International Journal of Combined Research & Development (IJCRD)
eISSN:2321-225X;pISSN:2321-2241 Volume: 6; Issue: 5; May -2017

A File Sharing System with Performance Progression Techniques using Hadoop

Yash A. Bonde
UG Student, Computer Department
Jayawantrao Sawant College of Engineering
Pune, India
yashbone999@gmail.com

Gunjan D. Deswal
UG Student, Computer Department
Jayawantrao Sawant College of Engineering
Pune, India
gunjansingh122@gmail.com

Sumit S. Shitole
UG Student, Computer Department
Jayawantrao Sawant College of Engineering
Pune, India
sumitshitole1994@gmail.com

Vrushali D. Khodade
UG Student, Computer Department
Jayawantrao Sawant College of Engineering
Pune, India
khodadevd@gmail.com

Prof. Mohan Pawar
Asst. Professor
Jayawantrao Sawant College of Engineering
Pune, India
Mohanpawar2006@gmail.com

Prof. Ravindra P. Bachate
Asst. Professor
Jayawantrao Sawant College of Engineering
Pune, India
bachateravi@gmail.com

***Abstract*— The usage of Hadoop has been increasing greatly in recent years. Hadoop adoption is widespread. Hadoop is mainly used by some big users such as Yahoo, Facebook, Netfix and Amazon for data analysis of unstructured data. As Hadoop can deal with both structured and unstructured data. Hadoop distributed file system (HDFS) is meant for storing large files but when large number of small files need to be stored, HDFS has to face few problems as all the files in HDFS are managed by a server. Hadoop, an open source java framework deals with big data. It has mainly two core components: HDFS (Hadoop distributed file system) which stores large amount of data in a reliable manner and another is MapReduce which is a programming model which processes the data in parallel and distributed manner. As large number of small files puts heavy load on NameNode of HDFS Hadoop does not perform well for those small files and MapReduce is encountered for increase in execution time. Hadoop is designed to handle huge size files and hence suffers a performance penalty while dealing with large number of small files. This research work gives an introduction about HDFS, small file problem and existing ways to deal with it these problems along with proposed approach to handle small files. In proposed approach, merging of small file is done using MapReduce programming model on Hadoop. This approach improves the performance of Hadoop in handling of small files by ignoring the files whose size is larger than the block size of Hadoop and also reduces the memory required by NameNode to store them.**

Keywords— HDFS, Hadoop, Map-Reduce, Small Files, Encryption

I. INTRODUCTION

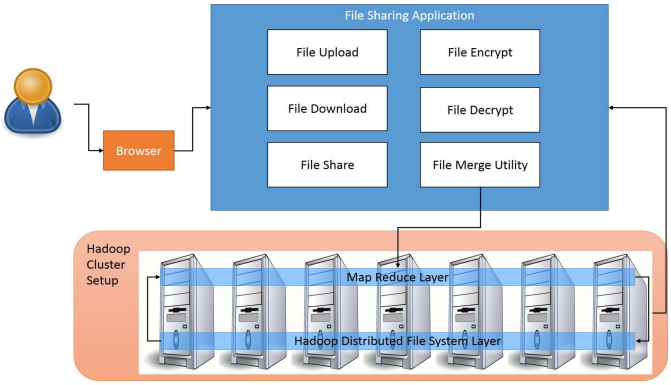
As huge amount of data is generated daily it is very difficult to deal with such huge data. So to handle this huge amount of data Hadoop is being used. Hadoop being an open source and java programming framework it supports storage and processing of extremely large data sets. It is possible to execute applications on system with Hadoop using thousands of commodity hardware nodes and it also handles thousands of terabytes of data. Hadoop being distributed file system it facilitates rapid data transfer rates within nodes and continue to operate system even in case of failure of node. With this approach the catastrophic system failure and unwanted data loss is preserve in case of number of nodes become inoperative. Hadoop is constructed using numerous functional modules as it is a software framework. Hadoop uses kernel to provide libraries for framework essentials, and other modules used in Hadoop are Hadoop distributed file system (HDFS) which can store data across number of servers to obtain high bandwidth within the nodes. Hadoop MapReduce is used to provide the programming model which can tackle large data sets and mapping of that data and reducing to obtain the desired result. [1]

II. LITERATURE SURVEY

HDFS have drawback with handling small files. It causes high consumption of NameNode and low efficiency of file reading. In this paper, SFS is used and also include merging approach. Here it cannot use the mapreduce features. Now we are increase performance of mapreduce for handling small files[3]. HDFS cannot work proper with storing and merging small files for this problem Hbase and Avro is used also it can store middle small files of different size. In future faster merging technique will be generated [4]. NameNode have its own RAM and it store metadata in it to get max Efficiency. HAR require for automatic scaling data. The following technique is used Hadoop Archive Plus (HAR+) using sha256 as the key, which is addition to HAR. Hadoop Archive plus have drawback of Overhead and it is time consuming [5]. To Avoid higher memory usage, flooding network, requests overhead and centralized point of failure (single point of failure "SPOF") of the single Namenode, HDFSX is introduce by AmrM Sauber. In future, this technique implement in real environment [6]. User access task is defined for improve and even solve the small file problem.it use PLSA technique which shows the correlations among the access tasks, applications and access files. It work from file-level to task-level. The proposed strategy effectively reduce the MDS workload and the request response delay [7]. In This avoid the weakness of hFS with help of new techniques like Fast File System and Log Structure File system with new update strategies (strategies—update-in-place and update-out-of-place).It is better technique to solve problem of handling small files in HDFS[8].

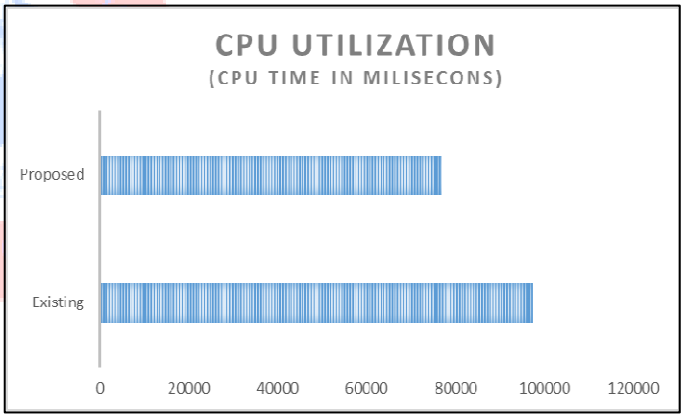
III. SYSTEM ARCHITECTURE

In this system, we have implemented File Sharing Application. This application is responsible for handle all the task. File Sharing Application consist of 6 individual blocks which are responsible for handling various types of individual tasks. These blocks are i). File Upload, ii). File Download, iii). File Sharing iv). File Encryption, v). File Decryption, vi). File Merge utility. File uploading and downloading in HDFS is handle by File Upload and File Download modules respectively. File Merging utility is main utility which optimized the storage scheme of HDFS. As per proposed system [1], this utility merge the same extension files in large one merged file. This approach will reduce the burden on NameNode. Which ultimately improves the Hadoop performance.



User use the browser to upload and download the files.Browser is connected to hadoop cluster setup via file sharing application.File sharing application can use six main method for store and merge the various types of small files.

Fig. System Architecture



IV. RESULTS AND OBSERVATION

The following graphs shows the improvement done in system. The following graph shows that CPU Utilization is reduced by 79% using Proposed System.

Fig. CPU Utilization of System

The following graph shows the comparisons between the Execution Time of Proposed System and Existing system throughout various stages of execution.

Graph state that performance is increased by (reduced
Fig. CPU Execution time

execution time) 25%,53% during Map, Reduce phase respectively. Which will cause total 44% improvement in execution time i.e. Execution time is reduced by 44%.
Overall throughput of system is also increased in proposed approach than the existing system.

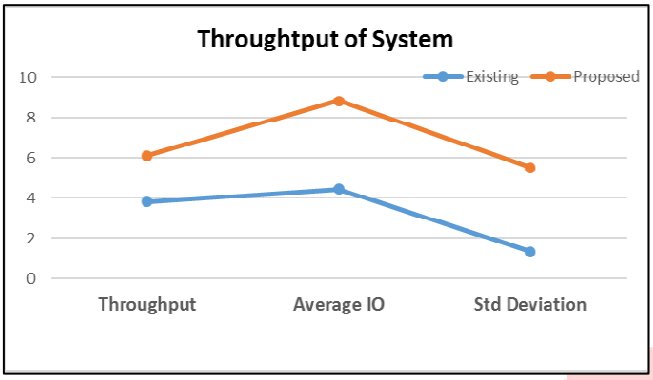


Fig. Throughput of System

V. Conclusion & Future Scope

Currently Hadoop Distributed File System (HDFS) facing problem with small files. Various methods have been proposed to deal with this problem. In terms of overhead as it involves slight overhead by combining multiple files into single split compare to other methods where sending each file to a map-reduce task will cause too much overhead. It also increases reading efficiency of small files to a great extent. HAR (Hadoop Archive) Files – HAR has been introduce to deal with small file issue. HAR is basically dependent on hadoop archive command. HAR has introduced a layer on top of HDFS, which provided interface for file accessing. A HAR file is designed by hadoop archive command runs a MapReduce job to pack the files being archived into a small number of HDFS files. Though it slow down the performance as compare to directly accessing file from HDFS. Each and every block has an index value therefore HAR create one more master index layer for lot of small file index values. So Namenode will hold master index value and in turn solves small file problem. In Future, as we did our background research into solutions to the small files problem, these following can be done to avoid produce small files (or perhaps files at all). In other words, if small files are create problem small files problem in Hadoop: 1.Change your "feeder" software so it doesn't, change your upstream code to stop generating them2.Run an offline aggregation process which aggregates your small files and re-uploads the aggregated files ready for processing3.Add an additional Hadoop step to the start of your job flow which aggregates the small files

Acknowledgement

We preconceived our commend in the vicinity to our project guide Prof. M. V. Pawar, Assistant Professor Computer Department for his precious helpfulness and navigation that he gave us throughout our Project. We specially thank our project coordinator Prof. A. V. Devare for boosting us and for arranging us all the lab readiness. We would also like to preconceive our appreciation and thanks to HOD Prof. H. A. Hingoliwala and Principal Dr. M. G. Jadhav and all our friends who have served us all through our hard work.

References

[1] Sumit S. Shitole, Yash A. Bonde, Vrushali D. Khodade,Gunjan D. Deswal, Mohan V. Pawar, Ravindra P. Bachate, "Allusive Study of Performance Progression Techniques for Hadoop," International Engineering Research Journal (IERJ), Volume 2 Issue 6 Page 2063-2068, 2016 ISSN 2395-1621

[2] Parth Gohil, B. Noble, and 1. S. Dhobi, "A Novel Approach to Improve the Performance of Hadoop in Handling of Small Files," 978-1-4799-608S-9/1S/$31.00©201S IEEE

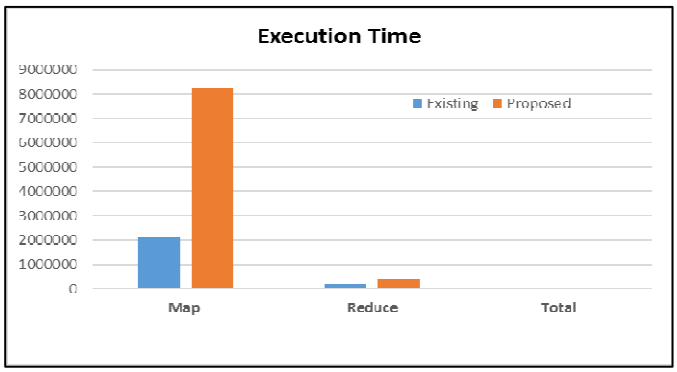
[3] Yonghua Huo, Zhihao Wang, XiaoXiao Zeng,Yang Yang, Wenjing Li, ZHONG Cheng,"SFS:a massive small file processing middleware in hadoop,"IEICE – The 18th Asia-Pacific Network Operations and Management Symposium (APNOMS) 2016

[4] Shuo Zhang, Li Miao, Dafang Zhang, Yuli Wang,"A strategy to deal with mass small files in hdfs," 978-1-4799-4955-7/14 $31.00 © 2014 IEEE DOI 10.1109/IHMSC.2014.87

[5] D. Dev; R. Patgiri. HAR+: Archive and metadata distribution! Why not both?. IEEE International Conference on Computer Communication and Informatics (ICCCI). 2015. pp.1-6.

[6] Passent M EIKafrawy, AmrM Sauber, Mohamed M Hafez,"HDFSX:Big Data Distributed File System with Small Files Support" 9781509028634/16/$31.00 ©2016 IEEE

[7] Tao Wang, Shihong Yao, Zhengquan Xu*, Lian Xiong, Xin Gu, Xiping Yang,"An effective strategy for improving small file problem in distributed file system", 978-1-4673-6850-6/15 $31.00 © 2015 IEEE



DOI 10.1109/ICISCE.2015.35

[8] Zhihui Zhang, Kanad Ghose, "hFS: A Hybrid File System Prototype for Improving Small File and Metadata Performance," Proc. SIGOPS/EuroSys European Conference on Computer Systems (EuroSys 07), ACM New York, NY, USA, June. 2007, vol. 41, pp. 175-187, doi: 10.1145/1272998.1273016
[9] Debajyoti Mukhopadhyay, Chetan Agrawal, Devesh Maru, Pooja Yedale and Pranav Gadekar, "Addressing NameNode Scalability Issue in Hadoop Distributed File System using Cache Approach",2014 IEEE.
[10] Andre Oriani and Islene C. Garcia, " From Backup to Hot Standby: High Availability for HDFS"2012 IEEE.
[11] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung," The Google File System", October 19–22, 2003, Bolton Landing, New York, USA, 2003 ACM.
[12] Yin Zhang, Weili Han, Wei Wang,Chang Lei, " Optimizing the storage of massive electronic pedigrees in HDFS" ,2012 IEEE
[13] Petros Zerfos, Hangu Yeo, Brent D. Paulovicks and Vadim Sheinin," SDFS: Secure Distributed File System for Data-at-Rest Security for Hadoop-as–a-Service",2015 IEEE
[14] Runqun Xiong, Junzhou Luo, Fang Dong, " SLDP: a Novel Data Placement Strategy for Large-Scale Heterogeneous Hadoop Cluster",2014 IEEE
[15] Jeffrey Shafer, Scott Rixner, and Alan L. Cox," The Hadoop Distributed Filesystem: Balancing Portability and Performance",2010 IEEE
[16] Bo Dong1, 2, Jie Qiu3, Qinghua Zheng1, 2, Xiao Zhong3, Jingwei Li1, 2, Ying Li3," A novel approach to improving the efficiency of storing and accessing small files on Hadoop: a case study by powerpoint files",2010 IEEE
[17] Chandrasekar S, Dakshinamurthy R, Seshakumar P G, Prabavathy B, Chitra Babu," A novel indexing scheme for efficient handling of small files in Hadoop distributed file system",2013 IEEE
[18] MENG Bing, GUO Wei-bin, FAN Gui-sheng,QIAN Neng-wu," A novel approach for efficient accessing of small files in HDFS: TLB-MapFile",2016 IEEE.
[19] Aishwarya K, Arvind Ram A, Sreevatson M C, Chitra Babu, and Prabavathy B," Efficient Prefetching Technique for Storage of Heterogeneous small files in Hadoop Distributed File System Federation",2013 IEEE.
[20] Liu Jiang, Bing Li , Meina Song,"The optimization of HDFS based on small files",2010 IEEE.
[21] Xuhui Liu1,Jizhong Han1,Yunqin Zhong,Chengde Han," Implementing WebGIS on Hadoop: A Case Study of Improving Small File I/O Performance on HDFS",2009 IEEE.
[22] Yonghwan KIM,Tadashi ARARAGI,Junya NAKAMURA and Toshimitsu MASUZAWA," A Distributed NameNode Cluster for a Highly-Available Hadoop Distributed File System",2014 IEEE.
[23] Liu Changtong," An Improved HDFS for Small File",ICACT 2016.
[24] Grant Mackey, Saba Sehrish, Jun Wang," Improving Metadata Management for Small Files in HDFS",2009 IEEE.
[25] Ankita Patel, Mayuri A. Mehta," A Novel Approach for Efficient Handling of Small Files in HDFS",2015 IEEE.
[26] Tanvi Gupta, Prof. SS Handa," An Extended HDFS with an AVATAR NODE to handle both small files and to eliminate single point of failure.",2015 IEEE.
[27] Bo Dong, Qinghua Zheng, Mu Qiao, Jian Shu, and Jie Yang," BlueSky Cloud Framework: An E-Learning Framework Embracing Cloud Computing", 2009 Springer-Verlag Berlin Heidelberg.
[28] Chi-yi Lin and Ying-chen Lin,"A load balancing algorithm for Hadoop distributed file system",2015 IEEE.
[29] ChatupornVorapongkitipun and Natawut Nupairoj," Improving Performance of Small-File Accessing in Hadoop",2014 IEEE.
[30] Xiayu Hua, Hao Wu and Shangping Ren," Xiayu Hua, Hao Wu and Shangping Ren",2014 IEEE.
[31] Wei Dai ,Ibrahim Ibrahim, Mostafa Bassiouni," A New Replica Placement Policy for Hadoop Distributed File System",2016 IEEE.
[32] E.Sivaraman and Dr.R.Manickachezian," High Performance and Fault Tolerant Distributed File System for Big Data Storage and Processing using Hadoop",2014 IEEE.
[33] Hadoop archives ,http://hadoop.apache.org/common/docs/current/hadoop_archives.html.
[34] Sequence File Wiki, http://wiki.apache.org/hadoop/SequenceFile.
[35] Map files, http://hadoop.apache.org/common/docs/current/api/org/apache/hadoop/io/MapFile.html.
[36] Tom White, The Small Files Problem,
[37] http://www.cloudera.com/blog/2009/02/02/the-small-files-problem/.
[38] Tom White. Hadoop: The Definitive Guide. O'Reilly Media, Inc. June 2009.
[39] SlideShare site, http://www.slideshare.net/.
[40] Http://issues.apache.org/jira/browse/HADOOP-1687.
[41] J. Hendricks, R. Sambasivan, S. Sinnamohideenand, and G.Ganger, "Improving small file performance in object-based storage,"
[42] Technical report, Tech. Report CMU-PDL-06-104, May 2006.
[43] The website, http://www. exascale.info