing,100191)

**Abstract：Building and running the data warehouse ETL is the kep step, while in ETL data cleaning and conversion is the most critical step. Under the background of current data explosion, the challenges of data cleaning and conversion is mainly from the complexity of data and huge amount of data. For the problem of huge amount of data, mapreduce framework of Hadoop system has become the standard in processing massive data at present.**

**In this article, we analyzed the complexity of the data source in the data cleaning work in the big data environment and proposed a Hadoop-based scalable massive data cleaning framework which is easy to use and transparent to developers, it can do some complex massive data cleaning work with only a few codes. A user behavior log data from an internet company was used as an example for introducing this framework in addition, in this example,** **compared to the previous solution, this framework has greatly improved the accuracy and efficiency of mass data cleaning. this framework can also be widely used in cleaning the mass of unstructured data.**

1 引言

随着互联网和其他行业的快速发展，信息的急剧增长，海量的数据处理和分析变得越来越普遍，传统的数据清洗技术面临严峻的挑战。海量的数据让传统的数据清洗手段变得越来越费时，但是留给数据清洗的时间越来越少；为了适应环境的快速变化，用户对快速得到数据有更迫切的要求；复杂多变的社会环境，提高了数据的复杂性和多变性，需要具有更好的扩展性。应对这些挑战获得更好性能和扩展性的关键是并行化的运用。近年来，大数据处理平台Hadoop以及其中的MapReduce框架广泛应用于数据密集型领域，一个MapReduce框架程序由map和reduce函数组成，在多个实例下并行处理键值对，有很好的水平扩展性，很好地解决了海量计算速度慢的问题，Hadoop可以做海量数据清洗的很好的基础平台。MapReduce编程模式见图1。

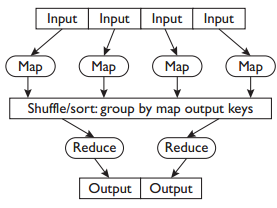


图1 MapReduce编程模式

MapReduce作为海量数据处理框架被广泛使用，特别是TB级和PB级数据处理方面有着天然的优势，越来越多的ETL清洗过程都在使用MapReduce框架编程模式，需要对MapReduce的运行机制、算法、性能优化、编程框架设计等方面进行研究定制，在连接算法方面，主要有日志处理的Join算法、Theta-Joins算法和Set-Similarity Join算法，使得频繁的连接操作变得简单；在编程模型改进方面， MapReduce Online模型可以用流水线的形式处理数据，提高了数据处理的效率和响应速度；然而在实际应用中，使用Hadoop平台需要有大量的底层开发，效率十分低下，实现一个通用性强的MapReduce程序并不是一件容易的事，由于各种数据源的复杂性和需求的不确定性，一般都意味着大量的重复开发，并没有关于这样减少开发的框架被设计和提出，本文将主要研究探讨这样的一个框架的具体设计和实现。

本文主要展示一个基于Hadoop的数据清洗框架，本框架的主要贡献有：

* 使用Hadoop提高数据清洗的性能，良好的架构组织和封装隐藏MapReduce程序开发的复杂性，对不熟悉Hadoop框架的开发人员帮助较大。
* 具有良好的扩展性，大大提高了开发效率。通过良好的架构组织，使用过滤器、工厂方法、策略、监听器等模式，以及良好的xml配置文件，做到很好的扩展性；将本框架应用于新的项目或者数据源，只需要实现几个类，并且在xml配置文件中做好配置即可使用。
* 提供配置文件，使用多路输出，将结果按一定标准分配到不同的HDFS目录中，方便建立hive分区表做后期的数据分析或者其他的相关计算。

示例说明：为了说明本文所要展示的数据清洗框架，例举一个互联网公司的真实数据，数据是非结构化的，经过数据清洗处理以后变成结构化数据，方便以后分析。如图2所示。



图2 示例数据处理前后对比图

示例中的源数据，以日志的形式记录了互联网用户的行为，包括：IP、cookie信息、来源网页、访问页面、客户端信息、访问时间等等信心。从图2中可以看到源数据初看起来杂乱无章，无法直接用于后续的分析。

2 设计概述

本框架主要由一个MapReduce任务组成，图3展示了数据清洗使用MapReduce的数据流，在数据清洗的数据流中，Map阶段是数据清洗的核心,数据清洗主要在本阶段完成，对本阶段程序的详细设计详见第三部分；合理设计partitioner数据分区程序，将数据比较均匀的分配到Reduce端；Reduce阶段主要负责解析分配目录的配置文件，通过MultipleOutputs这个hadoop自带的类,将清洗后的数据分配到合适的目录中。



图3在MapReduce上进行数据清洗

每个MapReduce工作会产生大量的Map/Reduce任务并行执行。每个task由很多步组成：从分布式文件系统(HDFS)中读取数据，执行Map函数，为Map结果做partition数据分区，不同的partition数据分区结果分配到不同的Reduce任务中，组合Map输出结果，执行Reduce函数，最后通过MapReduce的多路输出MultipleOutputs将结果写到多个目录中。

3 清洗程序详细描述

为了提高系统的可扩展性,对Map函数、Reduce函数以及Partition函数做了精心的设计，采用了xml配置,相应的清洗类动态获取、清洗规则设置等。

3.1 Map函数的设计

Map函数负责本框架最核心的部分，将数据从原始状态清洗出来，并格式化为结构化数据。Map的设计采用了过滤器、工厂方法、策略、监听器等模式，让系统有很好的扩展性。图4是Map函数的核心设计。

图4 Map函数的核心设计

图4所示，LogProcess类是负责Map阶段基础配置文件的准备和使用策略模式调用Handler类，进行实际的数据处理逻辑。Handler类是整个处理逻辑的实际控制类，这些都需要在Map阶段的初始化部分完成的。实际的处理是由Handler调用图4中下面部分所示的流程完成的，处理的过程通过FilterChain以类似管道的形式一步步解析和清洗，其中FilterChain是用来配置过滤清洗条件的，例如在FilterChain中配置添加TimeFilter后，可以将时间转换成相对应的格式。Map设计中，所有的类均采用了可配置的方式，可以根据实际数据清洗的需要进行随意的替换和修改。

日志处理流程如图5所示，原始日志经过JournalClean类进行基础处理产生一些基础字段，并以table的形式存储在Jounal中，组装成Journal日志类，将这个类在过滤器链中按照具体要求的逻辑进行解析产生最终需要的字段，并通过JournalWriter日志写入类读取输出字段配置文件，输出最终的数据。其实可以不用JournalClean类，只需要在进入处理逻辑前将日志原文存储到journal表中即可，然后将journalclean的相关逻辑用Filter过滤器的形式进行处理，这更符合逻辑，但是出于可重用的角度，Filter类大多是可以重用的，而这里的Journalclean类的逻辑一般都是根据实际的逻辑关系进行重写，不同的业务会有很大的不同，只需要根据需求将此处的设计修改到过滤器中即可。

本文采用的示例数据是互联网行为数据，包括用户的标识、ip、客户端信息、访问时间、url、关键词等多种信息以非结构化日志的形式存储，经过本框架的提取需要转变为结构化数据，将相应的字段提取出来。首先看一下JournalClean类的清洗。



图5 日志的初始清洗和提取

日志的初步处理解析出了用户id、时间戳、客户端、ip等信息。但是没有解析出年月日周等时间信息，根据定制的规则获得移动端和PC端、浏览器、浏览器版本、操作系统及其版本；地域相关信息-省、事、区、县等信息。

图6 过滤器链中过滤器的调用

对过滤器和过滤器链的设计，由于不同的种类日志处理方式不同，由Handler根据journalclean解析出的Jounal的类型不同，调用配置文件中不同的过滤器链。Journal根据其类型调用一个过滤器链。过滤器中含有多个过滤器，都是可配置的。TimeFilter类实现了将timestamp时间戳解析为年、月、日、小时、周等信息；针对很多定制的清洗规则，指定了一套规则解析库，含有相等Equal、前缀Index、后缀End、包含Include、正则Regrex等规则解析器，从配置文件读取规则，通过本套解析器解析出相应的字段解析规则，达到很好的可扩展性和易用性；其他的过滤器也都是根据某种解析规则或者查找字典解析相应的信息。以初步解析好的Journal类经过时间解析器TimeFilter和规则解析器JournalContentFilter解析为例，TimeFilter解析了时间的各种维度，JournalContentFilter解析了客户端各个维度的信息，该解析器按照规则解析并不仅限于客户端信息，解析规则的示例如图8所示。根据其中的Index,Regex等标识由各自的IndexRule，RegexRule等类解析，并清洗出相应的各个字段；采取这种方式，有很大的灵活性，只需要在配置文件中配置相应需要解析的规则，就可以将需要的信息解析出来。

 图7 日志通过Filter类清洗

 图8 规则解析器示例

经过过滤器链处理Journal类中已经含有我们关注的所有信息，对缺少关键信息的Journal也已经将状态设置为False，下面主要是Map端输出。需要根据Journal的类型从全局的配置文件中读取相应的格式化字段列表，组装结果输出。如本文示例中a类型的格式化输出配置文件为：

<property>

<name>a.log.format.text

</name>

<value>$keyword,$referer\_domain,$referer\_type,$referer\_word,area,$country,$provinces,$city,$area\_scope,$browser\_type,$browser\_version,$os,$os\_version,$mobile

</value>

</property>

将这些字段按照预先设定的分隔符隔开组装MapReduce的value字段进行输出。对输出Key的组装和设计，如果不需要进行多目录输出就不需要Partitioner和Reducer，可以用NullWritable,本例需要进行多路输出，为了在Reducer中更好的分组，本例根据当前数据的特点设计了实现writable接口的类。

3.2 Partitioner数据分区设计

Partitioner数据分区在MapReduce框架设计中十分重要，它主要分发Map端的输出数据作为Reducer端的输入数据，如果处理不好，通常会造成Reducer端的数据不均匀，导致有些Reduce任务运行慢，从而影响整体速度，在本框架中，Partitioner数据分区的设计是为了更均匀的分配数据，以字段本身或者设计的key键进行分区会造成数据的偏颇，因此默认设计了与key无关的完全随机分配的Partitioner数据分区，如图9所示，用以保证数据分配的均匀性。Partitioner数据分区作为MapReduce计算框架的默认组件，用户可根据自己的业务需要实现自己Partitoner类。

 图9 LogTypePartitioner源码图

3.3 Reducer设计

本Reducer的主要目的是将数据清洗的结果多路输出到相应的目录，如果只需要将结果分配的单一目录下可以完全不用Reducer更能提高性能。本文示例的日志有多种类型，不同的类型不只需要不同的过滤器链和输出字段，它们的输出目录也不同。Reducer需要读取x.log.write.category配置项指定的配置文件，采用Hadoop提供的MultipleOutputs将结果输出到不同的目录中去，为后续的数据分析提供默认的分区支持。

Reducer中根据Key对输入的数据进行分组，所以对Map的Key要进行精心的设计。本例的map输出的Key采取的是解析出的type和channel组合做Key。

4 结束语

数据清洗框架通过合理的架构，良好的配置文件安排，实现了扩展性较好的数据清洗框架，优缺点分析见表1.

表 1 校验方法优缺点比较

|  |  |  |
| --- | --- | --- |
| 序号 | 优点 | 缺点 |
| 1 | 采用hadoop做基础设施大大提高了处理速度，隐藏mapreduce开发的复杂性 | 仍然需要定制开发。 |
| 2 | 采用丰富的xml配置，具有良好的扩展性 | 需要对体系有一个基本的了解，才能针对本公司的数据特点修改 |
| 3 | 输出多目录，自然建好分区，方便后续使用hive外部表进行数据分析 | 仍然不是上层的，方便数据分析人员使用的框架，需要开发人员介入 |
| 4 | 实现了多种数据类型同时处理，对多种数据类型设置不同的处理规则和输出目录 | Xml文件配置规则需要很好理解才能做出。 |

本数据清洗框架已经开始在生产环境上使用，在100台普通PC服务器的Hadoop集群中，每20分钟运行一次本清洗程序进行数据清洗，3分钟即可完成对3000万条左右数据的处理，很好的满足了数据清洗在性能上的要求；而且Hadoop集群具有良好的水平扩展性，在处理性能达不到要求的情况下增加服务器规模即可。

本数据清洗框架使用后，在数据源变化不大的情况下，增加了几类日志，也增加过几类需求，都只需要修改一下配置文件就可以完成对新需求和新日志的清洗。

如果一个新公司使用本框架，可能需要开发一下相应的类；首先需要实现JournalClean抽象类，实现本公司数据源的基础清洗规则；复用过滤器中的某些类，比如TimeFilter/JournalContentFilter类，可能重新实现新的ValidFilter类等类，实现自己公司需要的一些Filter类，如图7，在配置文件中配置一下即可。

作者联系方式：

靳丹，（1975 - ），男，教授级高工，研究方向：电力信息化。

张磊，（1981 - ），男，工程师，研究方向：计算机及应用。

王洪军，（1985 - ），男，工程师/硕士，研究方向：软件工程。

王宝会，（1973 - ），男，教授级高工/硕士，研究方向：软件架构。

**参考文献:**

[1]Xiufeng Liu, Christian Thomsen, and Torben Bach Pedersen.ETLMR: A Highly Scalable Dimensional ETL Framework Based on MapReduce.Dept. of Computer Science, Aalborg University.2011

[2] Condie T, Conway N, Alvaro P, et al. Online aggregation and continuous query support in mapreduce[C]//Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. ACM, 2010: 1115-1118.

[3] Vernica R, Carey M J, Li C. Efficient parallel set-similarity joins using MapReduce[C]//Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. ACM, 2010: 495-506.