Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 5, Issue. 2, February 2016, pg.224 – 229

An Improved Version of Big Data Classification and Clustering using Graph Search Technique

Sindhujaa N[1] , Vanitha C N[2], Subaira A S[3]

[1,2,3] Assistant Professor, Mahendra College of Engineering, Salem, Tamilnadu, India
sindhujaan@mahendracollege.com
vanithacn@mahendracollege.com
subairaas@mahendracollege.com

Abstract: The Big data is categories as its sheer Volume, Variety, Velocity and Veracity. Most of the data is unstructured, quasi structured or semi structured and it is heterogeneous in nature. The volume and the heterogeneity of data with the speed it is generated, makes it difficult for the present computing infrastructure to handle Big Data. Traditional data management, warehousing and study systems fall short of tools to analyze this data. Due to its precise nature of Big Data, it is stored in distributed file system architectures. Hadoop and HDFS by Apache is widely used for storing and managing Big Data. Analyzing Big Data is a challenging task as it involves large distributed file systems which should be fault tolerant. Map Reduce is widely been used for the efficient analysis of Big Data DBMS techniques like Joins and Indexing and other techniques like graph search is mainly used for classification and clustering of Big Data. This techniques are being adopted to be used in Map Reduce. Map Reduce is a technique which makes use of file indexing with mapping, sorting, shuffling, reducing etc.

Keywords: Big Data Analysis, Big Data Management, Map Reduce HDFS.

1. Introduction

Big Data is a heterogeneous mix of data both structured (traditional datasets –in rows and columns like DBMS tables, CSV's and XLS's) and unstructured data like e-mail attachments, manuals, images, PDF documents, medical records such as x-rays, ECG and MRI images, forms, rich media like graphics, video and audio, contacts, forms and documents. Businesses are primarily concerned with managing unstructured data, because over 80 percent of enterprise data is unstructured [2] and require significant storage space and effort to manage.“Big data” refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyse [3].

Big data analytics is the area where advanced analytic techniques operate on big data sets. It is really about two things, Big data and Analytics and how the two have teamed up to create one of the most profound trends in business intelligence (BI) [4]. Map Reduce by itself is capable for analysing large distributed data sets; but due to the heterogeneity, velocity and volume of Big Data, it is a challenge for traditional data analysis and management tools [1] [2]. A problem with Big Data is that they use NoSQL and has no Data Description Language (DDL) and it supports transaction processing. Also, web-scale data is not universal and it is heterogeneous. For analysis of Big Data, database integration and cleaning is much harder than the traditional mining approaches [4]. Parallel processing and distributed computing is becoming a standard procedure which are nearly non-existent in RDBMS. Map Reduce has following characteristics [9]; it supports Parallel and distributed processing, it is simple and its architecture is shared-nothing which has commodity diverse hardware (big cluster).Its functions are programmed in a high-level programming language (e.g. Java, Python) and it is flexible. Query processing is done through NoSQL integrated in HDFS as Hive tool [10]. Analytics helps to discover what has changed and the possible solutions. Second, advanced analytics is the best way to discover more business opportunities, new customer segments, identify the best suppliers, associate products of affinity, understand sales seasonality[5] etc. Traditional experience in data warehousing, reporting, and online analytic processing (OLAP) is different for advanced forms of analytics [6]. Organizations are implementing specific forms of analytics, particularly called advanced analytics. These are an collection of related techniques and tool types, usually including predictive analytics, data mining, statistical analysis, complex SQL, data visualization, artificial intelligence, natural language processing. Database analytics platforms such as MapReduce, in-database analytics, in-memory databases, and columnar data stores [6] [9] are used for standardizing them.

With big data analytics, the user is trying to discover new business facts that no one in the enterprise knew before, a better term would be “discovery analytics. To do that, the analyst needs large volumes of data with plenty of detail. This is often data that the enterprise has not yet tapped for analytics example, the log data. The analyst might mix that data with historic data from a data warehouse and would discover for example, new change behaviour in a subset of the customer base. The discovery would lead to a metric, report, analytic model, or some other product of BI, through which the company could track and predict the new form of customer behavioural change.

2. CURRENT CHALLENGES IN BIG DATA

“Big Data” started to become a major issue in the late 1990‟ s due to the impact of the world-wide Web and a resulting need to index and query its rapidly mushrooming content. Database technology (including parallel databases) was considered for the task, but was found to be neither well-suited nor cost-effective [5] for those purposes. The turn of the millennium then brought further challenges as companies began to use information such as the topology of the Web and users‟ search histories in order to provide increasingly useful search results, as well as more effectively-targeted advertising to display alongside and fund those results. Google’ s technical response to the challenges of Web-scale data management and analysis was simple, by database standards, but kicked off what has become the modern “Big Data” revolution in the systems world [3]. To handle the challenge of Web-scale storage, the Google File System (GFS) was created [13]. GFS provides clients with the familiar OS-level byte-stream abstraction, but it does so for extremely large files whose content can span hundreds of machines in shared-nothing clusters created using inexpensive commodity hardware [5]. To handle the challenge of processing the data in such large files, Google pioneered its Map Reduce programming model and platform [1][11]. This model, characterized by some as “parallel programming for dummies”, enabled Google developers to process large collections of data by writing two user-defined functions, map and reduce, that the Map Reduce framework applies to the instances (map) and sorted groups of instances that share a common key (reduce) – similar to the sort of partitioned parallelism utilized in shared-nothing parallel query processing.

Driven by very similar requirements, software developers at Yahoo!, Facebook, and other large Web companies followed suit. Taking Google‟ s GFS and Map Reduce papers as rough technical specifications, open-source equivalents were developed, and the Apache Hadoop Map Reduce platform and its underlying file system (HDFS, the Hadoop Distributed File System) were born [1] [12]. The Hadoop system has quickly gained traction, and it is now widely used for use cases including Web indexing, clickstream and log analysis, and certain large-scale information extraction and machine learning tasks. Soon tired of the low-level nature of the Map Reduce programming model, the Hadoop community developed a set of higher-level declarative languages for writing queries and data analysis pipelines that are compiled into Map Reduce jobs and then executed on the Hadoop Map Reduce platform. Popular languages include Pig from Yahoo! [13], Jaql from IBM [8], and Hive from Facebook [3]. Pig is relational-algebra-like in nature, and is reportedly used for over 60% of Yahoo!‟ s MapReduce use cases; Hive is SQL-inspired and reported to be used for over

90% of the Facebook Map Reduce use cases. Microsoft‟ s technologies include a parallel runtime system called Dryad and two higher-level programming models, Dryad LINQ and the SQLlike SCOPE language [8], which utilizes Dryad under the covers. Interestingly, Microsoft has also recently announced that its future "Big Data" strategy includes support for Hadoop[14].

3. HADOOP DISTRIBUTED FILE SYSTEM

Hadoop is a scalable, open source, fault-tolerant Virtual Grid operating system architecture for data storage and processing. It runs on commodity hardware, it uses HDFS which is fault-tolerant high-bandwidth clustered storage architecture. It runs MapReduce for distributed data processing and is works with structured and unstructured data.

Figure1Illustrates the layers found in the HDFS clusters [3] [4]. At the bottom of the Hadoop software stack is HDFS, a distributed file system in which each file appears as a (very large) contiguous and randomly addressable sequence of bytes. For batch analytics, the middle layer of the stack is the Hadoop Map Reduce system, which applies map operations to the data in partitions of an HDFS file, sorts and redistributes the results based on key values in the output data, and then performs reduce operations on the groups of output data items with matching keys from the map phase of the job. For applications just needing basic key-based record management operations, the HBase store (layered on top of HDFS) is available as a key-value layer in the Hadoop stack.

As indicated in the figure, the contents of HBase can either be directly accessed and manipulated by a client application or accessed via Hadoop for analytical needs. Many users of the Hadoop stack prefer the use of a declarative language over the bare MapReduce programming model.

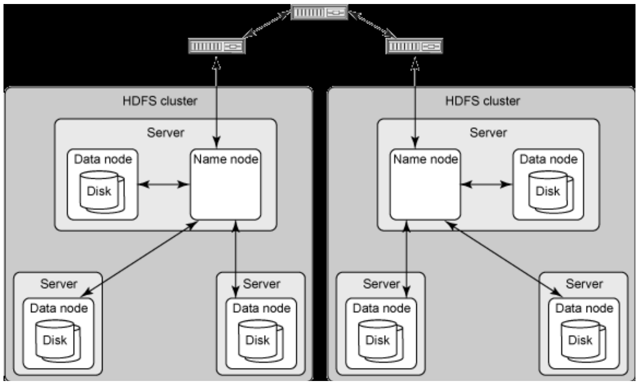

Figure 1: HDFS Clusters

High-level language compilers (Pig and Hive) are thus the topmost layer in the Hadoop software stack for such clients.

The relevancy between the traditional experience in data warehousing, reporting, and online analytic processing (OLAP) and advanced analytics with collection of related techniques like data mining with DBMS, artificial intelligence, machine learning, and database analytics platforms such as MapReduce and Hadoop over HDFS [4] [9]. Figure 1 shows the HDFS clusters implementation with Hadoop. It can be seen that HDFS has distributed the task over two parallel clusters with one server and two slave nodes each. Data analysis tasks are distributed in these clusters.

4. ANALYSIS OF BIG DATA

Heterogeneity, scale, timeliness, complexity, and privacy problems with Big Data hamper the progress at all phases of the process that can create value from data. Much data today is not natively in structured format; for example, tweets and blogs are weakly structured pieces of text, while images and video are structured for storage and display, but not for semantic content and search: transforming such content into a structured format for later analysis is a major challenge [15]. The value of data enhances when it can be linked with other data, thus data integration is a major creator of value.

For handling the velocity and heterogeneity of data, tools like Hive, Pig and Mahout are used which are parts of Hadoop and HDFS framework. It is interesting to note that for all the tools used, Hadoop over HDFS is the underlying architecture. Oozie and EMR with Flume and Zookeeper are used for handling the volume and veracity of data, which are standard Big Data management tools. The layer with their specified tools forms the bedrock for Big Data management and analysis framework.

Big Data analysis is a clear bottleneck in many applications, both due to lack of scalability of the underlying algorithms and due to the complexity of the data that needs to be analysed. Finally, presentation of the results and its interpretation by non-technical domain experts is crucial to extracting actionable knowledge as most of the BI related jobs are handled by statisticians and not software experts.

The Big Data analysis tools which are used for efficient and precise data analysis and management jobs. The Big Data Analysis and management setup can be understood through the layered structured defined in the figure. The data storage part is dominated by the HDFS distributed file system architecture; other mentioned architectures available are Amazon Web Service (AWS) [9], Hbase and CloudStore etc. The data processing tasks for all the tools is Map Reduce; we can comfortably say that it is the de-facto Data processing tool used in the Big Data paradigm.

5. MAP REDUCE

MapReduce [1-2] is a programming model for processing large-scale datasets in computer clusters. The MapReduce programming model consists of two functions, map() and reduce(). Users can implement their own processing logic by specifying a customized map() and reduce() function. The map() function takes an input key/value pair and produces a list of intermediate key/value pairs. The MapReduce runtime system groups together all intermediate pairs based on the intermediate keys and passes them to reduce() function for producing the final results.

Map (in_key, in_value)->list(out_key,intermediate_value) **Reduce** (out_key,list(intermediate_value))>list(out_value)
The signatures of map() and reduce() are as follows :

map (k1,v1) ! list(k2,v2) and
reduce (k2,list(v2)) ! list(v2)

A MapReduce cluster employs a master-slave architecture where one master node manages a number of slave nodes [13]. In the Hadoop, the master node is called JobTracker and the slave node is called TaskTracker as shown in the figure 2. Hadoop launches a MapReduce job by first splitting the input dataset into even-sized data blocks. Each data block is then scheduled to one TaskTracker node and is processed by a map task. The TaskTracker node notifies the JobTracker when it is idle. The scheduler then assigns new tasks to it. The scheduler takes data locality into account when it disseminates data blocks.

Map Reduce Architecture and Working It always tries to assign a local data block to a TaskTracker. If the attempt fails, the scheduler will assign a rack-local or random data block to the TaskTracker instead. When map() functions complete, the runtime system groups all intermediate pairs and launches a set of reduce tasks to produce the final results. Large scale data processing is a difficult task, managing hundreds or thousands of processors and managing parallelization and distributed environments makes is more difficult. Map Reduce provides solution to the mentioned issues, as is supports distributed and parallel I/O scheduling, it is fault tolerant and supports scalability and i has inbuilt processes for status and monitoring of heterogeneous and large datasets as in Big Data [6].

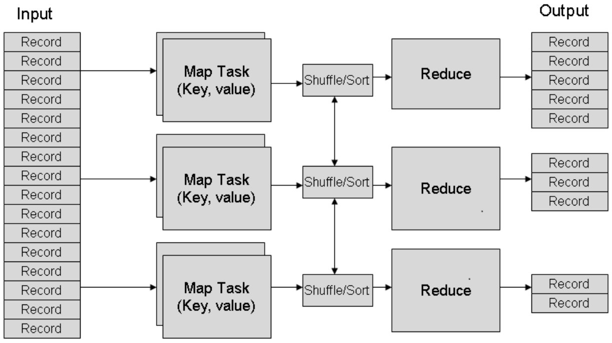

Figure 2. Map Reduce Architecture and Working

6. CONCLUSION

The need to process enormous quantities of data has never been greater. Not only are terabyte- and petabyte-scale datasets rapidly becoming commonplace, but there is consensus that great value lies buried in them, waiting to be unlocked by the right computational tools. In the commercial sphere, business intelligence, driven by the ability to gather data from a dizzying array of sources. Big Data analysis tools like Map Reduce over Hadoop and HDFS, promises to help organizations better understand their customers and the marketplace, hopefully leading to better business decisions and competitive advantages [6]. For engineers building information processing tools and applications, large and heterogeneous datasets which are generating continuous flow of data, lead to more effective algorithms for a wide range of tasks, from machine translation to spam detection. In the natural and physical sciences, the ability to analyse massive amounts of data may provide the key to unlocking the secrets of the cosmos or the mysteries of life. MapReduce can be exploited to solve a variety of problems related to text processing at scales that would have been unthinkable a few years ago [15]. No tool no matter how powerful or flexible can be perfectly adapted to every task. There are many examples of algorithms that depend crucially on the existence of shared global state during processing, making them difficult to implement in MapReduce (since the single opportunity for global synchronization in MapReduce is the barrier between the map and reduce phases of processing). Implementing online learning algorithms in MapReduce is problematic [14]. The model parameters in a learning algorithm can be viewed as shared global state, which must be updated as the model is evaluated against training data. All processes performing the evaluation (presumably the mappers) must have access to this state. In a batch learner, where updates occur in one or more reducers (or, alternatively, in the driver code), synchronization of this resource is enforced by the MapReduce framework. However, with online learning, these updates must occur after processing smaller numbers of instances. This means that the framework must be altered to support faster processing of smaller datasets, which goes against the design choices of most existing MapReduce implementations. Since MapReduce was specifically optimized for batch operations over large amounts of data, such a style of computation would likely result in insufficient use of resources [2]. In Hadoop, for example, map and reduce tasks have considerable start-up costs.

7. ADVANCEMENTS

Streaming algorithms [9] represent an alternative programming model for dealing with large volumes of data with limited computational and storage resources. This model assumes that data are presented to the algorithm as one or more streams of inputs that are processed in order, and only once. Stream processing is very attractive for working with time-series data (news feeds, tweets, sensor readings, etc.), which is difficult in MapReduce (once again, given its batch-oriented design). Another system worth mentioning is Pregel [11], which implements a programming model inspired by Valiant's Bulk Synchronous Parallel (BSP) model. Pregel was specially designed for large-scale graph algorithms, but unfortunately there are few published details at present.

Pig [15], which is inspired by Google [13], can be described as a data analytics platform that provides a lightweight scripting language for manipulating large datasets. Although Pig scripts (in a language called Pig Latin) are ultimately converted into Hadoop jobs by Pig's execution engine through joins, allow developers to specify data transformations (filtering, joining, grouping, etc.) at a much higher level. Similarly, Hive [10], another open-source project, provides an abstraction on top of Hadoop that allows users to issue SQL queries against large relational datasets stored in HDFS. Hive queries, in HiveQL are compiled down to Hadoop jobs by the Hive query engine. Therefore, the system provides a data analysis tool for users who are already comfortable with relational databases, while simultaneously taking advantage of Hadoop's data processing capabilities [11]. The power of MapReduce derives from providing an abstraction that allows developers to harness the power of large clusters but abstractions manage complexity by hiding details and presenting well-defined behaviours to users of those abstractions. This process makes certain tasks easier, but others more difficult, if not impossible. MapReduce is certainly no exception to this generalization, even within the Hadoop/HDFS/MapReduce ecosystem; it is already observed the development of alternative approaches for expressing distributed computations. For example, there can be a third merge phase after map and reduce to better support relational operations. Join processing mentioned n the paper can also tackle the Map Reduce tasks effectively. The future directions in Big Data analysis gives a very encouraging picture as the tools are build on the existing paradigm of HDFS and Hadoop, overcoming the existing drawback of the present systems and the advantages it provides over the traditional data analysis tools.

REFERENCES

[1] Jefry Dean and Sanjay Ghemwat, MapReduce:A Flexible Data Processing Tool, Communications of the ACM, Volume 53, Issuse.1,January 2015, pp 72-77.

[2] Jefry Dean and Sanjay Ghemwat,.MapReduce: Simplified data processing on large clusters, Communications of the ACM, Volume 51 pp. 107–113, 2015.

[3] Brad Brown, Michael Chui, and James Manyika, Are you ready for the era of „big data‟ ?,McKinseyQuaterly,Mckinsey Global Institute, October 2014.

[4] DunrenChe, MejdlSafran, and ZhiyongPeng, From Big Data to Big Data Mining: Challenges, Issues, and Opportunities, DASFAA Workshops 2013, LNCS 7827, pp. 1–15, 2013.

[5] MarcinJedyk, MAKING BIG DATA, SMALL, Using distributed systems for processing, analysing and managing large huge data sets, Software Professional‟ s Network, Cheshire Data systems Ltd.

[6] OnurSavas, YalinSagduyu, Julia Deng, and Jason Li,Tactical Big Data Analytics: Challenges, Use Cases and Solutions, Big Data Analytics Workshop in conjunction with ACM Sigmetrics 2013,June 21, 2013.

[7] Kyuseok Shim, MapReduce Algorithms for Big Data Analysis, DNIS 2013, LNCS 7813, pp. 44–48, 2013.

[8]Raja.Appuswamy,ChristosGkantsidis,DushyanthNarayanan,OrionHodson,AntonyRowstron, Nobody ever got fired for buying a cluster, Microsoft Research, Cambridge, UK, Technical Report,MSR-TR-2013-2

[9] Carlos Ordonez, Algorithms and Optimizations for Big Data Analytics: Cubes, Tech Talks,University of Houston, USA.

[10] Spyros Blanas, Jignesh M. Patel,VukErcegovac, Jun Rao,Eugene J. Shekita, YuanyuanTian, A Comparison of Join Algorithms for Log Processing in MapReduce, SIGMOD‟ 10, June 6–11, 2012, Indianapolis, Indiana, USA.

[11] Tyson Condie, Neil Conway, Peter Alvaro, Joseph M. Hellerstein,JohnGerth, Justin Talbot,KhaledElmeleegy, Russell Sears, Online Aggregation and Continuous Query support in
MapReduce, SIGMOD‟ 10, June 6–11, 2011, Indianapolis, Indiana, USA.

[12] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in USENIXSymposium on Operating Systems Design and Implementation, San Francisco, CA, Dec. 2014, pp. 137–150.

[13] S. Ghemawat, H. Gobioff, and S. Leung, "The Google File System." in ACM Symposium on Operating Systems Principles, Lake George, NY, Oct 2013, pp. 29 – 43.

[14] HADOOP-3759: Provide ability to run memory intensive jobs without affecting other running tasks on the nodes. https://issues.apache.org/jira/browse/HADOOP-3759

[15] VinayakBorkar, Michael J. Carey, Chen Li, Inside "Big Data Management":Ogres, Onions, or Parfaits?, EDBT/ICDT 2014 Joint Conference Berlin, Germany,2014 ACM 2014, pp 3-