*International Journal of Scientific Engineering and Applied Science (IJSEAS) - Volume-1, Issue-8,November 2015*
*ISSN: 2395-3470*
*www.ijseas.com*

# Comparison study on Hadoop's HDFS with Lustre File System

Sagar S. Lad [#1],NaveenKumar[*2] ,Dr. S.D. Joshi[*3]

[#] *Research Scholar , * Research Scholar, * Professor*
*Department of CSE, BharatiVidyapeeth's Collage of Engineering,*
*Pune, MS, India*

***Abstract*— Map/Reduce is a distributed computational algorithm, which is originally designed by Google, Mapreduce is expanding in** popularity **and is being utilized for many large-scale jobs. The open-source Hadoop system has the most common implementation of Map/Reduce. For the fundamental storage backend, Hadoop by default manages the Distributed File System (HDFS), however Hadoop originally was planned to be compatible with other FS (file systems). Apart from HDFS, Hadoop does provide few other types of FS i.e. KFS, S3. Hadoop uses Java interface provided by these file systems. Lustre doesn't contain JAVA wrapper. Lustre doesn't accept like hadoop does. Lustre provides a POSIX-complaisant interface for UNIX file system. Common problems with Hadoop plus HDFS as a platform can be solved with Lustre as a backend system.**

***Keywords*—** Hadoop, Lustre, HDFS, MapReduce, File Systems (FS)

## I. INTRODUCTION

When the amount of data is very large and it cannot be handled by the conventional database management system, then it is known as Big data. Big data is creating new challenges for the data analyst. There can be three types of data like unstructured form, semi structured form and structured form. In big data, most of part is in unstructured format. Unstructured data is difficult to handle. The Apache Hadoop project yields better tools and techniques to handle this huge amount of data. Hadoop gives a HDFS (Hadoop distributed file system) for storage and the MapReduce techniques for processing this data can be used. Hadoop uses HDFS as the fundamental storage backend, however Hadoop was originally planned to work with other FS too. Other types of FS are also supported by Hadoop i.e. KFS, S3 etc. Hadoop with HDFS file system have few challenges, one of them is Lustre as backend for Hadoop.

## II. LITERATURE SURVEY

The MapReduce programming paradigm has been successfully utilized at Google for many different purposes. **Dean, Jeffrey, and Sanjay Ghemawat[1]** implemented MapReduce framework is highly scalable.Mapreduceruns on a large cluster of commodity machine. The implementation makes efficientutilization of these machine resources.Because of this Mapreduce is suitable for use on many of the huge computational problems encountered at Google. **Shvachko, Konstantin** [3] explained in paper that HDFS is designed to store very large data sets easily, and to flow hose data sets at high bandwidth to user applications. Thousands of servers both host directly attached storage and execute user application tasks in a large cluster. By distributing storage and computation across many servers, the resources can increase with demand while remaining economical at every size. High Performance Computers (HPCs) which is usually deal with problems in traditional scientific computing area, **Schwan[2] described** that Lustre is a GPLed cluster file system for Linux that is currently being tested on three of the world's largest Linux supercomputers, each with more than 1,000 nodes. In the past 18 months authors have tried many tactics to scale to these limits. In this paper, authors discussed some of our successes and failures. **Nathan Rutman[4]** compared HDFS and Lustre architectural drivers and resulting system performance of Map/Reduce computations on HPC hardware. Author evaluated theoretical and actual performance of Lustre and HDFS for a variety of workloads in both traditional and Map/Reduce-based applications. Further, examined the additional benefits (cluster efficiency, flexibility, and cost) of using Lustre, on a Hadoop compute cluster. **JieYu[7]** discussed drawbacks efficiently utilize the HPC relatively lower availability and usability. Authors proposed algorithm to implement MapReduce framework on HPC to solve discussed problems and extensively expand the application field of HPC. Author designed a workable strategy to deploy

Hadoop on HPC with a Lustre file system, and adapt Lustre to a better performance based on the nature of data access in Hadoop.

## III. HADOOP'S HDFS VS LUSTRE FILE SYSTEM

### HDFS

Map/Reduce is a framework to easily write applications that process large amounts of data in parallel on clusters of compute nodes. The compute and storage nodes are the same in a Map/Reduce

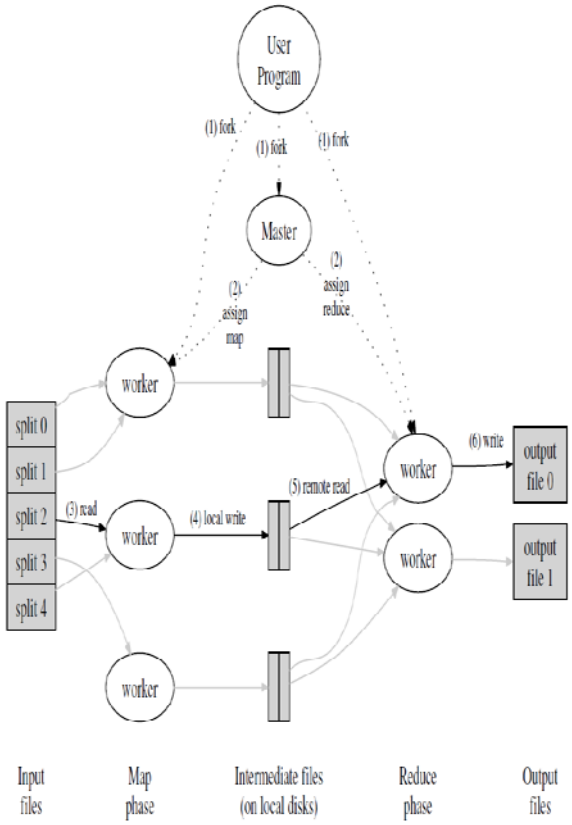


Fig 1.1 : Mapreduce Execution digram

In traditional Map/Reduce framework, input and output data was kept on the HDFS, with intermediate data collected in a local. Temporary file system stored on the Mapper nodes, and shuffled as needed (via HTTP) to the nodes running the Reducer tasks.The Map/Reduce framework is written in Java, which makes it easy to deploy across operating systems and (commodity) hardware platforms. Map/Reduce framework executes a library of functions and default values at each processing stage.

HDFS is not a POSIX-compliant file system. In HDFS, once data is written it is not modifiable (a write-once, read-many access model). HDFS protects data by replicating disk data blocks across multiple devices. By default, HDFS data blocks are replicated three times: (local, "nearby", and "far away") to minimize the impact of a data center catastrophe. Facebook created a new network striping module for HDFS, which is now available.

### Lustre

Lustre is a client/server based cluster file system. In Lustre, Data is stored on Object Storage Servers (OSSs) and metadata is stored on Metadata Servers (MDSs). Lustre is designed for large-scale compute and I/O-intensive, performance-sensitive applications. It is developed to operate efficiently on many types of high-end network fabrics, including InfiniBand, Elan, Myrinet, as well as TCP/IP, using advantage of RDMA where available. Files in Lustre are broken into stripes, which are typically stored on multiple Object Storage Targets (OSTs), allowing parallel read and write access to different parts of the file. Being POSIX-compliant Lustre mounted remotely in a manner similar to NFS.

### Issues with Hadoop and HDFS as backend

1. Hadoop principle: Moving data vs moving computation

As stated by task-assign method, Hadoop can't prepare tasks that depend on locally stored data. i.e. in Fig1.2, whenever node-A calls a task, but if all the tasks preassign to this node have already completed, JobTracker will then assign nodeA a task preassigned to other nodes in rack 2. In this condition, node-A will execute a few tasks whose data is not local. This violates the Hadoop principle: "Moving Data is Costier than Moving Computation". These tasks generate a large amount of net I/O when they have given massive inputs.

2. Output data distribution

Hadoop+HDFS storage strategy, for high computational complexity applications, temp/intermediate data is not good. Such applications produces large and expanding MapTask outputs. If huge MapTask results saved on Linux FS locally, there are chance of OS, disk or IO bottleneck. To avoid such problem, such input/output operations need to be distributed but such distribution is no allowed in Hadoop.

3. MapTask output

Before real task starts, Reduce node require to utilize HTTP so as to shuffle all related big MapTask outputs. Very high number of net I/O and merging or spilling operation is produced. Memory will be exhausted due to the bursting shuffle stage. It will make OS-kernel to kill few important processes like SSH. As a result cluster may imbalanced and could become unmanageable.

4. HDFS cannot be applied as a normal FS. So extending HDFS file system is difficult.

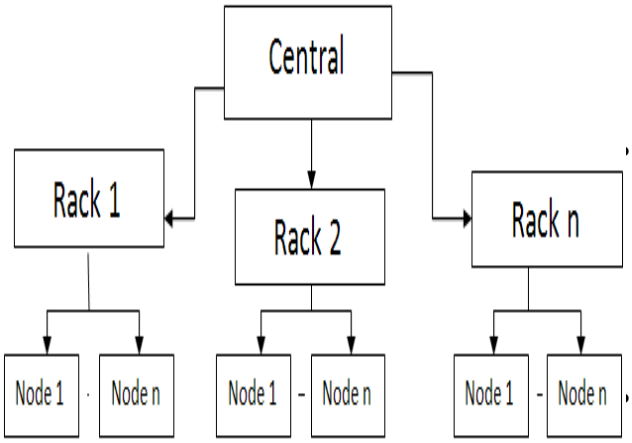


Fig 1.2 : Topology Map of all cluster nodes

5. For small data/file HDFS usually takes long time.

**Using Lustre for Hadoop**

Considering above issues with HDFS, using Lustre as backend file system instead of HDFS is better option. Hadoop with Lustre provides several benefits, including:

1. When inputs data are not locally then each HadoopMapTask can read data parallelly.

2. In Lustre, unlike HDFS big intermediate output data can be distributed which causes minimal disk or I/O traffic.

3. While doing data shuffle, Lustre generates a hard links for the reducer node. These hard links are nothing but delay in n/w transmission time which could be effective/efficient.

4. Unlike HDFS, read/write operation on small data/files can be more efficient in Lustre as it can be scaled as a normal POSIX FS.

5. Lustre is a real parallel file system. Lustre allows temporary or intermediate data to be kept inparallel on multiple nodes that will minimizing the load on single nodes.

6. Lustre provides its own network protocol, which will be more efficient for bulk data transfer than the HTTP protocol. As Lustre is a shared file system, each client sees the same file system image, so hard links can be used in Lustre to avoid data transfer between nodes.

Challenges of Lustre

MapReduce framework is not compatible with shared storage system and thus cannot be directly deployed on a High Performance Computer (HPC) system. Running MapReduce applications on HPC is still a challenge. Even if we implement MapReduce framework on the system, the application cannot avoid the I/O bottleneck of the shared storage system.

Making Lustre systems reliable is also a challenge. For luster system, finding stable Lustre versions is problematic. To find stable storage hardware is difficult. Also to identify misbehaving applications and recover from disaster is challenge.

- JAVA wrapper is not available in Lustre and thus Lustre cannot be adopted like KFS, S3 FS (file systems)

## IV. Conclusion

Hadoop uses HDFS as the fundamental storage backend though Hadoop was originally planned to work with other FS too. Other types of FS are also supported by Hadoop i.e. KFS, S3 etc. Hadoop with HDFS file system have few challenges, one of them is Lustre as backend for Hadoop.

Given the disadvantages of HDFS, Lustre looks much promising storage backend for Hadoop when compared with HDFS.

## References

[1] Dean, Jeffrey, and Sanjay Ghemawat, "MapReduce: simplified data processing on large clusters," Communications of the ACM 51.1 (2008): 107-113.

[2] Venner, Jason. *Pro Hadoop*. Apress, 2009.

[3] Lustre, [Online], Available: http://wiki.lustre.org

[4] White, Tom, *Hadoop: the definitive guide*. O'Reilly, 2012.

[5] Lam, Chuck, *Hadoop in action*, Manning Publications Co., 2010.

[6] Schwan, Philip, "*Lustre: Building a file system for 1000-node clusters*," Proceedings of the 2003 Linux Symposium, vol. 2003, 2003.

[7] Shvachko, Konstantin, et al, "*The hadoop distributed file system*," Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on, IEEE, 2010.

[8] He, Huang, et al, "Implementation and Evaluation of Massive DataProcessing Paradigm on High Performance Computers," Journal of Computer Research and Development (2012): S1.

[9] Jie Yu, Guangming Liu, Wei Hu, Wenrui Dong, Weiwei Zhang“ Mechanisms of Optimizing MapReduce Framework on High Performance Computer “ 2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing.

[10] “Using Lustre with Apache Hadoop“ by Sun Microsystems Inc.

[11] Pal, A.; Agrawal, S., "An experimental approach towards big data for analyzing memory utilization on a hadoop cluster using HDFS and MapReduce," Networks & Soft Computing (ICNSC), 2014 First International Conference on , vol., no., pp.442,447, 19-20 Aug. 2014