CS561 - Term Paper 1
Big Data and Map Reduce

Abhishek, *B15103, B.Tech CSE 2015-19 Batch, IIT Mandi*

***Abstract*—Big Data has become a huge trendy topic in the world of computing. Computation on big data, of course, is different than computation on a normal data, but the principles remain quite the same. Many technologies have come up for computing big data, but Hadoop Mapreduce has been one of the most popular technologies. In this term paper, we aim to analyze the current available technology for computation in big data, describe the Hadoop Mapreduce framework, and discuss some of the limitations of Mapreduce framework.**

***Index Terms*—Big Data, Apache Hadoop, Mapreduce framework**

I. Introduction

BIG data refers to the data sets which are of the size beyond the current database technology. [1] In this day and age of computing, the numbers on the size of data has increased very fast, and have gone outside the range of capability of a single computer. Although, it's size is huge, big data offers exciting aspects of computing as the sheer amount of information that can be extracted with this amount of data is very valuable for this day and age. Currently, all forms of huge amount of data can be classified as big data. Whether it be the data generated from online transactions, videos, YouTube, social media sites such as Twitter, Facebook, etc., sensors, blogs, search queries, health records all have the potential to be classified as big data. [2]

It is estimated that almost 5 exabytes of data was created until the late 2003s. In comparison to that of today's data generation, such an amount of data can be generated in just a matter of two days. Intel has expected this amount to double every two years. [3] A normal computer today has a capacity of at most couple of terabytes. If we need to store this amount of data on such computers, it would require about a million number of computers to store that amount of data. But it can be seen with such amount of data, that the amount of information that could be extracted from the big data is really golden. Big data analysis has helped achieving great leaps of progress in sectors such as healthcare, finance, research, etc.

With so many advantages, it does seem that big data is quite the way forward. But all is not so easy. Extraction of information from Big data poses a huge challenge some of which will be discussed in the next section. Many technologies have come up with solutions, but each have its own advantages and disadvantages. In the next section of this term paper, we analyze deeper into big data extraction and its challenges by the help of literature review. In the further sections we bring forward the technologies present today to deal with big data. Then we move forward to list some limitations of the popular Mapreduce framework, following a conclusion on the topic.

II. Literature Review

A. Important Challenges posed by Big Data

Big Data is characterized by its 3V features, namely, Volume, Velocity and Variety. The characteristics are explained as below -

- **Velocity** - Extraction of information from big data has to be in a timely fashion. If the information is not able to be extracted in that time, then the data becomes useless in context of extractable information.
- **Volume** - As described earlier, the volume of big data has gone past the limit of capability of complex database systems. Hence, a new way of handling such a huge amount of data is required.
- **Variety** - Big data can be generated from almost every domain of life. But this data is not inherently structured or sorted as such. The data could vary in terms of representation, structure, etc. Hence, handling such type of data makes extraction of information from big data very difficult.

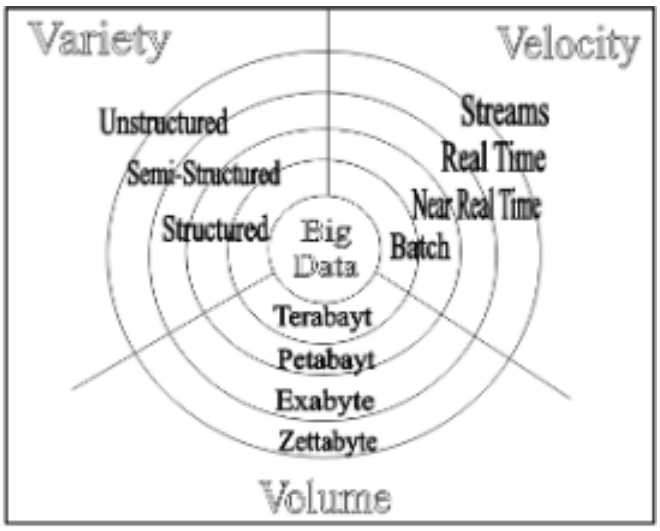

Fig. 1. The 3V's of big data. [2]

B. Data Collection

Big data can be collected from various sources such as astronomy, medicine, engineering, research, business, etc. and can be analyzed and can also help in solving real problems. McKinsey Global Institute has found out the potential usage of big data in five main topics [4] :

- Healthcare

- Public sector
- Retail
- Manufacturing
- Personal location data

Though the data can be collected very easily, there comes a question of data privacy for users. This amount of data comes from the millions and billions of people who use the above five services, and the question of ethics prevail in this matter. As long as the data is being collected in consent of the people, data collection poses no great problem. [5]

III. Current Technologies and Platforms

There have been many technologies in the market for computing on big data. The technologies can be grouped mainly on the basis of scalability. Scalability is the capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged to accommodate that growth. [6] It can be viewed in terms of big data processing as the ability to cater huge amounts of data efficiently. Hence, the big data platforms can be classified into the following two types of scaling -

- Horizontal scaling - This type of scaling generally means adding more nodes over the distributed network to evenly distribute the load.
- Vertical scaling - This type of scaling generally refers to adding of additional resources on a single computer.

Based on the two types of scaling, we will now see the platforms associated with it. [7]

A. Horizontal Scaling

*1) **Peer-to-Peer Networks**:* Peer-to-Peer networks [7] [8] [9] is a network with a huge amount of computer nodes connected to it for computation purposes. It is an architecture which is decentralized as well as distributed as the nodes in the network (peers) serve as well as consume resources.

Message Passing Interface (MPI) is the communication medium used in Peer-to-Peer Networks. Each node can be vertically scaled and the the whole system or network could be easily horizontally scaled.

Although, Peer-to-Peer networks is quite simple, there is a huge bottleneck within the system which lies in the communication between different nodes. This could be solved by broadcasting the messages, but the accessing the accumulated data/results would be very expensive.

*2) **Apache Hadoop and MapReduce Framework**:* Apache Hadoop is an open source framework that is used for distributed processing on large data sets across a cluster of computers. The various components of a Hadoop Stack are shown in Figure 2. The Hadoop platform contains the following two important components:

- **Distributed File System (HDFS)** [10] is a distributed file system that is used to store data across cluster of commodity machines while providing high availability and fault tolerance.
- **Hadoop YARN** [11] is a resource management layer and schedules the jobs across the cluster.

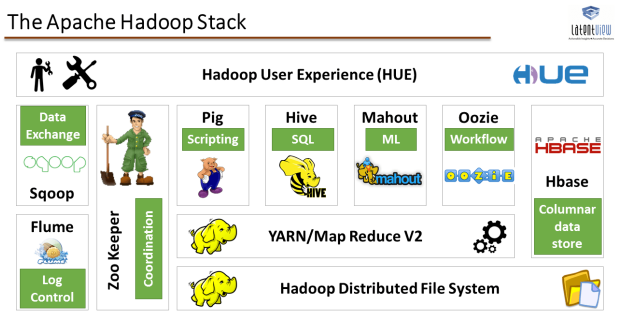

Fig. 2. Apache Hadoop Stack [12]

Hadoop uses the **MapReduce** framework [13] which was proposed by Dean and Ghemawat at Google, for its computation. MapReduce breaks the entire task into two parts, known as mappers and reducers. For a broad understanding, mappers read some input to generate some intermediate results, and these intermediate results are then passed to reducers, where the primary aggregation of results. Results are written to HDFS storage under the Hadoop stack. Mappers and reducers are distributed over the whole cluster of computing nodes. MapReduce can be divided into two stages [14]

- *Map Step* - The mapper's job is to simplify the input data into chunks of data which is stored on the local storage where the reducer can access it. [2]
- *Reduce Step* - This step takes the intermediate results and merge them in a parallel fashion to produce a set of output which is then stored on the HDFS. [2]

The Hadoop technology was inspired by the BigTable technology which is Google's data storage system, Google File System and the MapReduce framework. Hadoop is a Java based framework and an open source platform. Although not designed for real-time systems and data streaming, Hadoop offers [15] (also shown in Figure 2) -

- *HDFS* - A highly fault tolerant distributed file system that is responsible for storing data on the clusters.
- *MapReduce* - A powerful parallel programming technique for distributed processing on clusters.
- *HBase* - A scalable, distributed database for random read/write access.
- *Pig* - A high level data processing system for analyzing data sets that occur a high level language.
- *Hive* -A data warehousing application that provides a SQL like interface and relational model.
- *Sqoop* - A project for transferring data between relational databases and Hadoop.
- *Avro* - A system of data serialization.
- *Oozie* - A workflow for dependent Hadoop jobs.
- *Chukwa* - A Hadoop subproject as data accumulation system for monitoring database systems.
- *Flume* - A reliable and distributed streaming and log collection service.
- *ZooKeeper* - A centralized service for providing distributed synchronization and group services.

*3) **Apache Spark**:* Spark is a technology for big data computation developed by researchers at the University of California at Berkeley. Spark helps solve Hadoop's preformance issue regarding disk access and improves the performance of the older Hadoop systems. Since Hadoop is not efficient with iterative tasks, Spark allows the data to be cached in memory, which helps overcoming the Hadoop's limitation for iterative tasks. Spark supports Java, Scala and Python. It also supports almost all Hadoop plugins. This makes it extremely versatile to run on different systems. [7]

The Spark developers have also come up with an entire framework stack called Berkeley Data Analytics Stack (BDAS) [16]. At the lowest level of this stack, there is Tachyon which is based on HDFS. It helps enable file sharing at faster I/O rates. It works with cluster frameworks such as Spark and MapReduce. The major advantage of Tachyon over Hadoop HDFS is its high performance. Tachyon caches frequently read files thus minimizing the disk access. This enables the cached files to be read at memory speed. Since Tachyon is basically based on HDFS, it can support MapReduce programs as well. The other advantage of using Tachyon is its support for raw tables. Raw tables are cached in memory hence making its access faster.

B. Vertical Scaling Platforms

*1) **High performance computing (HPC) clusters**:* HPC clusters [17], are machines with thousands of cores. These machines have very powerful hardware with a variety of cache, processors, etc. depending on the requirement, which helps in optimized performance. Hardware failures are the least common problems in such tye of setups because of the high-end hardware. But the cost of deploying such a system can be very high because of the use of the high-tech hardware. They are not vertically scalable as Hadoop or Spark clusters but they are still at par with the performance levels. The communication scheme used for such platforms is typically MPI. We already discussed about MPI in the peer-to-peer systems. Figure 3 shows some of the comparison between HPCC and MapReduce framework.

*2) **Other technologies**:* There are some other type of vertically scalable technologies such as -

- Graphic Processing Unit (GPU)
- Field Programmable Gate Arrays (FPGA)
- Multicore CPUs

All of them have their own advantages and disadvantages, but all the technologies are mainly based on parallel computing.

IV. Limitations of MapReduce and Beyond

When the Hadoop stack was first introduced, YARN was designed to overcome some of the major limitations of the MapReduce framework. Although MapReduce is a very important framework in big data analysis, there are still performance issues, reduced optimization, reduced usability, etc. We will now see some of the limitations of MapReduce framework. [19]

A. Limitations of MapReduce

1) Performance Issues: MapReduce advocates of providing a highly scalable, fault-tolerant and a high performance framework. But in reality, the performance highly depends on the application and use of the framework. A quite large time of execution in Hadoop MapReduce is used in task initialization. scheduling, coordination and monitoring. Also, one big issue is that MapReduce does not employ any pipelining between Map stage and Reduce stage. This is a real performance blockage in terms of fast execution.

2) Programming Model Issues: MapReduce is a very elegant approach for formulating Big Data computations. But in reality, MapReduce task formulations require deep understanding of the system architecture, and what goes on behind the scenes of the computer. Even the most trivial concepts are very hard to implement in the MapReduce framework.

3) Iterative Programming Bottleneck: MapReduce is very inefficient when it comes to running iterative algorithms. In an iterative algorithms, Mappers read the same data again and again from the disk, and the results need to be written to the disk for further iterations. This poses a huge bottleneck with disc accesses and this degrades the performance.

4) Configuration and Automation Issue: Hadoop Stack (which houses the MapReduce framework) is not very easy to deploy. Each configuration option affects the performance significantly. Proper tuning of these parameters requires knowledge of both available hardware and workload characteristics, while misconfiguration might lead to inefficient execution and underutilization of resources.

B. Beyond MapReduce

Many solutions have come up in order to deal with the above mentioned limitations. Some are optimization methods over the existing MapReduce Framework, others are a different use of technology altogether. Some workarounds such as forward scheduling (setting up the next MapReduce job before the previous one finishes) have been proposed. However, these approaches introduce additional levels of complexity in the source code. One such work called HaLoop [20] extends MapReduce with programming support for iterative algorithms and improves efficiency by adding caching mechanisms. CGL MapReduce [21], [22] is another optimization that focuses on improving the iterative computation performance of MapReduce. Other examples of iterative MapReduce include Twister [23] and imapreduce [24]. There are also platforms such as MapReduce Online [25] which solves pipelining limitations, but again making the source code quite complex. Also, there are some estimation result methods of MapReduce, where we obtain the estimated results of the computation, but overcome the performance overheads.

V. Conclusion

In this term paper, we have seen how Big Data can become a golden repository of information. There are many challenges for extracting information out of Big Data, but many technologies solve this concept block. We also looked how MapReduce has become a great framework for computation over Big Data.

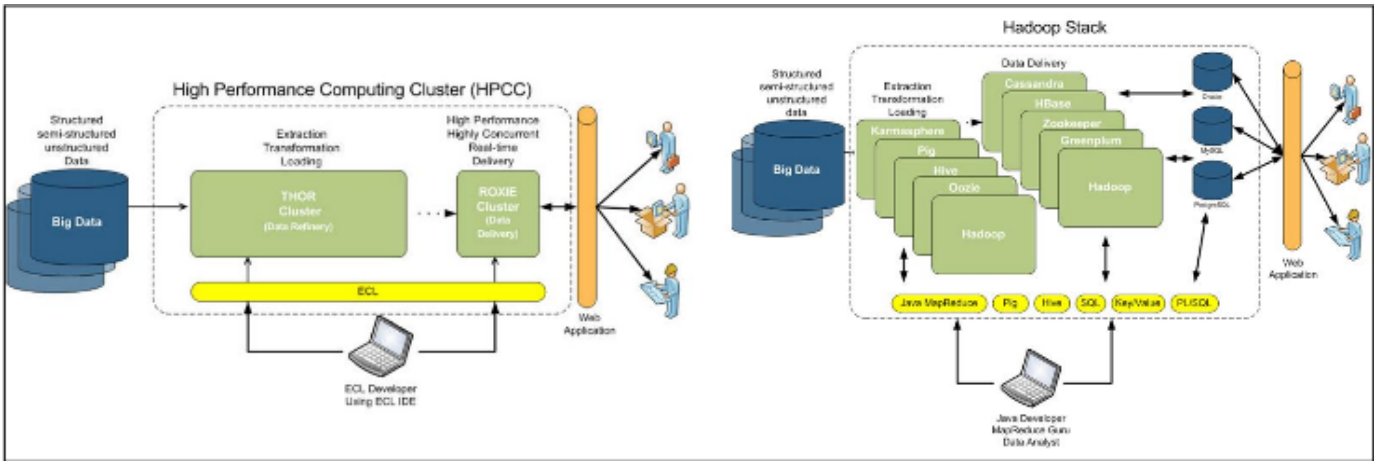

Fig. 3. Comparison between HPCC System Platform and Hadoop architecture [18]

MapReduce provides a sophisticated and distributed platform and helps in making Big Data more comprehensible. Though there are limitations of MapReduce, but there are also many other solutions which would offer an alternative over the current framework depending on case-to-case and idea-to-idea. Big Data is a very vast topic of study which would require still a few years to develop in a more extensible fashion. But the concept of 3Vs of Big Data would remain the same irrespective of any technology.

ACKNOWLEDGMENT

I would like to thank our faculty Dr. Arti Kashyap for guiding throughout the Term Paper.

REFERENCES

[1] R. Akerkar, *Big data computing*. Crc Press, 2013.
[2] S. Sagiroglu and D. Sinanc, "Big data: A review," in *2013 International Conference on Collaboration Technologies and Systems (CTS)*, May 2013, pp. 42–47.
[3] I. I. Center, *"Planning Guide: Getting Started with Hadoop"*, June 2012.
[4] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. H. Byers, "Big data: The next frontier for innovation, competition, and productivity," 2011.
[5] P. Ceravolo, A. Azzini, M. Angelini, T. Catarci, P. Cudré-Mauroux, E. Damiani, A. Mazak, M. Van Keulen, M. Jarrar, G. Santucci *et al.*, "Big data semantics," *Journal on Data Semantics*, vol. 7, no. 2, pp. 65–85, 2018.
[6] A. B. Bondi, "Characteristics of scalability and their impact on performance," in *Proceedings of the 2Nd International Workshop on Software and Performance*, ser. WOSP '00. New York, NY, USA: ACM, 2000, pp. 195–203. [Online]. Available: http://doi.acm.org/10.1145/350391.350432
[7] D. Singh and C. K. Reddy, "A survey on platforms for big data analytics," *Journal of Big Data*, vol. 2, no. 1, p. 8, Oct 2014. [Online]. Available: https://doi.org/10.1186/s40537-014-0008-6
[8] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu, "Peer-to-peer computing," 2002.
[9] R. Steinmetz and K. Wehrle, *Peer-to-peer systems and applications*. Springer, 2005, vol. 3485.
[10] D. Borthakur, "Hdfs architecture guide. hadoop apache project (2008)."
[11] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth *et al.*, "Apache hadoop yarn: Yet another resource negotiator," in *Proceedings of the 4th annual Symposium on Cloud Computing*. ACM, 2013, p. 5.
[12] L. A. Salazar, "Big data: El ecosistema básico en las empresas."
[13] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.
[14] K. Bakshi, "Considerations for big data: Architecture and approach," in *Aerospace Conference, 2012 IEEE*. IEEE, 2012, pp. 1–7.
[15] "Apache hadoop, http://hadoop.apache.org/."
[16] "Berkley data analysis stack, https://amplab.cs.berkeley.edu/software/."
[17] R. Buyya *et al.*, "High performance cluster computing: Architectures and systems (volume 1)," *Prentice Hall, Upper SaddleRiver, NJ, USA*, vol. 1, p. 999, 1999.
[18] "http://hpccsystems.com/."
[19] V. Kalavri and V. Vlassov, "Mapreduce: Limitations, optimizations and open issues," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*. IEEE, 2013, pp. 1031–1038.
[20] Y. Bu, B. Howe, M. Balazinska, and M. D. Ernst, "Haloop: efficient iterative data processing on large clusters," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 285–296, 2010.
[21] J. Ekanayake, S. Pallickara, and G. Fox, "Mapreduce for data intensive scientific analyses," in *eScience, 2008. eScience'08. IEEE Fourth International Conference on*. IEEE, 2008, pp. 277–284.
[22] I. Palit and C. K. Reddy, "Scalable and parallel boosting with mapreduce," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 10, pp. 1904–1916, 2012.
[23] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, and G. Fox, "Twister: a runtime for iterative mapreduce," in *Proceedings of the 19th ACM international symposium on high performance distributed computing*. ACM, 2010, pp. 810–818.
[24] Y. Zhang, Q. Gao, L. Gao, and C. Wang, "imapreduce: A distributed computing framework for iterative computation," *Journal of Grid Computing*, vol. 10, no. 1, pp. 47–68, 2012.
[25] T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, K. Elmeleegy, and R. Sears, "Mapreduce online." in *Nsdi*, vol. 10, no. 4, 2010, p. 20.