A Survey of Applications of Big Data Hadoop's Mapreduce Technique and its Limitations

Ch.Srilakshmi
Asst Professor
Department of Information Technology
R.M.D Engineering College,Kavaraipettai,Tamil Nadu,Chennai

Abstract : We are living in an age when an explosive amount of data is being generated every day. Data from sensors, mobile devices, social networking websites, scientific data & enterprises all are contributing to this huge explosion in data. This sudden bombardment has created a vast volume of data in the last few years. Big Data- large chunks of data has become one of the hottest research trends today. Research suggests that tapping the potential of this data can benefit businesses, scientific disciplines and the public sector contributing to their economic gains as well as development in every sphere. The need is to develop efficient systems that can exploit this potential to the maximum, In the Big Data processing, MapReduce technique has been one of the main approaches used for meeting continuously increasing demands on computing resources imposed by massive data sets. The MapReduce paradigm uses massively parallel and distributed execution over a large number of computing nodes.

Keywords: Hadoop MapReduce,Parallel and distributed computing

I.INTRODUCTION

Nowadays, dealing with datasets in the order of terabytes or even petabytes has become a reality. Therefore, processing such big datasets in an efficient way is needed. Hadoop MapReduce is a big data processing framework that has rapidly become the de facto standard in both industry and academia to process large-size data. The main reasons of such popularity are the ease-of-use, scalability, and parallelism of Hadoop MapReduce.

A. HADOOP

Hadoop is an open-source software framework for storing and processing big data in a distributed fashion on large clusters of commodity hardware. Essentially, it accomplishes two tasks: massive data storage and faster processing of that data.

Hadoop Has three core components:

- HDFS – the Java-based distributed file system that can store all kinds of data without prior organization.
- MapReduce – a software programming model for processing large sets of data in parallel.
- YARN – a resource management framework for scheduling and handling resource requests from distributed applications.

Other components are:

- Pig – a platform for e in HDFS. It consists of a compiler for MapReduce programs and a high-level language called Pig Latin. It provides a way to perform data extractions, transformations and loading, and basic analysis without having to write MapReduce programs.
- Hive – a data warehousing and SQL-like query language that presents data in the form of tables. Hive programming is similar to database programming. (It was initially developed by Facebook.)
- HBase – a nonrelational, distributed database that runs on top of Hadoop. HBase tables can serve as input and output for MapReduce jobs.
- Zookeeper – an application that coordinates distributed processes.
- Ambari – a web interface for managing, configuring and testing Hadoop services and components.
- Flume – software that collects, aggregates and moves large amounts of streaming data into HDFS.
- Sqoop – a connection and transfer mechanism that moves data between Hadoop and relational databases.
- Oozie – a Hadoop job scheduler.

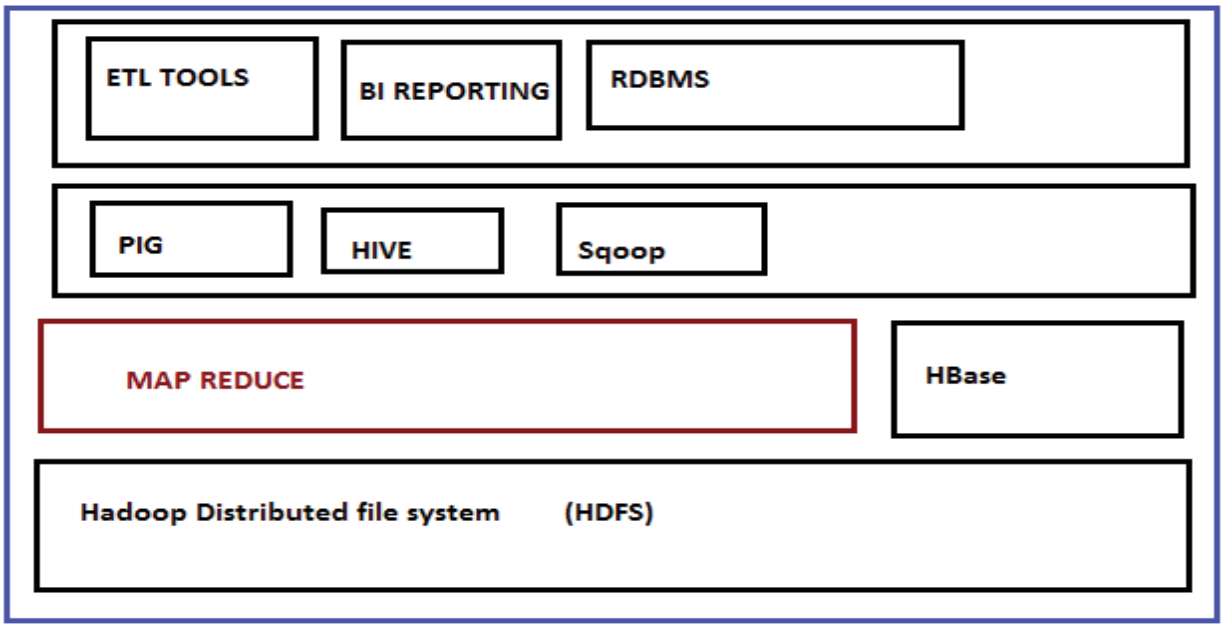

Fig-Hadoop Framework

B.HADOOP's MAPREDUCE

Hadoop MapReduce, is the most popular open source implementation of the MapReduce framework proposed by Google. The essence of Hadoop MapReduce is that users have to define the map and reduce functions. The framework takes care of everything else such as parallelization and scalability.

A Hadoop MapReduce job mainly consists of two user-defined functions: map and reduce. The input of a Hadoop MapReduce job is a set of key-value pairs (k, v) and the map function is called for each of these pairs. The map function produces zero or more intermediate key-value pairs (k′, v′). Then, the Hadoop MapReduce framework groups these intermediate key-value pairs by intermediate key k′ and calls the reduce function for each group. Finally, the reduce function produces zero or more aggregated results.

Map Job: The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The map function produces zero or more intermediate key-value pairs (key′, value′). Map function takes one pair of data with a type in one data domain, and returns a list of pairs in a different domain. The *Map* function is applied in parallel to every pair in the input dataset. This produces a list of pairs. After that, the MapReduce framework collects all pairs with the same key from all lists and groups them together, creating one group for each key.

$$\texttt{Map(k1,v1)} \rightarrow \texttt{list(k2,v2)}$$

Reduce job: The second is the reduce job, which takes the output from a map as input and combines those data tuples into a smaller set of tuples.

The *Reduce* function is then applied in parallel to each group, which in turn produces a collection of values in the same domain. Each *Reduce* call typically produces either one value v3 or an empty return, though one call is allowed to return more than one value. The returns of all calls are collected as the desired result list.

$$\texttt{Reduce(k2, list (v2))} \rightarrow \texttt{list(v3)}$$

As the sequence of the name Map Reduce implies that reduce job is always performed after the map job.Fig.2 shows the MapReduce Framework.

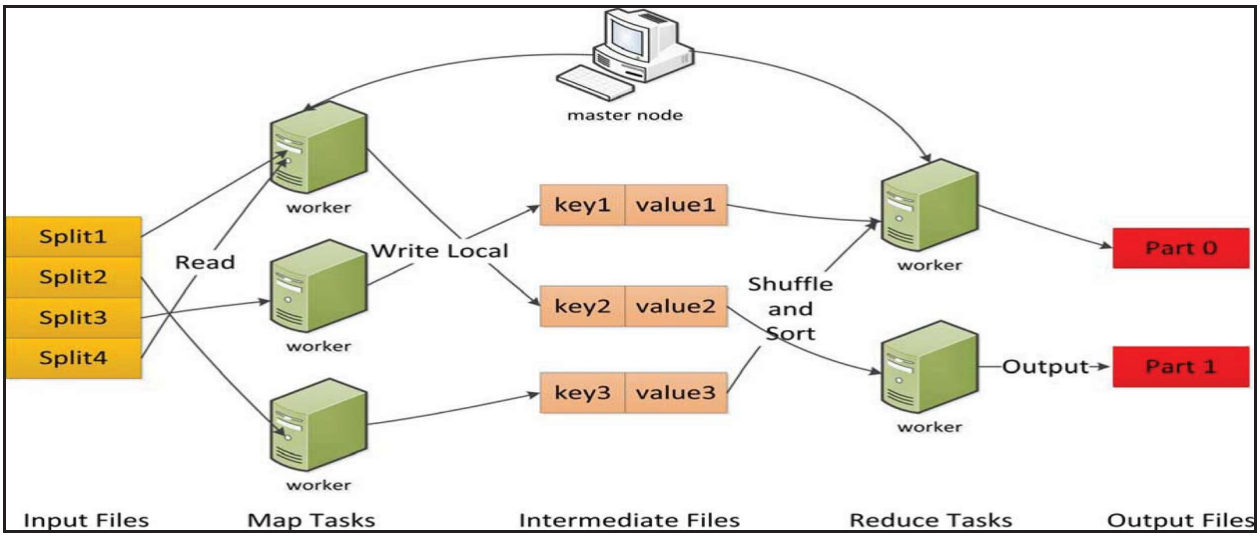

Fig.2-Overview of MapReduce framework

C.CHOOSING BETWEEN M AND R

Let M be Number of tasks and R be Number of reduce tasks .A small Small M and R value are inefficient as speed of processing is low. A Large M and R causes several smaller tasks ,this enabling easier load balancing and faster recovery in terms of failed machine(nodes) which is the advantage. The drawback is that it causes excess overhead is caused and increases complexity as O(M+N) due to scheduling decisions and O(MxN) inter-memory states at master node. This also increases set up cost.

Hence is better to choose M so that the split size is of 64 MB and choose R based as a small multiple of number of workers. Or alternatively can choose R as less than no of workers to finish reduce phase in one wave.

D.AN EXAMPLE OF MAPREDUCE

Assume you have five files, and each file contains two columns ,a key and a value in Hadoop terms that represent a city and the corresponding temperature recorded in that city for the various measurement days. This example is made very simple so it's easy to follow. You can imagine that a real application contain millions or even billions of rows.

Delhi, 31
Mumbai, 32
Chennai, 33
Calcutta, 32
Delhi, 24
Calcutta, 34
Chennai, 38
Delhi, 27
Calcutta, 33
Chennai, 37

Out of all the data collected, to find the maximum temperature for each city across all of the data files (note that each file might have the same city represented multiple times). Using the MapReduce framework, we can break this down into five map tasks, where each mapper works on one of the five files and the mapper task goes through the data and returns the maximum temperature for each city. The results produced from mapper task for the above data would look like this:
(Delhi, 31) (Mumbai, 32) (Chennai, 38) (Calcutta, 34)

II.OVERVIEW

The following sections thows light on how Hodoop's Mapreduce parallel and distributed framework can be used for the following applications

1. Distributed GREP
2. Geospatial Query Processing
3. Ultrafast and Scalable Cone-Beam CT Reconstruction
4. Map reduce for Digital Evaluation Model
5.Other Applications

1.DISTRIBUTED GREP

Distributed grep is used to search for a given pattern in a large number of files. For example, a web administrator can use distributed grep to search web server logs in order to find the top requested pages that match a given pattern,Map function would take input as (input file, line) and generate one of the following outputs.

- An empty list [], if there is no match found
- A key value pair [(line, 1)] if a match is found
  The reduce function would take input as (line, [1, 1, 1, ...]) and generate output as (line, n) where 'n' is the number of 1's in the list.Consider the following grep operation on files in a directory.

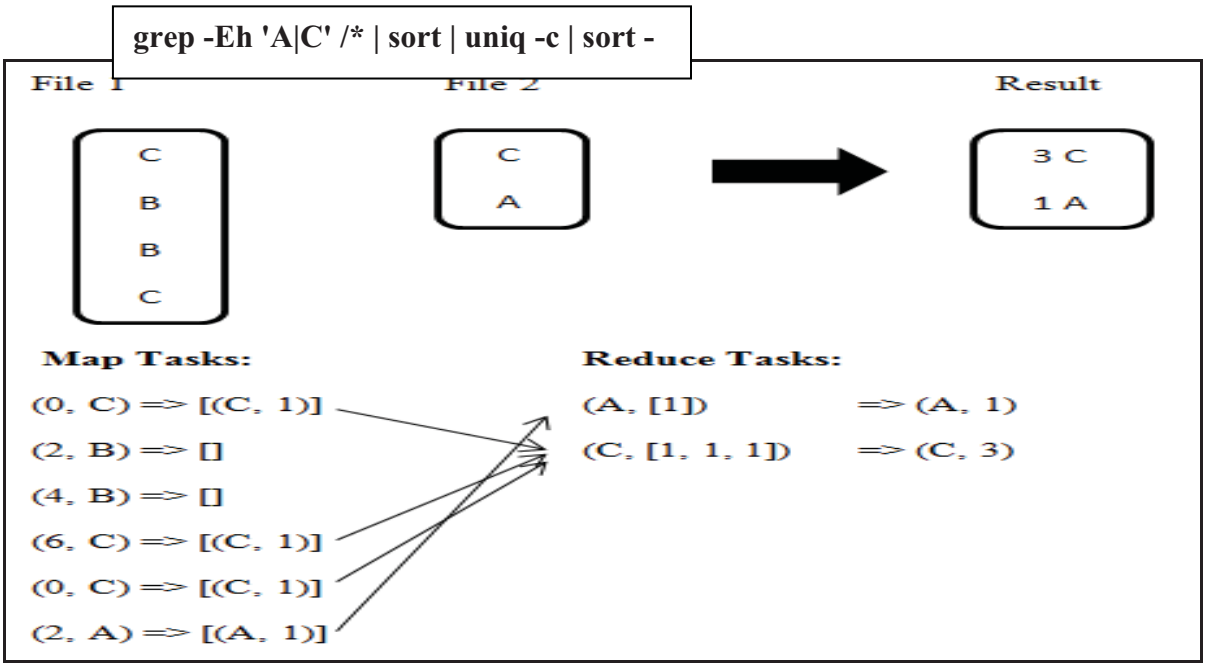

Fig.3-Distributed GREP execution overview

2.GEOSPATIAL QUERY PROCESSING

With the technological advancements in location-based services, there is a huge surge in the amount of geospatial data. Geospatial queries(nearest neighbor queries and reverse nearest neighbor queries) consume lot of computational resources and it is observed that their processing

is heavily parallelizable.

Google Maps uses MapReduce to solve problems like

- Given an intersection, find all roads connecting to it.
- Rendering of the tiles in the map.
- Finding the nearest feature to a given address or current location.

3. ULTRAFAST AND SCALABLE CONE-BEAM CT RECONSTRUCTION

Four-dimensional CT (4DCT) and cone beam CT (CBCT) are widely used in radiation therapy for accurate tumor target definition and localization. However, high-resolution and dynamic image reconstruction is computationally demanding because of the large amount of data processed. Efficient use of these imaging techniques in the clinic requires high-performance computing. To develop a novel ultrafast, scalable and reliable image reconstruction technique for 4D CBCT=CT using a parallel computing framework MapReduce technique can be used. The Feldcamp–Davis–Kress (FDK) algorithm is accelerated by porting it to Hadoop. Map functions were used to filter and back project subsets of projections, and Reduce function to aggregate those partial into the whole volume. MapReduce automatically parallelized the reconstruction process on a large cluster of computer nodes.

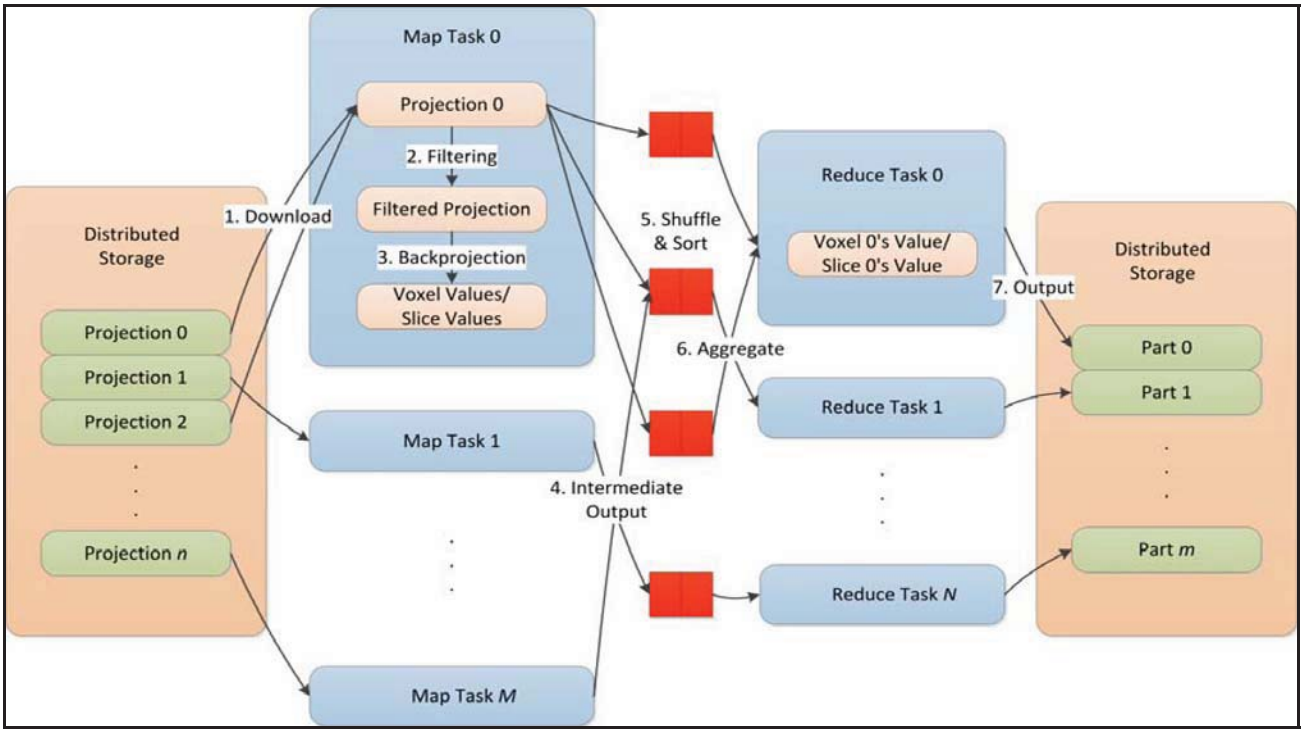

Fig.5-Ultrafast and scalable cone-beam CT reconstruction using FDK algorithm

Voxel-based FDK (FDK-VB), uses Key=Value pairs to transfer individual voxels. In this scheme, the Key encodes the voxel position (x,y,z) using an index calculated as ($z * N^2 + y * N + x$), and the Value is the backprojected image intensity V position (x,y,z). Each Map task emits Key=Value pairs voxel by voxel. The Reduce function accumulates all the intermediate values that share a common same key, producing the final reconstructed volume.

4.MAP REDUCE FOR DIGITAL EVALUATION MODEL

Digital Elevation Models(DEM) are digital 3D representation of the landscape, where each (X, Y) position is represented by a single elevation value. DEMs are used for a range of scientific and engineering applications, including hydrologic modeling, terrain analysis, and infrastructure design. One of the fundamental processing tasks

in the Open Topography system is generation of DEMs from very dense (multiple measurements per square meter) LIDAR (Light detection and Ranging) topography data.

In the Map phase, input points are assigned to corresponding grid cells, and in the Reduce phase the corresponding elevations for each grid cell are computed from the local bins. The reduced outputs are merged, sorted and the DEM is generated in the Arc ASCII grid format.

5.OTHER APPLICATIONS

Table-1: Few other prominent applications of MapReduce

APPLICATIONS	MAP FUNCTION	REDUCE FUNCTION
Count of URL Access Frequency To find the total number of times a URL is accessed.	The map function processes logs of web page requests and outputs <URL, 1>.	The Reduce function adds together all values for the same URL and emits a <URL,total count>pair.
Reverse Web-Link Graph For each URL find all pages (URL) pointing to it. (incoming links).	The map function outputs <target, source> pairs for each link to a target URL found in a page named "source".	The reduce function concatenates the list of all source URLs associated with a given target URL and emits the pair: <target, list(source)>.
Term-Vector per Host A term vector summarizes the most important words that occur in a document or a set of documents as a list of <word, frequency> pairs.	The map function emits a <hostname, term vector> pair for each input document (where the hostname is extracted from the URL of the document).	The reduce function is passed all per-document term vectors for a given host. It adds these term vectors together; throwing away infrequent terms, and then emits a final <hostname, term vector> pair.
Inverted Index Inverted index is an index data structure storing a mapping from content, such as words or numbers, to its locations in a database file, or in a document or a set of documents.	The map function parses each document, and emits a sequence of <word, document ID> pairs.	The reduce function accepts all pairs for a given word, sorts the corresponding document IDs and emits a <word, list(document ID)> pair. The set of all output pairs forms a simple inverted index. It is easy to augment this computation to keep track of word positions.

III.LIMITATIONS OF MAPREDUCE

There are few scenarios where MapReduce programming model cannot be employed. If the computation of a value depends on previously computed values, then MapReduce cannot be used. One good example is the Fibonacci series where each value is summation of the previous two values. i.e., f(k+2) = f(k+1) + f(k). Also, if the data set is small enough to be computed on a single machine, then it is better to do it as a single reduce operation rather than going through the entire map reduce Process. However there are special implementations of MapReduce like the MRLite which are optimized to process moderate size data with dependencies among the various computational steps.

IV.CONCLUSION

Learning from the application studies, we explore the design space for supporting data-intensive and compute-intensive applications on large data-center-scale computer systems. Traditional data processing and storage approaches are facing many challenges in meeting the continuously increasing computing demands of Big Data. This work focused on MapReduce, one of the key enabling approaches for meeting Big Data demands by means of highly parallel processing on a large number of commodity nodes.

REFERENCES

[1] Jens Dittrich, Stefan Richter and Stefan Schuh, " Efficient OR Hadoop: Why Not Both?," Datenbank-Spektrum, Volume 13, Issue 1 , pp 17-22

[2] Humbetov, S, "Data-Intensive Computing with Map-reduce and Hadoop," in Proc. 2012 Application of Information and Communication Technologies (AICT), IEEE ,6th International Conference pp. 5

[3] Hadoop Tutorial, Apache Software Foundation, 2014, Available:http://hadoop.apache.org/

[4] Sherif Sakr, Anna Liu and Ayman G. Fayoumi, " The family of mapreduce and large-scale data processing systems," ACM Computing Surveys, Volume 46 Issue 1, October 2013, Article No. 11

[5] Aditya B. Patel, Manashvi Birla and Ushma Nair, " Addressing Big Data Problem Using Hadoop and Map Reduce," in Proc. 2012 Nirma University International Conference On Engineering, pp. 1-5.

[6] Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D., Rasin, A., and Silberschatz, A. 2009. HadoopDB: An architectural hybrid of mapreduce and dbms technologies for analytical workloads. Proc. VLDB Endow. 2, 1,922–933.

[7] Jyoti Nandimath, Ankur Patil, Ekata Banerjee,Pratima Kakade and Saumitra Vaidya, " Big Data Analysis Using Apache Hadoop," IEEE IRI 2013, August 14-16, 2013, San Francisco, California, USA

[8] MapReduce: Simplified Data Processing on Large Clusters. Available at http://labs.google.com/papers/mapreduceosdi04.pdf

[9] Stephen Kaisler, Frank Armour, J. Alberto Espinosa, William Money,“Big Data: Issues and Challenges Moving Forward”, IEEE, 46th Hawaii International Conference on System Sciences, 2013.