ISSN 2395-1621

Allusive Study of Performance Progression Techniques for Hadoop

[#1]Sumit S. Shitole, [#2]Yash A. Bonde, [#3]Vrushali V. Khodade, [#4]Gunjan D. Deswal
[#5]Prof. Mohan V. Pawar, [#6]Prof. Ravindra P. Bachate

[1]sumitshitole1994@gmail.com
[2]yashbone999@gmail.com
[3]khodadevd@gmail.com
[4]gunjansingh122@gmail.com
[5]mohanpawar2006@gmail.com
[6]bachateravi@gmail.com

[#1234]UG Student, Computer Department
[#56]Asst. Professor, Computer Department

Jayawantrao Sawant College of Engineering
Pune, India

ABSTRACT

In today's digital world big data is on a requisition for research purpose. As every day trillion bytes of data is accomplishing in giant chunk by tremendous resources like social media, Healthcare, Stock market, etc. To box the data and run the application on clusters and process big data in distributed environment across clusters of computers. Core Hadoop consist of a storage part known as HDFS and processing part known as MapReduce. This paper spotlight on various processes used to grasp the files in HDFS, Retrieve the files from HDFS, various file operations in HDF various performance degradation while handling files, algorithms, techniques and systems to boost the work of Hadoop.

Keywords: HDFS, Performance improvement, MapReduce

ARTICLE INFO

Article History

Received: 28th November 2016

Received in revised form :

28th November 2016

Accepted: 30th November 2016

Published online :

3rd December 2016

I. INTRODUCTION

Apache Hadoop was created out of necessity as the data is flame up from the web and raise the ability to deal with the system. Hadoop is influenced in strong processing and analysing thousands of terabytes, and also petabytes of data. Hadoop accredit distributed parallel processing of huge quantity of data. Hadoop can box both storing and processing of data and also scales without any limit. By using MapReduce design queries of Hadoop are overhead on dataset and then these queries are split into various sub-sections and then are run on many parallel nodes. Apache Hadoop also consist of Hadoop Distributed File System (HDFS) [1] [30].

The architecture of HDFS is such as that is stores the huge files but inefficiently lacks in storing the large amount of tiny files due to large consumption of memory and inadmissible cost. Tiny files are files which are smaller in size then the block size of HDFS. Hence, boxing and controlling of large number of tiny file is provocation. The paper focusses to polish tiny files on HDFS to enhance the performance of Hadoop in accessing tiny files [1].

This paper includes various fields. Field II consist literature survey carried out for this paper. Field III is discussion about comparative study of all papers. Field IV shows proposed method. Field V concludes the paper.

II. LITERATURE SURVEY

The memory of Name Node is filled up completely there are no chances of extending the cluster holding power. So the concept of the cache memory is been used for dealing with the issue of Name Node scalability [1]. For the master of HDFS, a hot standby is achieved by (i)with the help of Check pointer helper and Backup Node the performance of master's state of replication is extending and it also took benefit of message replica methods build up by some other available solutions for HDFS named as Avatar Nodes. (ii)Introducing the automatic failover mechanism which deployed another Hadoop software known as Zookeeper which is a distributed coordination service. Further in future by reducing the size of the block received messages the resource consumption can be reduced. Supporting multiple standbys and reconfiguration can also be possible [2].

A file system interface extensions are composed of allowance of distributed applications, a different perspective of design, and address measurements from both Nano-

benchmarks and real world use [3]. To recover the problem of enormous storage of electronic pedigrees, by performing optimization of the repository and accessing huge tiny XML files in HDFS. For this the correlation of tiny files of the same cover are merged into large files to reduce the metadata at NameNode and then prefetching mechanism and remerging mechanism are applied for better efficiency performance for accessing tiny XML files. May be in future optimization solutions in HDFS with the proposed system can be compared and more application scenarios can be applied and other signature and cryptography technologies might be used to express the electronic pedigrees [4]. When the cloud is offered as a cloud service the efforts are taken to provide the security for data-at-rest .Then the requirements and architecture were analyzed and describe a new distributed file system developed for Hadoop called SDFS. And further analyze the parameter tuning for SDFS and with experiments on real test-bed performance is evaluated [5].

A modernistic snakelike data placement mechanism is proposed for huge scale heterogeneous Hadoop cluster which acquires heterogeneity-aware algorithm to separate different nodes into virtual storage tiers (VST), and then data blocks are placed across nodes in each VST according to the hotness of data.to reduce disk space utilization and effective power control function SLDP uses hotness proportional replication. The optimal value of block default size can be figure out by accomplishing a number of experiments and MapReduce performance can also be fine-grained optimized [6].

The performance of HDFS is analyzed and different problem issues are tap. There exist an architectural bottlenecks in Hadoop accomplishment which result in insufficient usage of HDFS due to MapReduce scheduling delay. The root causes of performance bottlenecks evaluate trade-offs between performance and portability in HDFS. Further enhancement can be done by reducing fragmentation and cache overheads to reduce portability in future [7].

BlueSky has mostly used the resource in China which uses HDFS for storing their study materials especially in form of small videos or PowerPoint Presentations. PPT courseware solution is the solution for this problem. Merging the correlated PPT files will reduce the metadata size, so ultimately it improves the performance of NameNode. It can also be used to improve the performance of storing and retrieving data from the HDFS [8].

The main problem of a small file is the amount of memory of Data Node is used. Extended HDFS (EHDFS) proposed by Dong et al is modified version of HDFS which improves the indexing and fetching process of NameNode. The followings are four techniques of EHDFS that support for enhancing the efficiency of HDFS's small file handling as file merging, file mapping ,file accessing ,file extraction. In future, by using improved techniques memory usage, time of writing and time of reading may be reduced [9].

The new policy TLB-MapFile improves the reading and download speed of small files. TLB-MapFile approach access frequency of files and generate the HDFS audit file. It also uses merging approach like HAR but based on access frequencies of files. It maintains TLB table which contains index based on audit file and directly locates the small files. In future, we improve the process refreshing of table files which will be caused for more improvement in performance [10].

Proposed Solution based on Dong et al work focuses on reducing the time for reading the files using retrieving techniques which depend on access patterns of cluster's users. This technique also uses the same merging approach of merging files received from heterogeneous users. In this approach by doing grouping at webserver layer transparency of cluster's users is enabled. This technique is progressively learned access patterns of files. We may reduce the loop holes of this techniques by providing features like appending the new files in existing one [11]. Metadata of Merged files storing on DataNode will optimize the performance of original HDFS. In advance of this, we may apply sorting of local metadata to access file faster by searching [12].

As we know Hadoop is also used to store geographical data. This approach will help to meet the requirement of WebGIS (Geographical Information System) like File accessing pattern of the system. This is achieved using grouping the file and merging consecutive files. This approach focuses on improvement in I/O operations in Hadoop. In future, improved approach may be used in many applications based on geographical data [13].

Engineers are attracted by new technology named as Hadoop in this case single skillful master node treated as NameNode .It manages the whole file system. Because of this single point of failure, NameNode limitations, load balancing, and commodity hardware problems occurred. To resolve limitations they introduced new HDFS as Griffa file system. From this study, we may introduce new techniques to reduce the latency for accessing small files which will be generalised for all the applications [14].

However, HDFS was basically developed for streaming access to large software, but it has low storage efficiency for massive small files. Hence, increases turn-around time while handling enormous number of small files. To solve this problem, the HDFS file storage process is improved. The files are checked before uploading to HDFS clusters. Small files are merged and the index information is boxed in the index file with the pattern of key-value pair. It uses Hadoop Archives (HAR files). Thus, it can improve the access efficiency [15].

We can put the limit on directories in HDFS. We can apply limit on number of files and amount of memory used by HDFS directories. Harball is a compression method provided by Hadoop. It is based on belief of amount of memory and number of files are assigned by every client. The new policy provide the feature of archiving the files without killing Jobtracker and MapReduce. Which enhances the operations of metadata data and efficient use of HDFS. In future we may use this policy along with other another efficient for making Hadoop for efficient. [16].

Further, it does not examine the correlation of the files is used to give a faster mechanism that is useful to increase access efficiency. The proposed approach merging the correlated files into an individual file to reduce the metadata storage on NameNode. Results show that the prefetching and caching mechanisms in this approach to improve access efficiency of small files. In future, we aim to evaluate the performance of the proposed approach with various cluster settings and different ranges of file sizes [17]. When the system load is low, then the synchronization overhead is larger than its effect, therefore we cannot apply the load balancing, however, if the system load becomes higher, the average processing time of each request in our system

becomes briefer. For removing the single point of failure, S. Chandra Mouliswaran.et.al introduce 'Avatar Node' but it failover of a primary Name Node. In future, the solution can be improved by advancing the prefetching framework. The framework should support better file correlation analysis for prefetching and a better file merging process that takes this File correlation information into consideration [18].

In BlueSky cloud framework, physical Hardware is virtualized and allocated on demand for E-Learning systems. BlueSky cloud framework combines load balancing and data caching which are traditional middleware functions that serve for E-Learning systems. It can manage resource pools. We improve utilization of resources [19]. They introduce the new algorithm which contains the BalanceNode to reduce/equate the load of DataNode. Future scope, we will introduce better techniques for load balancing & also try to introduce realistic client usage behavior into the simulation to prove the usefulness of our algorithm. [20] To improve file access performance, they redesigned indexing mechanism as New HAR without changing HDFS architecture. HAR has issues in handling small files like High NameNode's memory utilization, Unacceptable storing time, NameNode becomes a bottleneck. We may implement the best techniques for indexing mechanism [21]. The performance decreases while handling the relatively small sized files. The solution to this problem is improved HDFS architecture along with new storage allocation algorithm that improves the performance and minimizes the performance degradation problem. The proposed system had improved the HDFS throughput by 300% on an average and has no effect on large data set. The proposed system uses Particle Swarm Optimization (PSO) algorithm to improve the HDFS throughput [22]. The basic problem of Hadoop is the placement of data replicas. The existing system can acquire both fault tolerance and read/write efficiency but has to lean on balancing utility to evenly distribute replicas. In existing system the two replicas are placed on one random and the third replica is to be placed on another random rack. To overcome the above mentioned problem of existing system the author has used new policy called as Partition Replica Placement Policy (PRPP). In this policy the available nodes are split into three sections. Section 1 consist of two third of the replicas i.e. 2 of 3 replicas and places them on the same rack. Section 2 and 3 has about one-third of the replicas i.e. one replica. All the process are distributed basically in two phases: 1st is the section formation and 2nd is the replica distribution phase [23].

Hadoop allows the user to store, manipulate, operate, and analyze a large amount of data which was not allowed previously with the SQL-based system. Hadoop cluster is feasible to many organizations as it provides a various improvement in conventional compute and storage resources. A Huge amount of data in the form of logs, blogs, email and other technical structured and unstructured information streams are generated by the index web searches, social media, and recommendation engines. Hadoop Distributed File System uses write-once and read-many model. This model breaks data into blocks which are spread across many nodes for fault tolerance and high performance. The master-slave architecture is used by HDFS and Hadoop. The copy of NameNode is maintained by the Secondary NameNode which is used to restart the NameNode when a failure occurs. This may not be current thus the failure of Node may cause the data loss. MapReduce is another basic component of the Hadoop which provides the computational framework for data processing [24].

III. DISCUSSION

A. Year-wise Research

Research in this area is started from 2003 (As references in this paper). But this area comes mainly in focus after the introduction of Hadoop in 2008. As time passed by, 'Performance improvement of Hadoop' became interested area for researchers and computer scientists.

Year	Paper Count
2003	1
2009	3
2010	3
2012	2
2013	2
2014	6
2015	4
2016	3

Table 1 :- Year Wise Distribution of PapersCount

Once Hadoop was introduced the amount of research doubled. Most of the research from this paper was done in the year of 2014 with 6 papers. In this area, the research is still going on we referred 3 papers from the year 2016 i.e. 3.Table shows the years and the count of research papers. These statistics prove that there is an increase in the research of the performance improvement techniques in Hadoop. Figure 1 shows the pie chart for the papers published in this area of research.

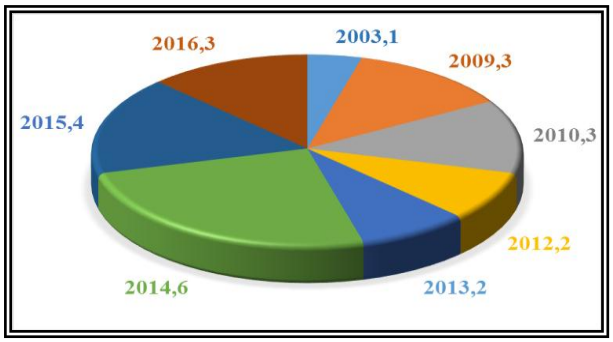

Figure 1 :- Year Wise Distribution of Papers

B. Comparative study

The following Table 2 shows the study of technologies and approaches used in various papers. According to this study, the major areas that are focused are Hadoop Distributed File System (HDFS), Meta Data Storage, File merging, Extended HDFS, GFS etc.

Sr. No.	PAPER TITLE	Areas Focused
1	Addressing NameNode Scalability Issue in Hadoop Distributed File System using Cache Approach.	Cache memory, Name Node stability.
2	From Backup to Hot Standby: High availability for HDFS	Hadoop Distributed File System, Message replica method, Zookeeper.
3	The Google File System	Google File System
4	Optimizing the storage of massive electronic pedigrees in HDFS	Hadoop File System, XML Files, Merging of small files
5	SDFS: Secure Distributed File System for Data-at- Rest Security for Hadoop-as–a- Service	Cloud services, Secure Distributed File System.
6	SLDP: a Novel Data Placement Strategy for Large-Scale Heterogeneous Hadoop Cluster	Virtual Storage Tiers, MapReduce
7	The Hadoop Distributed Filesystem: Balancing Portability and Performance	Hadoop Distributed File System, MapReduce
8	A novel approach to improving the efficiency of storing and accessing small files on Hadoop: a case study by PowerPoint files	Hadoop Distributed File System, PowerPoint Presentation, Merging of PPT
9	A novel indexing scheme for efficient handling of small files in Hadoop distributed file system	Extended Hadoop Distributed File System, file merging, file mapping file pre fetching, file extraction
10	A novel approach for efficient accessing of small files in HDFS: TLB- MapFile & quote	TLB-MapFile, TLB table, Hadoop Archive
11	Efficient Prefetching Technique for Storage of Heterogeneous small files in Hadoop Distributed File System Federation	File merging
12	The optimization of HDFS based on small files	Merging of Files, Sorting of local metadata.
13	Implementing WebGIS on Hadoop: A Case Study of Improving Small File I/O performance on HDFS.	WebGIS(Geographical Information System), Merging of files
14	A Distributed NameNode Cluster for a Highly-Available Hadoop Distributed File System.	HDFS as Griffa file system
15	An improved HDFS for Small File.	Hadoop Distributed File System, Hadoop Archives, merging of files
16	Improving Metadata Management for Small Files in HDFS.	Hadoop Archives
17	A Novel Approach for Efficient Handling of Small Files in HDFS.	Merging of files
18	An Extended HDFS with an AVATAR NODE to handle both small files and to eliminate a single point of failure.	NameNode, Load balancing.
19	BlueSky Cloud Framework: An E-Learning Framework Embracing Cloud Computing	BlueSky, E-learning systems.
20	A load balancing algorithm for Hadoop distributed file system.	Balance Node
21	Improving Performance of Small-File Accessing in Hadoop	Hadoop Archive, Hadoop Distributed File System.
22	A New Replica Placement Policy for Hadoop Distributed File System A New Replica Placement Policy for Hadoop Distributed File System	Partition-Replica Placement Policy(PRPP)
23	High Performance and Fault Tolerant Distributed File System for Big Data Storage and Processing using Hadoop	SQL, Hadoop Distributed File System, NameNode, MapReduce

Table 2 :- Study of Areas of focused in various papers

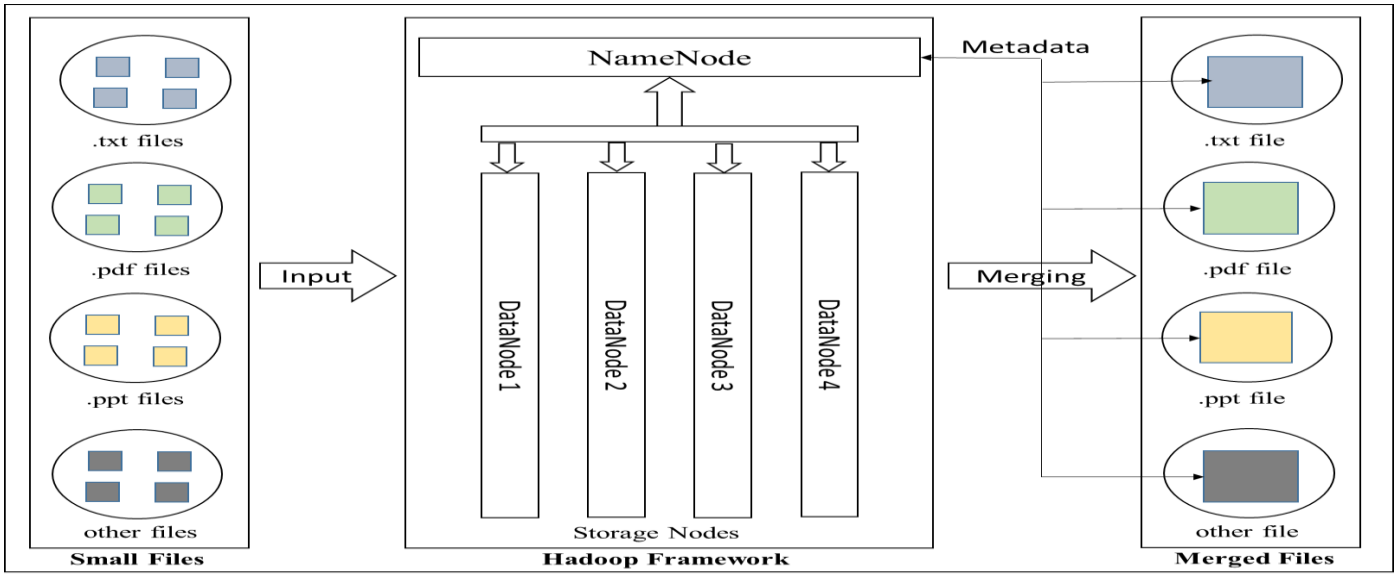

Figure 2 :- Proposed system

IV. PROPOSED SYSTEM

Figure 2 consist of three sections named as small files, Hadoop framework, and merged files. Small files consist of various files such as text files, pdf document files, Presentation files, etc. These massive number of small sized files are taken as input to the Hadoop framework. The Hadoop framework consists of one name node and different storage nodes called as data nodes. These data nodes are linked with NameNode. Then these massive number of small sized files are merged together on basis of their extension and are connected to the metadata of NameNode.

The output of the system is obtained by merging similar and relevant extension of files. The performance of Hadoop is improved and latency can be reduced. When unknown file extensions are given as input to the Hadoop framework the system fails to merge the files and the latency cannot be improved. When all the input files are correct in extension and the algorithm is followed properly then the system efficiently merges the file with metadata of Hadoop.

The proposed system also uses the modified structure of metadata and improved MapReduce programming. This modified architecture will help to improve performance of Hadoop framework

V. CONCLUSION AND FUTURE WORK

A Hadoop being wide network of research and one of the important matter of inquisition is handling bitty files in HDFS, so the following research aims on Map Reduce to deal with small files which include two aspects. Firstly, the execution time to run that file on Hadoop Cluster and secondly above mentioned aspects the determined algorithm elevate the results as compared to the current system. It handles the different files such as text file, image file, pdf file, Presentation file competently and avoid the files sizing more than threshold.

In future, work can be extended to work with different types of extension files and merging of all extension files can be done. For storing that files improved modified algorithms can be used to enhance the performance of Hadoop.

ACKNOWLEDGEMENT

We deliberate our praise in the vicinity to our project guide Prof. M. V. Pawar, Assistant Professor Computer Department for his valuable helpfulness and navigation that he gave us throughout our Project. We specially thank our project coordinator Prof. A. V. Devare for uplifting us and for arranging us all the lab readiness. We would also like to deliberate our acknowledgement and thanks to HOD Prof. H. A. Hingoliwala and Principal Dr. M. G. Jadhav and all our friends who have assisted us throughout our hard work.

REFERENCES

[1] 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.

[2] Andre Oriani and Islene C. Garcia, " From Backup to Hot Standby: High Availability for HDFS"2012 IEEE.

[3] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung," The Google File System", October 19–22, 2003, Bolton Landing, New York, USA, 2003 ACM.

[4] Yin Zhang, Weili Han, Wei Wang,Chang Lei, " Optimizing the storage of massive electronic pedigrees in HDFS" ,2012 IEEE

[5] 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

[6] Runqun Xiong, Junzhou Luo, Fang Dong, " SLDP: a Novel Data Placement Strategy for Large-Scale Heterogeneous Hadoop Cluster",2014 IEEE

[7] Jeffrey Shafer, Scott Rixner, and Alan L. Cox," The Hadoop Distributed Filesystem: Balancing Portability and Performance",2010 IEEE

[8] 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

[9] 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

[10] 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.

[11] 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.

[12] Liu Jiang, Bing Li , Meina Song,"The optimization of HDFS based on small files",2010 IEEE.

[13] 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.

[14] Yonghwan KIM,Tadashi ARARAGI,Junya NAKAMURA and Toshimitsu MASUZAWA," A Distributed NameNode Cluster for a Highly-Available Hadoop Distributed File System",2014 IEEE.

[15] Liu Changtong," An Improved HDFS for Small File",ICACT 2016.

[16] Grant Mackey, Saba Sehrish, Jun Wang," Improving Metadata Management for Small Files in HDFS",2009 IEEE.

[17] Ankita Patel, Mayuri A. Mehta," A Novel Approach for Efficient Handling of Small Files in HDFS",2015 IEEE.

[18] 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.

[19] 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.

[20] Chi-yi Lin and Ying-chen Lin,"A load balancing algorithm for Hadoop distributed file system",2015 IEEE.

[21] ChatupornVorapongkitipun and Natawut Nupairoj," Improving Performance of Small-File Accessing in Hadoop",2014 IEEE.

[22] Xiayu Hua, Hao Wu and Shangping Ren," Xiayu Hua, Hao Wu and Shangping Ren",2014 IEEE.

[23] Wei Dai ,Ibrahim Ibrahim, Mostafa Bassiouni," A New Replica Placement Policy for Hadoop Distributed File System",2016 IEEE.

[24] E.Sivaraman and Dr.R.Manickachezian," High Performance and Fault Tolerant Distributed File System for Big Data Storage and Processing using Hadoop",2014 IEEE.

[25] Hadoop archives ,http://hadoop.apache.org/common/docs/current/hadoop_archives.html.

[26] Sequence File Wiki, http://wiki.apache.org/hadoop/SequenceFile.

[27] Map files, http://hadoop.apache.org/common/docs/current/api/org/apache/hadoop/io/MapFile.html.

[28] Tom White, The Small Files Problem, http://www.cloudera.com/blog/2009/02/02/the-small-files-problem/.

[29] Tom White. Hadoop: The Definitive Guide. O'Reilly Media, Inc. June 2009.

[30] SlideShare site, http://www.slideshare.net/.

[31] Http://issues.apache.org/jira/browse/HADOOP-1687.

[32] J. Hendricks, R. Sambasivan, S. Sinnamohideenand, and G.Ganger, "Improving small file performance in object-based storage," Technical report, Tech. Report CMU-PDL-06-104, May 2006.

[33] The website, http://www. exascale.info.

