HADOOP MAPREDUCE - WORDCOUNT IMPLEMENTATION

[1]P. Deepika, [2]Prof. G. R. Ananatha Raman

Department of Computer Science and Engineering
ACE, Hosur

Abstract
Data is the hot topic in the present scenario. When the size of the data is small, then pen drives or CD's can be used for storage. For medium sized data, a hard disk can be used for storage but when the size of the data is very large we call it as Big Data. For managing this massive amount of data, Google found out a solution which is named as MapReduce. It simply breaks a large data into small parts and processes it, then combines it together. Hadoop is the tool for managing the Big Data where it uses the MapReduce algorithm where different CPU nodes are used for processing the Big Data. This paper is a study of Hadoop MapReduce and its related concepts.

Index terms: Storage, Big Data, Hadoop, MapReduce, CPU nodes

I. INTRODUCTION

Data can be either structured or unstructured. Structured data will always define a length and format for the particular data and it is organized properly. Unstructured data will not define any structure and it is also not organized properly. Big Data is the one where it will manage massive amount of both the structured and unstructured data. In the previous years, the organization will have a separate computer to storage the organization's content. This approach can be practiced when there is less volume of data. When the data size exceeds, it cannot processed with this approach. It is difficult to maintain the overwhelming data. For this problem, Google found a solution called MapReduce and it is OS Independent. MapReduce tasks have two steps. One is the Map Phase and the other one is the Reduce Phase. In the Map Phase, the input data is divided into splits for analysis by map tasks running in parallel across the Hadoop Cluster. The MapReduce framework gets the input from the HDFS. In the Reduce Phase, it uses the results from map tasks as inputs to a set of parallel reduce tasks. The result is again stored in HDFS.

The rest of the paper is organized as follows. Section III discusses the basic workflow of MapReduce. Section IV briefs the concept of HDFS. Section V relates to the implementation of Hadoop NameNode localhost. Section VI discusses the results obtained.

II. RELATED WORK

Hadoop is an industrial scale batch processing distributed computing tool. It has the capability to connect computers with multiple processor cores with a scale ranging from hundreds to thousands. Vast volumes of data can be efficiently distributed across clusters of computers using Hadoop. The Hadoop scale consists of hundreds of gigabytes of data at the least. Hadoop has been built with the capability to manage vast data sets whose size can easily lie between couple of gigabytes to thousands of petabytes. Hadoop provides its solution in the form of a Distributed File System which splits the data and stores it in several different machines. This enables parallel processing of the problem and efficient computation is possible. The design of Hadoop is such that it can efficiently manage vast quantity of data sets by taking advantage of clustered computing or by connecting hundred of machines with processing power in parallel. Theoretically speaking, a single, powerful thousand CPU machine would be much more expensive than thousands of machines with individual CPUs thus making it an easier investment. Hadoop offers a cost effective solution by tying these smaller and cheaper machines together.

After the data is loaded into clusters in Hadoop it is distributed to all the nodes. The HDFS then splits the data into sets which allow management by individual nodes within the cluster. To handle unavailability of data due to failure, each part is also replicated across the cluster. The data is also re-replicated in response to failure of the system. All these parts of data are easily accessible through a universal namespace, despite the parts being distributed and replicated on multiple machines.

III. MAPREDUCE

MapReduce is a tool implemented for managing and processing vast amounts of unstructured data in parallel based on division of a big work item in smaller independent task units. Programs which are Map Reduces are programmed to manage vast amounts of data in parallel. To achieve this, load shedding is required across multiple machines. The main leverage of MAPREDUCE is the tasks of similar nature are grouped together so that same type of data is placed on the same nodes. Doing this saves the sync overhead which might have been caused if tasks were grouped in a random order. MAPREDUCE data elements are immutable i.e if you change input (key, value) in a mapper then it will not be displayed in the input files. Rather it will be taken care in the next execution with the new output values.

- List Processing: In concept, programs which are map reduced convert an array of data coming in as input into an array of data which is the output. The program goes through this process two times, using two functions which are *mapping* and *reduction*.
- List based Mapping: In a map reduce context the first execution phase is the MAPPER which takes

the data elements as input and generates the corresponding output data elements

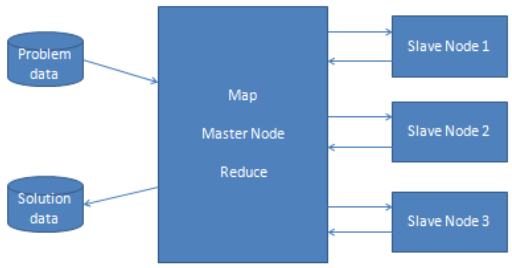

Fig. 1 MapReduce task

IV. HDFS

The HDFS is designed to run on clustered computing platform. It mirrors the already existing file nomenclature in many ways but its differences really make it stands out from existing file systems. One of the salient features of HDFS is that it is fault-tolerant to a very high degree and cost effective. The system allows for greater and faster access to data of an application which is an advantage for processes that require access to large amount of data. HDFS was designed by Apache Nutch project as an infrastructure extension and is now a core component of the project.

HDFS is based on a typical master - slave architecture. An HDFS cluster is made up of a single Name Node and a server acting as a master managing the file access and name space regulations. To simplify the system architecture a single name node exists in a cluster. The Name Node holds & manages whole metadata of HDFS. The design of the systems is such that the data does not flow through the Name Node.

HDFS supports an empherical file structure. Directories can be created by user or an application and files are stored inside those directories. The hierarchy of the file namespace is usually like the previously defined file systems. As such files can be created & removed, moved from one directory to another directory or renamed. HDFS has not yet implemented user quotas and access permissions.

The Name Node handles the file system namespace. It records alterations and its associated properties. A number can also be specified for replicas of a file by the application which must be maintained by the HDFS which is defined as the replication factor and the information is stored in the Name Node.

HDFS is programmed to manage last file stored in large cultures of data mines / structures while ensuring reliability. The way this is managed is by storing files in a sequence of blocks which are the same size, with the last block being an exception. These blocks are then replicated to test fault tolerance in which the size of the block and the replication factors are configurable. An application can then custom specify the number of copies of a file.

V. IMPLEMENTATION

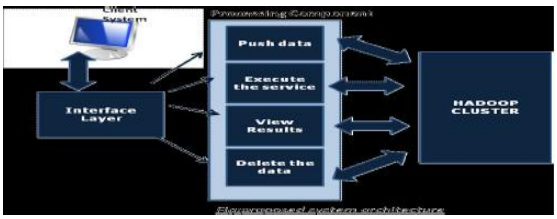

Figure 1. Architectural Design

The client machine is used for accessing the Hadoop Cluster. Interface Layer provides the interface between the client machine and the Processing component. Processing component provides different functionalities to perform the operations like reading, writing, modifying etc.. Hadoop Cluster is used for storing and managing huge volume of data.

Figure 2. Hadoop NameNode localhost

VI. RESULTS

Figure 3. WordCount result

The above result states that the developed system works faster to fetch the expected result.

VII. CONCLUSION

Hadoop with its efficient DFS & programming framework based on concept of mapped reduction, is a powerful tool to manage large data sets. With its map-reduce programming paradigms, overall architecture, ecosystem, fault- tolerance techniques and distributed processing, Hadoop offers a complete infrastructure to handle Big Data. Users must leverage the benefits of Big-Data by adopting Hadoop infrastructure for data processing. However, the issues such as lack of flexible resource management, application deployment support, and multiple data source support pose a challenge to Hadoop‟s adoption. Proper skill training is also needed for achieving large scale data analysis. These challenges must be overcome so that we can tap the full potential of Hadoop data management power.

REFERENCES

[1] Yahoo! Inc, Hadoop Tutorial from Yahoo! Available:http://developer.yahoo.com/hadoop/tutorial/index.html
[2] Jens Dittrich and JorgeArnulfo Quian´eRuiz, " Efficient Big Data processing in Hadoop Mapreduce," Proceedings of the VLDB Endowment, Volume 5 Issue 12, August 2012, Pages 2014-2015
[3] Arnab Nandi, Cong Yu, Philip Bohannon, and Raghu Ramakrishnan, Fellow, IEEE, "Data Cube Materialization and Mining over MapReduce" TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 6, NO. 1, JANUARY 2012
[4] Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D., Rasin, A., and Silberschatz, A. 2010. HadoopDB in action: Building real world applications. In Proceedings of the 36th ACM SIGMOD International Conference on Management of Data (SIGMOD"10).
[5] MapReduce: Simplified Data Processing on Large Clusters. Available at http://labs.google.com/papers/mapreduceosdi04.pdf
[6] Sam Madden, " From Databases to Big Data", IEEE, Internet Computing, May-June 2012.
[7] Yuri Demchenko, Zhiming Zhao, Paola Grosso, Adianto Wibisono, Cees de Laat, "Addressing Big Data Challenges for Scientific Data Infrastructure", IEEE , 4th International Conference on Cloud Computing Technology and Science, 2012.
[8] Jeffrey Dean and Sanjay Ghemawat. "Mapreduce:simplified data processing on large clusters", Commun. ACM, 51(1):107–113, 2008.
[9] Jiang, B.C. Ooi, L. Shi, and S. Wu, "The Performance of MapReduce: An In-Depth Study," Proc. VLDB Endowment, vol. 3, no. 1, pp. 472-483, 2010.
[10] Gillick et al., 2006, Gillick D., Faria A., DeNero J., MapReduce: Distributed Computing for Machine Learning, Berkley, December 18, 2006.
[11] Jeffrey Dean and Sanjay Ghemawat, " MapReduce: a flexible data processing tool," Communications of the ACM, Volume 53 Issue 1, January 2010, Pages 72-77
[12] Apache Hadoop - Petabytes and Terawatts [Online].Available:http://www.youtube.com/watch?v=SS27FhYWf U& feature=related
[13] "Big Data: The next frontier for innovation, competition, and productivity", McKinsey Global Institute, May 2011, p. 11: http://www.mckinsey.com/Insights/MGI/Research/Technology_and_Innovation/Big_data_The_next_frontier_for_innovation.