Hadoop Cluster Analysis and Assessment

Saliha KESKİN[1], Atilla ERGÜZEN[2]
[1](*Department of Computer Engineering, Kırıkkale University, Turkey)*

ABSTRACT:*Large amount of data are produced daily from various fields such as science, economics, engineering and health. The main challenge of pervasive computing is to store and analyze large amount of data.This has led to the need for usable and scalable data applications and storage clusters. In this article, we examine the hadoop architecture developed to deal with these problems. The Hadoop architecture consists of the Hadoop Distributed File System (HDFS) and Mapreduce programming model, which enables storage and computation on a set of commodity computers. In this study, a Hadoop cluster consisting of four nodes was created.Regarding the data size and cluster size, Pi and Grep MapReduce applications, which show the effect of different data sizes and number of nodes in the cluster, have been made and their results examined.*

KEYWORDS -*Bigdata, Hadoop, MapReduce, HDFS*

I. INTRODUCTION

We are in a Big Data era where large amounts of data are collected and processed in terabytes or petabytes from various sectors. Big Data refers data that can not be processed or analyzed using conventional methods.

These big data, which can not be processed by conventional data processing techniques and stored in the conventional database, lead institutions and organizations to a technology that provides a fast and efficient data processing model. Nowadays, Big Data has become a necessity in many areas.

In Chapter 2, Hadoop framework and its some components are examined.In Chapter 3, information about Hadoop cluster setup is given and the applications and results for the cluster test are presented. Finally, the results of the paper is indicated in Chapter 4.

II. MATERIALS AND METHODS

With the development of technology, hardware prices are falling, but storage costs are increasing as the volume and variety of the data increases. Day-by-day data's level can reach exabytes or even zettabyte. The data from many different sectors such as aeronautics, meteorology, IOT applications, health, distance education and energy sectors are obtained[1–2]. Dataset is growing, and RDBMSs are not suitable for storing and managing this large dataset[3].

New techniques and technologies are developing to store and analyze big data.Until now, scientists have developed various techniques and technologies to collect, cluster, analyze and visualize the big data.

Hadoop is one of the famous and powerful Big Data tools. It provides infrastructure and platforms for other specialized Big Data applications.A number of Big Data systems are built on top of Hadoop and have many uses in different areas such as data mining and machine learning [4]. Also, thanks to Hadoop's multi-node structure, the storage system has a stronger structure and more performance is achieved with the Hadoop file system [5].

2.1. HADOOP

Hadoop [6] is an open-source software framework that allows processing of large amounts of data in a distributed computing environment. Rather than expensive equipment and dependency on different systems to store and process data, it allows inexpensive parallel processing with Big Data.

Companies need to use terabytes or petabytes of data to understand specific queries and requests from users. Existing tools are insufficient to handle big data sets. Hadoop provided a solution to this large amount of data analysis problem.

Hadoop is a convenient tool for solving all the difficulties of big data. It can process large amounts of data in the petabyte range. It provides great convenience with scalability and also ensures continuous operation with high fault tolerance. It provides reliability for the data by creating multiple copies of the data (replication) in different locations (nodes) throughout the cluster.

Hadoop has two components, HDFS (Hadoop distributed file system) and Mapreduce.While Mapreduce is used for data processing, HDFS is used for data storage[2].

2.1.1 MAPREDUCE

Today, there are many different hardware architectures that support parallel computing. There are two popular examples: (1) Multi-core computing: A multi-core processor is a processor that contains more than one computer core on a single chip;and (2) Distributed computing: Distributed computing systems enable independent computers to communicate with a common network for parallel computing [7].

Mapreduce [8], is a programming model used to processlarge data sets in cluster by providing parallelization and distribution of calculations based on several processors.It was inspired by the map and reduce functions in the functional programming. In MapReduce tasks, data is handled by these two functions. (Fig 1). In the map phase, Hadoop splits the data into small pieces and distributes them to the slave nodes (datanodes). Therefore, when the map tasks are run in parallel, the data processing load is balanced. To further split the piece, a datanode can reapply the map task.Datanode handles the little problem it gets and returns a list in the form of a binary (key, value) as a result.Reduce phase, the outputs of the map task are collected and combined to form the result of the problem they want to solve[9].

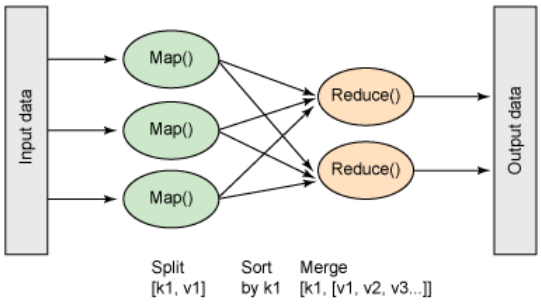

Figure 1. MapReduce diagram [10]

MapReduce greatly simplifies a large-scale analysis task on distributed data.It is much easier to process an application than to process it many times, with the MapReduce form being implemented more easily with thousands of machines in the cluster. This feature makes the MapReduce programming model more attractive.

The Mapreduce programming model [11] consists of two components that control the work execution process:the central component JobTracker and the distributed components TaskTrackers. When a MapReduce application is sent to Hadoop, the application is delivered to JobTracker. JobTracker does the job scheduling and distributes the job to TaskTrackers. Each TaskTracker performs the given operation and sends information to JobTracker about the progress of the job.

2.1.2. HDFS

Hadoop Distributed File System (HDFS), which allows distribution and storage of data between Hadoop clusters, is a file system that provides high speed access to large amounts of data [7].

HDFS architecture adopts a master / slave architecture that includes a single NameNode and all other nodes DataNodes.When the data are stored, they are stored separately as metadata and application data. While the NameNode stores the meta data, the DataNodes do the actual storage jobs. NameNode holds meta data for each file stored in HDFS in the main memory. These metadata includes a list of "stored filenames, corresponding blocks of each file, and Datanodes containing these blocks".For this reason, when a client reads a file, it first communicates with Namenode to get the locations of the data blocks that make up the file, and Namenode directs the client to the Datanode cluster hosting the requested file. The client then communicates directly with Datanode to perform file operations [12] [13].

The files in HDFS are divided into smaller blocks, typically 64 MB block size, and these divided blocks are duplicated and distributed to various Datanodes, as shown in Fig 2. Although HDFS has many similarities with existing distributed file systems, it is different.Compared to traditional distributed file systems, HDFS has two important advantages: (1) has high fault tolerance and is usually developed for low cost hardware.Datanodes send a "signal indicating that it is alive", which is called continuous heartbeat for storage space and malfunction detection. When a hardware or software problem occurs in a DataNode, Namenode detects it and reassigns the job to the other Datanode; (2) It provides effective access to data and is suitable for applications with big data sets.

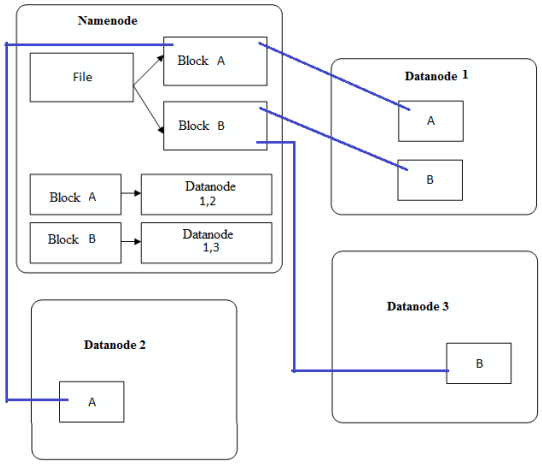

Figure2. HDFS file block structure

III. PERFORMANCE EVALUATIONS

Using Oracle VirtualBox, four virtual machines have been installed in the Ubuntu operating system, one of which have been master node and the other three have been slave nodes. While slave nodes are 4 GB memory, master node is 8 GB memory, and each nodes are also configured as 20 GB SSDs.

Each node has a Java Development Kit jdk.1.8.0 and Hadoop 2.7.2 installed.SSH (Secure Shell) is used to access other nodes in the cluster from the master node.Once the IP settings and Hadoop configurations are done, the Hadoop MapReduce function is run.In this section, Pi calculation and Grep algorithms are used.

3.1 PI CALCULATE

The area of the circle segment in the unit square shown in Fig. 3 is π / 4. If N points are randomly selected in the square, it means that approximately N * π / 4 of these points fall into the circle.This program selects random points inside the square and then checks whether the point is inside the circle (x2 + y2 <R2, it is in the circle, x and y points are the coordinates and R is radius of the circle).With the Quasi-Monte-Carlo method, the Pi estimation account is measured in terms of all points (N) and how many of these points are in the circle (M) (Pi = 4 * M / N) [14].

In this application, each map function forms some random points in the square, counting how many points are inside / outside the circle segment.The reduce function adds up all points and points in the circle to calculate the number of Pi.

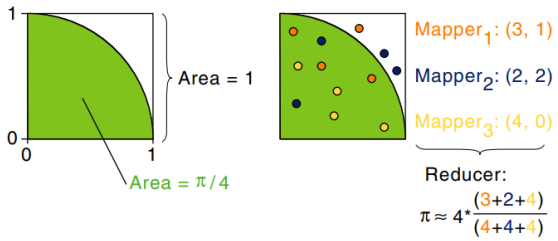

Figure 3. Monte Carlo Pi Calculation [15]

The first parameter entered in the application is the map number and the second parameter is the number of samples to be used for each map.Figure 7 shows the operation of the Pi application with 8 map tasks and 10000 samples.

1, 2 and 4 nodes were used for the test of Pi application, respectively. In all tests 8 map tasks and 1000 samples were used. The run times of the executed Pi application are given in Figure 4. According to the results, as the number of nodes increases, the duration of study decreases.

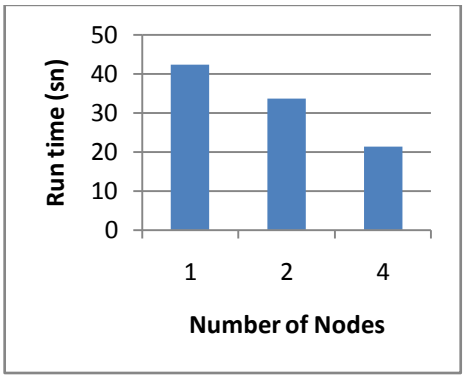

Figure4. Pi test on different node numbers

We also observed the results of Pi calculation using different map and sample numbers. According to at the working times shown in Fig. 5, it is seen that the working time increases when both the map number increases, and the number of samples run for each map increases.

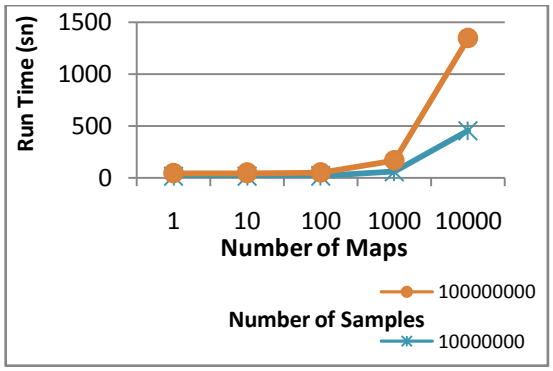

Figure 5. Pi test in different map / sample numbers

3.2. GREP APPLICATION

The Grep application, used as a search tool, counts matches of regular expressions in the given input. Each map task outputs lines containing one of the models. The reduce task collects the numbers from the map tasks and produces output in pairs.Figure 9 shows a Grep study that interrogates all expressions that start with "v" in the input.

The Grep application has been tested in file sizes of 100 MB, 500 MB and 1 GB. During the operation, first sorting is performed in the document and then the desired expression search is performed.The run times of the Grep application run in Figure 6 are shown. The Grep output of a 100 MB file is shown in Figure 10. According to the test results, as the file size increases, the processing time increases.

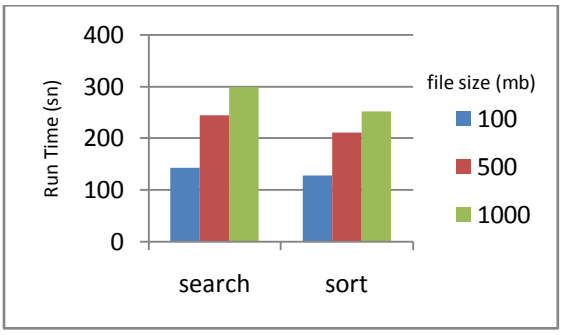

Figure 6. Execute times of Grep tests

IV. CONCLUSION

Today, the digital world is growing exponentially.Every second, data that cannot be directly managed or used with traditional data management tools are produced.In this article, we have reviewed the Hadoop architecture, which has several interfaces to create tools and applications for harmonizing data from many different data sources.The Hadoop MapReduce programming model and HDFS are increasingly being used to process and store big data sets.

In this study, a four-node hadoop cluster was created using virtualization platform.In these nodes, performance analysis was performed by applying various parameters with the benchmarking tools available under the Hadoop framework.We applied the Grep algorithm and Pi estimation algorithm for different node numbers and different file size and analyzed the results.Adding more nodes resulted in increased performance of the cluster; With a growing number of Data node, Hadoop cluster performance can be improved. Moreover, as the size of the input data increases, the performance of the runtime increases.

REFERENCES

[1] M. Ünver, E. Erdal, A. Ergüzen, "Big Data Example In Web Based Learning Management Systems", *International Journal of Advanced Computational Engineering and Networking*, ISSN(p): 2320-2106, ISSN(e): 2321-2063 Volume-6, Issue-2, Feb.-2018

[2] A. Ergüzen, E. Erdal, "Medical Image Archiving System Implementation with Lossless Region of Interest and Optical Character Recognition", *Journal of Medical Imaging and Health Informatics* Vol. 7, 1246–1252, 2017

[3] A. Erguzen, E. Erdal, M. Unver,"Big Data Challenges And Opportunities In Distance Education", *International Journal of Advanced Computational Engineering and Networking*, ISSN(p): 2320-2106, ISSN(e): 2321-2063 Volume-6, Issue-2, Feb.-2018

[4] P. Chen, C-Y Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data" *Information SciencesVolume 275, Page 314-347*,10 August 2014

[5] A. Ergüzen, E. Erdal, "An Efficient Middle Layer Platform for Medical Imaging Archives",*Journal of Healthcare Engineering*, 9 May 2018

[6] Harness the Power of Big Data - The IBM Big Data Platform, USA 2012, s.87-88

[7] T. Huang, L. Lan, X. Fang, P. An, J. Min, F. Wang "Promises and Challenges of Big Data Computing in Health Sciences" *Big Data ResearchVolume 2, Issue 1, March 2015, Pages 2-11*

[8] J. Dean, S. Ghemawat "MapReduce: Simplified Data Processing on Large Clusters", *Magazine Communications of the ACM,Volume 51 Issue 1, January 2008 Pages 107-113*

[9] R. Nanduri, N. Maheshwari, A. Reddyraja, V. Varma "Job Aware Scheduling Algorithm for MapReduce Framework" *Cloud Computing Technology and Science (CloudCom), IEEE Third International Conference*2011

[10] J. Kun, https://jeremykun.com/2014/10/05/on-the-computational-complexity-of-mapreduce/ October 5, 2014

[11] J. Dhok and V. Varma, "Using pattern classification for task assignment in Mapreduce" in *ISEC*, 2010

[12] S. J. Andaloussi, A. Sekkaki "Medical Content Based Image Retrieval by Using the Hadoop Framework", *Telecommunications (ICT), 2013 20th International Conference*, 2013

[13] M. B. Bisane , F. Pushpanjali, M. Chouragade, "Improving Access Efficiency of Small Files in HDFS" *International Journal of Scientific & Engineering Research, Volume 7, Issue 2, February-2016*

[14] J. H. Mathews, "Module for Monte Carlo Pi", http://mathfaculty.fullerton.edu/mathews/n2003/montecarlopimod.html 2005

[15] M. Völske, S. Syed, "Getting Started with Hadoop", https://www.uni-weimar.de/fileadmin/user/fak/medien/professuren/Webis/teaching/ss18/big-data-seminar/hadoop-tutorial-frame.pdf May 28, 2018

```
root@namenode1:~# time hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapre
duce-examples-2.7.2.jar pi 8 10000
Number of Maps  = 8
Samples per Map = 10000
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Wrote input for Map #4
Wrote input for Map #5
Wrote input for Map #6
Wrote input for Map #7
Starting Job
18/04/03 02:48:38 INFO client.RMProxy: Connecting to ResourceManager at namenode1/192.1
68.56.101:8032
18/04/03 02:48:39 INFO input.FileInputFormat: Total input paths to process : 8
18/04/03 02:48:39 INFO mapreduce.JobSubmitter: number of splits:8
18/04/03 02:48:40 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_152271083
5093_0009
18/04/03 02:48:40 INFO impl.YarnClientImpl: Submitted application application_152271083
5093_0009
18/04/03 02:48:40 INFO mapreduce.Job: The url to track the job: http://namenode1:8088/p
roxy/application_1522710835093_0009/
18/04/03 02:48:40 INFO mapreduce.Job: Running job: job_1522710835093_0009
18/04/03 02:48:48 INFO mapreduce.Job: Job job_1522710835093_0009 running in uber mode :
 false
18/04/03 02:48:48 INFO mapreduce.Job:  map 0% reduce 0%
18/04/03 02:49:08 INFO mapreduce.Job:  map 88% reduce 0%
18/04/03 02:49:09 INFO mapreduce.Job:  map 100% reduce 0%
18/04/03 02:49:15 INFO mapreduce.Job:  map 100% reduce 100%
18/04/03 02:49:16 INFO mapreduce.Job: Job job_1522710835093_0009 completed successfully
```

Figure 7. Pi application with 8 map tasks and 10000 sample

```
Job Finished in 37.729 seconds
Estimated value of Pi is 3.14145000000000000000

real    0m41.310s
user    0m4.188s
sys     0m0.228s
```

Figure 8.Output of Pi estimate

```
root@namenode1:~# time hadoop jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapre
duce-examples-2.7.2.jar grep /user/admin/grep_in /user/admin/grep_out 'v[a-z.]+'
18/04/03 03:46:15 INFO client.RMProxy: Connecting to ResourceManager at namenode1/192.1
68.56.101:8032
18/04/03 03:46:17 INFO input.FileInputFormat: Total input paths to process : 1
18/04/03 03:46:17 INFO mapreduce.JobSubmitter: number of splits:1
18/04/03 03:46:18 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_152271083
5093_0027
18/04/03 03:46:18 INFO impl.YarnClientImpl: Submitted application application_152271083
5093_0027
18/04/03 03:46:19 INFO mapreduce.Job: The url to track the job: http://namenode1:8088/p
roxy/application_1522710835093_0027/
18/04/03 03:46:19 INFO mapreduce.Job: Running job: job_1522710835093_0027
18/04/03 03:46:28 INFO mapreduce.Job: Job job_1522710835093_0027 running in uber mode :
 false
18/04/03 03:46:28 INFO mapreduce.Job:  map 0% reduce 0%
18/04/03 03:46:33 INFO mapreduce.Job:  map 100% reduce 0%
18/04/03 03:46:39 INFO mapreduce.Job:  map 100% reduce 100%
18/04/03 03:46:40 INFO mapreduce.Job: Job job_1522710835093_0027 completed successfully
18/04/03 03:46:40 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=102
                FILE: Number of bytes written=237533

18/04/03 03:46:53 INFO mapreduce.Job:  map 0% reduce 0%
18/04/03 03:46:59 INFO mapreduce.Job:  map 100% reduce 0%
18/04/03 03:47:05 INFO mapreduce.Job:  map 100% reduce 100%
18/04/03 03:47:05 INFO mapreduce.Job: Job job_1522710835093_0028 completed successfully
```

Figure 9. Grep application

Home / user / admin / grep_out / **part-r-00000**

11	ve
4	vrimi
2	videoyu
2	videoya
2	veya

Figure 10. Output of Grep application