JCSE International Journal of Computer Sciences and Engineering Open Access

Survey Paper Volume-3, Issue-10 E-ISSN: 2347-2693

A Survey on Map Reduce Algorithm for Big data analysis using Hadoop, Pig and Hive utility tools

Ankita Kadre[1*] and S.R Yadav[2]

Dept. of Computer Science and Engineering, MITS, Bhopal-India

www.ijcseonline.org

Received: Sep/28/2015 Revised: Oct/10/2015 Accepted: Oct /22/2015 Published: Oct /31/ 2015

Abstract-Data is growing at a rate which cannot be handled by the traditional methods of computing. To store and process such data new data analysis and storage techniques have emerged over the last few years. Hadoop is one such parallel processing open source framework which provides distributed storage and processing of big data. This paper introduces Big Data, a new platform which enables accessing, manipulating, analyzing, and visualizing data residing on a Hadoop cluster. In this paper a survey is done on big data analysis using Hadoop and other utility tools like Pig and Hive. The majority of large-scale data intensive applications executed by data centers are based on Map-Reduce or its open-source implementation, Hadoop. Such applications are executed on large clusters requiring large amounts of energy, making the energy costs a large fraction of the data center's overall costs. Therefore to minimizing the energy consumption when executing Map-Reduce jobs is a critical concern for data centers. In this survey Flight data has been analyzed in terms of the mentioned parameters such as time complexity and energy consumption information's are retrieved using Hadoop.

***Keywords*:** Map reduce, big data, Hadoop, HDFS, Pig & Hive, Flight data

1. Introduction

Big Data is varied, growing, moving fast, and it is very much in need of smart management. Data, cloud and engagement are energizing organizations across multiple industries and at present an enormous opportunity to make organizations more agile, more efficient and more competitive. In order to capture that opportunity, organizations require a modern Information Management architecture.

Key enablers of appearance and growth of Big Data are –

- Increase of storage capacities.
- Increase of processing power
- Availability of data
- Every day we create 2.5 quintillion bytes of data; 90% of the data in the world today has been created in the last two years alone.

The challenges include analysis, capture, creation, search, sharing, storage, transfer, visualization, and privacy violations. The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data.

Big data is difficult to work with using most relational database management systems and desktop statistics and visualization packages, requiring instead "massively parallel software running on tens, hundreds, or even thousands of servers". What is considered "big data" varies depending on the capabilities of the organization managing the set, and on the capabilities of the applications that are traditionally used to process and analyze the data set in its domain. Big Data is a moving towards target; which is considered to be "Big" today will not be so years ahead. "For some organizations, facing hundreds of gigabytes of 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.

3 V's of Big Data :

Volume: The quantity of data that is generated is very important in this context. It is the size of the data which determines the value and potential of the data under consideration and whether it can actually be considered Big Data or not.

Velocity: Data is streaming-in at unprecedented speed and must be dealt with in a timely manner. Reacting quickly enough to deal with data velocity is a challenge for most organizations.

Variety: Variety refers to the many sources and types of data both structured and unstructured. This variety of unstructured data creates problems for storage, mining and analyzing data.

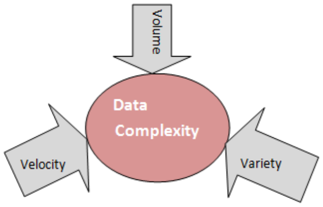


Figure no.1.1. Characteristic of Big data

Hadoop:

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage[7]. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

The paper includes these modules:

- **Hadoop Common:** The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS):** A distributed file system that provides high-throughput access to application data.

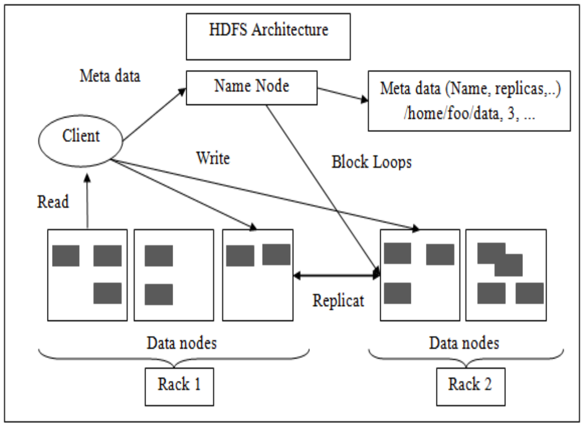


Figure no.1.2. HDFS Architecture

- **Hadoop YARN:** A framework for job scheduling and cluster resource management.
- **Hadoop Map-Reduce:** A YARN-based system for parallel processing of large data sets.

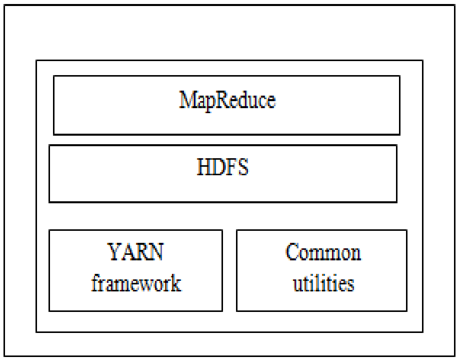


Figure 1.3: Architecture of Hadoop

Hadoop Map-Reduce:

Hadoop Map-Reduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

A Map-Reduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce tasks. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

Typically the compute nodes and the storage nodes are the same, that is, the Map-Reduce framework and the Hadoop Distributed 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 already present, resulting in very high aggregate bandwidth across the cluster.

The Map-Reduce framework consists of a single master Job tracker and one slave Task tracker per cluster-node. The master is responsible for scheduling the jobs' component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves execute the tasks as directed by the master.

Minimally, applications specify the input/output locations and supply map and reduce functions via implementations of appropriate interfaces and/or abstract-classes. These, and other job parameters, comprise the job configuration. The Hadoop job client then submits the job (jar/executable etc.) and configuration to the Job Tracker which then assumes the responsibility of distributing the software/configuration to the slaves, scheduling tasks and monitoring them, providing status and diagnostic information to the job-client.

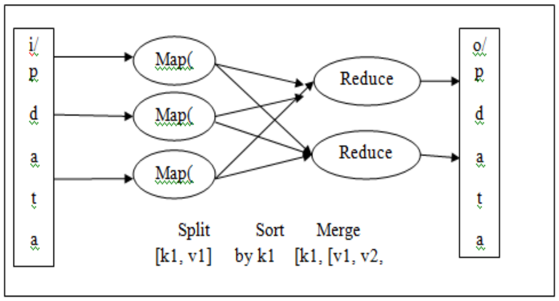


Figure 1.4: MapReduce Architecture

Components of Map-Reduce

There are three basic components of Map-Reduce :

- Driver
- Mapper
- Reducer

There are various ways to execute Map-Reduce operations:

- The traditional approach using Java Map-Reduce program for structured, semi-structured, and unstructured data.

- The scripting approach for Map-Reduce to process structured and semi structured data using Pig.

- The Hive Query Language (HiveQL or HQL) for Map-Reduce to process structured data using Hive.

Pig:

Pig is a Hadoop extension that simplifies Hadoop programming by giving a high-level data processing language while keeping Hadoop's simple scalability and reliability.

Pig has two major components:

- A high-level data processing language called Pig Latin .
- A compiler that compiles and runs your Pig Latin script in a choice of *evaluation mechanisms* .

The main evaluation mechanism is Hadoop. Pig also supports a local mode for development purposes.

Pig over Map-Reduce:

Pig runs over the top of Map-Reduce thus all the Hadoop daemons must be running before starting Pig. Grunt is the name of the shell which runs over Map-Reduce.

Hive:

It is a platform used to develop SQL type scripts to do Map-Reduce operations. Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analyzing easy. Initially Hive was developed by Facebook, later the Apache Software Foundation took it up and developed it further as an open source under the name Apache Hive. It is used by different companies. For example, Amazon uses it in Amazon Elastic Map-Reduce.

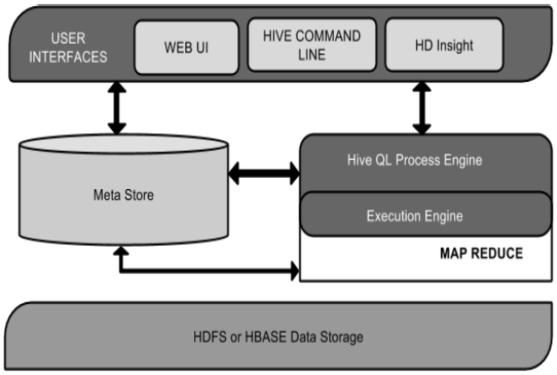


Figure.no.1.5.Architecture of Hive

	Pig	**Hive**
Type of Data	Semi Structured or structured both	Structured only.
Type of Tool	Scripting language	Query based (HiveQL)
Map-Reduce	Runs Map-Reduce in Background	Also Runs Map-Reduce in Background
Concept of data storage	Uses concept of Bags for data storage which are in turn stored over HDFS.	Uses concept of tables and databases which are in turn stored over HDFS.

Table no.(1):Comparative analysis of Hadoop and utility tools

2. Literature review

Toshimori Honjo in IEEE 2013 [1] proposed that Hardware acceleration of Map-reduce is proposed by utilizing multi core architecture of CPU. As map-reduce posses inherent parallelism and if multicore hardware can be utilized properly then acceleration can be achieved. Because sometimes number of map increases on a particular data node and CPU faces bottleneck. So to avoid this situation hardware acceleration is done in this paper.

In this paper, we conducted a performance study of Hadoop Map-Reduce algorithm. We showed that the CPU will become the bottleneck given the adoption of state-of-the-art storage and networking devices. To overcome the anticipated CPU bottleneck without significantly altering the Map-Reduce framework, we proposed a hardware acceleration approach. We implemented a prototype using a many core processor board developed by Tilera, and showed the feasibility of our proposal.

.

Ming Meng, Jing Gao, Jun-jie Chen in IEEE 2013[2] proposed that sequence alignment which is a basic method of processing information in Bioinformatics is done using Hadoop. It is used for finding sequences of proteins and nucleic acid. Most common local sequence alignment problem called BLAST is implemented in this paper. These algorithm posses some inherent parallelism so speedup is achieved when implemented on Hadoop.

Blast algorithm and the parallel experiment of the Blast algorithm which based on Hadoop, the performance of Blast algorithm has been significantly improved after parallelization. Especially when dealing with large-scale genomic data sets, the execution efficiency of the Blast algorithm which is improved on the base of Hadoop is far more than the serial Blast algorithm. In this paper, through the comparison of the improved Blast-Parallel algorithm and the Hadoop-Blast algorithm, the experiment achieves the desired goal, the matching speed of Blast-Parallel algorithm can achieves 1-1.5 times of the Hadoop-Blast algorithm.

Madhury Mohandas, Dhanya P.M in IEEE 2013[3] suggest that network failure detection system is built using Hadoop. As Hadoop have several daemons each for specific purpose. Apache Hadoop's Job tracker, Name node, Secondary Name node, Data node and task tracker all generate logs. This paper aims at building a failure monitoring system from the scratch, by parsing and analyzing the Hadoop log files generated in the cluster. The monitoring system gives all relevant details related to the application, and points out the specific reason for failure, that is, whether an application failure or a network failure (these are the most common failures in the cluster). All these daemons create logs on respective nodes. These logs can be used to detect failures in network and used for network failure monitoring system .

Job Tracker assigns task to each of the task trackers that have a free slot with it and this assignment is also based on computation near data. The Task Tracker that have a free slot and that is nearer to the data location is assigned the particular task that uses the data. A drawback of this approach is that it does not consider the work load, but only the network traffic. So a situation can arise were a node is highly loaded and others lightly loaded.

Ilja Kromonov, Pelle Jakovits, and Satish Narayana Srirama in IEEE 2014 [4] is also proposed that network failure monitoring system. To counter these drawbacks we presented a BSP-inspired parallel programming model that enables transparent stateful fault tolerance through check pointing. To validate the usefulness of the proposed model, we created a distributed computed framework, called NEWT.

NEWT supports a larger range of applications than the current BSP implementations and utilizes Hadoop YARN to perform automatic checkpoint/restart of programs.

Lena Mashayekhy, Mahyar Movahed Nejad, Daniel Grosu, Dajun Lu, Weisong Sh in IEEE [5] suggest that Energy-aware Map-Reduce Scheduling Algorithm, EMRSA, that schedules the individual tasks of a Map-Reduce application for energy efficiency while meeting the application deadline. EMRSA provides very fast solutions making it suitable for execution in real-time settings. We performed experiments on a large Hadoop cluster to determine the energy consumption of several Map-Reduce applications, and then used this data in an extensive simulation study to analyze the performance of EMRSA. The results showed that EMRSA is capable of obtaining significant energy savings compared to the standard make span minimization algorithms

As energy is a challenging issue in this era of computing. To save energy and to develop green algorithm is a big task. In this paper a new energy aware scheduling algorithm replaces traditional scheduling algorithm of Hadoop. Which saves about 40% energy?

Oscar D. Lara, Weiqiang Zhuang, and Adarsh Pannu in IEEE 2014 [6] proposed that big Data as a new framework coupling R and Hadoop to provide large scale distributed analytics. Compared to other products, Big Data offers clear usability advantages since users do not need to deal with Map-Reduce programming. Instead, Big Data extends R primitives to seeming less manipulates, visualize, and analyze big data. Moreover, any function from the extensive CRAN package repository can be pushed to the cluster via Big Data's partitioned execution. We summarize that data scientists and business analysts are hesitant to embrace Map-Reduce programming, even if the map () and reduce () operations are available in R. Most users are not interested in learning a new paradigm, especially if it involves dealing with the code parallelization. In this sense, Big Data looks appealing to the R community, since it inherits R primitives for big data analytics.

3. Proposed Methodology

We propose a technique which will take suitable features from dataset and perform the queries which are given below. At the initial stage, selected features are mapped as <key, value> pair with the help of record reader. Here key represents the group no. and the value represents the data related to the task. Each <key, value> pair is give as input to the mapper and a new <key, value> pair is generated for the further process. These data is shuffle and sorted in appropriate order and assign to the reducer. Reducer will take the intermediate key pairs and value set which will relevant to the keys. Reducer will merge these values to the small set of values. Here input for the reducer is the output of the mapper which is grouped according to their key because different mappers output may give same key. At last final output is generated from different reducer and combined the final output.

Proposed Queries on Flight Data Analysis

Pseudo code for the average delay per Airline Company is given below. Here we are selecting the feature, delay corresponding to every flight of each airline company. We can consider delay like extreme weather delay, late-arriving aircraft delay, security delay, arrival delay, departure delay, carrier delay, national aviation system delay etc. But we are considering the arrival and departure delay. Because these delay caused by most of the above given delay.

Algorithm 1.Calculate average delay per Airline Company.

```
class Mapper
    method Map(string s , integer i)
        Emit(string s, integer i)

class Reducer
    method Reduce (string s , integer [$i_1$, $i_2$, $i_3$, . . . . . $i_n$])
        sum ←0
        count ←1
forall values n ← integer[$i_1$,$i_2$, $i_3$, . . . . . $i_n$ ] do
            sum ←sum+i
            count ← count+1
```

```
avg← sum/count
end for
        Emit(string s, double avg)
```

Here we are selecting three features from the Data Set Airline Company, arrival delay and departure delay. These features are mapped in mapper code. For every flight of each airline company we are passing there total delay. Here Airline company name is mapped as key and total delay of the flight of particular Airline Company as value. Output of the map function is every flight of the Airline Company and corresponding delay as value. Then reduce will take the mapper output as the Airline company as a key and set of the delay of their flight as value. Than reducer calculate the average delay of each Airline company. So any traveler's wanted to make their trip, with the help of this query they can plan according to the flight data.

Flight data analysis using Hadoop and utility tools:-

In this paper Hadoop and all above studied utility tools are used to analyze flight data which is collected for every flight during an interval of time. It is a well-known fact that big data is a big part of the aviation industry. Each flight generates terabytes of data that requires real-time analysis to optimize flight operations, maintain safety and meet all compliance requirements. The ability to process terabytes of data in real-time, 'big fast data' is a huge competitive advantage for an airline as it leads to better service with lower operational costs. It also reduces the expenditure on storage hardware as not all data is important and by filtering it in real-time, only the relevant data can be stored.

NAME DESCRIPTION

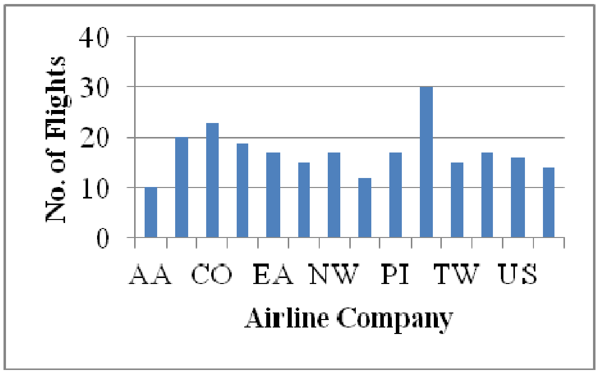
1	Year	1987-2008
2	Month	1-12
3	Day of Month	1-31
4	Day Of Week	1 (Monday) - 7 (Sunday)
5	DepTime	actual departure time (local, hhmm)
6	CRSDepTime	scheduled departure time (local, hhmm)
7	ArrTime	actual arrival time (local, hhmm)
8	CRSArrTime	scheduled arrival time (local, hhmm)
9	UniqueCarrier	unique carrier code
10	FlightNum	flight number
11	TailNum	plane tail number
12	ActualElapsedTime	in minutes
13	CRSElapsedTime	in minutes
14	AirTime	in minutes
15	ArrDelay	arrival delay, in minutes
16	DepDelay	departure delay, in minutes
17	Origin	Source airport code
18	Dest	Destination airport code
19	Distance	in miles
20	TaxiIn	taxi in time, in minutes
21	TaxiOut	taxi out time in minutes
22	Cancelled	was the flight cancelled?
23	CancellationCode	reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
24	Diverted	1 = yes, 0 = no
25	CarrierDelay	in minutes
26	WeatherDelay	in minutes

Following queries can be implemented using hadoop and tools :

- Calculate average delay per Airline company.
- Calculate maximum delay per Airline company.
- Calculate average delay per source to destination.
- Calculate maximum delay per source to destination.
- Calculate number of cancelled flights per Airline company.
- Calculate number of diverted flights per Airline company.
- Calculate maximum arrival delay per Airline company / per month.
- Calculate maximum departure delay per Airline company / per month.
- Calculate number of cancelled flights per month.
- Calculate maximum distance travelled per Airline Company.

4. Result analysis

Here we analyze the flight data using Hadoop framework which gives the different results between different attribute values. Using these values, we select better option and optimize the solution in order to select the flights. We are analyzing results between some of the important attributes here.



- Average delay per airline company
- Number of cancelled flights
- Max arrival delay per airline company
- Number of diverted fights per airline company

- Longest distance travelled per airline company
- Average traffic per month in different airports

5. Conclusions

Big data analysis trends these days are due to lots of applications areas such as world web wide, facebook, Amazon..etc. This survey focuses on Big Data analysis techniques. In this paper Hadoop, Pig and Hive have been studied and it is found that these tools are easy to use and are high level utility tools. So lots of customizations are not to be done in Mapper and Reducer and if customization is needed then Map Reduce code has to be written. Hive and Pig can be used to write above mentioned queries and Map reduce can be used to write customized and efficient code for the same.

In this survey Flight data has been analyzed in terms of the mentioned parameters such as time complexity and energy consumption information's are retrieved using Hadoop.

6. Scope of future work

The same process can be applied for different data sets like accident prediction using traffic data analysis and other utility tools like sqoop, oozie and Hbase can also be studied for map-reduce algorithm. This paper define to improve the services of flight and improve the delay of flight.

In future, we analyze the result using other distributed framework which is similar to Hadoop but much better in term of implementation speed like apache spark.

7. References

.

[1] Toshimori Honjo, Kazuki Oikawa "Hardware acceleration of Hadoop Map-Reduce" in 2013 IEEE International Conference on Big Data.

[2] Ming Meng, Jing Gao, Jun-jie Chen "Blast-Parallel: The Parallelizing Implementation Of Sequence Alignment Algorithms Based On Hadoop Platform" in 2013 6th International Conference on Biomedical Engineering and Informatics (BMEI 2013).

[3] Madhury Mohandas, Dhanya P M "An Approach for Log Analysis Based Failure Monitoring in Hadoop Cluster" in 2013 IEEE.

[4] Ilja Kromonov, Pelle Jakovits, Satish Narayana Srirama "NEWT - A Resilient BSP Framework for Iterative Algorithms on Hadoop YARN" in 2014 IEEE.

[5] Lena Mashayekhy, Mahyar Movahed Nejad, Daniel Grosu, Dajun Lu, Weisong Shi "Energy-aware Scheduling of MapReduce Jobs" in 2014 IEEE International Congress on Big Data.

[6] Oscar D. Lara, Weiqiang Zhuang, and Adarsh Pannu "Big R: Large-scale Analytics on Hadoop using R" in 2014 IEEE International Congress on Big Data

[7] Kiran M., Amresh Kumar "Verification and Validation of Parallel Support Vector Machine Algorithm based on Map-Reduce Program Model on Hadoop Cluster" in 2013 International Conference on Advanced Computing and Communication Systems (ICACCS -2013), Dec. 19 – 21, 2013, Coimbatore, INDIA