Big Data Analysis with Dataset Scaling in Yet another Resource Negotiator (YARN)

Gurpreet Singh Bedi
M.tech Student
Department of Computer Science Engineering
Thapar University, Patiala, India

Ashima Singh
Assistant Professor
Department of Computer Science Engineering
Thapar University, Patiala, India

ABSTRACT

The data is exceedingly large day by day. In some organizations, there is a need to analyze and process the gigantic data. This is a big data problem often faced by these organizations. It is not possible for single machine to handle that data. So we have used Apache Hadoop Distributed File System (HDFS) for storage and analysis. This paper shows experimental work done on the MapReduce Application on Health sector dataset. The result shows the behavior of the MapReduce application framework to map and reduce the big volume of the data. The main problem is to check the behavior of the MapReduce applications by increasing the size of dataset. Our analysis lies in understanding the Apache MapReduce application performance. We expect that execution time increases linearly with the dataset size but our analysis shows sometimes the execution time varies non-linearly with the increase in the dataset size. The experimental result shows that with scaling the datasets execution time distinguishes.

Keywords

Big Data, Hadoop, MapReduce, YARN, Single Node, Multi Node, Dataset Scaling.

1. INTRODUCTION TO BIG DATA

A human being is a kind of device that generates information with a large value. A person is able to generate terabytes of information in a day, millions of data generated could be used in various fields like health centers. Data is growing at phenomenal rate make it difficult to handle the data which is in petabytes. The main problem is that handling of the data is difficult as volume is increasing with the increase of the resources. It can be measured in the many ways variety, velocity, variability, complexity and the volume-

A) Variety- As the data is a traditional data or semi structured data like web pages, documents, email social networking etc. Unstructured data is very difficult to analyze and it is very difficult to handle. So a variety of data comes into picture like audio, video [1]. For example check the live records from surveillance cameras to target a particular point.

B) Volume- Big word means the big store. The data that today exists is in petabytes and it is supposed to be increased in zettabytes in the coming future. Social networking also continuously increasing the data in terabytes and that data is very easy to handle by using existing Big Data Technology. For example analyses of 12 terabytes of tweets created every day. To estimate power consumption of a particular area, conversion of 350 billion annual meter reading. So in this way big volume of data is one of the most important dimensions of Big Data.

C) Velocity- Velocity deals with the speed of the data that generates from the various resources. This doesn't limited to the speed of incoming data only but also the speed of the flow of the data. For example the data moves from sensor devices to the vast databases, so to handle the data which large enough our traditional system is not enough. Sometimes a minute late becomes too late to handle critical cases like fraud detection, scrutiny of millions of events to catch up the fraud activity. So in time sensitive data velocity is needed to prevent the delays because minute can led to the failure of the event. Big data also analyze millions of call records to check particular customer's data.

D) Variability- Variability means inconsistencies in the flow of the data. Data load is more critical as the data load increases or decreases with the usage of the social media applications.

E) Complexity- It is complex to transform the data coming from the various data sources. It is necessary to correlate the data.

F) Value- User run the queries and deduces the result by filtering the data according to their condition. The results help the people to find the latest trends by analyzing the market strategy. The main challenge that comes is to design such system that would be capable to handle the large amount of the data. The second challenge is to filet the data comes from the sources.

2. BIG DATA AND APACHE HADOOP

To build a search engine index was a difficult process as it was needed to process millions of the web pages. In the search engine the user asks a phrase and the search engine finds the best match in the billions of the web pages. Actually mapping is there by the Google to map the terms in the advance. This follows the Google's PageRank algorithm. Around 2000's Google invented the server architecture to implement the PageRank. By 2006, they became part of a separate Apache project, called Hadoop [2], the name comes from a stuffed toy which is an elephant, the Cutting's son owned at the time. MapReduce works by splitting the data into phases – map phase and reduce phase. Each phase of MapReduce with the input and output are in key/value pairs. In 2008, Hadoop broke the world record to sort terabytes of the data in less amount of time.

3. DATA ANALYSIS AND HADOOP

To handle the data in short time, new tools are being used. Apache Hadoop is one of such tool which is a java based framework for analyzing large amount of the data [3]. Hadoop is basically for processing large amount of data by using multiple nodes. Apache hadoop uses its own algorithm to break down the big data into smaller parts called chunks [4] and performs various operations on it and algorithm is called MapReduce [9]. Hadoop uses HDFS which is its own file system, helps fast transfer and avoid failures. Hadoop is used by many companies like yahoo, Google, IBM for storing and analyses of large amount of the data emphasis on efficiency and throughput.

4. YET ANOTHER RESOURCE NEGOTIATOR (YARN)

Yahoo started on Apache Hadoop framework in the year 2006. This replaces the WebMap Application [3] this was the technology that builds the graph of the web to index the search engine contents. At that time, nearly 100 billion nodes were processed by the graph with trillions of the edges. Dreadnaught , a framework reached the capacity of working at nearly 1000 machines. Dreadnaught applications resembled with the MapReduce programs [5], So Scalability arises as the number one requirement in Yarn. The second was the multi-tenancy, means optimizing the analytics. The early beginning of hadoop bring large number of nodes, and load their data onto the filesystem provided by hadoop HDFS and obtain the final results by processing the nodes. At YAHOO users load datasets that were adhoc datasets. To address the issues, YAHOO developed Hadoop on Demand(HOD) framework to allocate the clusters to manage independently MapReduce and the HDFS [6]. Torque and Maui was one of the framework where users submit their jobs to Torque and the Torque enqueue the job and waits. It enqueues until the nodes become available and ones it becomes available torque starts the process on HOD, interact with maui/torque to start the slave process which spawned the JobTracker to accept the jobs. Serviceability was the main issue. Yahoo finally drew back HOD due to lack of resource utilization. During the mapping phase, JobTracker makes excessive effort to place the tasks close to their input data in HDFS framework [7], mainly on the node stores the replicas of that data, but the torque allocates the nodes with concerning locality, the granted nodes contain only small amount of replicas. Locality awareness is the requirement which becomes the bases of Yarn. In another scenario, task running on a single node prevent the cluster from reuse. So jobs that held hundreds of nodes remain idle. Now the other priority in Yarn is resource utilization. The response time was overlooked by the time that spent on the allocation. So this makes the users to work on the shared clusters as the latency was high and the users offer awaited clusters with their partners. Hadoop On Demand (HOD) had little information to decide the allocation process. API gives the users harsh results. The clusters became larger and throughput has increased and also results in increase in the bugs. Many features added to the jobtracker. This results in the loss of all running jobs instead of losing a single workflow. It mainly requires the users to manually recover. It results into the pressure on the jobtracker which creates a backlog in pipelines. Restarts lead to kill the users mapreduce jobs until cluster recovered. Operating the cluster becomes hard. Availability issues rises, because the JobTracker allocates its structures for the job it initializes, its admission control logic protects its own availability; it may delay allocating the cluster resources to the jobs because the overhead of tracking them could overpower the JobTracker process. As Hadoop managed more tenants, vast datasets, its requirements for the isolation became more rigorous, but the authorization model lacked strong, scalable authentication, a feature for multitenant clusters. This was added to the multiple versions. Secure and auditable operation must be preserved in YARN. Developers gradually hardened the system to accommodate diverse needs for resources. While MapReduce supports a wide range of use cases, it is not the ideal model for all large-scale computation. For example to come to a result many machine learning programs need multiple iterations corresponding to a dataset. If certain program follow this sequence of jobs this will delay the overall execution time of the result. Now this delays the progress to user's productivity. The separation between the both map and reduce prevents the deadlock between the two. However, it can also responsible for the reduction in the flow of the process. In Apache Hadoop, the overlap between the map and reduce job is configured by the user for the job being submitted, starting the tasks later increases the throughput, starting early them reduces the latency [11]. The number of the map and reduce job's slots are however fixed by the operators so map can't be used to spawn the reduce tasks or vice versa. No configuration is perfectly balanced as both map and reduce tasks complete at definite rates respectively. So a flexible model is needed to serve the purpose. The shared clusters increases the utilization as compared with the Hadoop On Demand (HOD), it also fulfills the demand of serviceability and availability. To fix the issue in the MapReduce, operators first shut down the cluster, place the new bits to fix the bug, validate the upgrade and finally admit the jobs. Thus, to upgrade the cluster makes the user to wait, validate and then perform the jobs. The next requirement is the backward compatibility with the existing system. YARN is also backward compatible with the previous versions.

4.1. YARN Applications

YAHOO updates its processing flow from basic Hadoop framework to YARN. The experiment prior to upgrading the basic hadoop cluster and after upgrading the hadoop cluster to YARN is performed. After the up gradation to Yarn, Yarn is approximately processing five lakh jobs daily. The storage reportedly increased by 350 petabytes. Yahoo survey reported that prior to Yarn they don't even run clusters bigger than 4000 nodes. The main aim of YARN was to increase the scalability and improve the resource utilization. But this is limited to near 7000 nodes and can't scale further. The YARN component Log aggregation has increase the overhead of the Hadoop FileSytem Namenode. But the improvement is going on to increase the namenode throughput. Also the main issue is observed in the Hadoop framework with large amount of small applications. These issues will be yet to be solved. Apache Yarn, which gives a resource management framework has much more capability then the simple MapReduce. Apache Hadoop continues to go beyond the mapreduce. However, MapReduce is still the core of many tasks performed on Hadoop 1.x. Apache yarn has just increase the capability to grow beyond the simple MapReduce. The basic architecture of Hadoop 1.x. Mainly there are only two cores, Hadoop file system and the MapReduce which form the basis of all the process to be done. The remaining components must use MapReduce for the jobs to be performed. So Apache Hadoop provides the functionality for processing and has a system of various tools, vendors and the applications. Mapreduce limit the needs. To fulfill the needs, Hadoop was started be the Hadoop team to run the jobs even the non-Mapreduce within the same framework. MapReduce version 1(MRv1) is rearchitected and called MapReduce version 2(MRv2). YARN in addition to providing functionality to existing MapReduce Applications, it also provides a new support to the distributed application. It doesn't change the capability of existing MapReduce jobs. The new Yarn works the same with addition capability. For exploring the large datasets, Pig scripting language is used [12]. There is an issue in mapreduce that development cycle is large. The whole job includes map, reduce, shuffle, compiling and packaging the code, job submission and retrieving the final results is very time consuming process. Pig has the ability to process terabytes of the data with nearly half dozen lines of code. It was developed by Yahoo for mining the large datasets.

4.2. YARN Process

Yarn has new components with additional facilities. These components offer more capabilities and are more beneficial t the end user. Resource Manager and Application Master are responsible for the execution of the YARN Process [14].

a) Resource Manager (RM)- ResourceManager is a scheduler which is also called a pure scheduler for the available resources. It keeps the resources in use every time so it is suitable for cluster utilization against a number of constraints. For other policy constraints pluggable scheduler is used by the ResourceManager that uses various algorithms like capacity and fair scheduler.

b) Application Master (AM) - The important component in YARN is the Application Master. The Application Master is, an instance of the library and it is responsible for taking the resources from the ResourceManager. It also works with the NodeManager to execute and monitor the containers and their resource consumption. It tracks their status and monitoring progress. The ApplicationMaster design provides the number of features including scalability as the Application Master provides the functionality of the traditional ResourceManager so that the system can scale more dramatically. Jobs simulations scaling to thousands of node clusters doesn't give a specific issue. Resource Manager as a pure scheduler does not have to provide fault-tolerance for the various resources across the cluster. Control becomes local and not global, by shifting fault tolerance to ApplicationMaster instance. Every application in hadoop has its own instance of ApplicationMaster. However, it's feasible to implement an ApplicationMaster to manage the set of applications like ApplicationMaster for Pig, or Application Master of Hive to manage the set of MapReduce jobs.

4.3. YARN comparison with MapReduce

The main aim of YARN is to split the responsibilities of the JobTracker [15]. The responsibility of resource management and information monitoring is divided into separate chunks, a ResouceManager and an ApplicationMaster for each application. ResourceManager and the slave per node, NodeManager makes a new system for managing the applications in the distributed manner. The main authority is the ResourceManager which arbitrates the resources in the system. Application master per application is the framewok entity works with the NodeManager to monitor the tasks.

The Need for Non-MapReduce Workloads- MapReduce is efficient strategy for many applications but it is not suitable for everything. Many other programming models serves the requirements like graph processing. Google Pregel and Apache Giraph are one of the examples that serve graph processing. MapReduce is basically for batch processing but the real time processing is the main issue. A more robust framework is needed so that organizations can face an increased return on the investment. The processing power has also increasing vastly. Apache Mapreduce is efficient for operation upto 5000 nodes. In the current scenario, nodes are managed by the TaskTrackers and JobTracker. JobTreacker view the cluster as a group of nodes with distinct number of map and reduce slots. Now the utilization issues may occur if map slots are full while the reduce slots are empty or it could be vice versa. NodeManager act as a slave, which launches the applications containers, and also responsible to monitor the resource usage like memory, disk, CPU, network and this reports the same to Resource Manager [13]. Application Master is responsible for the arrangement of the relevant resource containers from the scheduler. It also tracks the status and monitors the progress of the MapReduce job. So an Application Master itself runs as a normal container. One of the main things within the new system Yarn is that MapReduce reuse the existing framework without any major changes. This step was very needed to ensure compatibility for existing MapReduce applications and users. MapReduce is paired with the Hadoop Distributed File System (HDFS) to provide a high bandwidth for the large clusters. The main aim of Hadoop is to move the tasks to the servers on which the data resides so the data movement to compute the servers is reduced. Mainly, MapReduce tasks can be put on the same node on which the data resides in HDFS. So this design keeps the Input/Output on the local disk or on a neighboring server on the same rack.

5. DATASET SCALING AND PERFORMANCE

Each map executes with the same duration with the increase in the input dataset but map execution time sometimes differs with the difference in the applications [16].

A. Scaling K- of Data -We can change the dataset size in the MapReduce tasks therefore we refer the increase in the dataset size as k-scaling which means enlarging the dataset size by k times.

B. K-Scaling and the Map Stage- K scaling means launching more map tasks i.e. increasing by k-factor. Map job normally present in the map slots, containers for amp tasks in java virtual machines. The number of slots limits the number of tasks that can run concurrently.

C. K-Scaling and the Reduce Stage- Reduce tasks has its own slots i.e. input data partition. This is locally stored. When an input data get scaled by k times which leads to the reduce stage to do the k times of the prescaling. Since the reduce tasks is fixed, therefore reduce tasks execution doesn't get changed at all.

5.1. Experiment Setup

For experiment setup, Apache Hadoop 2.2 has installed [17]. Hadoop cluster has then used with Ubunto 12.04, java 7 jdk and VMware workstation 8.0.0. First step is to import the data from data warehouse. Health Sector dataset has used which is in comma separated value (csv) format [18]. It has then scaled upto 100MB, 200MB and 400MB using random function in Microsoft excel. Next step is to feed the data into hadoop cluster and run it using java program which is made for the purpose to map the rows one by one in key value format and then to reduce the dataset row by row.

5.1.1. Experiment with increase in the nodes

First experiment has conducted to check the behavior of MapReduce Program with the increase in the number of nodes. So for this virtual machine workstation has used. Ubunto 12.04 has cloned 3 times for the purpose to increase the number of the nodes so to make one node as a master node whose task is to give the job to other slave nodes. The behavior of MapReduce application has then analyzed.

Table 1. Experiment Setup

Nodes	1	2	3
Database Size Used (MB)	100, 200, 300, 400	100	100

Experiment with the increase in the number of nodes has shown in Figure 1. Size of dataset used = 100 Mb. Following figure shows as number of nodes increases, the execution time decreases.

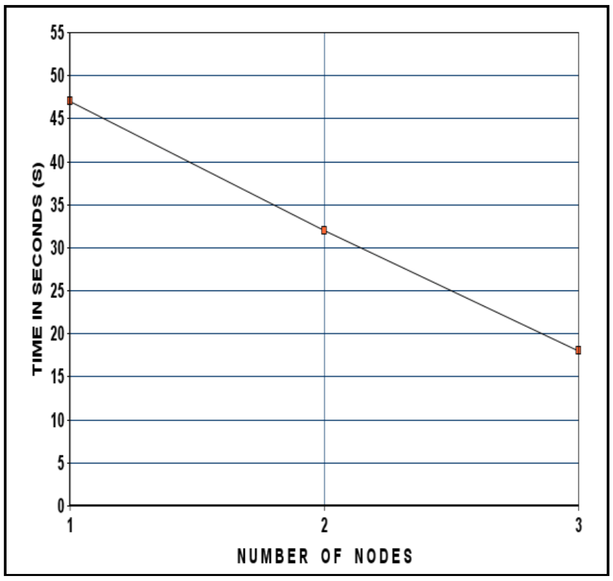

Figure 1: Execution time vs. Number of nodes

5.1.2. Experiment with the variation in the dataset size

Experiment with the increase in the number of nodes. Size of dataset used = 100 Mb, 200Mb, 300Mb and 400Mb. Following figure shows increase in the execution time with the increase in the dataset size.

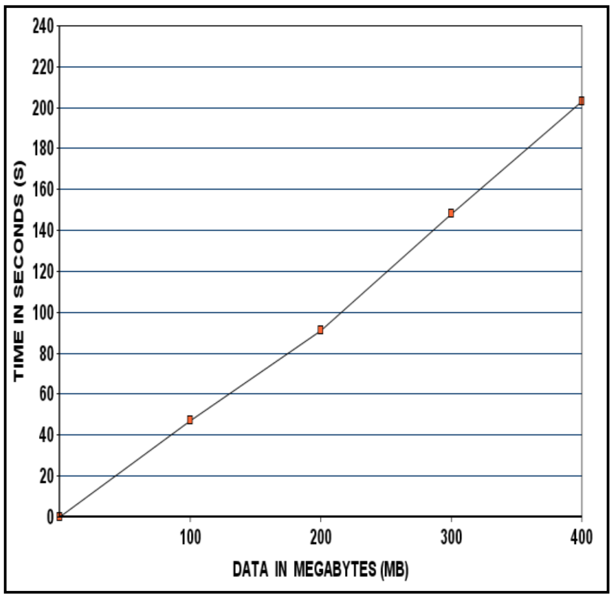

Figure 2: Execution time vs. Dataset size

Results- The above figure shows the relationship between execution time and the dataset size. We have used 100, 200, 300 and 400 MB of the dataset. So the actual relationship between dataset size and the time execution has shown in the graph. This shows the behavior of MapReduce application with the increase in the dataset size. So it shows the impact of increasing the input size. We expect that the mapreduce works linearly with the increase in the dataset but the results show something different behavior. Slight non-linear behavior with increase in the dataset size. To verify the result, we identify the map and reduce intensive programs. In this way, we anaylze the behavior of Apache MapReduce application with the dataset.

6. CONCLUSION

In this paper, best solution to Big Data by using Apache Hadoop Yet Another Resource Negotiator (YARN) using MapReduce programming framework which is based on java is examined. The main aim was to analyze the big volume of the Hospital data. The Hospital ratings with the large dataset are also examined. It was done to observe the best hospital where the patients got the recommended treatment to prevent the blood clots on the day they admitted to the hospitals. The solution has been proposed by using MapReduce which estimated the overall ratings of the hospital. The graph has been drawn to check the behavior of MapReduce application. First experiment was carried to check the behavior between execution time and scalability of the data. The second experiment was done to check the execution time of MapReduce paradigm with the single node and multi node cluster of Apache Hadoop. Non-linear relationship between the time and the dataset size is observed. In this, it has examined the behavior of the MapReduce Application and have considered the homogeneous Big Data as an input and have done the analysis on that part.

7. REFERENCES

[1] Big data with the three dimensions: Volume, Velocity and Variety". Available at: http://www-01.ibm.com/software/in/data/bigdata

[2] Tom White, "Hadoop: The Definitive Guide", O'Reilly Media 3rd edition, pp. 9-12.

[3] Jaliya Ekanayake, Shrideep Pallickara, and Geoffrey Fox "MapReduce for Data Intensive Scientific Analyses" in Fourth IEEE International Conference on eScience, 2008.

[4] Impetus white paper,"Planning Hadoop Projects for 2011", Available at: http://www.techrepublic.com/whitepapers/planninghadoopnosql-projects-for-201/292 3717,March, 2011.

[5] Tom White, "Hadoop: The Definitive Guide", 3rd edition, O'Reilly Media, pp. 41-82.

[6] Saumitra Vaidya, Jyoti Nandimath, Ankur Patil, "Big Data Analysis Using Apache Hadoop" in IEEE IRI 2013, California, USA, August 14-16, 2013.

[7] "The Hadoop Architecture and Design", Hadoop official website. http://www.apache.org/common/docs/r0.16.4/hdfs_d esign.html.

[8] P.Narayan, C.Neerdaels, T.Negrin, Ramakrishnan, U.Srivastava, "Building cloud for Yahoo" in IEEE Data Eng. Bull, page 36-42.

[9] Karthik Kambatla, Naresh Rapolu, Suresh Jagannathan, Ananth Grama, "Asynchronous Algorithms in

MapReduce", in 2010 IEEE International Conference on Cluster Computing.

[10] J.Li, C.Dyer, J.First "Data Intensive Text Processing With MapReduce", in Morgan Publishers, April 30,2010

[11] Aditya B. Patel, Manashvi Birla, Ushma Nair, "Addressing Big Data Problem Using Hadoop and Map Reduce", IEEE International Conference on Engineering, Dec, 2012.

[12] Karen Montgomery, "Big Data Now", 2nd edition, O'Reilly Media, page 83-93, 2012.

[13] Avita Katal, Mohammad Wazid, R H Goudar, "Big Data: Issues, Challenges, Tools and Good Practices", in IEEE 2013 conference in Graphic Era University, Dehradun, India, 2013.

[14] "Apache Hadoop YARN: Yet Another Resource Negotiator" in Santa Clara, California, USA, in October, 2013, ACM Publications.

[15] Arun Murthy, Jeffrey Markham, Vinod Vavilapalli, Doug Eadline, "Moving Beyond MapReduce and Batch Processing with Apache Hadoop 2", Addison Welson and Data Analytic Series, 2013.

[16] Fan Zhang, Majd Sakr, "Data Scaling And Map Reduce Performance", in 2013 IEEE International Conference on Parallel & Distributed Processing Workshops and Phd Forum.

[17] "Apache Mapreduce Setup", Available at: http:www. hadoop.apache.org/mapreduce.

[18] "Health Dataset". Available at: http://www.healthdata.g ov/dataset