International Journal of Engineering Research ISSN:2319-6890)(online),2347-5013(print) Volume No.5, Issue No.6, pp : 531-533 1 June 2016

Mapreduce Technique: Review and S.W.O.T Analysis

SaloniMinocha, Hari Singh
Department of Computer Science, N.C. college of Engineering, Panipat, Haryana India
Corresponding email: saloni.minocha2015@gmail.com

Abstract: In this paper, the review of Map Reduce Technique is given. The paper also presents the S.W.O.T analysis of Map Reduce technique.

Keywords:Big Data Analysis, MapReduce, Virtualization, Parallel Computing, Distributed Computing.

I. Introduction

There has been a surprising growth in the volumes of data produced by the organizations. Large amounts of data are captured by the organizations on daily basis. This data is known as Big Data. These are the datasets which continues to grow so much that it is difficult to manage it using existing database management concepts and tools[9]. Big data is mainly associated with three dimensions: Volume, variety of sources and the velocity with which it is processed. In recent years, several efficient parallel and concurrent algorithms got applied, so as to meet the performance requirements and scalability of Big Data. Several algorithms are put into action using different techniques of parallelization. However, this paper mainly focuses on MapReduce among those new algorithms.MapReduce was introduced by Dean *et al* in 2004. It is inspired by functional programming. It is a programming model initiated by Google for processing Big Data in a distributed environment.
The paper is divided into several sections for the ease of understanding. Section 2, introduces the MapReduce Technique. Section 3, presents the architecture of MapReduce. Section 4 presents the classification and example of MapReduce. Section 5, presents theS.W.O.T analysis of MapReduce Technique. Section 6, states the conclusion drawn from the paper.

II. MapReduce Technique

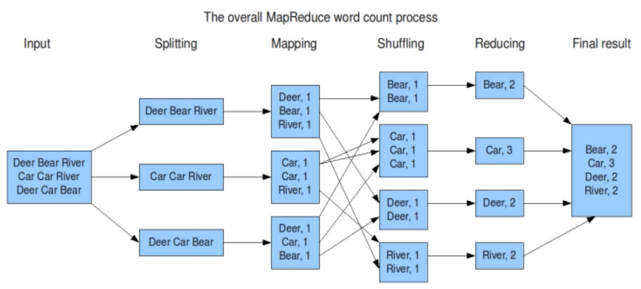
Over the last few years, for analysis of large amount of data in parallel and batch style MapReduce has emerged as the most popular paradigm. [6] MapReduceis a technique that can process large data files which are multi structured across massive data sets. The processing is broken into small units of work and is executed in parallel across several nodes. As a result, performance is improved. There are two primary functions in MapReduce: The Map function and the Reduce function. These functions are written by the user according to his requirements. Hence, one can easily achieve the goal of distributed computing by combining the MapReduce programming technique and an efficient distributed file system [6].

III.Architecture of MapReduce [2]:

In MapReduce one of the node acts master node and various other nodes which acts as worker nodes. The worker nodes take the responsibility for running map and reduce tasks whereas the master is responsible for assigning tasks to the idle workers. Initially, the input file is read and split into multiple pieces across distributed file system. Content of associated split is read by each map worker and passed to the user defined map function. The output from the map function is partitioned according to the number of reducers after buffering it in the local memory. Reduce workers are then notified by the master to read the data from the local memory of map workers. The output from the reduce worker is finally appended to the output file.
Technically, the inputs fed to the map tasks and results from the reduce tasks are of Key-Value pair form. They are executed without any communication with other tasks. Thus, there are no communication costs between tasks along with no contention arisen due to synchronization. Also, MapReduce follows run time scheduling. This means that there is no execution plan that specifies which task will run on which node before execution. For improving performance, the tasks on straggling nodes are executed redundantly on nodes which have finished the tasks assigned to them.

IV. Classification and Example of MapReduce

The MapReduce algorithms can be classified as follow [2]:
a) The first class includes algorithms that are adopted as a single execution MapReduce model.
b) The second class includes algorithms that are adopted for sequential execution of constant number of MapReduce model.
c) The third class includes iterative algorithms where iteration represents an execution of a single MapReduce model.
d) The fourth class includes complex iterative algorithms where iteration represents an execution of multiple MapReduce models.
Figure 1 below shows an example of MapReduce working:
Figure 1: MapReduce Example



In this example, number of occurrences of each word in a collection of documents is counted. The input to the map function will be documents, where documents represent a record. The Key for map function is the Document Id and the value will be the string representing the document contents. It then outputs a list of intermediate Key-Value pairs in which each Key will represent the work and the value will always be one. The combiner function can be used to aggregate all the repeated words occurring in the same document. After the map phase, shuffling and sorting takes place alphabetically. In this the intermediate data from each map phase is taken and sorted with intermediate data of other nodes. Division of this data is done according to the number of reducers. Finally, the reduce tasks takes keys and list of values to produce the desired results, i.e., the occurrences of words in each document.

V. S.W.O.T analysis of MapReduce Technique

The following are the advantages of MapReduce [2] [5]:

1. Simple and easy to use- The MapReduce model is simple but expressive. It is easy even for the programmers without any experience in parallel and Distributed Computing, since it hides the details of paralleization, fault tolerance, and locality of data. With MapReduce, a programmer only needs to define his job with only Map and Reduce Functions, without having to specify physical distribution of his job across the nodes.
2. Flexible- MapReduce does not have any dependency on data model and schema. With MapReduce a programmer can deal with irregular or unstructured data more easily than they do with DBMS.
3. Independent of the storage- MapReduce is basically independent of the storage layers. Thus, MapReduce can work with different storage layers.
4. Fault tolerance- MapReduce is highly fault tolerant. It detects the failed Map and Reduce tasks of failed nodes and reassign them to other idle nodes of the cluster.
5. High scalability- MapReduce has been designed in such a way that it can scale up to large clusters of machines. It supports runtime scheduling which enables dynamic adjusting of resources during job execution. Hence, offering elastic scalability.
6. Supports data locality by collocating the data with the compute node, so it reduces network communication cost
7. Ability to handle data for heterogeneous storage system, since mapreduce is storage independent, and it can analyze data stored in different storage system.
8. High level language support which was not there earlier. Microsoft scope, Apache Pig and Apache Hive all aim at supporting declarative query languages for the MapReduce Framework [5].

The following are the pitfalls in the MapReduce framework compared to DBMS[5]:

1. Earlier there was no high level language support like SQL in DBMS and any query optimization technique.
2. MapReduce is schema free and index free. An MR Job can work right after its input is loaded into its storage.
3. A single fixed dataflow which don’t support for algorithms that require multiple inputs. MapReduce is originally designed to read a single input and generate single output.
4. Low efficiency- With fault tolerance and scalability as its primary goals, MapReduce operations are not always optimized for I/O efficiency. In addition, MapReduce are blocking operations. A transition to the next stage cannot be made until all the tasks of the current stage are finished. Also, MapReduce has a latency problem that comes from its inherent batch processing nature. All of the inputs for an MR job should be prepared in advance for processing.
5. Very young compared to 40 years of DBMS.

The two major challengesin MapReduceare[6]:

1. Due to frequent checkpoints and runtime scheduling with speculative execution, MapReduce reveals low efficiency. Thus, how to increase efficiency guaranteeing the same level of scalability and fault tolerance is a major challenge. The efficiency problem is expected to be overcome in two ways: Improving MapReduce itself or leveraging new hardware.
2. Second challenge is how to efficiently manage resources in the clusters which can be as large as 4,000 nodes in multi user environment and achieving high utilization of MR clusters.

Another challenge is the energy issue [5]. Since the energy cost of data centers hits 23% of the total amortized monthly operating expenses. It is necessary to devise an energy efficient way to control nodes in data center when they are idle. Two strategies are proposed:

1) Covering Set Approach- It designates in advance some nodes that should keep atleast a replica of each data block and the other nodes are powered down during low utilization periods [5].
2) All-In strategy- It saves energy in all or nothing fashion. In this, all the MR jobs are queued until it reaches a threshold, then all the nodes are run to finish MR jobs and then all the nodes are powered down until new jobs are queued [5].

The following are the applications of MapReduce framework [5]:

1) Processing crawled documents, web request logs in order to compute various kinds of derived data, such as inverted indices.
2) Machine learning task, scientific simulation and large scale image processing tasks.
3) Opinion mining. It is technique for extracting estimation from the internet. It is known as sentiment classification.
4) MapReduce can be for clustering very large-moderate-to-high dimensionality dataset.
5) MapReducecan also be used for performing join task:

Two way joins- Joins which involve only two datasets. The two way join is classified into Map side join, Reduce side join and broadcast join.

Multi way joins- Where joins involve more than two datasets.

Figure 3 shows the summary of S.W.O.T analysis of MapReduce Technique.

Figure2: S.W.O.T Table of MapReduce

Strengths	Weaknesses
Simplicity and ease of use. Flexibility. Independent of the underlying storage. High Fault tolerance. High scalability. Supports locality of data. Reduces Network communication costs.	Schema Free and Index free. Single fixed data flow. Low efficiency. Very young compared to 40 years DBMS. Low Input Output cost optimization. Take more time to finish a job. All tasks should be completed

Ability to handle data for heterogeneous system. Support Big Data Analysis. High level Language support.	before moving to the next job, this causes performance degradation
Opportunities Processing crawled documents, web logs in order to produce various kinds of derived data. Machine learning tasks, Scientific simulation and large scale image processing. Opinion mining. Clustering high dimension dataset. Performing Joins.	Threats Technical and cultural challenges. Hidden Energy Cost issues. Heterogeneity of systems. Low efficiency due to various checkpoints. Difficulty in management of resources in cluster of nodes.

VI. Conclusion

After taking the various advantages and limitations into account, it is concluded thatMapReduce is simple, provides good scalability and fault tolerance but is unlikely to replace traditional DBMS. Instead it can compliment DBMS. Nonetheless, efficiency of MapReduce needs to be addressed for successful implications.

Acknowledgement

I take this opportunity to express my sincere thanks and deep gratitude to Hari Singh, my guide who extended his whole hearted co-operation and have helped me in completing this paper successfully.

References

i. *Dittrich J, Quiané-Ruiz JA. Efficient big data processing in HadoopMapReduce. Proceedings of the VLDB Endowment. 2012 Aug 1;5(12):2014-5.*

ii. *Elsayed A. et al, MapReduce: State of the art and research directions, IJCEE, vol6, No.1, February 2014.*

iii. *Garlasu, D, A Big Data Implementation based on Grid Computing, published in 11th Roedunet International Conference-2013.*

iv. *Gohil P, Garg D, Panchal B. A performance analysis of MapReduce applications on big data in cloud based Hadoop. InInformation Communication and Embedded Systems (ICICES), 2014 International Conference on 2014 Feb 27 (pp. 1-6). IEEE.*

v. *Lee KH, et al Parallel data processing with MapReduce: a survey. AcMsIGMoD Record. 2012 Jan 11;40(4):11-20.*

vi. *Maitrey S. &Jha C.K., Handling Big Data efficiently by using Map Reduce Technique,2015 IEEE International Conference on Computational Intelligence & Communication Technology.*

vii. *Manikandan SG, Ravi S. Big Data Analysis Using Apache Hadoop. InIT Convergence and Security (ICITCS), 2014 International Conference on 2014 Oct 28 (pp. 1-4). IEEE.*

viii. *Nandimath, J. et al, "Big data analysis using Apache Hadoop." In Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on, pp. 700-703. IEEE, 2013.*

ix. *Singh, S. & Singh, N., Big Data Analytics, 2012 International Conference on Communication, Information & Computing Technology, Oct 19-20, Mumbai, India*

x. *Tang B, Moca M, Chevalier S, He H, Fedak G. Towards mapreduce for desktop grid computing. InP2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2010 International Conference on 2010 Nov 4 (pp. 193-200). IEEE.*