Journal of Information Engineering and Applications
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol.7, No.3, 2017

Big Data Analysis: Implementations of Hadoop Map Reduce, Yarn and Spark

Fatimah Abdalla Al-Alem
Ministry of Education, Irbid, Educational Directorate of Bani Kinana

Abstract
Nowadays, with the increasingly important role of technology, the internet and huge size of data, it has become not only possible, but necessary for management and analyzing these data, where it is difficult to process and retrieve information related to that data. Moreover, the amount of memory consumed by such data reached to terabytes or petabytes, which make it difficult for processing, analyzed, and retrieving. Also, many techniques have been carried to process big data. The dealing with the statistical programs became very hard. There are a number of algorithms that is used in big data processing, such as Mapreduce. Many obstructions and challenges face the big data processing as: poor bounded-time performance in heavy activities and high-priced cost. In this study, different big data implementations are demonstrated, also, we propose open issues and challenges raised on big data implementations. The findings compares several big data platforms which are; Hadoop, Yarn and Spark. Finally, we provide useful recommendations for further research about the best one between these implementations to process the data according to specific bases.
Keywords: Big data, Mapreduce, Hadoop, Spark, Yarn.

1. INTRODUCTION

Big data is a collection of data sets so large and complex that it becomes difficult to process using a hand database management tools or traditional data processing.

We can also explain big data definition in other words like.

Big data is extremely large data sets, which mean data that cannot fit easily into standard relational data base. For identifying big data there are three characteristics known as 3 V's: Volume Variety and Velocity. Volume reflects a large amount of data that needs to be processed as a various data sets stack together the amount of data increases. Variety reflects different sources of data it can vary from web server. logs to structure data from the data bases to unstructured data from social media. Velocity reflects the amount of data which Keeps on accumulating times. All the three V's are very good description of big data problems and may help you to find it in.

2. Importance of database

International data corporation (IDC) 6th annual study [1] concludes that:

From 2005 to 2020 the digital universe will grow by a factor of 300, from 130 Exabyte's to 40,000 Exabyte's, or 40 trillion gigabytes.

More than 5200 gigabytes for every man, woman and child in 2020.

From now until 2020, the digital universe will about double every two years.

33% of the digital data might be valuable if analyzed, compared with 25% today.

For what we have already increased the size of the data and the importance of treatment, researchers studied several algorithms and methods for big data processing. One of the famous algorithms used for processing purposes is Mapreduce algorithm. Mapreduce is a framework uses large data sets which computed in parallel, and distributed on a cluster that is occupy a specific algorithm.

In the next section we will talk in details about Mapreduce algorithm, and then we will introduce three implementations: Hadoop, Yarn, and Spark. Followed by a comparison between them, and finally we will introduce some recommendations to users to choose the best implementation for the suitable application.

3. MAPREDUCE ARCHITCTURE

Mapreduce programming model [2] is used to process large datasets. The model itself is based on the concept of parallel programming. Based on the parallel programming notion, processing is decomposed into n sub-entities that are executed concurrently. The instructions for each sub-entity are executing simultaneously on different CPU's. Depending on the IT infrastructure setup, the CPU's may be available on the same server system or on remote nodes that are accessible via some form of interconnect/network. The Mapreduce model itself is derived from the map and reduce primitives available in a functional language such as Lisp. A Mapreduce job usually splits the input datasets into smaller chunks that are processed by the map task in a completely parallel manner. The outputs obtained from all the map tasks are then sorted by the framework and made available as input(s) into the reduce tasks. It has to be pointed out that a Mapreduce model is normally suitable for long running batch jobs, as the reduce function has to accumulate the results for the entire job, a process that is associated with overhead

concerning the collection and submission of the processed datasets. In the Mapreduce model, actual parallelism is achieved via a Map- Reduce (split/merge) process (see Fig. 1) that can be described as:

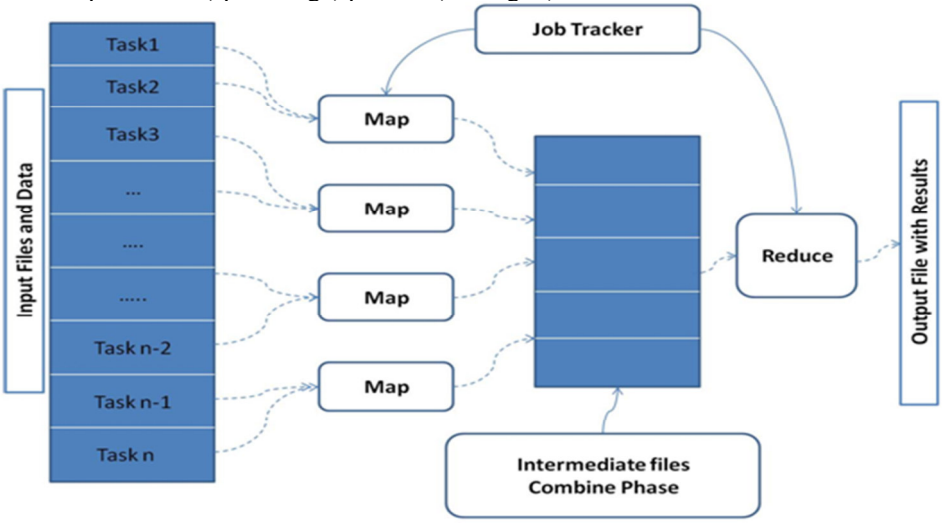

Fig. 1 *Mapreduce Architecture*

Initially, a master program (basically a central coordinator) is started by the Mapreduce job [1] that utilizes the predefined datasets as input (input file and data).

According to the Job Tracker, the master program initiates the Map and the Reduce programs on various systems. Next, it starts the input reader to retrieve the data from some directory (hosted in a distributed file system). The input reader then decomposes the data into small chunks (tasks) and submits them to randomly chosen mapper programs. This process concludes the intermediate phase and initiates the parallel processing stage.

After receiving the data, the mapper program executes a user supplied map function, and generates a collection of [key, value] pairs. Each produced item is sorted and submitted to the reducer.

The reducer program collects all the items with the same key values and invokes a user supplied reduce function to produce a single entity as a result.

The output of the reduce program is collected by the output file (effective join phase) and this process basically terminates the parallel processing phase.

4. MAPREDUCE IMPLEMENTATIONS

Since there are many real life applications for big data that have appeared [1], each application requires a different kind of map-reduce implementation to deal with it. Researchers discovered many of those implementations that can't be counted in one study, so we will propose some general-purpose Mapreduce algorithm frameworks such as Hadoop, Spark and Yarn. We will study the architecture of which features and determine appropriate framework for the applications depending on its nature. We have studies these techniques, then, we comparison with the traditional models. Finally, highlighting the advantages of Mapreduce Models into traditional models.

Hadoop

Hadoop [3] is a framework that is an open-source software which storing and processing big data in a distributed manner on large clusters of hardware. Basically, it establishes two tasks: big data storage (HDFS) and faster processing. In additional to its original goal of searching millions or billions of web pages and returning relevant results, many organizations are looking for Hadoop as a platform starts using input data and ends with output data as in the following figure(Fig. 2).

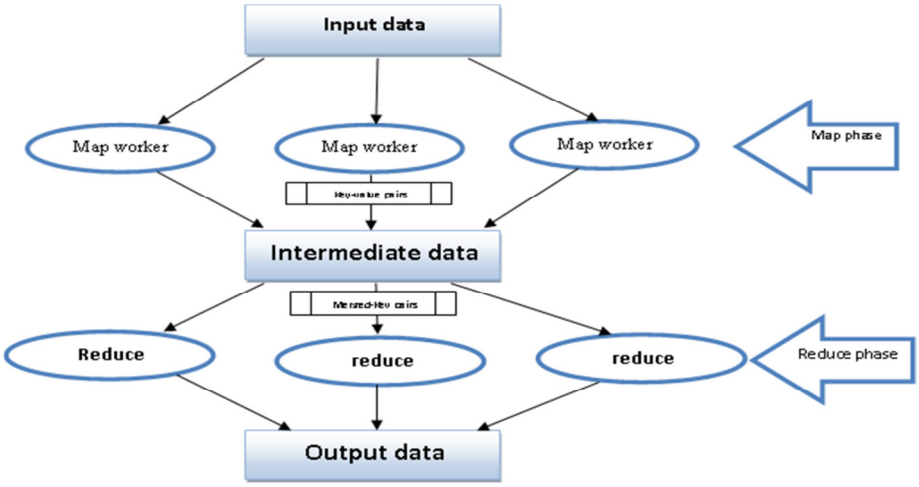

Fig.2 *Hadoop Architecture*

The above figure shows how input data transmitted and transformed to output in less overhead and time by, first the input data divided into parts and spreads them onto map workers by job trackers, the map worker breaks down the work into task vectors each with unique values (key –value) after map workers complete their tasks, the result work taken to the reduce workers to merge their keys of the data sets to introduce the final result minimum overhead and time as possible.

Our paper basically uses Hadoop, a tool specified by Apache Server, for information retrieval. Hadoop is a Java Software Framework that supports data intensive distributed applications and is developed under open source license. Many websites such as Facebook is depending on Hadoop [4].

A.1. the Advantages of Hadoop

Hadoop have been used extensively because of many advantages [4] [5], including:

Low cost: When we say open-source framework means it's free to use store large amount of data using particular hard ware.

Computing power: We know that it's distributed computing form fast process for very large volumes of data. Processing power increases by increasing the nodes used.

Scalability: There are no difficulties in scaling the system just by adding more nodes or platform. No much administration is required.

Storage flexibility: There is no need having preprocessed data before storing it as it done in (relational data base systems). Including unstructured data such as text, images and videos. You can store much data you want and then you can decide how to use it.

Inherent data protection and fault tolerant features: If a node failed, automatically jobs are goes to other nodes to save the distributed computing from fail. And multiple copies of all data automatically saved.

A.2. The challenges that faced Hadoop

In spite of its advantages, Hadoop faces some limitations such as: Mapreduce is not a good solution for all type problems. Its works well for simple requests for information and problems that can be splits into separate tasks. Although it is inefficient for iterative and interactive tasks. Map Reduce is a heavy file. Because the nodes don't intercommunicate except through sorts and shuffles, iterative algorithms asked for more than single map-shuffle/sort-reduce steps to finish. That makes multiple files between Map Reduce steps which is not very good for advanced computing. Second, there exist a skill perform gap. It is much easier to find programmers with SQL skills than Map Reduce skills. And, Hadoop administration seems part art and part science, requiring low-level knowledge of operating systems, hardware and Hadoop kernel settings. The third challenge about the fragmented data security subject in Hadoop, it does not have easy-to-use, full-feature tools for data management, missing tools for data quality and standardization.

Spark

Spark [6] data platform that shows the idea of in-memory cluster computing; where datasets can be go in memory to decrease their latency of access. Rises some performance compared to Hadoop, it implementation in and exploits the Scala language, which provides a different environment for data processing. Spark architecture have been illustrated in the following figure:

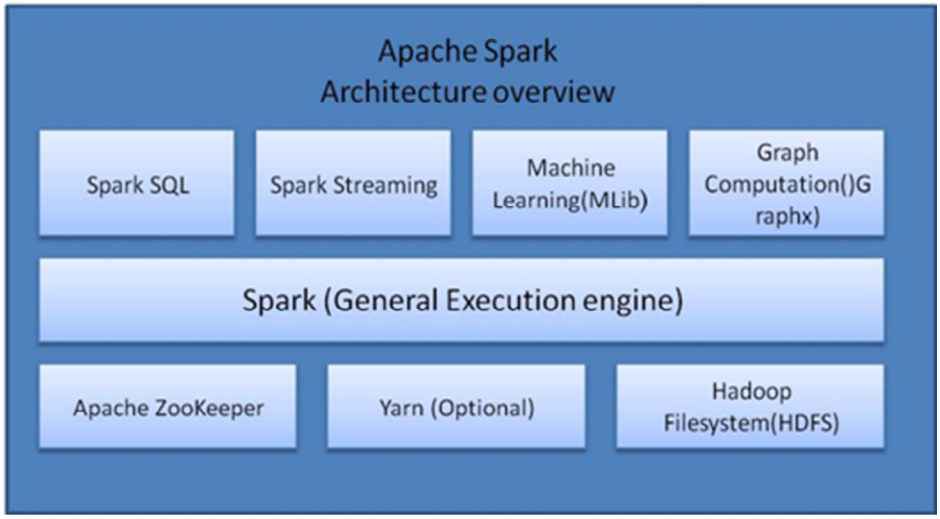

Fig. 3 *Spark Architecture*

The above figure shows that Spark platform includes: Spark Streaming which used for manipulating real time streaming data. Spark SQL: which enables spark datasets to be exposed and allow execution of SQL operations using traditional tools. Spark MLlib: is Spark's scalable machine learning library consisting of common learning algorithms and utilities, such as classification, collaborative filtering clustering, dimensional reduction, etc. Spark GraphX: a new (alpha) Spark API for graphs and graph-parallel computation.

B.1. Spark advantages

Spark [6] is used widely because it offers a compatible programming model for a broad range of applications, also it Achieves fault tolerance by providing coarse-grained operations and tracking lineage, Spark Provides definite speedup when data fits into the memory.

B.2. Spark Disadvantages

Despite the proliferation of Spark greatly, because it solves many problems, but it suffers from some restrictions such as the Problem in not fitting into memory. Saving as text files is too slow and didn't fit all scenarios; if data is larger than size for e.g. petabyte other implementations are faster than spark [6].

Yarn

Yarn [7] introduces a cluster resource management layer for more resource utilization. Previously versions of Hadoop it was not capable to share the same cluster among different task operations .Before Yarn, the only way to cooperate a cluster was to splitting the cluster into different partitions and execute different platform on these partitions. This doesn't insure efficient utilization of the cluster resources. So, Yarn handle different operational frameworks (fig. 4) in the same cluster.

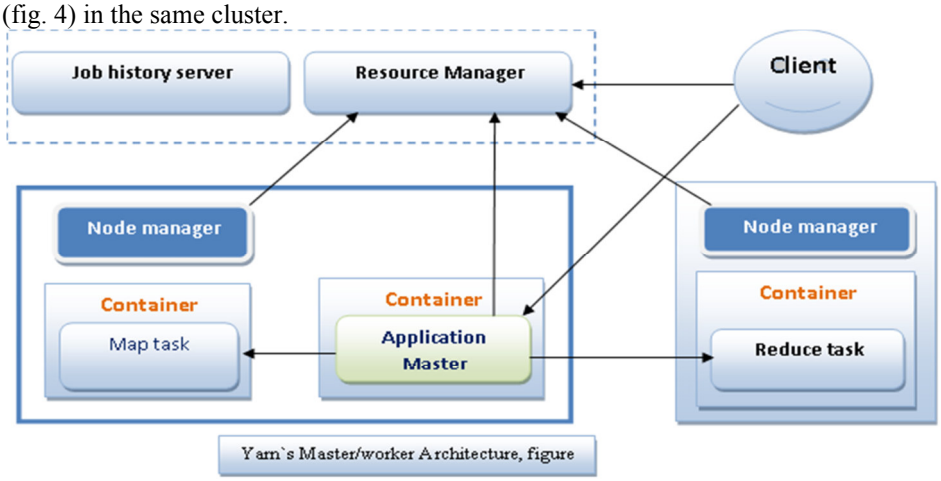

Fig.4 *Yarn Architecture*

As we see above in fig. 4, "Yarn`s Master/worker Architecture", YARN is designed to organize cluster resources, and share the resources between compute platform as Map Reduce, and Spark etc. The resources are being organized by the master (the Resource Manager) over the workers and schedules working in the cluster. Also, the Resource Manager deals with clients interactions.

C.1. Yarn Advantages

Yarn is a developed version from Hadoop with specific advantages. Yarn does more utilization of resources, no need for constant map reduced slots. Yarn provides a central shared manager that enables multiple applications

shared common resources. Yarn can run application that it's not necessary follow map reduce model [7].

C.2. Yarn vs. Hadoop

Table.1 shows a comparison between yarn and Hadoop platform from different perspectives:

Table.1 Hadoop vs. Yarn

Comparison field	Hadoop	Yarn
Framework Architecture	Map Reduce is Programming Model,	YARN is architecture for distribution cluster.
Resource management	Hadoop support programming model which support parallel processing that we know as Map Reduce. Before Hadoop 2.0, Hadoop 1.0 already support Mapreduce.[8]	Hadoop 2 using YARN for resource management.
Recourses retrieval	When user submit Mapreduce Job. Resource will be back to free.	Resource manager will give MR master all resources it needs or it is according to cluster computing capabilities.
Cluster resources	In Hadoop 1.0, there is tight coupling between Cluster Resource Management and Map Reduce programming model.[4]	In Hadoop 2.0, this is totally based on Cluster Resource Management.
Cluster management	Map Reduce job is divided between number of tasks called mappers and reducers. Each task runs on one of the machine (Data Node) of the cluster, and each machine has a limited number of predefined slots (map slot, reduce slot) for running tasks concurrently.	Job Tracker is responsible for both managing the cluster's resources and driving the execution of the Map Reduce job. It reserves and schedules slots for all tasks, configures, runs and monitors each task.
Resources Sharing	Multiple applications can run on Hadoop via Map Reduce and all application could not share common resource management.	Multiple applications can run on Hadoop via YARN and all application could share common resource management.
Comparison field	Hadoop	Yarn

From the above table, it can be concluded that Yarn platform is better than Hadoop for several applications, Yarn has central resource manager component resources and allocates the resources to the application, while Hadoop, does not have this type of resources to the application.

5. COMPARISON BETWEEN SEVERAL IMPLEMENTATIONS

Table.2 shows the comparison between Hadoop, spark and yarn, starting with Hadoop, we see that Hadoop maximum scalability size reaches 5,000 nodes in cluster [9], but Yarn exceeds 4,000 nodes that's a massive scale compared with Hadoop, also spark scale reached over 8,000 nodes [10], all of them supports data size up to petabyte. Also according to Data I/O we can see that Hadoop returns back to disk after map or reduce action while spark processes its data in memory sparks needs a lot memory, but yarn utilizes several resources. Real time processing in Hadoop uses batch processing but spark can perform batch processing as well as real time system that makes spark use a platform for different tasks instead of assign different tasks to different plat form , yarn perform Multiple master application job processing. Fault tolerance all have fault tolerance but differ in the methodology using. Spark and Hadoop Mapreduce both have acceptable failure tolerance, but Hadoop Mapreduce is a little more tolerant [11]. That spark have to restart a crashed process from the beginning but Hadoop because it relies on disk if a process crashed it will continued from the point it left off. But Yarn it provides mechanisms to restart the (application master) AM if AM fails. Each of , Resource Manager and Node Managers executes the AMs which is in charge of automatically detecting an AM fail and retry running it again. Hadoop supports iterative tasks but it's not good enough, but spark support iterative tasks uses Scala interpreter, and Yarn using Central resource manager [7].

Table.2 General Comparison between Mapreduce Implementations

Map reduce implementations [15]	Hadoop[12]	Spark[13] [14]	Yarn[8]
Scalability	max Cluster size almost 5,000 nodes	Over 8,000 of nodes	Exceeds 4,000 nodes
Data I/O	Usually on disk	In desk and in memory	Memory and CPU ,disk , NT
Real time process	Batch process	up to 10x faster on disk 100x in memory	master application job processing
Fault tolerance	Yes	Yes	Yes
Data size supported	Pet bytes of data	From mega to Pet bytes of data	Pet bytes of data
Iterative tasks support	Slow iterative (HDFS)read/write(Not ideal)	Support iterative tasks uses Scala interpreter	Central resource manager

RECOMMENDATIONS AND CONCLUSION

In this study, we described three popular techniques based on Mapreduce model; Hadoop ,Spark and Yarn ,and generalized a comprehensive comparison of these techniques. The study provided useful recommendations for further research about the best one between these implementations to process the data according to specific bases We also highlighted pros and cons of each algorithm. All the restrictions and disadvantages of traditional models cover with Mapreduce model, which increases the performance and reduce computational time. We summarized some tips about Mapreduce implementations depend on what we want to do: When size of data is from GBs to PBs we recommend in use Spark, where it got a good performance and high speed compared to other Mapreduce implementations. Hadoop architecture depends mainly on clusters, so it is costly in the step of building and developing clusters. It is worth mentioning that Yarn as a new technique compared Hadoop achieved good results in the processing of data at the lowest possible cost and In addition to using linear-scale storage.

In the future, we plan to examine the performance of the previous implementations experimentally using the same data set and compare each one with time, accuracy and storage; also we plan to study other implementations such as Mahout, Disco, Spring, etc.

6. References

Gantz, John, and David Reinsel. "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east." *IDC iView: IDC Analyze the future* 2007, no. 2012 (2012): 1-16.

Schomig, M.; Neuhauser, D.; Seidler, R.; Bucker, H.M., "Benchmarking Different MapReduce Implementations for Computer-Aided Hardware Development," Mathematics and Computers in Sciences and in Industry (MCSI), 2014 International Conference on , vol., no., pp.15,21, 13-15 Sept. 2014.

Fadika, Z.; Dede, E.; Govindaraju, M.; Ramakrishnan, L., "Benchmarking MapReduce Implementations for Application Usage Scenarios," Grid Computing (GRID), 2011 12th IEEE/ACM International Conference on, vol., no., pp.90, 97, 21-23 Sept. 2011.

Garg, Dweepna, Khushboo Trivedi, and B. B. Panchal. "A comparative study of clustering algorithms using mapreduce in hadoop." In International Journal of Engineering Research and Technology, vol. 2, no. 10 (October-2013). ESRSA Publications, 2013.

Dalavi, M.; Cheke, S., "Hadoop MapReduce implementation of a novel scheme for term weighting in text categorization," Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014 International Conference on , vol., no., pp.994,999, 10-11 July 2014.

Miller, John A., Casey Bowman, Vishnu Gowda Harish, and Shannon Quinn. "Open Source Big Data Analytics Frameworks Written in Scala." In *Big Data (BigData Congress), 2016 IEEE International Congress on*, pp. 389-393. IEEE, 2016

Shao, Yanling, et al. "Energy-Aware Dynamic Resource Allocation on Hadoop YARN Cluster." *High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2016 IEEE 18th International Conference on*. IEEE, 2016.

Guo Yucheng, "MapReduce Model Implementation on MPI Platform," Distributed Computing and Applications to Business, Engineering and Science (DCABES), 2014 13th International Symposium on , vol., no., pp.88,91, 24-27 Nov. 2014.

Gyftakis, Konstantinos; Anagnostopoulos, Iraklis; Soudris, Dimitrios; Reisis, Dionysios, "A MapReduce framework implementation for Network-on-Chip platforms," Electronics, Circuits and Systems (ICECS), 2014 21st IEEE International Conference on, vol., no., pp.120, 123, 7-10 Dec. 2014.

Yunhee Kang; Park, Y.B., "The performance evaluation of k-means by two MapReduce frameworks, Hadoop vs.

Twister," Information Networking (ICOIN), 2015 International Conference on, vol., no., pp.405, 406, 12-14 Jan. 2015.

Lee, Daewoo, Jin-Soo Kim, and Seungryoul Maeng. "Large-scale incremental processing with MapReduce." Future Generation Computer Systems 36 (2014): 66-79.

Vavilapalli, Vinod Kumar, et al. "Apache hadoop yarn: Yet another resource negotiator." Proceedings of the 4th annual Symposium on Cloud Computing. ACM, 2013.

Kaewkasi, C.; Srisuruk, W., "A study of big data processing constraints on a low-power Hadoop cluster," Computer Science and Engineering Conference (ICSEC), 2014 International , vol., no., pp.267,272, July 30 2014-Aug. 1 2014.

Sarazin, T.; Azzag, H.; Lebbah, M., "SOM Clustering Using Spark-MapReduce ," Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International , vol., no., pp.1727,1734, 19-23 May 2014.

S. Ancy and M. Maheswari, "Locality based data partitioning in Map reduce," *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, 2016, pp. 4869-4874.