IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

File Systems and Hadoop Distributed File System in Big Data

G Fayaz Hussain[1], K Tarakeswar[2]

Assistant Professor, CSE Dept, Ravindra College of Engineering for Women, Kurnool, Andhra Pradesh[1]

Assistant Professor, CSE Dept, G Pullaiah College of Engineering & Technology, Kurnool, Andhra Pradesh[2]

Abstract: Hadoop provides distributed file system and a framework for the analysis and transformations of very large data sets using MapReduce. In a large cluster, thousands of servers both host directly attached storage and execute user application tasks. By distributing storage and computation across many servers, the resource can grow with demand while remaining economical at every size .Over the last few years; organizations across public and private sectors have made a strategic decision to turn big data into competitive advantage. The challenge of extracting value from big data is similar in many ways to the age-old problem of distilling business intelligence from transactional data. At the heart of this challenge is the process used to extract data from multiple sources, transform it to fit your analytical needs, and load it into a data ware house for subsequent analysis, a process known as “Extract, Transform Load”. The nature of big data requires that the infrastructure for this process cost-effectively. Apache Hadoop has emerged as the de facto standard for managing big data. In past few years Hadoop Distributed File System (hdfs) has been used by many organizations with gigantic data sets and streams of operations on it. HDFS provides distinct features like, high fault tolerance, scalability; etc. The Name node machine is a single point of failure (SPOF) for a HDFS cluster. If the name node machine fails the system needs to be re-starting manually the system less available.

Keywords: Big data, Hadoop, HDFS.

I. INTRODUCTION

Big data usually includes data sets with sizes beyond the ability of commonly used software tools to capture, curate, manage, and process the data within a tolerable elapsed time. Big data sizes are a constantly challenges. Moving target, as of 2012 ranging from a dozen terabytes to many peta bytes of data in a single data set. In a 2001 research report report and related lectures, META Group (now Gartner) analyst Doug laney defined data growth challenges and opportunities as being three dimensional, i.e. increasing volume (amount of data), velocity (speed of data in and out), and variety (range of data types and sources).Gartner and now much of the industry, continue to use this “3Vs”model for describing big data. In 2012, Gartner updated its definition as follows:”big data is high volume, high velocity, and /or high variety information assets that require new forms of processing to enable enhanced decision making insight discovery and process optimization. Additionally, a new V”Veracity “is added by some organizations to describe it. Big data uses inductive statistics and concepts from nonlinear system identification to infer laws (regressions, nonlinear relationships, dependencies and perform predictions of outcomes and behaviors.

Big data can also be defined as “big data is a large volume unstructured data which cannot be handled by standard database management system like DBMS, RDMS or ORDBMS”. Big data is a blanket term for any collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications. The challenges include capture, duration storage, search, sharing, transfer, analysis and visualization. The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found to “spot business trends, determine quality of research, prevent diseases, link legal citations, combat crime, and determine real-time roadway traffic conditions.[1]

As of 2012, limits on the size of data sets that are feasible to process in a reasonable amount of time were on the order of Exabyte’s of data that is millions of terabytes. Scientists regularly encounter limitations due to large data sets in many areas, including meteorology, genomics,[2] connections, complex physics simulations[3], and biological and environmental research[4]. The limitations also affect Internet search ,finance and business informatics .Data sets grow in size in part because they are increasingly being gathered by ubiquitous information – sensing mobile devices aerial sensory technologies (remote sensing), software logs , cameras, microphones , radio-frequent identification (RFID)readers , and wireless sensor networks [5][6][7].The world’s technological per-capita capacity to store information has roughly doubled every 40 months since the 1980s,[8] as of 2012 every day 2.5 Exabyte ($2.5*10^{18}$) of data were created[9] . The challenge for large enterprise is determining who should own big data initiatives that straddle the entire organization [10]. Big data is difficult to work using most relational data base management systems and desktop statistics and visualization packages , requiring instead “massively parallel software running on tens, hundreds , or even thousands of servers[11].What is considered “big data” varies depending on the capabilities of organization managing the set , and on the capabilities of the applications that are traditionally used to process and analyze the data for the first time may trigger a need to reconsider data management options. For others, it may take tens or hundreds of terabytes before data size becomes a significant consideration [12]. Visualization created by IBM of Wikipedia widest .At multiple terabytes in size, the text and images of Wikipedia are classic example of big data.

II. HADOOP FILE SYSTEM

It seems any time any one emotions big data on the web these days, the conversation inevitably turns to hadoop. And why not there’s lot of elephant poop and while the philosophy and approach can be argued, it’s important to remember that the reason you have so many choices is that you need the right data base for the right job. In that context, let’s look at where hadoop fits. Hadoop, comprised at its core of the hadoop file system and map reduce, is very well designed to handle huge volumes of data across a large number of nodes. At a high level, Hadoop leverages parallel processing across a large number of nodes. At a high level, hadoop leverages parallel processing across many commodity servers to respond to client applications. The key difference is, rather than only looking at parallel computing, it looks at parallelizing the data access. This all sounds great, but in reality hadoop is designed for large files, not large quantities of small files, so if have millions of 50 kb documents that is not Hadoop’s sweet spot. Likewise, Hadoop stores its data on hard disks spread across the many nodes. This is opposite to the industry standard of storing the data on single (or a few) file servers, AS or SAN. So if you already have big data, then moving to a Hadoop system will require time and resources to re-architect. Even though Hadoop leverages many servers, each one requires a significant amount of memory (more than your typical desktop), and if the name node runs out of memory, you are looking at a crash.Also, at the moment Hadoop is open source, and that means you save money at the expense of time- it is a developer tool requiring client-side development, but growing and adapting .and that means it requires some patience.

Now let’s look at what big data really is: While the volumes of data are growing by leaps and bounds from many sources, such as social media, location data, locality information, operations and supply chain, the type of information is also an issue. It may be structured, semi-structured or unstructured .Making sense of and gaining knowledge from this data to achieve a competitive advantage should be the driving goal. So if you have Big data and need to search and sort through the bulk of that data, then hadoop may serve your purpose. If the majority of the data is structured or even unstructured but you are able to add structured metadata describing the unstructured portion and you want to run complex analyses on the data. to predict or ask questions outside of the standard reports , questions which cannot be prepared in advance(i.e. the types of queries most valuable to real business intelligence), then you probably need a column- based data store.

III. INTRODUCTION OF HDFS

The hadoop Distributed file system (HDFS) - a subproject of the Apache Hadoop project – is a distributed, highly fault tolerant file system designed to run on low-cost commodity hardware. HDFS provides high-throughput access to application data and is suitable for applications with large data sets. This article explores the primary features of HDFS and provides a high- level view of the HDFS architecture data. HDFS is an Apache Software Foundation project and a subproject of the Apache Hadoop project (see Resources). Hadoop is ideal for storing large amounts of data, like terabytes and petabytes, and uses HDFS as its storage system. Hdfs lets you connect nodes (commodity personal computers) contained within clusters over which data files is handled in a streaming manner, meaning that applications or commands are executed directly using the Map reduce processing model (again, see Resources).

a) NameNode: The HDFS namespace is a hierarchy of files and directories. Files and directories are represented on the NameNode by inodes which record attributes like permissions, modification and access times, namespace and disk space quotas. The file content is split into large blocks (typically 128 megabytes, but user selectable file by file) and each block of the file is independently replicated at multiple DataNodes (typically three, but user selectable file by file)

b) DataNodes: Each block replica on a DataNode is represented by two files in the local host’s native file system. The first file contains the data itself and the second file is block’s metadata including checksums for the block data and the block’s generation stamp. The size of the data file equals the actual length of the block and does not require extra space to round it up to the nominal block size as in traditional file systems. Thus, if a block is half full it needs only half of the space of the full block on the local drive

c) HDFS Client: Similar to most conventional file systems, HDFS supports operations to read, write and delete files, and operations to create and delete directories. The user references files and directories by paths in the namespace. The user application generally does not need to know that file system met a data and storage are on different servers, or that blocks have multiple replicas.

d) Checkpoint Node:
The NameNode in HDFS, in addition to its primary role serving client requests, can alernatively execute either of two other roles, either a CheckpointNode or a BackupNod.The role is specified at the node startup. The CheckpointNode periodically combines the existing checkpoint and journal to create a new checkpoint and an empty journal. The CheckpointNode usually runs on a different host from the NameNode since it has the same memory requirements as the NameNode. It downloads the current checkpoint and journal files from the NameNode, merges them locally, and returns the new checkpoint back to the NameNode

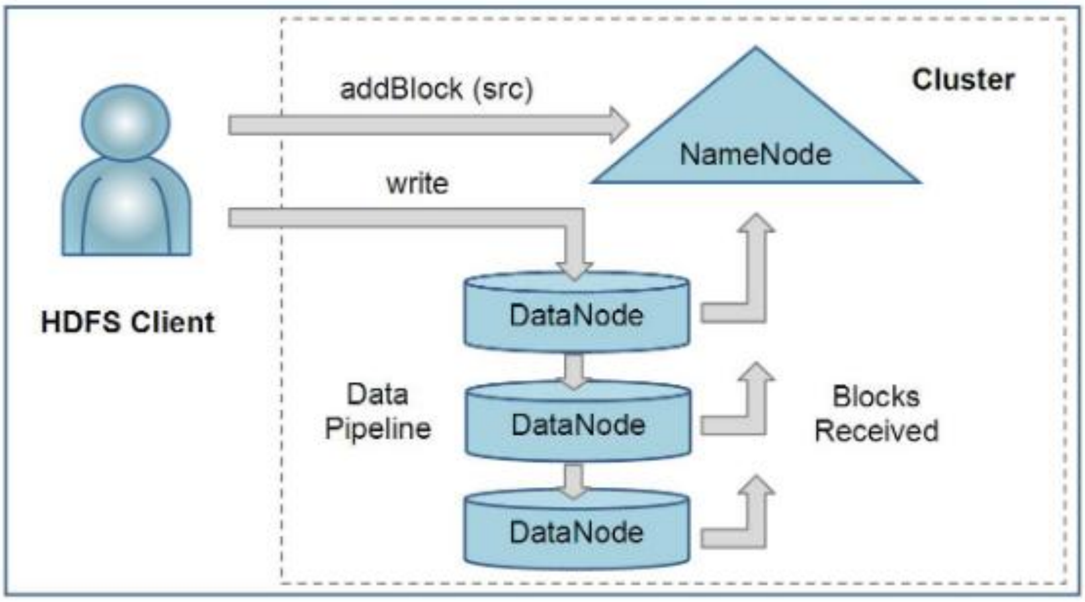

Figure 1: HDFS ARCHITECTURE

IV. HADOOP FILE SYSTEM & NORMAL FILE SYSTEM

The major two differences that is notable between HDFS and other File systems are:

Block Size:
Every disk is made up of a block size. And this minimum amount of data that is written and read from a disk. Now a file system also consists of blocks which are made out of these blocks on the disk. Normally disk blocks are of 512 bytes and those of file system are of a few kilobytes. In case of HDFS we also have the blocks concept.

But here one block size is of 64Mb by default and which can be increased in an integral Multiple of 64 i.e. 128MB, 256 Mb, 512Mb or even more in GB's. It all depends on the requirement and use-cases.

Metadata Storage:
In normal file system there is a hierarchical storage of metadata i.e. let's say there is a folder ABC, inside that folder DEF, and inside that there is hello.txt file. Now the information about hello.txt (i.e., Meta data info of hello.txt) file will be with DEF and again the Meta data of DEF will be with ABC. Hence this forms a hierarchy and this hierarchy is maintained until the root of the file system. But in HDFS we don't have a hierarchy of metadata. All the Meta data information resides with a single machine known as Name Node (or Master Node) on the cluster.

V. GOOGLE FILE SYSTEM & HADOOP FILE SYSTEM

Architecture of GFS: GFS is clusters of computers. A cluster is simply a network of computers. Each cluster might contain hundreds.

Properties:
- Chunks replicated on multiple chunkservers (default is 3) for reliability.
- Chunk size is 64MB which is much larger than normal file system blocks.
- Lazy space allocation avoids wasting space.

Advantages:
- Reduces interaction on master.
- Reduces metadata stored on master.

Disadvantages:
- Small files may become hotspots.

Client can be other computers or computer applications and make a file request. Request can range retrieving and manipulating existing files to creating new files on the system. Clients can think as customers of the GFS.

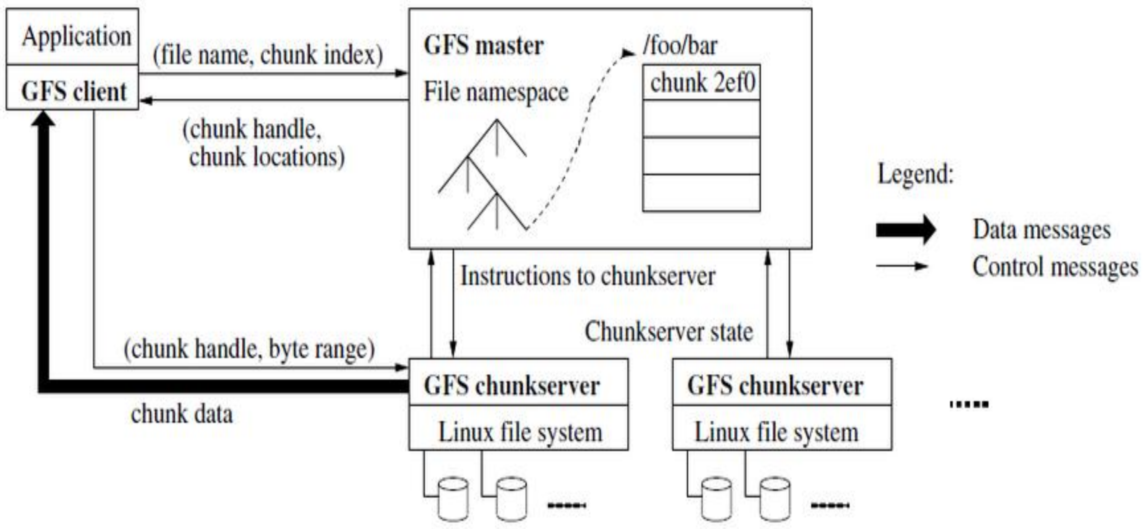

Figure 3: Architecture of GFS

Master Server is the coordinator for the cluster. Its task includes:-

1. Maintaining an operation log that keeps track of the activities of the activities of the cluster. The operation log helps keep service interruptions to a minimum if the master server crashes, a replacement server that has monitored the operation log take its place.

2. The master server also keeps track of metadata, which is the information that describes chunks. The metadata tells the master server to which files the chunks belong and where they fit within the overall file.

Chunk servers are the workhorses of the GFS. They store 64-MBfile chunks. The chunk servers don't send chunks to the master server. Instead, they send requested chunks directly to the client. The GFS copies every chunk multiple times and stores it on different chunk servers. Each copy is called a replica. By default, does GFS makes three replicas per chunk, but users can change the settings and makes more are fewer replicas if desired.
Having a single master enables the master to makes sophisticated chunk placement and replication decision using global knowledge. However the involvement of master in reads and writes must be minimized so that it does not become a bottleneck. Clients never read and write file data through the master. Instead a client asks the master which chunk servers it should contact. It caches this information for a limited and interacts with the chunk servers directly for many subsequent operations. File requests follow a standard work flow. A read request is simple; the client sends a request to the master server to find out where the client can find a particular file on the system. The server responds with responds with the location for the primary replica of the respective chunk. The primary replica holds a lease from the master server for the chunk in question.

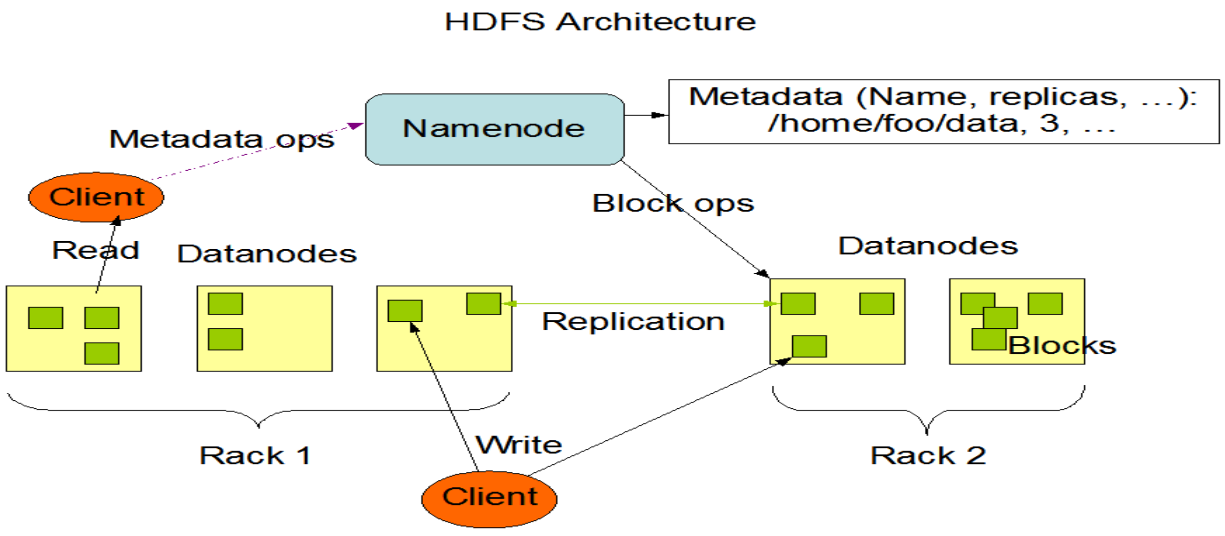

Figure 5: HDFS Process

VI. ADVANTAGE OF HDFS

It is easy to handle partial failure. Here the entire nodes can fail and restart. Flat scalability in executing limited amount of data on a small number of nodes.HDFS store large number of information. Data will be written to the hdfs once and the read several times.The overhead of cashing helps the data should simply be re read from hdfs source. Can be deployed on large clusters of cheap commodity hardware as opposed to expensive, specialized parallel-processing hardware

VII. DISADVANTAGES

Rough manner:- Hadoop Map-reduce and HDFS are rough in manner. Because the software under active development. Programming model is very restrictive- Lack of central data can be preventive. Joins of multiple datasets are tricky and slow- Often entire dataset gets copied in the process. Cluster management is hard- In the cluster, operations like debugging, distributing software, collection logs etc are too hard. Still single master which requires care and may limit scaling Managing job flow isn't trivial when intermediate data should be kept Optimal configuration of nodes not obvious. Eg: –#mappers, #reducers, mem.limits.

VIII. CONCLUSION

In this paper, we have focused on big data and hadoop related to big data. Also compare HDFS and other file systems and why we are using HDFS than other file systems, mainly we have compared with normal file system and Google file system and at the end, and the advantages and disadvantages in HDFS than the other file system.

REFERENCES

[1] "IBM What is big data? — Bringing big data to the enterprise". www.ibm.com. Retrieved 2013-08-26.
[2] Oracle and FSN, "Mastering Big Data: CFO Strategies to Transform Insight into Opportunity", December 2012
[3] Jacobs, A. (6 July 2009). "The Pathologies of Big Data". ACMQueue
[4] "Data, data everywhere". The Economist. 25 February 2010. Retrieved 9 December 2012.
[5] "Community cleverness required". Nature 455 (7209): 1. 4 September 2008. doi:10.1038/455001a.
[6] "Sandia sees data management challenges spiral". HPC Projects. 4 August 2009.
[7] Hellerstein, Joe (9 November 2008)."Parallel Programming in the Age of Big Data". Gigaom Blog.
[8] Segaran, Toby; Hammer bacher, Jeff (2009). Beautiful Data: The Stories Behind Elegant Data Solutions. O'Reilly Media. p. 257. ISBN 978-0-596-15711-1.
[9] Reichman, O.J; Jones, M.B; Schildhauer, M.P. (2011). "Challenges and Opportunities of Open Data in Ecology". Science 331 (6018): 703–5. doi:101126/science.1197962.PMID 21311007.
[10] "Data Crush by Christopher Surdak". Retrieved 14 February 2014.
[11] Hilbert & López 2011