ISSN: 0974-5823 Vol. 6 No. 1 January-June, 2021

International Journal of Mechanical Engineering

ADJURATIONS ON BIG DATA APPLICATIONS TO LESSEN THE SPACE AND TIME PARAMETERS IN STORAGE AND PROCESSING

SIVA RAMA KRISHNA DODDAKA[1*], Dr. O NAGARAJU [2*]

[1]Research Scholar, Dept of CSE, ACHARYA NAGARJUNA UNIVERSITY, Guntur, India.
[2]Asst. Prof, Dept of CS, Govt. Degree College, Macherla, AP, India.

Abstract:.

The main concepts that are involved in frame work of Hadoop are MR (Map Reduce) and HDFS(Hadoop Distributed File System). Various scenarios of big data include twitter analytics like processing and storing the tweets and data processing of FB(Facebook) that depend on framework of Hadoop to perform operations like processing and storing of data from where we can investigate in future.

The point here is the usage of space and time in the processing of the above-mentioned huge amounts of the data definitely leads to higher amounts of space and time consumption of the Hadoop framework. The problem here is usage of huge amounts of the space and at the same time the processing time is also high which need to be reduced to get the fastest response from the framework.

The attempt is important as all the other eco system tools also depends on HDFS and MR to perform the data storage and processing of the data and alternative architecture to improve the usage of the space and effective utilization of the resources so as to reduce the time requirements of the framework.

The point is that the use of time and space while processing the massive amounts of data mentioned above would results in consumption of time and space in framework of Hadoop. The issue here is that the framework consumes a lot of space while also having a long processing time, which needs to be lowered to get the best response from it.

The effort is critical because all other eco system tools rely on HDFS and MR for data storage and processing, as well as alternative architecture to enhance space consumption and resource use to minimize time.

The outcome of the work is faster data processing and less space utilization of the framework in the processing of MR along with other eco system tools like Hive, Flume, Sqoop and Pig Latin. The work is proposing an alternative framework of the HDFS and MR and the name we are assigning is Unified Space Allocation and Data Processing with Metadata based Distributed File System (USAMDFS).

Keywords: Analytics, Hadoop Framework, Meta Data based File system, Eco System, Unified Space Allocation.

I. INTRODUCTION

MapReduce (MR) and other eco system tools such as Pig Latin, Hive, Flume and Sqoop rely on HDFS (Hadoop Distributed File System. In HDFS, the Map Reduce is a distributed as well as parallel processing programming model that is totally reliant on DFS (Distributed File System).

In the Hadoop eco system, all the other eco system tools rely on HDFS and MR to store and process data. Because all of these tools use HDFS as well as MR, transferring the data and storing the files requires more time because the data is traversed from the tool storage to Hadoop Distributed File System.

The current work is concerned with minimizing the time required to process and store data through the effective use of a proposed unified method to data storage, which in turn improves the framework's performance.

We feel that the work represents a paradigm shift in the study of large data scenarios and analytics. The basic requirements to store the data and processing time for large volumes of data were extremely high up to this point. The effort we're making to provide an alternative method of storing and processing data in the context of HDFS and MR will undoubtedly pave the way for new research possibilities.

The suggested design includes storage provisions apart from HDFS, as well as alternate way to use log files, the possibility of a local file system, and various other provisions to make use of time and space for trail as well as pilot run projects within the industry.

The work is organized as follows. In section 2 issues which that are related with present architecture are observed. Section 3 includes proposed work that includes Unified framework including programming and data storage can be seen including architecture model. Results are discussed in section 4. Conclusion is discussed in section 5.

II. MR AND HDFS MEMORY USAGE ISSUES

All tools of eco system including Map Reduce must rely on Hadoop Distributed File System storage for storing and accessing data, according to the framework of Hadoop. The issue with this practice is that when ever block-level data needs to be accessed and used processing must be completed first, followed by uploading the data into HDFS.

The other level of action is that once the data has been stored into Hadoop Distributed File System through Map Reduce or any other tools of eco system it must be accessed only through Hadoop Distributed File System. As a result, a significant amount of time will be consumed, as well as additional time needs for typical data processing.

The time it takes for transition between the interface and HDFS is more and experience extra utilization of resources to complete procedure including storing and processing of data.

- The initial point is data transmission from the interface of MR/other eco system tools need to traverse up to Hadoop Distributed File System.
- The temporary data/ log files must be saved in a more efficient manner, as analyzing big data necessitates a large quantity of storage, even for log files.
- A greater amount of time is spent during traversing, storing, and retrieving data between Hadoop Distributed File System and Map Reduce logics.
- There is no standard process for Flume/Hive/ MR/Pig/Sqoop to follow when it comes to the point of data access and storage.
- The MR and HDFS logics have root-based access, therefore there is no need to configure them for the other tools of eco system.
- The issues that relate with HDFS reliance and replication will lead to problems that occur while using additional memory within the file system.
- The design of MR and HDFS while dealing with individual user configuration include usage of more resources, necessitating the need of distinguishing tools that are used for monitoring to keep track of cluster operations.

To resolve all the issues, the present work proposes a unified framework method that uses the Local File System in conjunction with HDFS to manage all of the available activities.

As a result, we should expect the utilization of extra space within Local File System and decrease the factor of time necessary to switch between the Local File System and Hadoop Distributed File System.

To solve most of these issues, the current study proposes several solutions as well as a unified architecture for data storage and processing to save time and space.

III. UNIFIED METHODOLOGY PROPOSED FOR UTILIZATION OF TIME AND SPACE OF FRAMEWORK.

We used the below experimental set - up to carry out the current research. Operating system used is Ubuntu 20.04 LTS, and Hadoop 3.2.1 is configured with MR and HDFS. core-site.xml, hdfs-site.xml, hadoop-env.sh, yarn-site.xml and mapred-site.xml were the files that were configured.

We evaluated data storage using the Lfs work1 (local file system) and the time factor is estimated, as well as same file is imported into HDFS to estimate the time factor.

Obviously, when comparing the time for importing the file to HDFS, LFS storage takes a little less time. This is because with LFS storage, one can utilize storage directly for placing source data to the file system.

The configuration of Hadoop plays a critical role in HDFS import, and the time for same import (same file, same size) using LFS to store into HDFS is more. The reason for it is not based on the volume of data consumed by the file, but things involve the configuration checking and traversal and rechecking the activities, which add more space and time requirements to HDFS.

The observations can be seen in below figures.

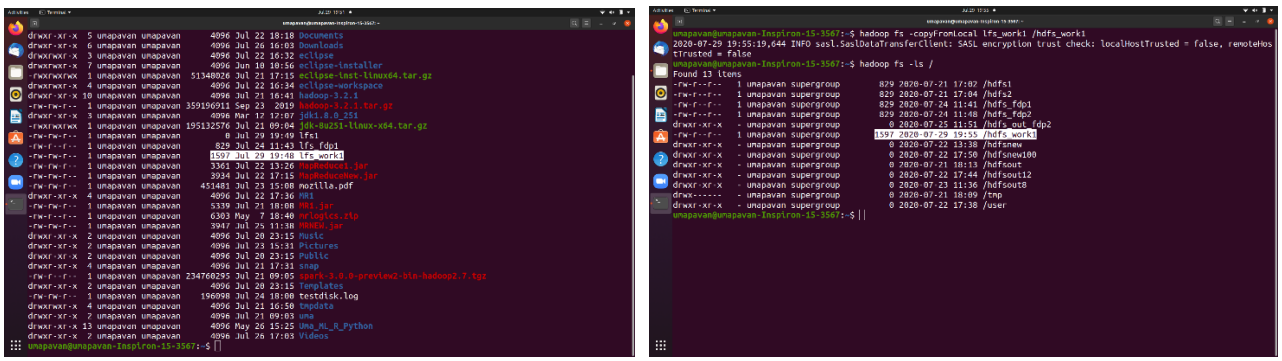

Figure 1: Time required for storing the file in Local File System and Hadoop Distributed File System.

The above figure 1shows that the time required to store a file in LFS(Local File System) is minimum when compared to the time taken for importing the file with same size into HDFS. We use LFS to launch temporary files, processing of log files instead of using HDFS which in turn will reduce the time required for file importing and executing the program.

Another observation we'll offer today is the many factors that are required to complete Map Reduce (MR) logic processing from the point of launch, as well as the time factor, which can be seen in the accompanying graph.

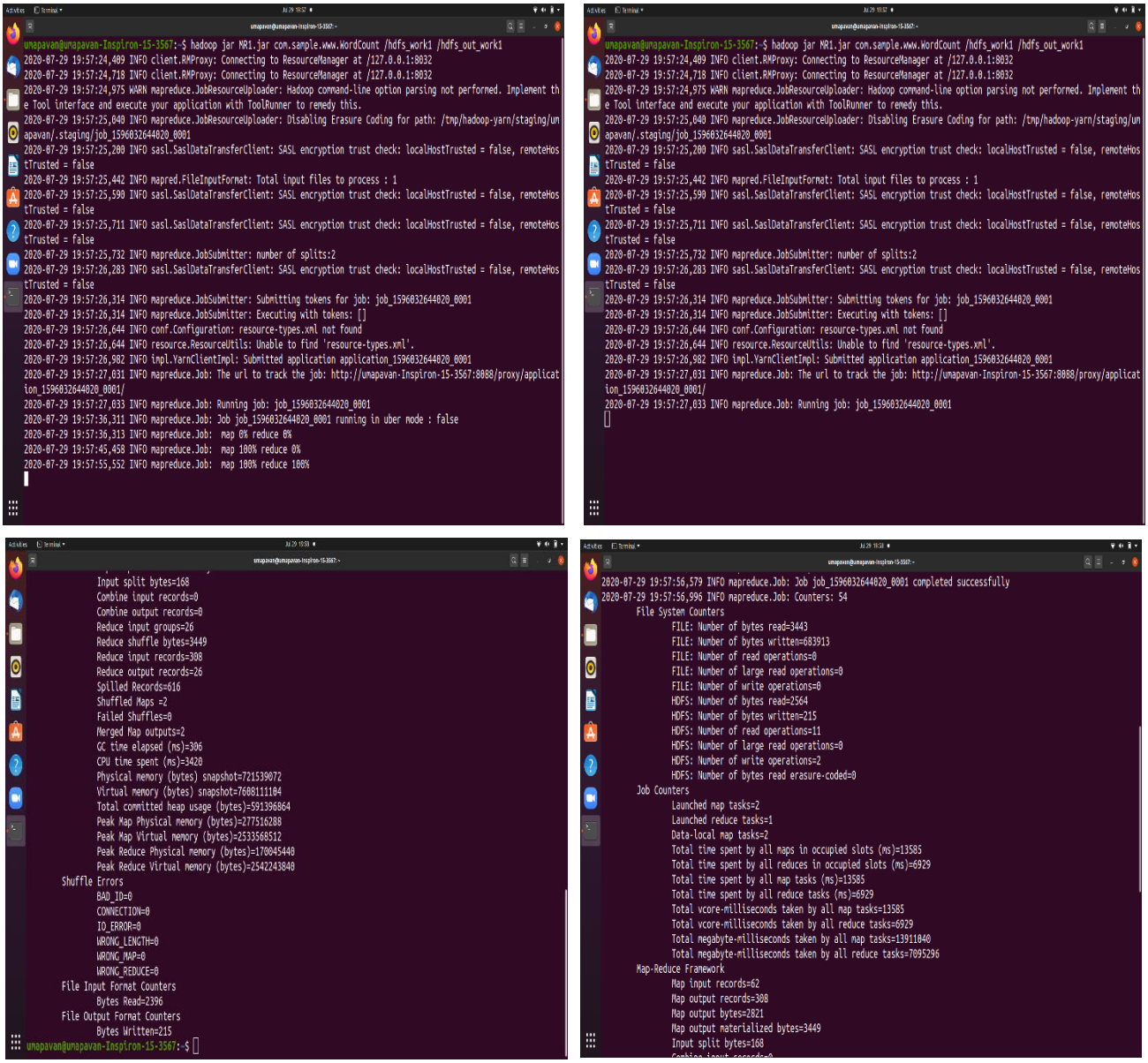

Figure 2: .jar file processing in Map Reduce (MR) including various other parameters.

The main point to note here is that file access from HDFS necessitates the use of various other parameters, traversing data from Local File System (such as Hadoop API support, .jar files and JDK) to HDFS (input Directory/file and output directory) necessitates additional configuration and support in the processing.

As a result, the proposed methodology focuses on two key issues.

- The use of Local File System while storing the file from external sources such as MySQL, data from Facebook, or data from Twitter, as shown in Figure 1, required less time than HDFS import, thus the suggestion is to use LFS to store external files.
- The second significant improvement is to use LFS not only for Java and .jar, but also Hadoop API as the whole build route, including Mapper, Driver logics and Reducer.
- Instead of moving to HDFS every time, the factors like input directory or file and output directory should use LFS for the temporary file information and log file.

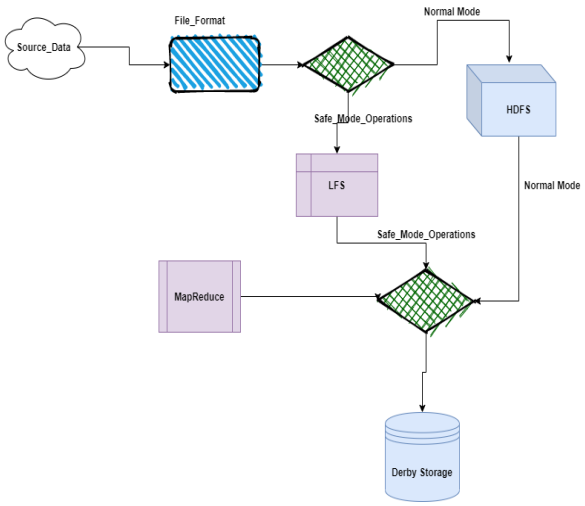

Figure 3: Unified Model of Hadoop Storage and Processing.

The design we propose is the core of the work; its central topic is to fix the space requirements for additional configuration used by framework of Hadoop, which successively solves the requirement of time because we are providing alternative ways to map the memory and relying on same combinations and with necessary changes in configuration files of Hadoop, one can meet the required concept of lowering parameters like time and space while using HDFS File System.

MR, HDFS, additional tools used in eco system (as users of data base may use Hive and users of Scripting can use Pig Latin, etc.,) and importing and exporting simple tools like Flume and Sqoop are included in the design to handle unstructured and structured data.

The work's scope is not related to our current research, but it does have an impact on the researcher community, encouraging them to come up with new ideas in the field of Bigdata and Hadoop.

IV. CONCLUSION

The present article explained how MR and HDFS work in the setting of massive volumes of data. We calculated the space and time requirements for performing storage using the LFS and HDFS.

When comparing the time required accomplishing storage using LFS and HDFS, it is evident that LFS consumed less time. The other experiment we conducted was to use Map Reduce logic to execute the distributed and parallel and use of the HDFS.

Observations made here is LFS is used in process of Map Reduce for. jdk, Hadoop API, .jar and HDFS is used for input file and output directory. As a result, the suggested unified approach method suggests using LFS to reduce the space and time needs while processing MR.

The article discusses file system use in terms of HDFS and LFS, as HDFS devotes space and time to configure and other activities related to data export/import/ parallel processing, and after the data is processed again, it is moved back to distributed storage with extra parameters.

The article has given the possibilities of utilizing the space of the LFS whenever there is chance of skipping the HDFS heavy configuration and context switch based on the services mentioned here.

The Hadoop Framework's file system and processing safe mode option is best suited for prototype projects or projects that require cluster performance testing. The safe mode will allow the cluster to execute the same functions as before, such as viewing the file system of HDFS and performing tasks of MR with input which has light weight, with the output of the job being estimated in directory of LFS directory, reducing resource usage and saving time.

We ensure that this work substantially aids us in improving time efficiency and reducing space utilization by traditional usage of framework of Hadoop, and that the Unified framework is a ground-breaking attempt in Hadoop research including distributed and parallel concepts of MR.

REFERENCES

1. Ivanliton Polato, "A Comprehensive view of Hadoop Research- A Systematic Literature Review, Volume 46, November 2019, PP:1-25.
2. Konstantin Shvachko," The Hadoop Distributed File System" IEEE 2010.
3. Mohd Rehan Ghazi, "Hadoop, MapReduce and HDFS: A Developers Perspective" Procedia Computer Science ,2015.
4. Wu Jun ," Study of New Materials Design based on Hadoop", MATEC Web of Conference 61,07016(2016).
5. Himi Egemen Ciritogulu, "A Heterogeneity –aware replica deletion for HDFS" Journal of Big data, October 2019.
6. Sujit Roy,"Hadoop Periodic Jobs Using Data Blocks to Achieve Efficiency", Indian Journal of Research in Computer Science and Information Technology,Vol:3, Issue:3,2018.
7. Ronald C.Taylor, " An Overview of the Hadoop/Map Reduce/HBase/ framework and its current applications in bio informatics ",BMC Bio Informatics,2010.
8. Jason.C.Cphen, "Towards a Trusted HDFS storage platform",ACM Digital Library,Vol:19,No:3,July 2014.
9. Aibo Song, "A memory-Based Hadoop Framework for Large Data Storage",Resource management in Virtaulized Clouds,Hindawi publishers,Volume 2016
10. Tafiq Hassanin, "Severly Imbalanced Big data challenges: Investigating Data Sampling approaches",Springer, 30 November 2019.
11. Konstantin, The Hadoop Distributed File System,"Symantic Scholar.org,October 2013.
12. D.Borthakur," The Hadoop Distributed File System Architecture and Design", 2017, Apache.org.
13. H.Liao, "Multi-Dimensional index on Hadoop Distributed File System" 2010, IEEE explore ieee.org.
14. J.Zhang "A Distributed Cache for Hadoop Distributed File system in real-cloud services,ACM 2012.
15. S.Jin,"Design of trusted file System based on Hadoop,2012, Springer.