International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 06 | June -2017 www.irjet.net p-ISSN: 2395-0072

A SURVEY ON DIFFERENT FILE HANDLING MECHANISMS IN HDFS

Revathi S[1], Saniya Kauser[2], Sushmitha S[3], Vinodini G[4]

[1]Assistant Professor, Dept of CSE, Dr.TTIT, Karnataka, India
[2]UG Student, Dept of CSE, Dr.TTIT, Karnataka, India
[3] UG Student, Dept of CSE, Dr.TTIT, Karnataka, India
[4]UG Student, Dept of CSE, Dr.TTIT, Karnataka, India

Abstract - *Hadoop is a software framework for distributed processing of large datasets across large clusters of computers. Hadoop framework consists of two main layers. They are Hadoop Distributed file system (HDFS) and Execution engine (Map Reduce). HDFS has the property of handling large size files (in MB's, GB's or TB's), but the performance of HDFS degrades when handling small size files. The huge numbers of small files impose heavy burden on NameNode of HDFS, correlations between small files were not considered for data placement. There are three common mechanisms to handle the small files in HDFS like Hadoop Archive (HAR), Sequence File and TLB MapFile. In order to improve the access efficiency and to quickly locate a small files, a common strategy is to merge small files into large ones. This paper discusses the different small file handling mechanism like Hadoop Archive (HAR), Sequence Files, TLB-MapFile and compares them.*

Keywords: Hadoop, HDFS, HAR, Sequence File , TLB-MapFile

1. INTRODUCTION

Hadoop is an open-source software framework[1], which offers cost-efficient solution to store, manage and analyze a large amount of data, it provides distributed processing and storage of huge data across thousands of computers[2]. Google initiated the idea of hadoop to store and process a large information through web and now it is adapted by other web giants like, Facebook, Twitter, Linkedin, Yahoo, etc, The Hadoop comes with two layers called MapReduce framework and Hadoop Distributed File System (HDFS).

1.1 MapReduce framework: MapReduce is a core component of the Apache Hadoop software framework[3]. It is parallel programming model for processing and generating large data sets. generally it is the execution unit of hadoop framework[4]. It uses map function to process a key/value pair in order to generate a set of key/value pairs, and a reduce function that combines all intermediate values associated with the same intermediate key. It is based on two functions called map and reduce. MapReduce provides good fault-tolerant, with each node periodically reports its status to a master node.

1.2 HDFS: is a distributed file system designed to store and process large datasets [3]. HDFS is scalable and fault-tolerant, which is organized on low-cost hardware. HDFS provides efficient access to application data and is suitable for applications that have large data sets. HDFS provides a stable storage layer for the distributed application. HDFS has a master/slave architecture and consisting of three main components [4]: NameNode, DataNodes and Clients, as shown in Figure 1.

1. Name node: a master server that manages the file system namespace and regulates access to files by clients.
2. Date Nodes: a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on.

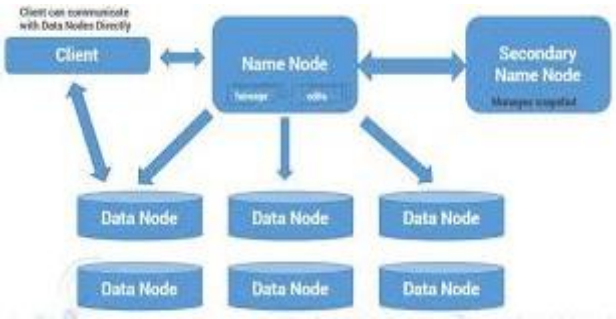

Figure 1: Components of HDFS

However storing and accessing a large number of small files impose a big challenge to HDFS because of two main issues:

1. The Namenode Memory is highly consumed by large numbers of files
2. Doesn't considers file correlations for data placement.

Based on the analysis of small file problem, an efficient approach is designed for HDFS to reduce the memory consumption of NameNode, and to improve the storage and access efficiency of small files.

II. Small File Handling mechanisms:

The three common mechanisms are HAR, sequence files and TLB-MapReduce [5].

a. HADOOP ARCHIVE FILES:

Hadoop archive or HAR is an archiving facility that pack files in to HDFS blocks efficiently. Thus HAR are used to tackle the small files handling problem in hadoop. Usually the HAR files are created through collection of files and the archiving tool runs a mapreduce in order to process the input files in parallel and create an archive files.

Advantages:

1. Hadoop Archives (HAR) can be used to address the namespace limitations associated with storing many small files.
2. HAR packs a number of small files into large files so that the original files can be accessed transparently.
3. HAR increases the scalability of the system by reducing the namespace usage and decreasing the operation load in the NameNode.
4. This improvement is orthogonal to memory optimization in the NameNode and distributing namespace management across multiple NameNodes.
5. Hadoop Archive is also compatible with MapReduce,it allows parallel access to the original files by MapReduce jobs.

Limitations of HAR files:

1. Once an archive file is created, you cannot update the file to add or remove files. Hence the files are immutable.
2. Archive file will have a copy of all the original files so once a .har is created it will take as much space as the original files. .har files are not the compressed files.
3. When a .har file is given as an input to Map Reduce job, the small files inside the .har file will be processed individually by separate mappers which is inefficient.

b. SEQUENCE FILES:

Sequence files is a flat file consisting of binary key/value pairs. It is extensively used in Map Reduce as input/output formats and also the temporary results of maps are stored using Sequence File. The Sequence File provides a Writer, Reader and Sorter classes for writing, reading and sorting respectively.

Advantages

1. As binary files, these are more compact than text files
2. Provides optional support for compression at different levels- record, block.
3. Files can split and processed in parallel.
4. As HDFS and MapRecude are optimized for large files, Sequence Files can be used as containers for large number of small files thus solving Hadoop's drawback of processing used number of small files.
5. Extensively used in MapReduce jobs as input and output formats. Internally, the temporary outputs of maps are also stored using sequence file format.

Limitations

1. Similar to other Hadoop files, Sequence Files are append only.
2. As these are specific to Hadoop, as of now, there is only Java API available to interact with sequence files. It does not supports multi language.

c. TLB- MapFile

TLB-MapFile consists of three parts

- Small files merge module
- The audit log mining module and
- Small files prefetching module

1. Small files merge module

By accessing HDFS audit logs, TLB-MapFile obtains the access frequency of small files. Then, small files are merged into large files according to the order of the level of access frequency. Finally, the file block is kept into HDFS.

2. Audit log mining module

It analysis HDFS audit log, obtains the strength of the association of any two small files, and gets the access frequency of small files within the specified time. Also, the module creates a TLB table and a high frequency access table based on the correlation strength and the access frequency.

3. Small file prefetching module

The module gets the mapping information between block and small files of being read in TLB table. If the module retrieves the mapping relationship, the location of small files is directly located and the content of small file is read.

Meanwhile, the module will pre read small files to the cache according to the association strength of small files, and the number of small files of pre-reading can be controlled by the ratio between the user's waiting time threshold and the reading time of small files.

Advantages of TLB-MapFile

1. The TLB-MapFile improves the Name Node memory efficiency.
2. It also improves the file download scheme in case of mass number of files.
3. File reading speed is increased.

III. Comparison between three small file handling mechanisms

Functions	HAR	Sequence File	TLB-MapFile
Working	Provides archiving facility to pack the file into HDFS blocks	It is a flat binary file consisting of key/value pair	Merges a massive small files into large files based on high frequency access log files.
Purpose	To tackle the small file handing problems in HDFS	To tackle the small file handing problems in HDFS	To tackle the small file handing problems in HDFS
MapReduce	Used to efficiently create a hadoop archives	Uses mapreduces as input/output format	Uses mapfile to store, merge the mapping information
NameNode memory consumption	Reduces the namenode memory usage	Also, reduces namenode memory usage	Efficiently reduces the namenode consumption than HAR
Reading efficiency of small files	Very low as it has to read two index file and data file	Reading efficiency is still slow than HAR	Reading speed is high as it enhances analyzing speed
File updating	Files are immutable. Once created cannot be updated	Binary key/value pair will be updated	The files are updated regularly
Support for compression	Does not support for compression	Optional support	Supports compression
Retrieval efficiency	higher	Based on the binary key processed	Higher than HAR
Download Speed	Less compare to TLB	Still less	Improves download speed
Capability to split	Splittable hence used in MapReduce	Files can split and processed in parallel	It's a fast table structure storing a log files, which cannot be splitted

Table 1: Comparison File Handling Mechanism

IV. Conclusion

Hadoop is a framework to handle the big data, hadoop software always process the big files efficiently but it results in inefficient handling of small file, to overcome the problem of small file accessing in HDFS layer of hadoop, there are three mechanism called HAR, sequence files and TLB-Mapfile. The paper outlines the advantages and disadvantages of all three files, and also paper compares the three small file handling mechanism with respect to its working and efficiency in HDFS.

References:

[1] Apache org.Hadoop Distributed File System. http://hadoop.apache.org

[2] Apache Hadoop in Heterogeneous Distributed File System. http://www.cloudera.com/hadoop/

[3] K. Kiruthika1, E. Gothai, An Efficient Approach for Storing and Accessing Small Files in HDFS, Karpagam Journal Of Engineering Research (KJER) Volume No.: II, Special Issue on IEEE Sponsored International Conference on Intelligent Systems and Control (ISCO'15).

[4] Deeksha S, R Kanagavalli, Dr. Kavitha K S, Dr. Kavitha C, Efficient Resolution for the NameNode Memory Issue for the Access of Small Files in HDFS, International Research Journal of Engineering and Technology (IRJET) ,Volume: 04 Issue: 04 Apr -2017.

[5] Bing Meng, Wei-bin Guo, Gui-sheng Fan, A Novel Approach For Efficient Accessing Of Small Files In HDFS: TLB-Mapfile, International Conference on International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), IEEE/ACIS, 21 July 2016