Asian Journal of Computer Science and Technology
ISSN: 2249-0701 (P) Vol.8 No.S3, 2019, pp.35-40

DOI: https://doi.org/10.51983/ajcst-2019.8.S3.2091

A Survey on Big Data Analytics Using HADOOP

S. Mamatha[1] and T. Sudha[2]
[1]Research Scholar, [2]Professor, [1&2]Department of Computer Science,
Sri Padmavati Mahila Visva Vidyalayam (Women's University), Tirupati, Andhra Pradesh, India
E-Mail: thatimakula-sudha@yahoo.com
(Received 26 March 2019; Revised 7 April 2019; Accepted 17 April 2019; Available online 24 April 2019)

Abstract **- In this digital world, as organizations are evolving rapidly with data centric asset the explosion of data and size of the databases have been growing exponentially. Data is generated from different sources like business processes, transactions, social networking sites, web servers, etc. and remains in structured as well as unstructured form. The term — Big data is used for large data sets whose size is beyond the ability of commonly used software tools to capture, manage, and process the data within a tolerable elapsed time. Big data varies in size ranging from a few dozen terabytes to many petabytes of data in a single data set. Difficulties include capture, storage, search, sharing, analytics and visualizing. Big data is available in structured, unstructured and semi-structured data format. Relational database fails to store this multi-structured data. Apache Hadoop is efficient, robust, reliable and scalable framework to store, process, transforms and extracts big data. Hadoop framework is open source and fee software which is available at Apache Software Foundation. In this paper we will present Hadoop, HDFS, Map Reduce and c-means big data algorithm to minimize efforts of big data analysis using Map Reduce code. The objective of this paper is to summarize the state-of-the-art efforts in clinical big data analytics and highlight what might be needed to enhance the outcomes of clinical big data analytics tools and related fields.**

Keywords: **Big Data, Mining, Heterogeneity, HDFS, Map Reduce, HADOOP, Cluster, Name node, Data Node**

I. INTRODUCTION

Big data is the term used to describe huge datasets having the - 3 V's definition: volume, variety, velocity and value (e.g. medical images, electronic medical records (EMR), biometrics data, etc.). Such datasets present problems with storage, analysis, and visualization [1,2]. To deal with these challenges, new software programming frameworks to multithread computing tasks have been developed [2-4].

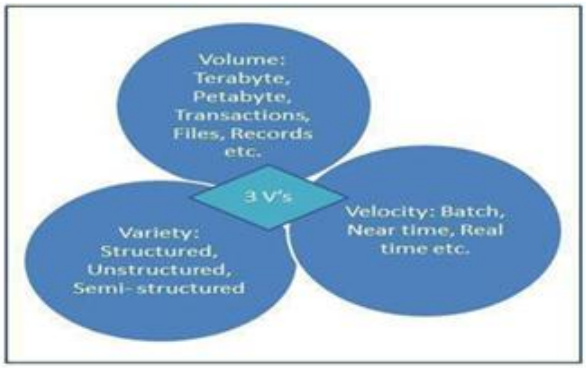

Fig. 1 Three V's of Big Data

These programming frameworks are designed to get their parallelism not from a supercomputer, but from computing clusters: large collections of commodity hardware, including conventional processors (computing nodes) connected by Ethernet cables or inexpensive switches. These software programming frameworks begin with a new form of file system, known as a distributed file system (DFS) [3,4], which features much larger units than the disk blocks in a conventional operating system. DFS also provides replication of data or redundancy to protect against the frequent media failures that occur when data is distributed over potentially thousands of low cost computing nodes [3].

The goal of this review is to summarize the potential and expanding usage of MapReduce on top of the Hadoop platform in the processing of clinical big data.

A secondary objective is to highlight the potential benefits of predictive and prescriptive clinical big data analytics. These types of analytics are needed for better usage and optimization of resources [5,6]. Types of analytics Analytics is a term used to describe various goals and techniques of processing a dataset.

II. COMPUTING SYSTEMS

A. Distributed System: A distributed system [3] is a setup in which several independent computers (computing nodes) participate in solving the problem of processing a large volume of and variety of structured/semi-structured/unstructured data.

B. Grid Computing System: The grid computing system [7] is a way to utilize resources (e.g. CPUs, storage of computer systems across a worldwide network, etc.) to function as a flexible, pervasive, and inexpensive accessible pool of computing resources that can be used on demand by any task. Furthermore the system architecture of GPUs may not be suitable for the MapReduce architecture and may require a great deal of modification [9]. The basic differences between grid computing and distributed computing systems are:

1. A distributed computing system manages hundreds or thousands of computer systems, which are limited in processing resources (e.g. memory, CPU, storage, etc.). However the grid computing system is concerned

about efficient usage of heterogeneous systems with optimal workload management servers, networks, storage, etc.

2. A grid computing system is dedicated to support computation across a variety of administrative domains, which makes it different from the traditional distributed computing system.

C. Map Reduce Programming Using HADOOP: On top of the DFS, many different higher-level programming frameworks have been developed. The most commonly implemented programming framework is the MapReduce framework [4, 11, 12]. MapReduce is an emerging programming framework for data-intensive applications proposed by Google. MapReduce borrows ideas from functional programming [12], where the programmer defines Map and Reduce tasks to process large sets of distributed data. Implementations of MapReduce [11] enable many of the most common calculations on large-scale data to be performed on computing clusters efficiently and in a way that is tolerant of hardware failures during computation. However MapReduce is not suitable for online transactions [11, 12].

The key strengths of the MapReduce programming framework are the high degree of parallelism combined with the simplicity of the programming framework and its applicability to a large variety of application domains [4,11]. This requires dividing the workload across a large numberof machines. The degree of parallelism depends on the input data size. The map function processes the input pairs (key1, value1) returning some other intermediary pairs (key2, value2). Then the intermediary pairs are grouped together according to their key. The reduce function will output some new key-value pairs of the form (key3, value3). Figure shows an example of a MapReduce algorithm used to count words in a file. In this example the map input key is the provided data chunk with a value of 1.

The map output key is the word itself and the value is 1 every time the word exists in the processed data chunk. The reducers perform the aggregation of the key-values pair output from the maps and output a single value for every key, which in this case is a count for every word. High performance is achieved by breaking the processing into small units of work that can be run in parallel across potentially hundreds or thousands of nodes in the cluster. Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system [3,4]. MapReduce programs are usually written in Java; however they can also be coded in languages such as C++, PHP Perl, Python, Ruby, R, etc. These programs may process data stored in different file and database systems.

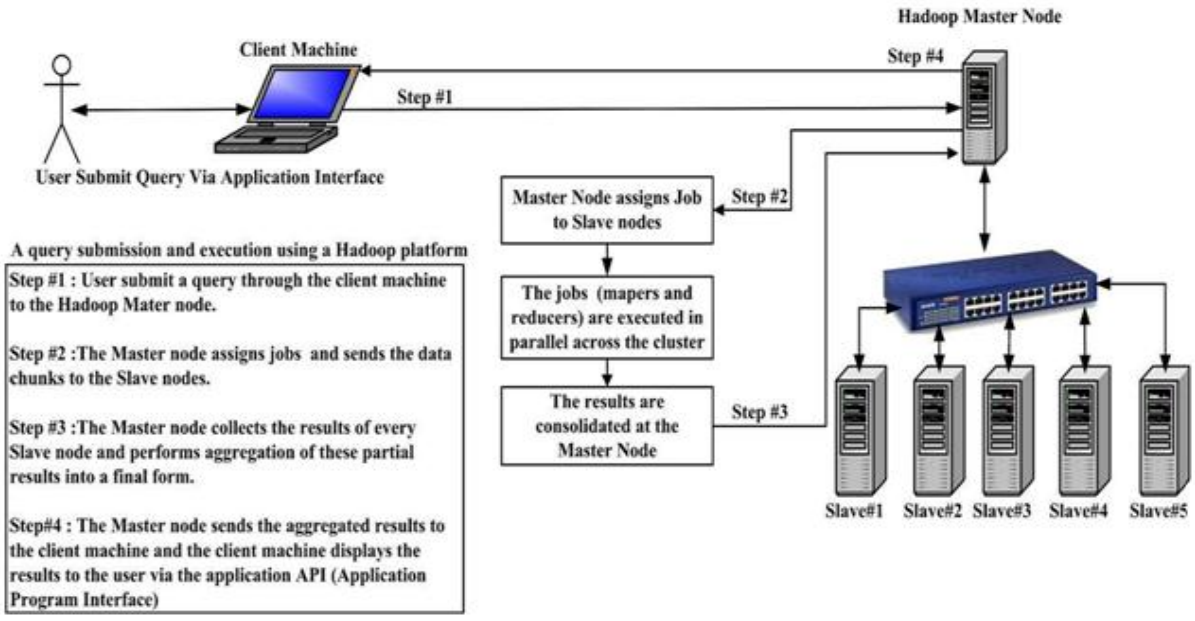

Fig. 2 The Architecture of the Hadoop HDFS cluster

The hadoop platform Hadoop [13-15] is an open source software implementation of the MapReduce framework for running applications on large clusters built of commodity hardware from Apache [16]. Hadoop is a platform that provides both distributed storage and computational capabilities. Hadoop differs from other distributed system schemes in its philosophy toward data. A traditional distributed system requires repeat transmissions of data between clients and servers [3]. This works fine for computationally intensive work, but for data-intensive processing, the size of data becomes too large to be moved around easily. Hadoop focuses on moving code to data

instead of vice versa [13, 14]. The client (NameNode) sends only the MapReduce programs to be executed, and these programs are usually small (often in kilobytes). More importantly, the move-code-to-data philosophy applies within the Hadoop cluster itself. Data is broken up and distributed across the cluster, and as much as possible, computation on a chunk of data takes place on the same machine where that chunk of data resides.

D. Proposed Work: Data mining is a process of extracting information from the raw data and Cloud computing provides scalable and flexible infrastructure which provides everything as a service. By integrating data mining and cloud computing provides agility and quick access to technology. The traditional database system and RDBMS are not able to mine large data set and theexisting data mining algorithms are not capable to mine big data.HADOOP is a distributed master–slave architecture that consists of the Hadoop Distributed File System (HDFS) for storage and the MapReduce programming framework for computational capabilities. The HDFS stores data on the computing nodes providing a very high aggregate bandwidth across the cluster. Traits inherent to Hadoop are data partitioning and parallel computation of large datasets. MapReduce framework [4,11,12]. MapReduce is an emerging programming framework for data-intensive applications proposed by Google. MapReduce borrows ideas from functional programming [12], where the programmer defines Map and Reduce tasks to process large sets of distributed data. Implementations of MapReduce [11] enable many of the most common calculations on large-scale data to be performed on computing clusters efficiently.

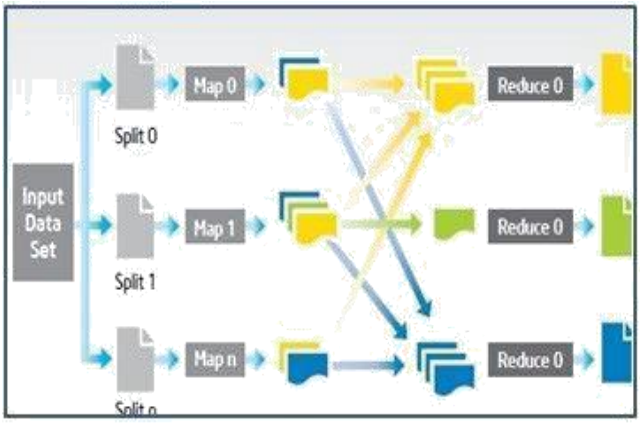

Fig. 3 MapReduce Framework

Map Reduce framework allows users to define two functions, map and reduce, to process a large number data entries. Here a scalable parallel clustering algorithm is used to overcome the problem in clustering large dataset with high dimension. The clustering is the Fuzzy C-Means (FCM) clustering algorithm which is applied to the each randomly divided set of input data[22]. Then finally the resultant cluster is obtained at the output. The fig. shown below represents the architecture of the proposed scalable parallel clustering algorithm.

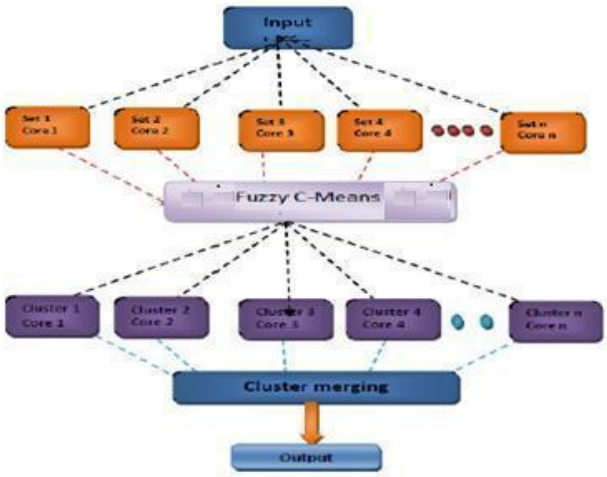

Fig. 4 Parallel architecture of the proposed algorithm

E. Partitioning the Input Large Dataset: Let the input be the large dataset with a size of M * N. In this processing, input large dataset using fuzzy c-means clustering algorithm is difficult. So dividing the input dataset randomly in to small subsets of data with equal size will make system better. So further in this proposed system the input large data set is divided into N number of subset, based on number of cores available in the system, S={S1, S2,.....,Sn }where N is the total number of sets with equalsize. Here the each subset of data is clustered into clusters using a standard and efficient clustering algorithm called Fuzzy C-Means.

Programmatically used in fork method in Java. The each single data subset S consist of a vector of d measurements, where X = (x1 , x2 , ,.....). The attribute of an individual data set is represented as i x and d represents the dimensionality of the vector. The Fuzzy C-Means [30] is applied to the each subset of dataset for clustering the input dataset n d in to k-clusters. Fuzzy C-Means clustering method is applied to divided subset of data.

The PFCM is one of the most efficient parallel clustering methods. Let the unlabelled data set is S= {S1, S2... Sn}, which is further clustered in to a group of k-clusters using CMeans clustering method.

III. IMPLEMENTAION OF BIG DATA MINING ON HDFS, MAPREDUCE

After configuration HDFS, we are configuring our application for big data mining. In this step, we are using configured an iso image file of our virtual machine. This setup can be run over any virtual platform; we need not to configure it again for other systems. Over this architecture we configure our Big Data Mining application. This application will process map reduce's huge data according to the filter criteria given to the application. We need not to use upper layer tools of HDFS architecture for further implementation. We are using lamp server which is configured on this architecture for our application for data retrieval.

A. Configuring Hadoop: Configuring and getting Hadoop up and running is quite straightforward. However, since this process requires editing multiple configuration and setup files, make sure that each step is properly followed.

1. Install Java: Hadoop requires Java to be installed, so let's begin by installing Java: apt-get update apt-get installdefault-jdk. These commands will update the package information on your VPS and then install Java. After executing these commands, execute the following command to verify that Java has been installed:

2. Configure HDFS: First let's fetch Hadoop from one of the mirrors using the following command: Wgethttp://www.motorlogy.com/apache/hadoop/common/ current/hadoop-2.3.0.tar.gz after downloading the Hadoop package, execute the following command to extract it: tar xfz hadoop-2.3.0.tar.gz this command will extract all the files in this package in a directory named hadoop-2.3.0.

For this tutorial, the Hadoop installation will be moved to the /usr/local/hadoop directory using the following command: mv hadoop-2.3.0/usr/local/hadoop. After completing all the configuration outlined in the above steps, the Hadoopfilesystem needs to be formatted so that it can start being used. This is done by executing the following command: hdfsnamenode –format

3. Start Hadoop: All that remains to be done is starting the newly installed single node cluster: start-dfs.sh Executing this command should show you something similar to the following:

```
root@tutorials:~# jps
1778 Jps
1744 NodeManager
1474 SecondaryNameNode
1146 NameNode
1621 ResourceManager
1272 DataNode
root@tutorials:~#
```

Fig. 5 Showing Name and Data Node of Hadoop

After configuration hadoop, we are configuring our application for big Data Analysis. In this step, we are using configured an iso image file of our virtual machine. This setup can be run over any virtual platform; we need not to configure it again for other systems.

4. Starting the multi-node cluster: Starting the cluster is performed in 2 steps.
i) Starting the HDFS Daemons – the NameNode daemon is started on the master (Hadoopmaster) and the DataNode daemons are started on all slaves (in our case Hadoopmaster and hadoopnode). ii) Start the MapReduce Daemons - The JobTracker is started on the master (Hadoopmaster), and the TaskTracker daemons are started on all slaves (in our case Hadoopmaster and hadoopnode)

5. HDFS Daemon: Start hdfs by executing the following command on the master (Hadoopmaster). This will bring up the HDFS with the NameNode up on the master and, the DataNode on the machine listed in the conf/slaves file. bin/start-dfs.sh

```
hduser@Hadoopmaster:~$ start-dfs.sh
Warning: $HADOOP_HOME is deprecated.

starting namenode, logging to /usr/local/hadoop/libexec/../logs/hadoop-hduser-na
menode-Hadoopmaster.out
hadoopnode: starting datanode, logging to /usr/local/hadoop/libexec/../logs/hado
op-hduser-datanode-hadoopnode.out
Hadoopmaster: starting datanode, logging to /usr/local/hadoop/libexec/../logs/ha
doop-hduser-datanode-Hadoopmaster.out
Hadoopmaster: starting secondarynamenode, logging to /usr/local/hadoop/libexec/.
./logs/hadoop-hduser-secondarynamenode-Hadoopmaster.out
```

Fig. 6 Showing HDFS Daemon

6. Mapred Daemon: Start mapreduce by executing the following command on the master (Hadoopmaster). This will bring up the MapReduce cluster with the JobTracker up on the master and, the TaskTracker on the machine listed in the conf/slaves file. bin/start-mapred.sh

```
hduser@Hadoopmaster:~$ start-mapred.sh
Warning: $HADOOP_HOME is deprecated.

starting jobtracker, logging to /usr/local/hadoop/libexec/../logs/hadoop-hduser-
jobtracker-Hadoopmaster.out
Hadoopmaster: starting tasktracker, logging to /usr/local/hadoop/libexec/../logs
/hadoop-hduser-tasktracker-Hadoopmaster.out
hadoopnode: starting tasktracker, logging to /usr/local/hadoop/libexec/../logs/h
adoop-hduser-tasktracker-hadoopnode.out
hduser@Hadoopmaster:~$
```

Fig. 7 Showing MapReduce Daemon

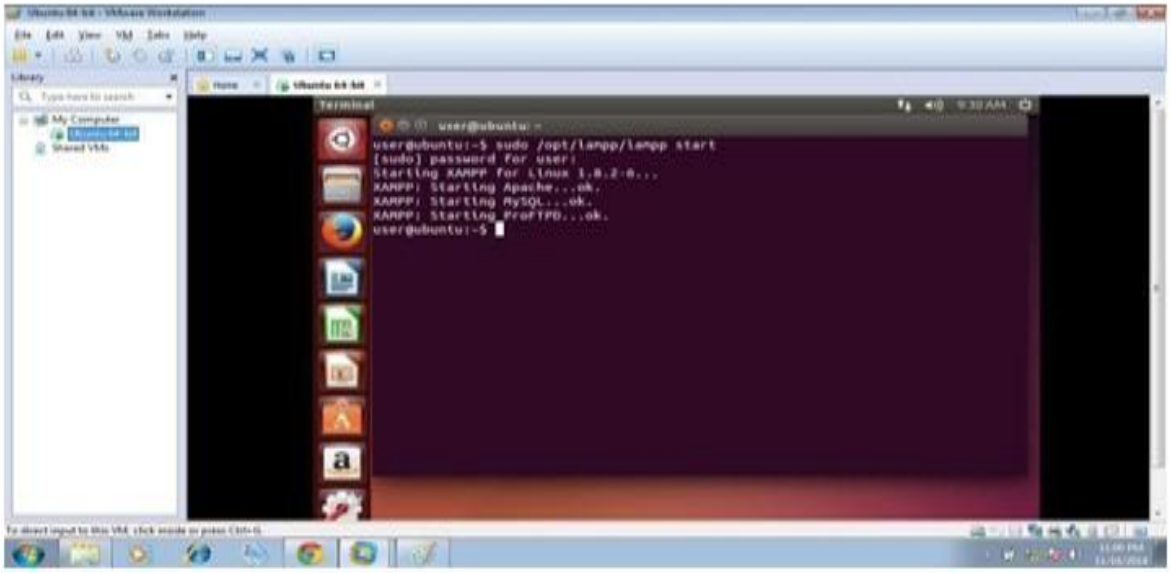

Fig. 8 Configured LAMPP and Services

To start and run the mining application, we use browser. Call the application start page by giving the url on address bar.After successfully call and running the index page, the application visualize as the following screen.

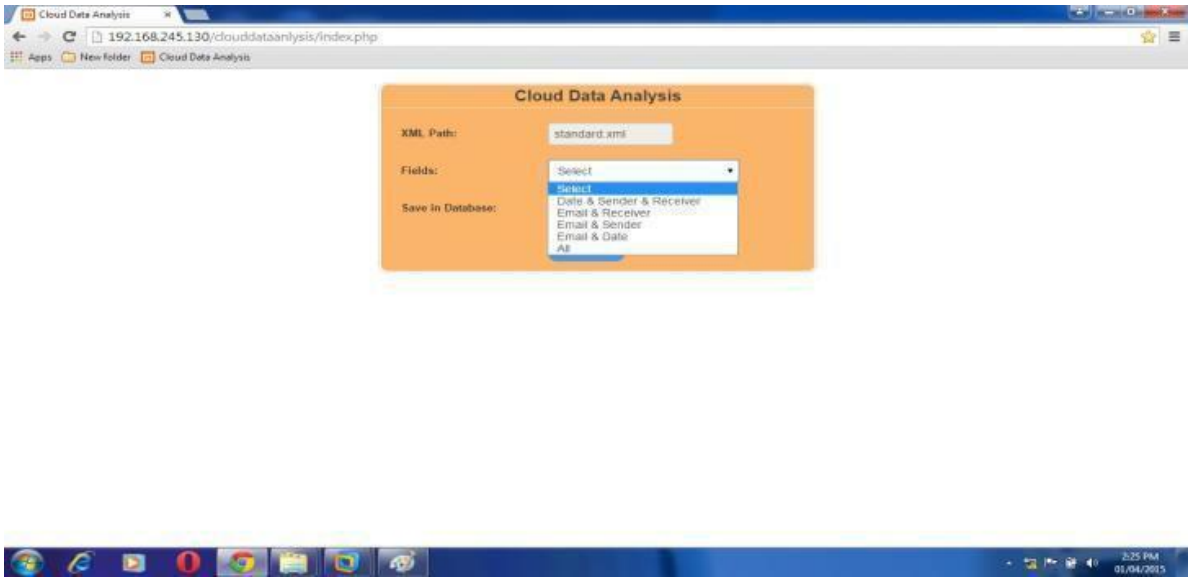

Fig. 9 Big Data Mining Application

IV. CONCLUSION

This paper presents a review of need data mining services in cloud computing along with a case studyon the integrated approach of data mining and cloud computing and mining. The data mining in cloud allows organization to centralize the management of software and data storage with assurance of efficient, reliable and secure services for their users. The implementation of data mining techniques through cloud computing will allow the users to retrieve meaningful information from virtually integrated data warehouse that reduces the costs of infrastructure and storage. This approach also reduces the barriers that keep small companies from benefiting of the data mining instruments. The emergence of cloud computing brings new ideas for data mining. It increases the scale of processing data. This approach is deployed in HDFS, MapReduce and C-Means big data mining algorithm framework of Hadoop. Big data discloses the limitations of existing data mining techniques, resulted in a series of new challenges related to big data mining. Big data mining is a promising research area, still in its infancy. In spite of the limited work done on big data mining so far, we believe that much work is required to overcome its challenges related to heterogeneity, scalability and execution speed.

REFERENCES

[1] S. Shuman, "Structure, mechanism, and evolution of the mRNA capping apparatus", *Prog Nucleic Acid Res MolBiol*, 2000.

[2] A. Rajaraman and J. D. Ullman, "*Mining of Massive Datasets*. Cambridge – United Kingdom: Cambridge University Press, 2012.

[3] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed Systems: Concepts and Design*: Pearson Education; 2005.

[4] M. De Oliveira Branco, *Distributed Data Management for Large Scale Applications*. Southampton – UnitedKingdom: University of Southampton; 2009.

[5] W. Raghupathi and V. Raghupathi, "Big data analytics in healthcare: promise and potential", *Health Inform Sci Syst.*, Vol. 2, No. 1, pp. 3, 2014.

[6] D. E. Bell, H. Raiffa, and A. Tversky, "Descriptive, normative, and prescriptive interactions in decision making", *DecisMak*, 1988.

[7] I. Foster, and C. Kesselman, "The Grid 2: Blueprint for a new Computing Infrastructure", Houston – USA, *Elsevier*, 2003.

[8] J. D. Owens, M. Houston, D. Luebke, S. Green, "Stone and JC. Phillips: GPU computing",*Proc IEEE*, Vol. 96, No. 5, pp. 879–899, 2008.

[9] N. Satish, M. Harris and M. Garland, "Designing efficient sorting algorithms for manycore GPUs", *In Parallel &Distributed Processing*, 2009 IPDPS 2009 IEEE International Symposium on: 2009, *IEEE*, pp. 1–10, 2009.

[10] B. He, W.Fang, Q. Luo, NK. Govindaraju, and T. Wang, "Mars: a MapReduce framework on graphics processors", *InProceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques*, 2008,

[11] J. Dean, S. Ghemawat, "MapReduce*:* simplified data processing on large clusters", *Commun ACM 2008*, Vo. 51, No. 1, pp. 107–113.

[12] S. L. Peyton Jones, The *Implementation of Functional Programming Languages* (Prentice-Hall International Series in Computer Science). New Jersey – USA: Prentice-Hall, Inc; 1987.

[13] R. E. Bryant, "Data-intensive supercomputing: The case for DISC", Pittsburgh, PA – USA: School of Computer Science, Carnegie Mellon University; 2007, pp.1–20.

[14] T. White: Hadoop: *The Definitive Guide*. Sebastopol – USA: ― O'Reilly Media, Inc.‖; 2012.

[15] K. Shvachko, H. Kuang, S. Radia and R. Chansler, "The hadoop distributed file system. In Mass Storage Systems and Technologies (MSST)", *2010 IEEE 26th Symposium, IEEE*, pp.1-10, 2010.

[16] The Apache Software Foundation. [http://apache.org/]

[17] M. Olson, "Hadoop: Scalable, flexible data storage and analysis", IQT Quart, No. 3, pp. 14–18, 2010.

[18] J. Xiaojing, "Google Cloud Computing Platform Technology Architecture and the Impact of Its Cost", In 2010 Second WRI World Congress on Software Engineering, pp. 17–20, 2010.

[19] A. Thusoo, JS. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P Wyckoff, and R Murthy, "Hive: a warehousing solution over a map-reduce framework", *Proc VLDB Endowment*, Vol. 2, No. 2, pp.1626–1629, 2009.

[20] C. Olston, B. Reed, U. Srivastava, R. Kumar, A. Tomkins, "Pig latin: A not-so-foreign language for data processing", In Proceedings of the 2008 ACM SIGMOD *International Conference on Management of Data*: 2008, ACM; 2008, pp. 1099–1110.

[21] S. Prabha and P. Kola Sujatha, "Reduction Of Big Data Sets Using Fuzzy Clustering", *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, Vol. 3 No. 6, June 2014.

[22] R. Madhuri, M R Murty, J. V. R. Murthy, PVGD Prasad Reddy and S.C Satapathy, "Cluster analysis on di_erent data sets using k-modes and k-prototype algorithms, ICT and Critical Infrastructure", *Proceedings of the 8thAnnual Convention of Computer Society of India,Springer*, Vol. 2, pp. 137-144, 2014.

[23] X. F. Jiang, "Application of parallel annealing particle clustering algorithm in data mining*", TELKOMNIKA Indonesian Journal of Electrical Engineering*, Vol. 12, No. 3, pp. 2118-2126, 2014.

[24] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering", *IEEE Transactions on Fuzzy Systems*, Vol. 1, pp. 10-12, 1993.

[25] N. Janardhan, T. SreePravallika and SowjanyaGorantla, "An efficient approach for integrating data mining into cloud computing", *International Journal of Computer Trends and Technology (IJCTT)*, Vol. 4, No. 5, May 2013.