Computer Life (CPL) ISSN: 1819-4818
DELIVERING QUALITY SCIENCE TO THE WORLD

Design of Log Data Distributed Storage Analysis System Based on Hadoop Cluste

Mingkuan Li

Changchun University of Science and Technology, China

892138450@qq.com

Abstract: there are huge amounts of Web log data on Internet network. Massive Web log data contain a lot of available information. By analyzing the Web log, you can know the amount of visits to the website, the IP path of the visitors, and the maximum number of web pages visited. Medium sized websites (above 100 thousand of PV) generate log data of more than 1G Web per day. Large websites may generate 10G data per hour. The storage and analysis of mass data is a difficult and complicated task. Because of the bottleneck of computing, it can not keep up the pace of the fast analysis and calculation of the data. For mass data, using Hadoop distributed cluster for storage and analysis is an inevitable trend and the most appropriate.
Keywords: Log Data; Hadoop; Distributed.

---

1. INTRODUCTION

The log file is the user's behavior information data produced when using the APP application or browsing the web page, and the analysis of the log files can get a lot of valuable data. With the further popularization of the Internet, the Web2.0 era of "all people are interconnected" has come, and the number of users' access to the website has increased, and the user behavior log files have increased greatly. The single log file analysis method has not followed the needs of Shanghai log analysis. This paper, taking Web log as an example, uses Hadoop[1] to build a distributed cluster computing platform, storing and parallel analysis and calculation of mass data to improve the efficiency of analysis and processing of log data.

2. OVERVIEW OF HADOOP DISTRIBUTED CLUSTER

Log files record the original data of Internet users' Internet behavior, and the data are very valuable. As time goes on, the user log behavior data will increase continuously. Naturally, it makes a high demand for the speed of mass storage and processing of data. The purpose of this design is to solve the difficult problem of massive Web log data analysis and high storage cost. Based on the Hadoop distributed architecture built on the virtual machine, the data is distributed through the distributed HDFS file system, and then the Map-Reduce programming model of parallel processing large data is used to filter and analyze the log data files. The Hadoop ecosystem has made full use of the

advantages of Hadoop distributed storage and distributed computing by virtue of the two major components of HDFS and Map-Reduce. The data analysis system based on distributed structure improves the efficiency of analysis.

The target log data used in this design is derived from the user access records of a forum. Figure 2.1 shows the record format of the log data, in which there are 5 fields in each line: visitor IP, access time, access resources, access status (HTTP state code), and this visit use traffic.

```
20 8.35.201.163 - - [30/May/2013:17:38:21 +0800] "GET /uc_server/data/avatar/000/04/87/94_avatar_middle.jpg HTTP/1.1" 200 5117
21 8.35.201.165 - - [30/May/2013:17:38:21 +0800] "GET /uc_server/data/avatar/000/01/01/03_avatar_middle.jpg HTTP/1.1" 200 5844
22 8.35.201.160 - - [30/May/2013:17:38:21 +0800] "GET /uc_server/data/avatar/000/04/12/85_avatar_middle.jpg HTTP/1.1" 200 3174
23 8.35.201.164 - - [30/May/2013:17:38:21 +0800] "GET /uc_server/avatar.php?uid=53635&size=middle HTTP/1.1" 301 -
24 8.35.201.163 - - [30/May/2013:17:38:21 +0800] "GET /static/image/common/arw_r.gif HTTP/1.1" 200 65
25 8.35.201.166 - - [30/May/2013:17:38:21 +0800] "GET /static/image/common/px.png HTTP/1.1" 200 210
26 8.35.201.144 - - [30/May/2013:17:38:21 +0800] "GET /static/image/common/pmto.gif HTTP/1.1" 200 152
27 8.35.201.161 - - [30/May/2013:17:38:21 +0800] "GET /static/image/common/search.png HTTP/1.1" 200 3047
```

The record format of log data in Figure 2.1

3. DESIGN AND IMPLEMENTATION OF THE SYSTEM

Log files are analyzed and filtered by distributed architecture, and log data are analyzed according to time. The analysis starts with page PV (page view), IP, request status, traffic and so on. First, the HDFS file system is used to store the log data in a distributed way, then Map-Reduce is used to clean the logs in parallel.

3.1 Hadoop cluster deployment

This design uses Centos Linux virtual machine to simulate the deployment of Hadoop cluster composed of 3 servers. Figure 3.1 shows the basic structure of Hadoop deployment, the main control job node of the Master in the MapReduce model is called JobTracker, and all of the jobs (Job) under this framework are managed by JobTracker, which is the only existence. TaskTracker, responsible for the execution of every specific task. Task is the basic unit of specific execution. Each job is split into a number of tasks, assigned to the appropriate task node, and the task node performs the assigned task while reporting the state of the task to the JobTracker, in order to help JobTracker understand the overall situation of the job execution. The operation of assigning new tasks to idle nodes.

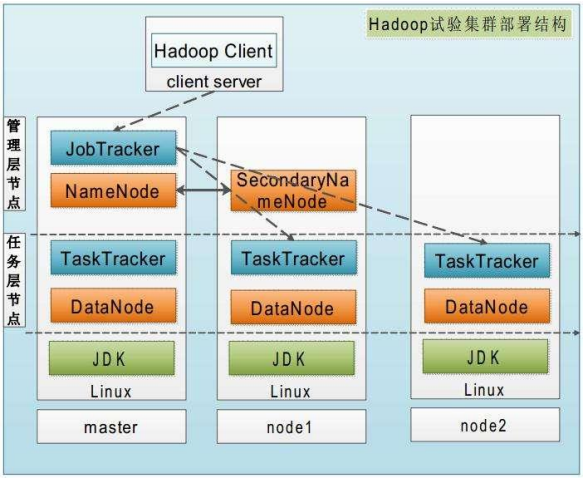

Figure 3.1 Hadoop cluster deployment structure diagram

3.2 HDFS storage of log data

As shown in Figure 3.2, when writing data to HDFS[2], the client client needs to do two things. First, the data files are partitioned into different data blocks. The size of the block is 64MB or 128MB by default, and the block size is configurable. Next is to request a batch of datanode to store data blocks to namenode. Of course, client does not choose three datanode randomly. It selects three three datanode closest to client. What client wants to do next is to transmit data to the 3 datanode returned by namenode. When it writes the first block, it writes data to the nearest datanode. So how does client know whether it has succeeded in writing data to the first datanode? In this way, when client passes data to datanode, the accounting calculates the data fast check sum, and this check sum will also pass to datanode. After storing the data, datanode will check the data block and compare with client check sum. If the data is consistent, the data is saved successfully, and then it will make a to client. CK tells client that the data has been saved successfully, and it will also tell namenode that the data block is saved successfully. The remaining data blocks to the next two datanode processes are similar to the first process, but the data transmission is not all passed by the client, but is passed between the datanode, and each datanode succeeds in storing the data to the client and notifies the namenode to complete the data. When client accepts all the ack of datanode, client will tell the namenode data block to be all written. When namenode accepts messages to client, what namenode wants to do is maintain two tables, one table for each data block corresponding to the datanode address, and the other to store the data copies of those pipelines.

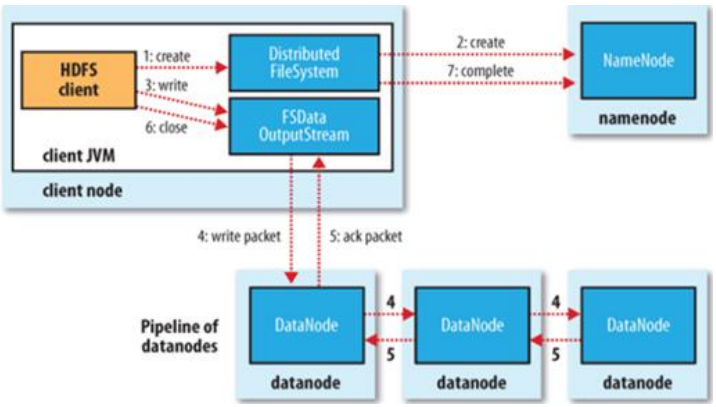

Figure 3.2 Write operation of HDFS

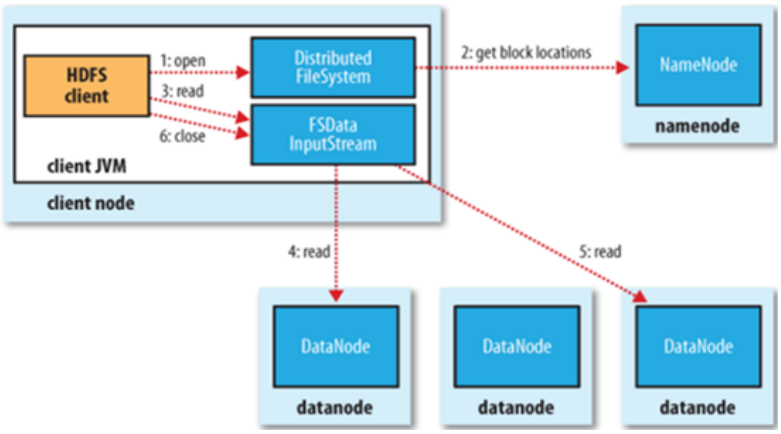

Figure 3.3 The reading operation of HDFS

As shown in Figure 3.3, when client is to read data from HDFS, client will ask namenode for the file address of the data, and namenode will return to the client about the data block, which is the datanode on which the data blocks are stored, and which data blocks are stored for each datanode, and of course these datanode are also based on the client distance. Out of order. When client gets the information, it will download the data first to the nearest datanode.

3.3 Hadoop parallel algorithm model

In Figure 3.4, the Application business system is on the left and the Hadoop HDFS[3], Mapreduce on the right. The log is generated by the business system. It can set up the system timer CRON, and import the log files to HDFS at 0 after night. After importing, the system timer is set up, the MapReduce program is activated, and the statistical indicators are extracted and calculated. After completing the calculation, set up the system timer and export the statistical index data from HDFS to the database.

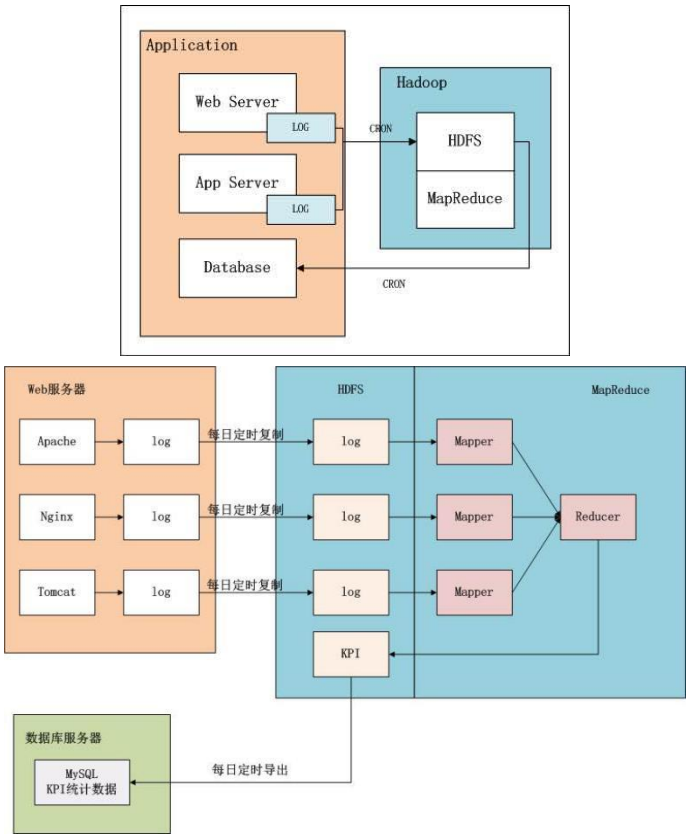

Figure3.5 Log data flow chart

From figure 3.5, we can see clearly how the log data flow in Hadoop cluster processing. The blue background is in Hadoop, and the next task is to complete the Map-Reduce program implementation.

3.4 Log KPI analysis of Map-Reduce

Map-Reduce[4] is a distributed computing model proposed by Google, which is mainly used in the search domain to solve the problem of massive data computation. MR consists of two phases: Map and Reduce, which only need to implement Map () and reduce () two functions to achieve distributed computing.

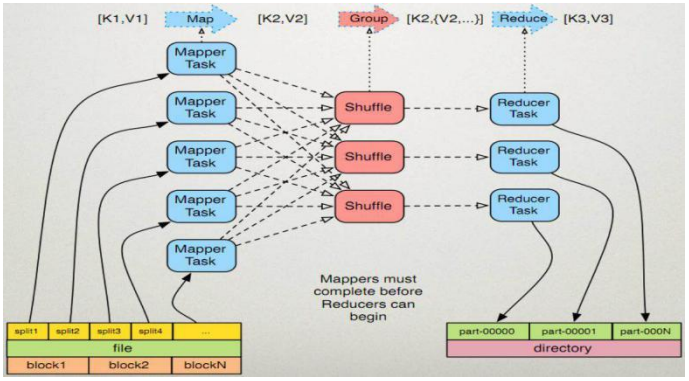

Figure3.6 Programming model diagram

The execution steps of the Map-Reduce are shown in Figure 3.6.1, Map task processing: read the files in the HDFS. Each line is parsed into a <k1, v1>. Each key value is called a map function. Cover map (), receive the <k2, v2> generated from the previous step, process it, convert it into new <k2, v2> output. Using partitioner to partition <k2 and v2> (default is divided into one area), sorting data in different partitions (according to k) and grouping. Grouping refers to the value of the same key placed in a collection.

2, Reduce task processing: the output of multiple map tasks, according to the different partitions, through the network copy to the different ruduce nodes, this process is completed by the shuffle mechanism. After the Reduce function is processed, a new <k3 and v3> output are generated. Finally, the reduce output <k3 and v3> are written to the distributed storage in HDFS, and the files can also be exported to the database, and finally displayed according to the statistical data.

4. RESULT ANALYSIS

Figure 4.1 shows the deployment of the Hadoop cluster.

Datanode Information

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
localhost (192.168.137.53:50010)	1	In Service	15.65 GB	696 KB	2.18 GB	13.46 GB	17	696 KB (0%)	0	2.6.4
lmk2 (192.168.137.52:50010)	0	In Service	15.65 GB	696 KB	2.22 GB	13.43 GB	17	696 KB (0%)	0	2.6.4

Figure 4.1 Hadoop deployment profile

Browse Directory

/log Go!

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	lmk	supergroup	58.25 MB	2	128 MB	KPI.log

Figure 4.2 HDFS log data storage diagram

There are 3 virtual servers in this cluster, one of which is master (namenode), the management of metadata for log distribution; the other two is slaver (datanode), which is responsible for the physical storage of log data. The most common way to upload log data to HDFS is to upload data to HDFS

directly under linux using shell command. Figure 4.2 is the result graph after the log data upload is successful.

Next is to write the Map-Reduce program cleaning log: (1) write the log parsing class to separate five components of each row; (2) write the Map-Reduce program to filter all records of the specified log file; (3) export the execution program jar package and upload it to the Linux server specified directory. . The log data in HDFS is viewed through the Web interface: the unfiltered log data stored: 192.168.137.51:50070/explorer.html#/log. See Figure 4.3.

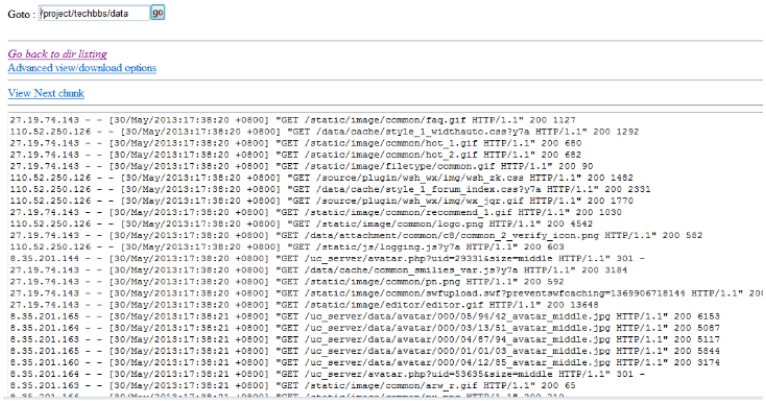

Figure 4.3 Figure 4.3 Log data before unprocessed

The filtered log data that is stored is shown in 4.4.

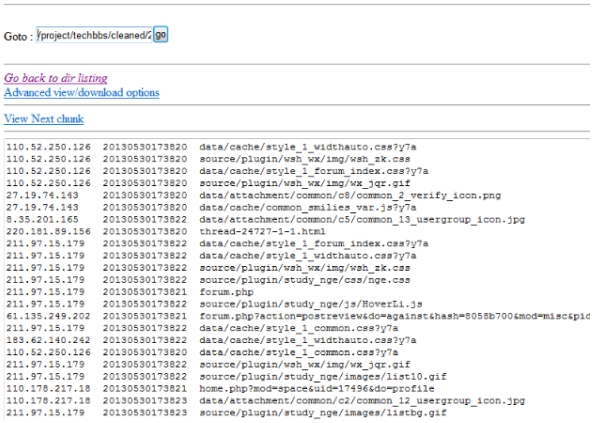

Figure 4.4 Figure 4.4 Log data after statistics

5. SUMMARY

As an increasingly mature technology, distributed computing has gradually revealed its advantages. Hadoop distributed cluster framework is a good integration of distributed file system HDFS and Map-Reduce programming model. The use of Hadoop can easily build a distributed cluster environment in a poorly configured server. It has good performance and scalability for distributed task control and scheduling. Based on this example, it can be further expanded to further and detailed analysis and mining of large data tasks, so as to provide reference for large data processing in various fields.

REFRENCES

[1] W.Jiang,V.Ravi，and G.Agrawqal.A Map-Reduce system with an alternate API for multi-core environments.CCGRID,2010.

[2] A.Rowstron,D.Narayanan,A.Donnelly,G.O'Shea,and A.Douglas.Nobady ever got fired for using

Hadoop on a cluster.HotCloud,2012.
[3] White T.Hadoop:The Definitive Guide[M].O'Reilly Media,2009.
[4] Dean,Jeffrey,and Sanjay Ghemawat."MapReduce:simplified data processing on large clusters."[J]Communications of the ACM 51.1(2008):107-113.