International Journal of Engineering Science Invention
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org ||Volume 5 Issue 9|| September 2016 || PP. 82-86

Introduction to Big Data and Hadoop using Local Standalone Mode

K.Srikanth [1], P.Venkateswarlu [2], R.Roje Spandana[3]
[1,2,3]*(Information Technology, Jawaharlal Nehru Technological University Kakinada, India)*

***Abstract**: Big Data is a term defined for data sets that are extreme and complex where traditional data processing applications are inadequate to deal with them. The term Big Data often refers simply to the use of predictive investigation on analytic methods that extract value from data. Big data is generalized as a large data which is a collection of big datasets that cannot be processed using traditional computing techniques. Big data is not purely a data, rather than it is a complete subject involves various tools, techniques and frameworks. Big data can be any structured collection which results incapability of conventional data management methods. Hadoop is a distributed example used to change the large amount of data. This manipulation contains not only storage as well as processing on the data. Hadoop is an open- source software framework for dispersed storage and processing of big data sets on computer clusters built from commodity hardware. HDFS was built to support high throughput, streaming reads and writes of extremely large files. Hadoop Map Reduce is a software framework for easily writing applications which process vast amounts of data. Wordcount example reads text files and counts how often words occur. The input is text files and the result is wordcount file, each line of which contains a word and the count of how often it occurred separated by a tab.*

***Keywords:** Big Data, Hadoop, HDFS, Map Reduce, WordCount*

I. Introduction

Big data is generalized as a large data which is a collection of it is a collection of big datasets [6] that cannot be processed using traditional computing techniques. Large data is not purely a data, rather than it is a complete subject involves various techniques, tools and frameworks. Due to the onset of new advanced technologies, creation, and communication means like media, social networking sites, the amount of data produced by mankind is growing speedily every year [2]. The large amount of data beginning of time till 2003 was 6 billion gigabyte. Hadoop is a distributed exemplar used to change the large amount of data. This direction contains not only storage as well as processing on the data. In addition to simple storage, Google File System was built to data-intensive, support large-scale, distributed processing applications. The coming year, new paper, related Map Reduce Simplified Data Processing on Large Clusters, was presented, defining a programming model framework that provided automatic parallelization and the scale to process thousands of terabytes of big data in a one job no limitation thousands of machines. When paired, these two systems could be used to build large data processing [5] clusters on relatively inexpensive, commodity machines. This paper directly inspired the development of HDFS and Hadoop Map Reduce, respectively. Within the Apache Software Foundation, real time projects that explicitly make use of integrate with, Hadoop are springing up regularly. Some of these real time projects make authoring Map Reduce work easier and more jobs accessible, while others focus on getting data in and out of HDFS, simplify operations, enable deployment in cloud environments, and so on.

II. Big Data

Big Data is a term defined for data sets that are extreme and complex where traditional data processing applications are inadequate to deal with them. The data is large data, moves too fast, or does not fit the structures of our current big data architectures [7]. Big Data is typically large volume of unstructured and structured data that gets created from various organized and disordered applications, activities and channels such as emails etc. The main adversity with Big Data includes search, capture, storage, sharing, analysis, and visualization [2]. The core of Big Data is Hadoop which is a platform for administer computing problems across a number of servers. It is first developed and released as open source by web application. It implements the Map Reduce approach pioneered by Google in compiling its search indexes [8]. To store data, Hadoop utilizes its own distributed file system, HDFS, which makes data available to multiple computing nodes. Big data outbreak [4], a result not only of increasing web usage by people around the world the connection of billions of devices to the web.

III. Hadoop

Hadoop is a platform that provides both shared storage and computational capabilities. At the time Google had published papers that described its different distributed file system, the Google File System and Map Reduce, a computational framework for parallel processing. The successful implementation of these paper concepts in Nutch resulted in its split into more projects, the second of which became Hadoop, in this section will look at Hadoop architectural perspective, examine how much company's uses it, and consider some of its weaknesses. Once we have covered Hadoop background, will look at how to install Hadoop and run a Map Reduce job. Hadoop is a distributed master and slave architecture that consists of the Hadoop Distributed File System for storage and Map Reduce [10] for computational capabilities. Traits intrinsic to Hadoop are data partitioning and parallel computing of large datasets. Its storage and computational capabilities scale with the more number of hosts to a Hadoop cluster, and can reach volume sizes in the pet bytes on clusters with thousands of hosts. In the first step in this section will examine the Hadoop Distributed File System and Map Reduce architectures.

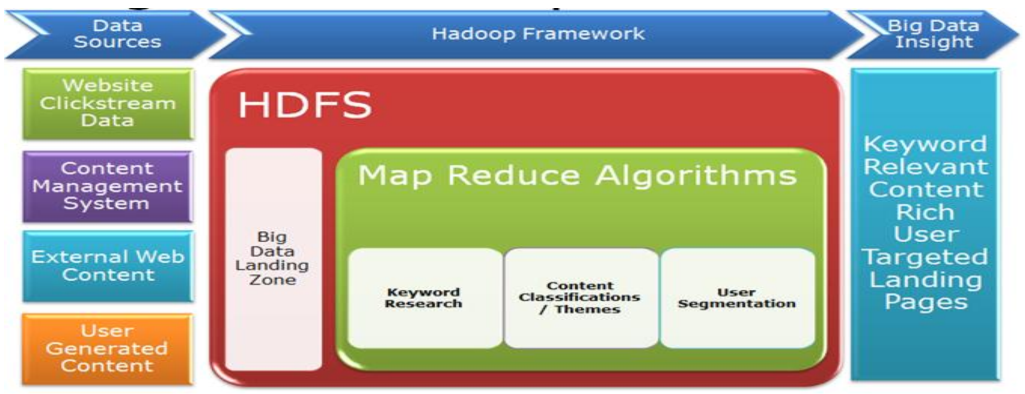


Figure 1. Big Data and Hadoop Architecture

IV. Hdfs

The Apache Hadoop is a file system called the Hadoop Distributed File System or simply HDFS. HDFS was built to support big throughput, processing read and write of extremely large files. Traditional large Network Attached Storage (NAS) and Storage Area Networks (SANs) offer centralized [1], low-latency access to either a block device or a file system on the order of terabytes in size. These systems are unusually as the backing store for large databases, content delivery systems and same types of data storage needs because they can support full-featured POSIX semantics, scale to meet the size requirements of these systems, and offer lesser-latency access to data. Imagine for a second [2], though, hundreds or thousands of machines all awakening at the same time and pulling hundreds of terabytes of data from a central control storage system at once. This is where conventional storage does not necessarily scale. By creating a system composed of independent machines, each with its own input output subsystem, network interfaces, hard disks, RAM, and CPU, for the specific type of workload we are interested in.

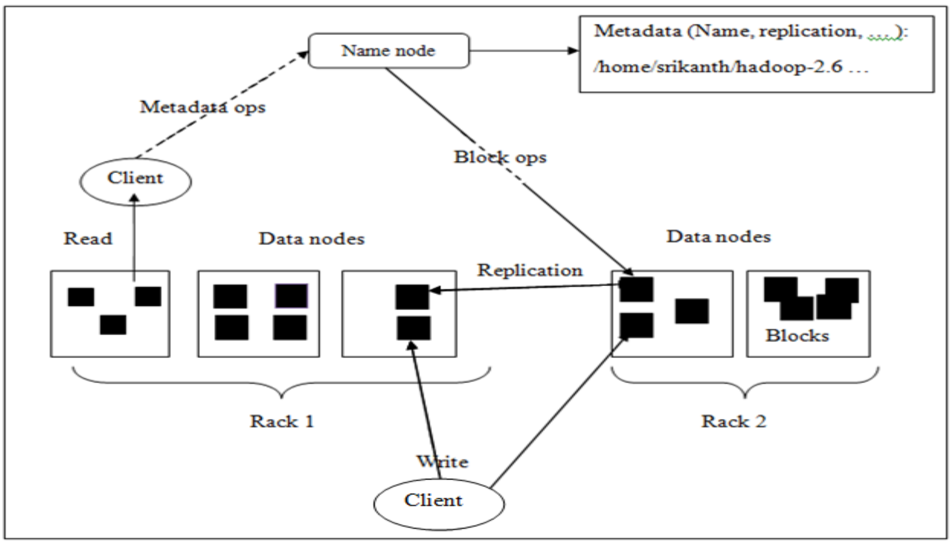


Figure 2. HDFS Architecture

4.1 There are a number of specific goals for HDFS:

- Store millions of huge files, each greater than tens of gigabytes, and file system sizes embracing tens of petabytes.
- Use a scale out model based on inexpensive commodity servers with internal JBOD rather than RAID to achieve large scale storage. Accomplish possibility and high throughput through application level reproduction of data.
- Optimize for large streaming reads and writes rather than low suspension access to many small files. Batch performance is more important than reciprocal response times.
- Gracefully deal with component failures of machines and disks.
- Support the performance and scale requirements of Map Reduce processing.

V. Mapreduce

Hadoop Map Reduce is a software framework for freely writing applications which process large amounts of data in parallel on large clusters of product hardware very trustful, fault-tolerant manner. A Map Reduce job usually splits the input dataset into self directed collections which are processed by the map tasks in a completely parallel manner. The framework distinguished the outputs of the maps, which are then input to the reduce tasks. Typically both the input and the output stored in a file system. The framework takes care of scheduling tasks, monitoring them and reproduces the failed tasks. Typically the storage nodes and the compute nodes are the same [3], that is, the Map Reduce framework and the widespread file system are running on the same set of nodes. This configuration allows the framework to effectively schedule tasks on the nodes where data is presented before, resulting in very high aggregate bandwidth across the cluster.
The Map Reduce framework consists of a single master [9] Job detector and one slave Taskdetector per cluster-node. The master is responsible for scheduling the jobs' element tasks on the slaves, monitoring them and reproduces the failed tasks. The slaves execute the tasks as assisted by the master. Minimally, applications specify the input and output locations and provide Map Reduce functions via implementations of perfect interfaces and or abstract-classes, and other job parameters, composed of the job configuration. The Hadoop job client then submits the job and configuration to the Jobdetector which then assumes and answerable of distributing the software configuration to the slaves, scheduling tasks and monitoring them, providing status and characteristic information to the job client.

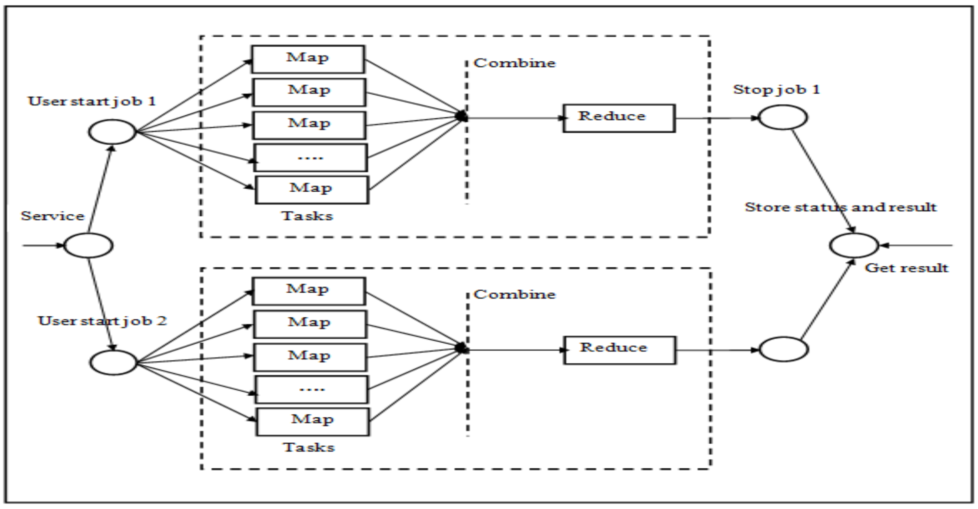


Figure 3. MapReduce Architecture in Big Data

5.1 Key Features of Map Reduce Framework

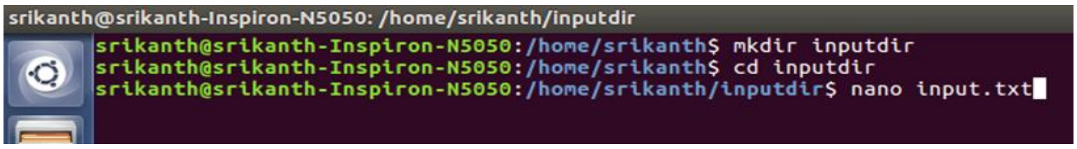
- Implement Framework for Map Reduce Execution
- Intellectual Developer from the complexity of Distributed programming
- Partial failure of the refine cluster is expected and tolerable
- Redundancy and fault-tolerance is built in. so that programmer does not have to worry
- Map Reduce programming model is language self-sufficient
- Self starting parallelization and distribution
- Failing tolerance
- Enable data local refine
- Shared nothing architecture

VI. Wordcount Example

Wordcount example reads text files and counts how frequently words occur. The input is text files and the output is word count, each line contains a word and the count of how oftentimes it occurred, divided by a tab. Each mapper takes a line as input and breaks it into words. It then spit out a key, value pair of the word and 1. Each reducer sums the counts for each word and spit out a single key, value with the word and sum. As for improvement of the reducer is also used as a incorporate on the map output. This reduces the amount of data sent across the network by combining each word into a single record.

Step 1: Running Worcount

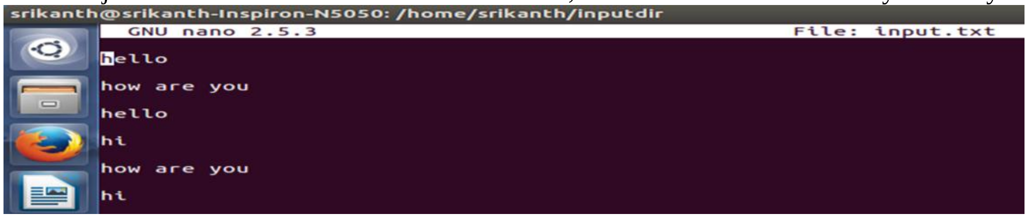
```
$mkdirinputdir
$cd inputdir
$nano input.txt
```



```
srikanth@srikanth-Inspiron-N5050: /home/srikanth/inputdir
srikanth@srikanth-Inspiron-N5050:/home/srikanth$ mkdir inputdir
srikanth@srikanth-Inspiron-N5050:/home/srikanth$ cd inputdir
srikanth@srikanth-Inspiron-N5050:/home/srikanth/inputdir$ nano input.txt
```

Figure 4.Create a file

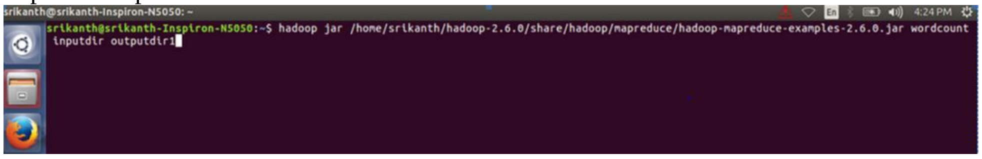
Step 2:Create the jar file and enter some text and save the file, it will be there in local file system only.



```
srikanth@srikanth-Inspiron-N5050: /home/srikanth/inputdir
  GNU nano 2.5.3                                   File: input.txt
hello
how are you
hello
hi
how are you
hi
```

Figure 5. Input to a text file

$hadoop jar /home/srikanth/hadoop-2.6.0/share/hadoop/mapreduce/hadoop-mapreduceexamples-2.6.0 wordcount inputdir outputdir to

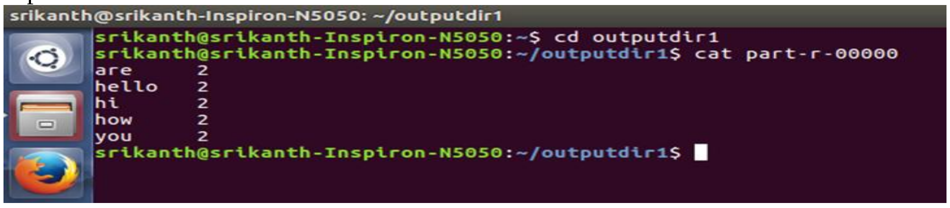


```
srikanth@srikanth-Inspiron-N5050: ~
srikanth@srikanth-Inspiron-N5050:~$ hadoop jar /home/srikanth/hadoop-2.6.0/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar wordcount
 inputdir outputdir1
```

Figure 6. Create a Jar file

Step 3: wordcount inputdit ouputdir

```
$ cd outputdir
$ cat part-r-00000
```



```
srikanth@srikanth-Inspiron-N5050: ~/outputdir1
srikanth@srikanth-Inspiron-N5050:~$ cd outputdir1
srikanth@srikanth-Inspiron-N5050:~/outputdir1$ cat part-r-00000
are     2
hello   2
hi      2
how     2
you     2
srikanth@srikanth-Inspiron-N5050:~/outputdir1$
```

Figure 7. Output for a wordcount

VII. Conclusion

Using big data large volumes of data get created from various applications and activates where these common technical challenges are very common in various application domains. The data is extremely large where it does not fit to the structure of current database architectures. The paper describes Local StandaloneMode with Hadoop which is an open source software used for processing of large volumes of data. The paper directly is based on the development of HDFS and Hadoop Map Reduce. The Interest and investment in Hadoop has led to an entire ecosystem of related software both open source and commercial. Within the Apache Software Foundation, projects are integrated with Hadoop are springing up recurrently. Some of these projects make Map Reduce jobs easier and more accessible while others focus on getting data in and out of HDFS, simplify operations, and so on. Based on this paper we can extend the paper with the help of YARN and DFS to execute the word count as a web based application.

References

[1]. S.VikramPhaneendra&E.Madhusudhan Reddy "Big Data- solutions for RDBMS problems- A survey" In 12thIEEE/IFIP Network Operations & Management Symposium (NOMS 2010)(Osaka, Japan, Apr 19{23 2013).
[2]. Kiran kumara Reddi&Dnvsl Indira "Different Technique to Transfer Big Data: survey" IEEE Transactions on 52(8) (Aug.2013) 2348 {2355}.
[3]. Jimmy Lin "MapReduce Is Good Enough?" The control project.IEEE Computer 32 (2013).
[4]. Umasri.M.L, Shyamalagowri.D,Suresh Mining Big Data: - Current status and forecast to the future" Volume 4, Issue 1, January 2014 ISSN: 2277 128X.
[5]. B. J. Doorn, M. V. Duivestein, S. Manen and Ommeren, ―Creating clarity with Big Data‖, Sogeti, (2012).
[6]. Bernice Purcell "The emergence of "big data" technology and analytics" Journal of Technology Research 2013.
[7]. Defining Big Data Architecture Framework: Outcome of the Brainstorming Session at the University of Amsterdam, 17 July 2013.Presented at NBD-WG, 24 July 2013 [Online].
[8]. AhmedEldawy, Mohamed F. Mokbel "A Demonstration of Spatial Hadoop: An Efficient Map Reduce Framework for Spatial Data" Proceedings of the VLDB Endowment, Vol. 6, No. 12Copyright 2013 VLDB Endowment 21508097/13/10.
[9]. A Workload Model for Map Reduce by Thomas A. deRuiter, Master's Thesis in Computer Science, Parallel and Distributed Systems Group Faculty of Electrical Engineering, Mathematics, and Computer Science. Delft University of Technology, 2nd June 2012.
[10]. HDFS, http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html.