Small Files Consolidation Technique in Hadoop Cluster

Amarendra Mohanty [1], P. Ranjana [2] and D. Venkata Subramanian [3]

[1,2],*School of Computing Sciences,* Hindustan Institute of Technology & Science, Chennai, India.
[3] *School of Computing Sciences,* Velammal Institute of Technology, Chennai, India.

***Abstract* - Hadoop distributed file system (HDFS) usually has large number of small files which causes negative impact on Hadoop performance. The performance tuning in Hadoop jobs are difficult because of the lack of single performance tuning techniques. The performance of tuning is also based on the amount of data that is transferred during the production of hadoop jobs. To overcome this and to improve the performance, a tuning is performed on hadoop. This is achieved with a Small File Consolidation(SFC).The objective of the proposed Small File Consolidation (SFC) is to overcome some of the current challenges with respect to performance of Hadoop Cluster. SFC will consolidate many number of smaller files currently being generated in Hadoop impacting the performance. It will merge all the smaller files into a single file of preset size. The size can be dynamic based on the environment. Thus the proposed SFC will improve the query execution time by generating the result set quickly which will result the effective management of cluster usage.**

Keywords - Hadoop, HDFS, SFC, Cluster, Performance.

I. INTRODUCTION

Due to the rapid development of wireless communication, people are accessing the internet form anywhere more quickly and conveniently for various services. Due to this the accessing nodes are also increased. The data is streamed and analyzed to get the useful information. The data processed through this cloud computing environment is big data which cannot be handled by traditional systems. Big data consists of data of different types like structured, unstructured and semi structured. The big data is used to meet the challenges faced in the business [1]. A frame work to handle the big data is developed know as Hadoop. A Hadoop cluster n-level architecture which consists of a rack mounted server system

A Hadoop cluster has an n-level architecture which is comprised of rack-mounted server systems. Many researches attempted to modify the version of the hadoop using a map reduce framework to improve the efficiency in terms of caching and task scheduling [2]. Currently in Hadoop HDFS, data of a table is stored in a number of files. A table can contain multiple files in which the table's data is stored. The size of the files is not constant or preset. This causes a large number of small files to be present as part of a table. As detailed above, due to presence of large number of small files, the query executed against a table needs to traverse through these files to generate the result set. This results in high query execution time when the number of files are more, hence increased cluster usage and turnaround time. HDFS file systems are used for storing larger files. There are situation where there is a need to sore number of smaller file. If all those smaller files are managed by a single server the HDFS faces some problems .Some of the problems are given below

1. Since every block is capable of handling a single file. The smaller files will be less than the block size. To store these file more blocks are needed and there blocks are read one after the other, due to this to store the small files it takes more time.
2. The name node stores the record of each file and block in the memory. If more blocks and more files are used then the memory space need is also more and it reduces the memory complexity.

To overcome this many methods have been proposed in HDFA. In this paper we give a comparative analysis of these methods and propose and methods to deal with the small file problem in HDFS.

A. Problem Statement

The default size of HDFS block is 64 MB. If any files is less than that size it is considered as small file. If this small files are stored in a HDFS there is a bottle neck situation in terms of disk utilization. Due to many smaller files there will be a serious problem in storing the metadata in the name node in terms of memory usage. This in turn increases the main memory usage. This also creates overhead in CPU time. A table can contain multiple files in which the table's data is stored. The size of the files is not constant or preset. This causes a small files of large number to be present in the table as one part. As detailed above, due to presence of large number of small files, the query executed against a table needs to traverse through these files to generate the result set. This results in high query execution time when the number of files are more, hence increased cluster usage and turnaround time.

II. LITERATURE REVIEW AND RELATED WORK

A. Techniques for Managing Small Files in Hadoop

Hadoop is for analyzing the data in cloud computing environment for serving the parallel distributed computing. The main advantage of the hadoop is due to the reasonable structure which is based on the network topology. It is also an open source platform and can also be used in data in distributed and parallel computing. The HDFS architecture has two layer the Map reduce and the programming model. The main hadoop commodity and extremely used for large file streaming. The architecture of hadoop has two nodes known as the name node which is the master node and several other data nodes known as the slave nodes. Since there is only one master node and several slave nodes this becomes a bottleneck in using the hadoop[3]. Hadoop Archive (HAR) is to pack many smaller files into a large HDFS block. Due to this the original file can be accessed transparently in parallel without expanding the files of the metadata and data files. [4].A sequential file consists of a binary key vale pairs and it is a flat file. The file name is used as key and the file content is used as the value. Small files can be made into a single sequence file using program and it can be process using the Map reduce operations [5].To handle the small files in cloud computing environment many scheduling algorithms were proposed one among that was a genetic algorithm for computing the larger number of tasks[6]Hadoop uses a map reduce techniques . Map reduce is used to process large dataset. It consists of two phases the Map Phase and Reduce phase. Every job has a map function and a reduce function. As the map reduce does not have a built-in-support for iterative programs. To improve the efficiency of the iterative programs a technique call Ha Loop is introduced which is the modified version on hadoop map reduce. This is used in parallel and distributed system with for large scale data analysis applications. [7]. There is a need for data replication in a distributed environment several methods are proposed for efficient replication and storage strategy in dynamic environment. [8, 9]. Hadoop Map reduce is a framework used for creating the application, It enables to execute large quantity of information simultaneously. This is applicable for a large group of commodity hardware with high performance. The map reduce job consists of the mapper and reducer function. Many techniques were used to manage smaller files in Hadoop some of them are

- Hadoop Archive (HAR): It merges the smaller files into HDFS blocks for performance improvement. The disk space for this HAR system is equal to the original file. The reading of files in this systems is easy due to two indexing, but the compared to other files systems it is less efficient

- Sequence files is normally used based on the key sequence. Since it takes time the map reduce takes the sequence file splits the files into pieces and process it individually. Though the access time is less the converting the file to sequence file takes time

The problem in small file storage are creating an indices [10]. The small files are formed as clusters . The smalles files are merged and stored in HDFS separately in various blocks [11[.In this a cut of point need to be deceied while merging the file. So there is a need for enhancement method in this merging cutoff for efficent merging. [12]. Proposed an efficient merging technology but it was difficult to find the cutoff point. Thus finding the cutoff point is very important than merging the file.

B. Related Work

From the different studies mentioned in the reference, below are the following techniques used for the consolidations of smaller file and to reduce their sizes.

C. Hadoop Archive (HAR)

Hadoop archive is a technique used in packing the smaller files in many numbers in HDFS block in an efficient manner.

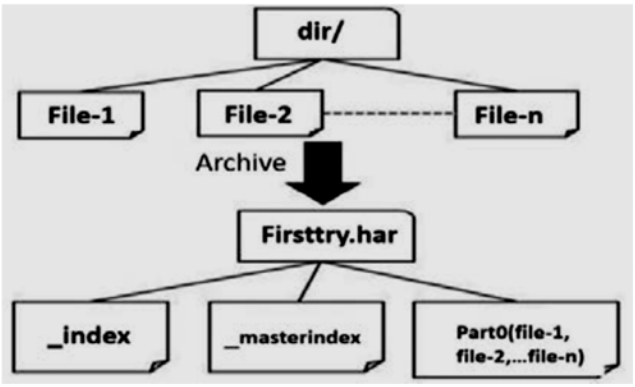

Figure 1: Hadoop Archiving smaller file

The advantages of HAR is to reduce the storage overhead of the data on the name node and reduce the operations in map reduce program.

There were several drawbacks of the HDFS systems they are

1. Reading HAR is less efficient and slower than reading the files using HDFS.
2. It is difficult to operate the MaP process on HAR
3. To upgrade the HAR there should be changes done in HDFS system. Upgrading HDFS architecture is a tedious work

D. New HAR

To overcome the drawbacks of HAR the new HAR is proposed.

The NHAR has two characteristics:

1. New HAR is proposed the basic idea of this is merging small files to a large files. With that the file number is reduced and the performance to access the file is optimized.
2. The file management with HAR is same as other systems where the functionality can be extended.

In NHAR the indexing structure is improved to store the metadata information in HDFS. It access the performance of HDFS without changing its architecture. The NHAR also adds additional file to the exiting archive which is the limitation of HAR.

To read a file from HAR, there is an overhead in reading the indexes. So the performance is improved in NHAR using a single level index. This creates a table instead of the master index technique. The information is split over multiple files using NHAR is shown in Figure 2.

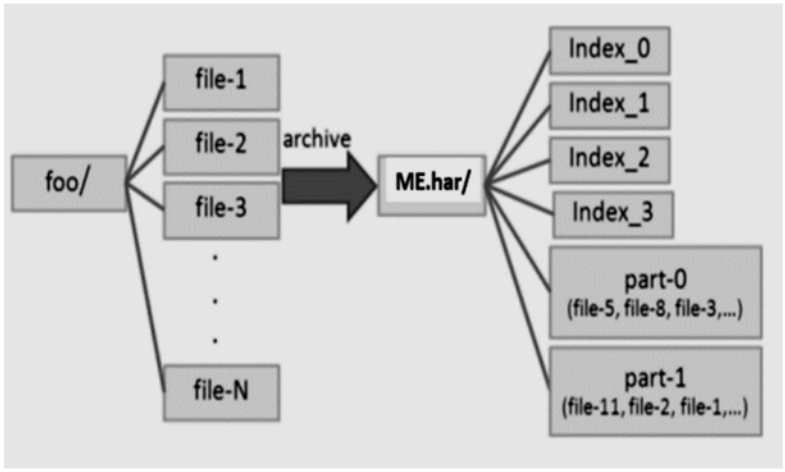

Figure 2: New Hadoop Archiving Files

The index file containing the meta data also depends on the number of index files. The file names are used with their mods in addition to index in index files. The actual file is sorted in part file. The hashing mechanism is same NHAR. Figure 3 represents the hashing mechanisms of NHAR.

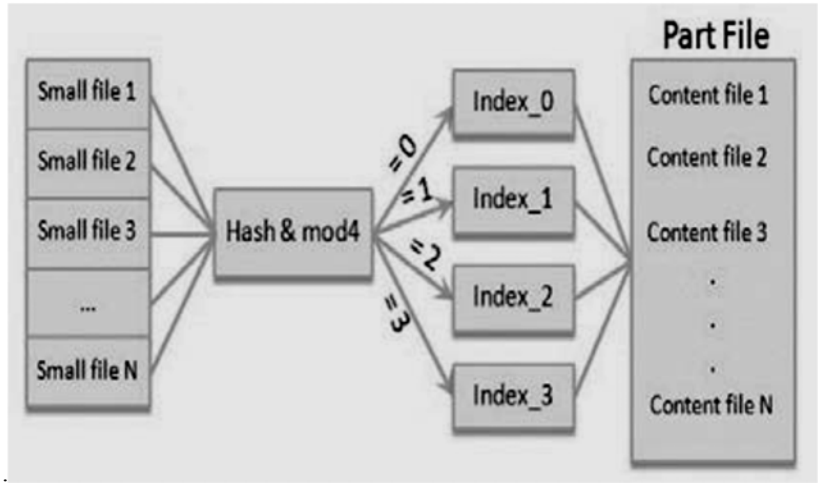

Figure 3: Hashing Mechanisms in NHAR

Though there are many advantages in NHAR. It is inefficient in terms of reconstructing new NHAR files. There are some additional process involved in inserting the file like achieving new files, merging and moving the new part of the file

E. CombineFileInputFormat

Due to the use of multiple mappers and single reduce. The input format is used for splitting the files and the map reduce frame work. The combinefileinputformat system

improves the performance by modifying the input format by splitting. The map will get more input to process than the other systems. The time required to process the large files of smaller size is reduced. It also achieves parallelism using the map reducers.

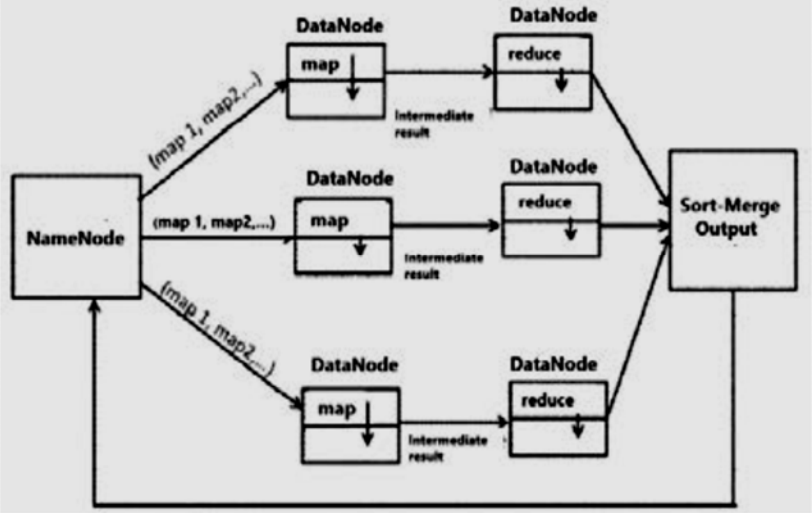

Figure 4. Combined Input output format.

III. PROPOSED NEW APPROACH FOR SMALL FILE CONSOLIDATION

The small files merging is performed in HDFS by setting the file average as n MB, where n = 256 or 512 or 1024 based on the required scalability.

The command used is:

Set hive.merge.smallfiles.avgsize=n000000;

where n= 256 or 512 or 1024.

After this, execute insert overwrite statement against all tables in the DB to merge the files. The command used is INSERT OVERWRITE <<table name>> select * from <<table name>>;

This exercise reduces small file count by over 85% there by significantly improving cluster performance.

In this approach, the small files are consolidated which will reduce the turnaround time (TRT) for the queries executed against the table. Consider a scenario where there were 27K files in the DB (Cumulative for all the tables) and upon consolidation using this technique will reduce the files to 1.5K. So any query run against the DB will need to traverse through only a fraction of file count when compared to unconsolidated state resulting in optimizing cluster performance increasing overall health of the system

The small consolidation reduces the time taken by each Hive job to complete. On a holistic note, this reduces the cluster usage.

VI. EXPERIMENTAL RESULTS AND DISCUSSION

The SCF approach was implemented in the production databases. These databases had varied number of tables. This approach significantly reduces the small files in the databases. Detailed below is the performance improvement of the small file consolidation using SFC approach Table 1 represents the tables in the database and the files before and after consolidation.

TABLE I: SCF APPROACH IN PRODUCTION DATA BASE

Number of tables in the DB	Number of files before consolidation	Number of files after consolidation
1670	27100	1500
396	1500	80
2780	34200	1830
590	1690	91
1290	25500	1364
1754	29500	1579
2945	36000	1927
453	1450	78
983	1900	102
1634	2700	145

With this consolidation, we will now have a look at the cluster usage optimization benefits due to implementation of SCF. The figure shows the implementation of SCF and the cluster usage is improved as represented.

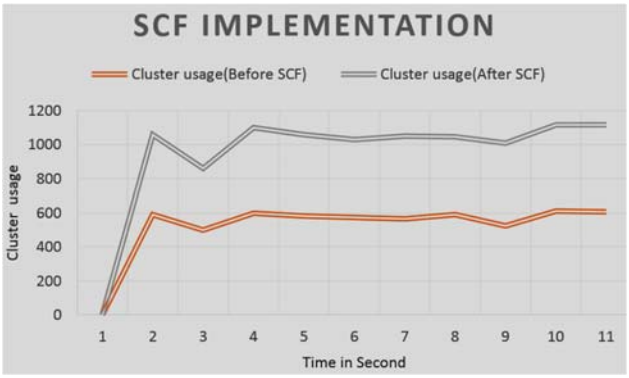

Figure 5: SCF Implementation

V. CONCLUSION AND FUTURE WORK

This paper presents a solution to handle small files problem in Hadoop based on small file consolidation techniques. This technique is focused on reducing the no. of files than recusing the size of the files. It is observed that the processing performance of small files has been improved up to 85 % by the proposed approach. The experimental evaluation has demonstrated that the proposed technique has reduced the cluster usage and has improved the efficiency of accessing small files in HDFS. From the above experiments, 161540 files were reduced to 8696 achieving an efficiency of 95%. As for future work, automatic monitoring tool to be developed to set the small file size based on the environment and number of files. The cut-off point between large and small files will be further studied.

REFERENCES

[1] Bernice Prucell, (2013) "The emergence of big data technology and analytics", Journal of Technology Research.

[2] Yingyi Bu , Bill Howe, Magdalene Balazinska, Michael D. Ernst(2010), " The Haloop Approach to large scale iterative Data Analysis"VLDB paper 2010 .

[3] Shafer.J, Rixner. S and Cox. A(2010), " The hadoop distributed file system: Balancing portability and performance , proceedings of the IEEE international symposium on performance Analysis of system and software USA March 2010, PP 122-133.

[4] C.Shen, W. Lu, J. WU and B. Wei, (2010), "A digital library architecture supporting massive small files and replica maintenance. ", Proceedings of the 1-the annual joint conference on digital libraries, ACM Press pp 391-394.

[5] T. White (2010), Hadoop the definitive guide, Yahoo press

[6] J. Huang (2014), " The workflow task scheduling algorithm based on the GA model in the cloud computing environment:, Journal of software vol.9 pp. 873-880.

[7] Yingyibu, Bill howem Magdalene Balszinska (2010) "Efficient iterative data processing on large clusters VLDB paper.

[8] T. SudalaiMuthu and K. RameshKumar, "A Value based dynamic replica replacement strategy in datagrid," International Journal of Control Theory and Applications, vol. 10, no. 26, pp. 448-462, 2017.

[9] T. S. Muthu, R. Vadivel, A. Ramesh and G. Vasanth, "A novel protocol for secure data storage in Data Grid environment," Trendz in Information Sciences & Computing (TISC2010), Chennai, 2010, pp. 125-130.

[10] B. Gupta, R. N. (2016). An efficient Approach for storing and Accessing Small files with Big data technology. International Journal of computer Applications, 36-39.

[11] Nupriroj, C. C. (2014). Improving performance of small file accessing in Hadoop. 11th international joint conference on computer science and international journal of pure and applied mathematics.

[12] Z. Gao, Y. Q. (2016). An Effective merge strategy based hierarchy for improving small file problem in HDFS. International conference on cloud computing and Intelligence systems cces (pp. 327-331). IEEE.

.