International Journal of Trend in Scientific Research and Development (IJTSRD)
International Open Access Journal
ISSN No: 2456 - 6470 | www.ijtsrd.com | Volume - 2 | Issue – 4

A Comparative Study on Apache Spark and Map Reduce with Performance Analysis Using KNN and Page Rank Algorithm

Himanshu Suhas Mone[1], **Shilpa Deshmukh**[2]
[1]Student, [2]Assistant Professor
Master of Computer Application, SIES College of Management Studies,
Nerul, Navi Mumbai, Mumbai India

ABSTRACT
With the unremitting advancement of internet, IT and enhancement of technology, tremendous growth of data has been observed. Data is getting generated at very tremendous speed, referred to as Big Data. Big Data has gained more prominence in recent times with continuous explosion of data resulting from various sources. The major focus of this paper is to compare performance between Hadoop and Spark on iterative and machine learning algorithm. Hadoop and Spark both are processing model for analysing big data and their performance varies significantly based on the use case under implementation. In this paper, we compare these two frameworks along with providing the performance analysis using a standard machine learning algorithm for classification (knn) and Page Rank algorithm.

Keywords: *Big data, Hadoop, Map Reduce, Spark, Mahout, MLib, Machine Learning, KNN, Page Rank*

I. INTRODUCTION

Big data refers to data sets or mixture of data sets whose size, complexness and rate of growth creates them terribly hard to be captured, managed, processed or analyse it using conventional tools such as relational databases within the time necessary to convert them into useful.

Characteristics of Big Data

Big data is not just about the size of the data but also based on data variety and data velocity. These Five V's are very important to identify Big Data andthey are :

1) **Volume:** Big data denotes huge amount of data Involved. Data is rising exponentially. Data volumes are expected to grow more than 300 times by 2020 [10].
2) **Variety:** Variety refers to the type of data. Data can be of any type that is structured, semi-structured and unstructured [10]. On twitter 400 million tweets are sent per day and there are 200 million active users on it. [11]
3) **Velocity:** Velocity is the speed at which data is generated and processed.
4) **Value:** This is very important V of big data. The problem is how to extract useful data. You have to make sure that analysis you have done it is of some value, it will help your business to grow.
5) **Veracity:** Big data has a lot of inconsistencies when you are dumping such huge amount of data some data package are bound to lose in process such as data may be incorrect and missing.

Map Reduce

Apache Hadoop [1] is an open source framework. It provides solutions for handling big data along with extensive processing and analysis. It is used for storing and processing Big Data in a distributed manner on large clusters of commodity hardware. Hadoop is licensed under the Apache license 2.0.

It was created by Doug Cutting and Mike Cafarella in 2005 when Doug was working for Yahoo at the time for the Nutch search engine project. Hadoop has two major components named HDFS (Hadoop distributed file system)[2] and the Map Reduce[3] framework.

Hadoop Distributed File System is inspired by Google's File System (GFS) [4] and provides scalable, efficient and replica based storage of data at various nodes that form a part of a cluster.

HDFS is a master slave architecture where 'name node' is the master and 'data nodes' are slave nodes where actual or replicated data presents. The second component is Map Reduce. It is a programming model of Hadoop, this allows a parallel and distributed processing and generating large data sets. A map function processes key/value pairs and gives a result into a set of intermediate key/value pairs and a reduce function that combine and aggregate data to find the result to primary problem statement.[3].

Apache Spark

Although recently, the world of Big Data has seen a dynamic shift from this computing model with the introduction and stable release of Apache Spark [5], that provides a user friendly programming interface to lessen coding effects and gives better performance in a most of cases with issues related to big data. Spark not simply provides an alternative to Map Reduce, but also provides shark which is like sql querying and machine learning library referred as MLib. The performance and working of spark is considerably different from map reduce.

Apache Spark [6] started as a research project at UC Berkeley in AMP Lab. It was started with an aim to build a programming model which will support a much wider class of applications than Map Reduce, while maintaining its automatic fault tolerance.

Spark offers an abstraction referred to as Resilient Distributed Datasets (RDDs)[8] to support these type of applications efficiently. RDDs can be stored in memory without requiring replication. They reconstruct missing data on failure using lineage, whereas all RDD knows how it was built from other datasets to rebuild itself. RDDs permits spark to exceed existing models by up to 100x faster in multiple pass analytics. RDDs can support a wide variety of iterative algorithms and interactive data mining and a highly efficient SQL engine Shark [9]

Spark was introduced by Apache Software Foundation for speeding up the Hadoop processes. Spark is not a modified version and reliant on Hadoop because it has its own cluster management. Hadoop is simply one of the way to implement spark. Spark uses Hadoop in two ways– one is storage and second is for processing. Since spark has its own cluster management computation, it uses Hadoop only for storage purpose. The main feature of Spark is in-memory cluster computing that improves the processing speed of an application.

Spark is intended to process batch applications, iterative algorithms, interactive queries and streaming.

II. DIFFERENCE BETWEEN APACHE SPARK AND MAP REDUCE

1. Introduction –

Apache Spark – Apache Spark is an open source big data framework which provides faster data processing engine. Spark is specifically build for faster computation and can processes data batch, interactive, iterative and streaming mode.

Hadoop Map Reduce – Map Reduce is an open source framework for analysing large datasets. It also processes structured and unstructured data. Map Reduce can process data in batch mode.

2. Speed –

Apache Spark – Spark is lightning fast computing tool than Map Reduce which runs programs 100x faster in memory and 10x faster on disk. Because it reduces the number of read and write operations to disk and storing intermediate data in-memory Spark.

Hadoop Map Reduce – It slows down the processing speed because it reads and writes from disk.

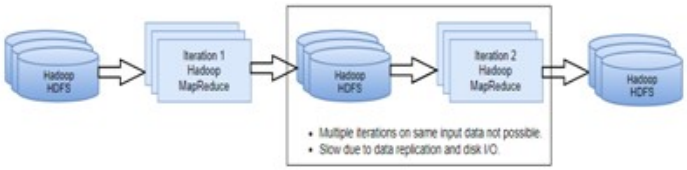


Fig1.MapReduce

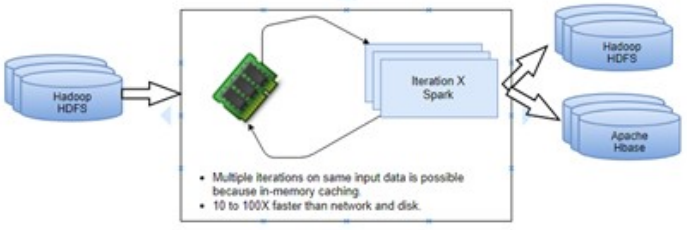


Fig 2. Apache Spark

3. Real-time analysis –

Apache Spark – It processes **real-time data** i.e. data coming from the real-time applications at the rate of millions of events per second like Twitter data for instance or Face book sharing, posting & stock market etc. Spark's one of the big strength is the ability to process live streams efficiently.

Hadoop Map Reduce – Map Reduce disappoints when it comes to real-time data processing as it was build to execute batch processing on immense amounts of data.

4. Easy to Manage-

Apache Spark – It is a complete data analytic engine because spark is capable of performing batch, interactive, streaming all in the same cluster. Therefore no need to manage various component for various needs. Spark on a cluster will be enough to handle all the requirements.

Hadoop Map Reduce – As Map Reduce only provides the batch engine. Hence, we are dependent on different engines. For example- Storm, Giraph, Impala, etc. for other requirements. So, it is very difficult to manage many components.

III. PERFORMANCE EVALUATION

3.1 Machine Learning and KNN

1) Machine Learning Introduction

Machine Learning is an important branch of artificial intelligence that enable computers to grasp new patterns and instructions from data rather than being explicitly coded by developer. Machine learning permits system to enhance themselves based on new data that is added and to generate more efficient new data. [12]

2) **Knn algorithm**

K-Nearest Neighbour (KNN) is a classification algorithm which is used for Machine Learning. It comes under supervised learning. It is mostly used in the solution of classification problems in the industry. It is mostly used in pattern recognition, data mining. It stores all the available cases from training dataset and classifies the new cases based on distance function.

1. Load the data
2. Initialize the value of k
3. Iterate from 1 to total number of training data points for getting the predicted class,
   3.1 Calculate the distance between test data and each row of training data. For this we will use Euclidean distance
   3.2 Sort the calculated distances in ascending order based on distance values
   3.3 Get top k rows from the sorted array
   3.4 Get the most frequent class of these rows
   3.5 Return the predicted class

A. Methodology

In this paper, a systematic evaluation of Hadoop Map Reduce is done and its performance is compared with another Big Data framework that is Apache Spark. For this purpose, we performed a comparative analysis using these frameworks on a dataset that allows us to perform classification using Knn algorithm.

B. Experimental Setup

Dataset Description

In This example we have use K nearest neighbor classification method which comes under supervised machine learning and is applied to some sample data about car types and buyer characteristics, so that it classifies a buyer with a likely car model.

A sample of the data records are shown as below. The data record contains:

The data record contains:

1. Age
2. Income
3. Status
4. Gender
5. Children

Sample record:

67, 16668, Married, Male, 3

Tests were conducted on Hadoop and Apache Spark. Hadoop is built on a single node having Intel-core i5 processor with 4GBs of Ram 64-bit architecture. The operating system is Linux (Ubuntu 14.04). To benchmark the performance, the stable release of Hadoop and Spark namely Hadoop - 2.9.0 and Spark - 2.3.0 were chosen.

C. Results

The purpose of this study was to evaluate the Map Reduce performance with Spark performance under the same setup for knn algorithm. The teste were conducted for various datasets having different sizes

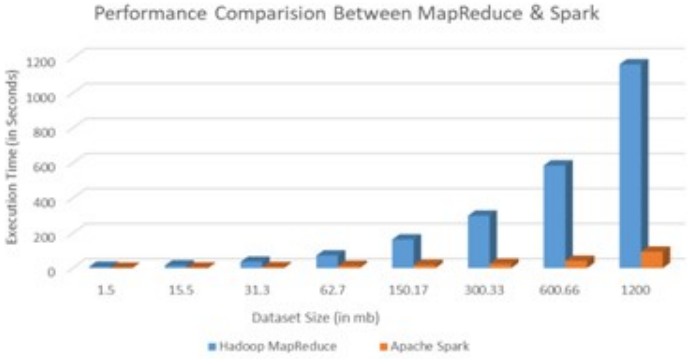
ranging from 1.5 MB(approx) to 1.2 GB (approx). Map Reduce job and Apache Spark job are run on these datasets one by one to get the desired output. The analysed result shows that it classifies buyer with a likely car model and time to get the desired result. The results can vary system to system.

The Performance Ratio can be calculated as:

$$PR = \frac{\text{Running time of data size in Map reduce}}{\text{Running time of data size in Spark}}$$

Dataset size (in MB)	Execution Time in (sec)		Performance Ratio
	Knn using Map Reduce (Mahout)	Knn using Spark (MLib)	
5	7.43	1.53	4.85
15.5	14.98	2.90	5.16
31.3	35.93	3.40	10.56
62.7	72.06	12.38	5.82
150.17	163.28	17:85	9.14
300.33	299:09	24:06	12.43
600.66	586.33	41:52	14.12
1200	1163.45	93.36	12.46

The results clearly show that the performance of Spark is considerably higher in terms of time.



The above figure represents the performance comparison of both the models in graphical manner. The figure clearly displays that Apache Spark is far better than Map Reduce.

3.1 Iterative Algorithm Page Rank

PageRank is selected to show the comparison between Hadoop and Spark because of the following reasons:

1. The implementation of Page Rank algorithm is involved in multiple iterations of computation.
2. In Hadoop, Map Reduce always writes immediate data back to HDFS after a map or reduce action for each iteration computation.
3. Spark processes immediate data in an in-memory cache. In this case, for such a particular application, it is clearly expected that Spark would completely overwhelm Hadoop on performance.

A. Algorithm Description

This paper primarily focus on the comparison of performance between spark and Hadoop, a standard Page Rank algorithm will be used.

Page Rank appears along with the development of web search engines. It is considered as the probability that a user, who is given a random page and clicks links at random all the time, eventually gets bored and jumps to another page at random[15]. As result, it is used to calculate a quality ranking for each page in the link structure of the web, and thus improves the precision of searching results. This is the way that Google searching engine evaluates the quality of web pages.

Basically, the core idea of Page Rank is as follows [15]:

1. If a page is linked by a large number of other pages, it is much more important than a page linked by a few others, and this page also owns higher rank value;
2. If a page is linked by another page with higher rank value, its rank value is improved;
3. The final goal of this algorithm is to find stable ranks for all of links after multiple iterations.

In this case study, a Page Rank value will be calculated by the following formula [15], which was proposed by the Google founders Brin and Page in 1998:

$$R_i = d * \sum_{j \in S} (R_j / N_j) + (1 - d)$$

R_i= The Page Rank value of the link i

R_j= The Page Rank value of the link j

N_j= The number of outgoing links of link j pointing to its neighbour links

S = the set of links that point to the link i

d = The dumping factor (usually d =0.85)

The sample datasets are used to evaluate the performance of the Page Rank application respectively running in Hadoop and Spark. All of the data source is from http://snap.stanford.edu/data

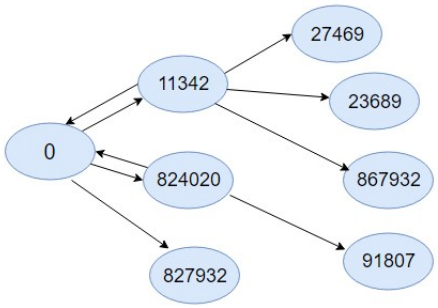
B. Experimental Setup

Dataset Description

The Data Sample shown below is from Web-Google.txt. The left column represents starting link points, and the right column is the ending link point

From LinkId ToLinkId

0	11342
0	824020
0	867932
11342	0
11342	27469
11342	23689
11342	867932
824020	0
824020	91807



After the first iteration result will be:

```
The PageRank value of link ID 0:
0.15 + 0.85(1.0/4 + 1.0/2) = 0.7875

Link ID 11342:  4 outgoing links
Link ID 824020: 2 outgoing links
Link ID 827932: No outgoing
links, which means no contribution to
other links. So, here ignore it.
```

Dataset for Page Rank Example

File Name	Size (MB)	Nodes	Edges
Web-Stanford	31.3	281,903	2,312,497
Web-Google	71.8	875,713	5,105,039
Web-BerkStan	105	685,230	7,600,595

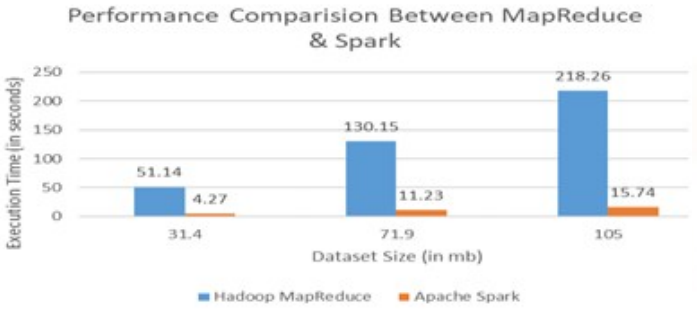
The Performance Ratio can be calculated as:

$$PR = \frac{\text{Running time of data size in Map reduce}}{\text{Running time of data size in Spark}}$$

C. Results

The results can vary system to system.

File Name	Size (MB)	Execution time in sec		Performance Ratio (PR)
		Map Reduce	Spark	
Web-Stanford	31.3	51.14	5.82	8.78
Web-Google	71.8	130.15	11.23	11.58
Web-Berks tan	105	218.26	19.74	11.05



The performance of Spark outperforms Hadoop because as discuss earlier, Spark utilizes memory-based storage for RDDs but Map Reduce processes disk-based operations.

However Spark allows to limit the memory usage of each executor by assigning spark. Executor. Memory.

The memory usage limit is varied between 1 and 3GB on each executor, comparable results are listed as follows:

Data Usage Size Memory (GB)	Web-Stanford 31.3 MB	Web-Google 71.8 MB	Web-Berk Stan 105 MB
1	5.82	11.23	19.74
2	7.18	10.56	16.23
3	6.51	9.37	14.11

Spark still performs better than Hadoop based on same memory usage.

The table clearly displays that

1. For small size of data or fewer iterations, increasing memory does not contribute to the improvement of performance.
2. As the growth of data size and multiple iterations are executed, then there is a significant performance improvement with increasing memory usage.

Map Reduce is not suitable for iterative processing. It is designed for batch processing of data and even if it do iterative processing it will be very slow because the results of each iteration will be written to disk and then read again (even on the same node) in the next iteration. Apache Spark is specifically build for iterative processing and can scale up and out. We believe that's what you need.

IV. CONCLUSION

In this paper two programming model Map Reduce and Apache Spark has been presented for analysing their performance. Each model has its own advantages and disadvantages. By employing this comparative study, concluded that Spark has better performance in terms of execution times as compared to Map Reduce.

Spark is generally faster than Hadoop because of in-memory computation. But Spark is not a good fit for if we do not have sufficient memory and the speed is not demanding requirement, Hadoop is a better choice. Spark is sure to be best fit for applications which are time sensitive or involved in iterative algorithms and there is abundant memory available.

V. REFERENCES

1. Apache Hadoop Documentation 2018- https://hadoop.apache.org/
2. Shvachko K., Hairong Kuang Radia S, Chansler, R The Hadoop Distributed File System Mass Storage Systems and Technologies (MSST), 2010 IEEE.
3. Jeffrey Dean and Sanjay Ghemawat. Map Reduce: Simplified data processing on large clusters. In OSDI'04 : Sixth Symposim on Operating System Design and Implementation, 2004.
4. Sanjay Ghemawat, Howard Gobioff and Shun-Tak Leung. The Google file system. 2003
5. Apache Spark Documentation 2016 https://spark.apache.org/documentation.html
6. Apache Spark Research 2018- https://spark.apache.org/research.html
7. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. Franklin, S. Shenkar and I. Stoica. Resilient distributed datasates: A fault-tolerant abstraction for in memory cluster computing. Technical Report UCB/EECS-2011-82, EECS Department, University of California, Berkeley, 2011
8. Reynold Xin, Joshua Rosen, Matei Zaharia, Michael J. Franklin, Scott Shenkar, Ian Stocia. Shark: SQL and Rich Analytics at Scale. SIGMOD 2013 june 2013.
9. SMITHA T. V. Suresh Kumar "Application of Big Data in Data Mining" in International Journal of Engineering Technology and Advanced Engineering Volume 3, Issue 7, July 2013
10. IBM Big Data Analytics HUB.
11. Machine Learning, Wikipedia 2016 https://en.wikipedia.org/wiki/Machine_learning
12. "Hadoop in Action" by Chuck Lam.
13. White, Tom, 2011. "Hadoop the Definitive Guide" O' Reilly media, Inc., CA
14. M. Bianchini, M. Gori and F. Scarselli, "Inside PageRank" ACM Trans. Inter. Tech. TOIT ACM Transactions on Internet Technology, pp. 92-128, 2005