International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 7 Issue: 2 11 - 17

Improving the Performance of Heterogeneous Hadoop Clusters Using Map Reduce

Vidisha Sharma
M.Tech Scholar, Department of Computer Science & Engineering
Kautilya Institute of Technology & Engineering, Jaipur
E-Mail: vidisha81@gmail.com

Satish Kumar Alaria
Assistant Professor, Department of Computer Science & Engineering
Kautilya Institute of Technology & Engineering, Jaipur
E-Mail: satish.alaria@gmail.com

Abstract: The key issue that emerges because of the tremendous development of connectivity among devices and frameworks is making such a great amount of data at an exponential rate that an achievable answer for preparing it is getting to be troublesome step by step. Thusly, building up a stage for such propelled dimension of data handling, equipment just as programming improvements should be led to come in level with such generous data. To enhance the proficiency of Hadoop bunches in putting away and dissecting big data, we have proposed an algorithmic methodology that will provide food the necessities of heterogeneous data put away .over Hadoop groups and enhance the execution just as effectiveness. The proposed paper intends to discover the adequacy of new calculation, correlation, proposals, and an aggressive way to deal with discover the best answer for enhancing the big data situation. The Map Reduce method from Hadoop will help in keeping up a nearby watch over the unstructured or heterogeneous Hadoop bunches with bits of knowledge on results obviously from the algorithm.in this paper we proposed new Generating another calculation to tackle these issues for the business just as non-business uses can help the advancement of network. The proposed calculation can help enhance the situation of data ordering calculation MapReduce in heterogeneous Hadoop groups. The exposition work and analyses directed under this work have copied very amazing outcomes, some of them being the selection of schedulers to plan employments, arrangement of data in similitude lattice, bunching before planning inquiries and in addition, iterative, mapping and diminishing and restricting the inner conditions together to stay away from question slowing down and execution times. The test led additionally sets up the way that if a procedure is characterized to deal with the diverse use case situations, one could generally lessen the expense of processing and can profit on depending on disseminated frameworks for quick executions.

*****

I. INTRODUCTION

1.1 General Introduction

The data burst over internet is no mystery today for anyone. Developers or users working on internet can easily tell about the significance of data and the power of internet today. The internet has become a huge place where one can easily find everything from a needle to human genome reports. The internet is a complex mesh of tools, frameworks, applications, algorithms, hardware commodity for storing, processing, and managing data across the world. A single data server or application is even, simply not enough to handle data generated by a single user. The role of deciding what applications suit best or how data could be managed is an area of constant research. The internet today is closely related to data as mentioned earlier, but the more general term to describe this kind of voluminous data would be through Big Data. Another question that arises in this scenario is how this data is generated [1].

1.2 Big Data

Big data is a collection of frameworks, large datasets, techniques, and tools. The amount of data considered to be big data is such that traditional computing techniques fails to process it. The amount generally equals in Petabytes of data that are generated through different systems or users. Big Data can also be defined as a collection of algorithms, infrastructure, and visualizations that can be used to make sense of data generated by users or machines in a fast and efficient manner.The three V's give a huge insight on how big data behaves, how big data is generated, and what is the quality of big data [6].

Variety: It refers to the different sources from which data can be generated. It includes onlinestore logs, phone data, X-ray data, medicinal research, sensor data, and many other such sources. The problem with older databases was the fact that data needed to be stored in some sorted or arranged ways but with different kinds of data generated with different behaviors, need of such data holder which can process any type of data was much needed [6].

Velocity: It refers to the speed at which data arrives. In a single day TB's of data can arrive andthat needs to be processed with the same speed as no data can be discarded [7].

Volume: It refers to the size of data that is generated each day. It could be from any of the placeslike transactions of banks, logs of user database, business information, share

market patterns, user information, sensor data, medical data and also data generated from social media [7].

1.3 Hadoop

Hadoop is used to store and process Big Data in a distributed environment with its open source frameworks. Hadoop uses simple programming models in a distributed environment to store and process data files. Hadoop can handle the infrastructure of Big Data by providing batch processing and storage to any kind of data. Hadoop is also an open source framework that is used to store as well as process data using different methods in a distributed environment. It can manage data movement in a distributed manner across various cluster of computers in such a manner that data access becomes fast, reliable, and secure [12]. It involves use of simple computing and programming models for handling data storage or distribution over large and widespread geographical areas. The correlation between Big Data and Hadoop are simpler than it seems. Going with the traditional databases was rather a good solution with predefined software written to use them. The developers and administrators had a very good chance of using traditional RDBMS systems like DB2, Oracle Databases, or MS SQL servers [12].

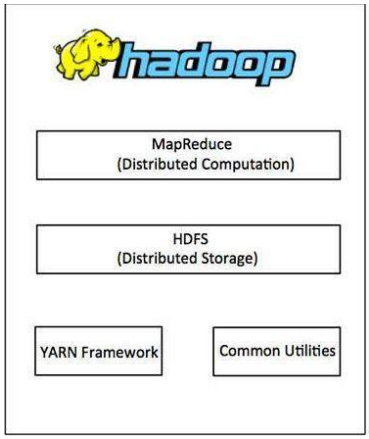

Fig.1.1. Architecture of Hadoop

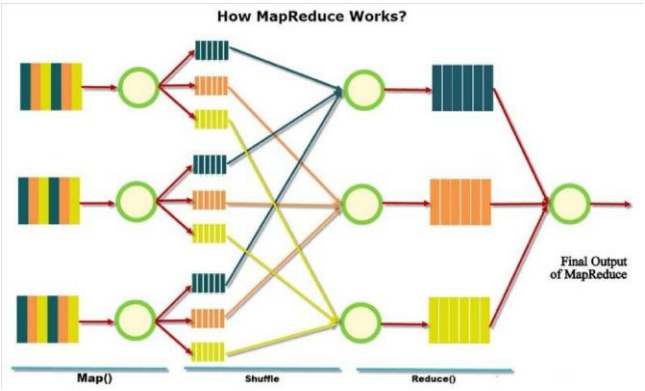

Fig.1.2. MapReduce at work [http://www.techquark.com/2014/07/hadoop-what-it-is-how-to-get-started.html]

1.4 MapReduce Algorithm

The basic task scheduling in a Hadoop cluster is taken care by Hadoop YARN and MapReduce. The algorithm consists of a master and slave dynamics which are responsible.

1.5 Problem Domain

Studies carried out so far indicate that, if correctly analyzed the data from the digital world constitutes about 30% can be used for a variety of useful purposes. But the only 0.5% of digitized data is utilized. It is clearly so because of the limited capacity of the existing system, lack of data in large areas brought about by the use of existing tools. Unstructured data resides on the server or cluster analysis reveals the impact of the current conflict in the system data. MapReduce model can provide high throughput, low-latency assignment or fairness between jobs.But it needs to be done in speed, in order to improve and increase the unorganized and unstructured data to cope with them. Another problem that needs to be taken care of, is the turnaround time and effective clustering algorithm of data sorting and retrieval of its structure with high throughput and low latency data. So that the digitized data could be fully utilized for various purposes.

II. LITERATURE SURVEY

Big data and its related applications require a lot of data movement around different data nodes in Hadoop's clusters. With simple homogeneous cluster configurations, data movement in and around the nodes is simple but with heterogeneous clusters the problem of data movement becomes a crucial issue as different nodes as data nodes have different computing speed. This in turn also increases the job completion times. On running data intensive jobs, this mismatch of physical hardware impacts the performance of applications that require huge amount of data. The MapReduce framework used in Hadoop over Hadoop Distributed File System or other file systems proves to be successful in homogeneous clusters but not in heterogeneous cluster environment. Author says that the distinguished more profiled critical execution of queries have equitability issues existing. For present MapReduce-based information warehousing framework, particularly, prediction bases systems are suggested, built inquiry planning framework, which scaffolds those semantic hole between MapReduce runtime. Furthermore inquiry compiler empowers productive inquiry planning to quick and more reasonable enormous information analytics. The author has performed works to overcome the drawbacks of design to improve performance of large scale clusters. A cross layer scheduling framework is used so that Hive queries can be percolated, semantically extracted, and a multivariate execution-time prediction and two level query scheduling is performed [1].

The authors in paper have out rightly discussed their ideas on the parallel processing of data in heterogeneous clusters [2]. Authors in their paper [3], have revealed the crucial importance of Big Data in today's world. As far as applications are concerned, Big Data finds a lots of usage but poses challenges too. In the paper [4], author mention the process of classification of automatic documents is used for base of analyzing applications [5]. The type of classification used here is Naïve Bayes because they are simple, efficient and effective for big data documents. They have certain limitations when compared to some of the statistical methods but they are able to help categorize big documents into simpler sets.

III. PROPOSED WORK

3.1 Methodology

The dissertation work focuses on improving the performance of heterogeneous clusters on Hadoop by following a set of phases that improves the data input/ output queries, improves the routing of algorithm to heterogeneous clusters, and then improving the performance of processing of queries in such a way that they are easily connected to the right part of execution in minimal time as possible without increasing the costs at computing level. The proposed work follows a series of steps to handle these scenarios which are described and depicted in the following pages, each considering the effects and processes it takes along.

3.1.1 Solving Problems Through Iteration and Phasing:

This work includes dividing the query processing for MapReduce in consecutive iterative phases for filtering, clustering, and improving the query for faster execution processes. Initially the phase involves iterative steps to clean the query and find dependencies among the query so that once query is executed, tasktracker doesn't have to depend on computing query dependencies again once it was calculated in the initial steps. The problem here is to minimize the query execution time, improve the efficiency of clusters and including the iterative processing where the queries are improved through clustering them into similar steps, applying TF-IDF [19], calculating weights, managing the similarity matric and finally submitting the query for execution. The data used here for performing experiments are log files of websites which is in form of unstructured data. It is the best way through which one can determine the efficiency of methods and it is also closely related to the real world environment where data is captured in such state. The data hence also helps in visualizing the scenario of actual usage of processes that are closely related to the real world scenario where one has to deal with data which can come in any form and shape. The data as log files have helped in accessing the query submission process for an e-commerce website that has to deal with millions of search keywords, requires clustering of data items to buy and providing accurate results in very short spans of life.

3.1.2 Query Improvisation

This is where clustering also comes in scene; when queries are fetched into the parser and semantic analyzer, they invariably compute the dependencies among the queries. But once this parsed query is sent to the Hadoop's MapReduce [23] [24] to execute the dependencies that were calculated for the Hive query is lost between the transitions. Once the dependencies are calculated they can be used for semantics extractions in Hive QL processor [24]. In the second step we can use these dependencies such as logical predicates and input tables to process the dependencies among different queries to be closely attached to each other during transition. When the sematic gap between Hive and Hadoop is bridged by these intermediary steps, they can be easily used for clustering similar queries at query level, and hence by minute steps improving the query job execution.

3.1.3: Applying Clustering in Initial Phases:

A lot of applications and systems use clustering as a part of dividing similar and dissimilar data items together. Clustering of data sets here can help achieve focusing on the items that are useful and discarding items which are generally not useful [23]. Looking from the view of a log file and applying clustering on it, we get clusters of user prone data that categorize the behavior of the item buying selections. The interactions of a user with an e-commerce website can identify good buying patterns easily. When this single end user data is collected from the multi million users of the website and analyzed, the results tell a whole lot of insight on the consumer behavior, selection, buying options, and fields which need improvisation.

The log files used here symbolizes the unstructured data category and recording the data from different fronts, a presentation of heterogeneous Hadoop clusters. In this iterative clustering model when the data is captured from user end, it is optimized to remove static and dynamic parts of it in the clustering process itself where static parts are the decisions where dynamic parts are variants of time stamping, invoice numbers, delimiters etc. It could be done through iteration process but applying a standard clustering algorithm with changes improves the efficiency as well [25].

3.1.4 **Proposed** Algorithm

Step 1:- We will read the log transition.

Step 2:- We store the transition value in Qi, Where i=1, 2, 3……..n

Where n is equal to the log transition value.

Step 3:- We store the query which is request from client and store in array by get Query () method.

IQ = I get query. Where I is number of query request and .We store the query in array to create cluster. By method table – put (null, array, parameter1, parameter 2 ….n);

To convert the array in object they are

Qi= Table – get query ();

Step 4:- Merge the pair of most similar queries (qi,qj) that does not convert the same queries. If

(Qi is not in Ci) then store the frequency of the item and the increase the value.

Step 5:- Compute the simulating Matrix if Qiin Ci.

Step 6:- If Qiis not Cithen compute new cluster IQ=New (Ci).

Step 7:- Go to Step 3.

Step 8:- Step go to Step 2.

IV. RESULTS AND CONCLUSION

4.1. Comparison of productivity among Hadoop Schedulers:

Comparison among the most efficient schedulers used in Hadoop i.e. FIFO, HFS and HCS. The following graphs establish the point that schedulers can overtake other schedulers which are used in HDFS and internal mechanisms to share a file or job over the several default systems to execute and store them reliably.

Scheduling Algorithm	Data Size	Previous Methodology Execution Time	Proposed Methodology Execution Time
FIFO	500 MB	4000	3950
	1GB	7800	7333.33
	2GB	9900	9165
	5GB	14500	11130
	10GB		**12780**
	500 MB	3950	3925
	1GB	7200	7065
	2GB	9200	9085
	5GB	13500	10980
	10 GB		**12400**
HCS	500 MB	3900	3875
	1GB	7000	6998
	2GB	9000	8912
	5GB	12000	10260
	10 GB		**12610**

4.2 PERFORMANCE EVALUATION

4.2.1 Screenshots of Heterogeneous cluster in work:

The clusters that were created for this dissertation work are working and derived results as formulated. The average I/O rate of data movement, standard deviations, and time taken to execute the loaded files prove that the formulated method is working as expected.

The following images depict the step by step processes of MapReduce at work and clusters working under the stipulated load scheduling of big data sets.

4.2.1.1 Namenode and Datanode stature

The namenodes and datanodes could be seen from the admin reports that depict each node as working.

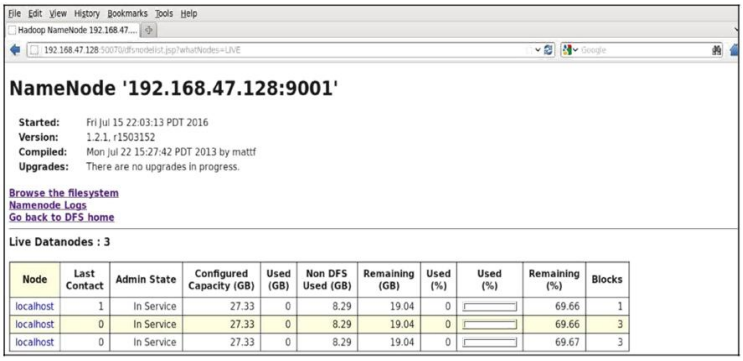

Fig.4.1. Namenode demo

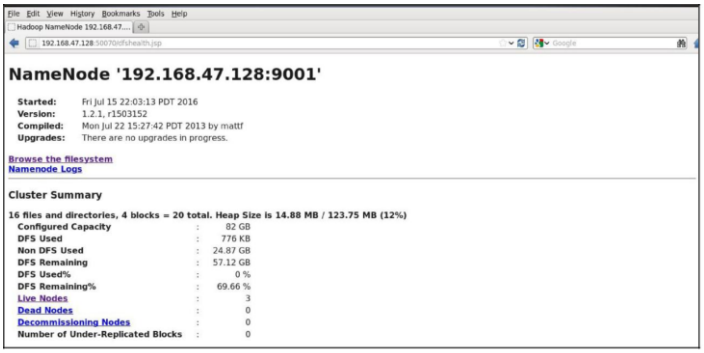

Fig.4.2. Namenode summary

4.2.1.3 MapReduce at work:

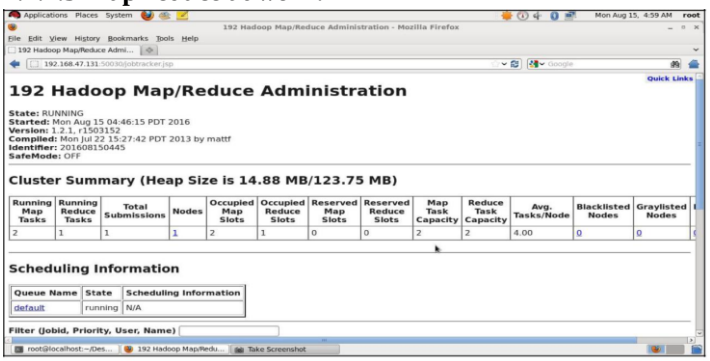

Fig.4.3. MapReduce Administration

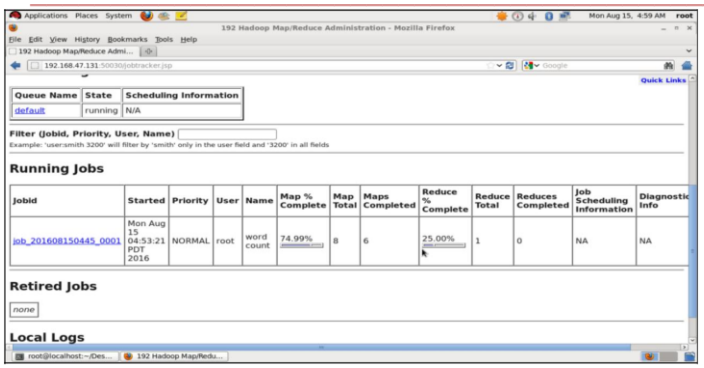

Fig.4.4 Status of Running Jobs

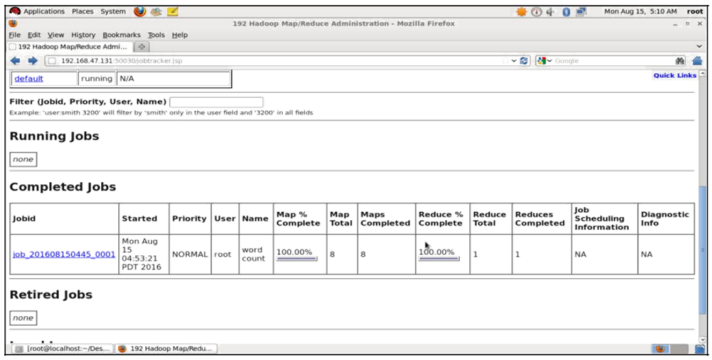

fig.4.5. Job Completion

4.3 Schedulers at work

4.3.1 First In First Out: The scheduler used here is FIFO which is used to schedule theresources of big data sets in the heterogeneous clusters and concurrently running TestDFSIO as benchmarking. The following figures depict them working and deriving results for it.

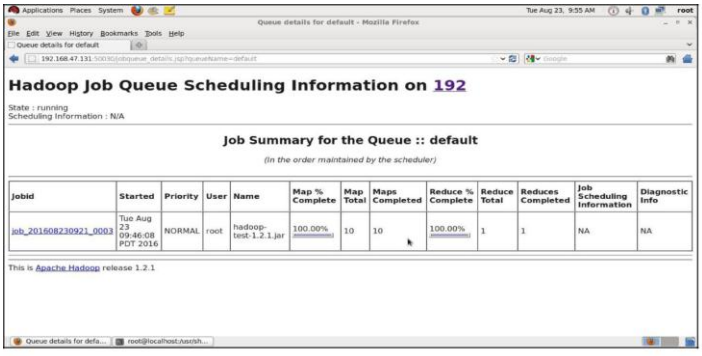

Fig.5.6. Job Queue for FIFO Scheduler

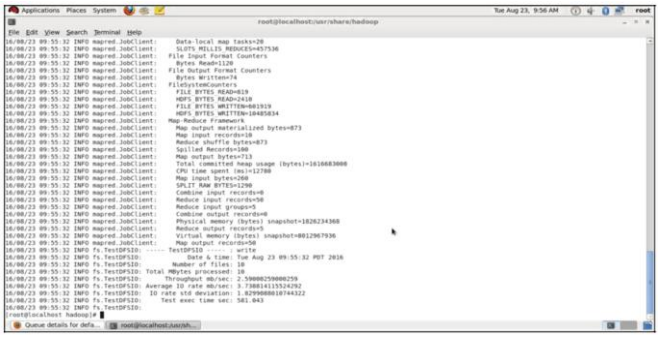

Fig.4.7. TestDFSIO Benchmarking for FIFO

4.3.2 Hadoop Fair Scheduler: The following test results are the resultant of the working of fair scheduler algorithm on the Big Data sets for heterogeneous clusters in Hadoop environment.

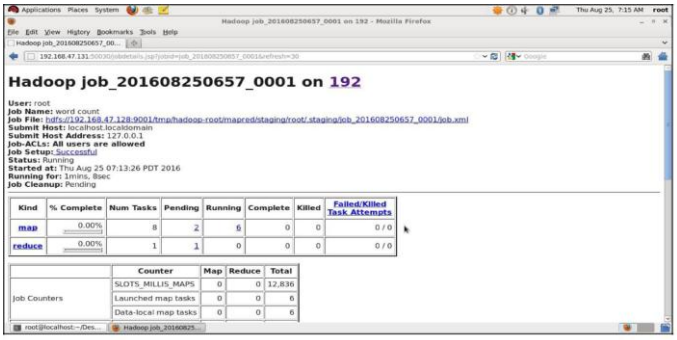

Fig.4.8. Hadoop Fair Scheduler at work

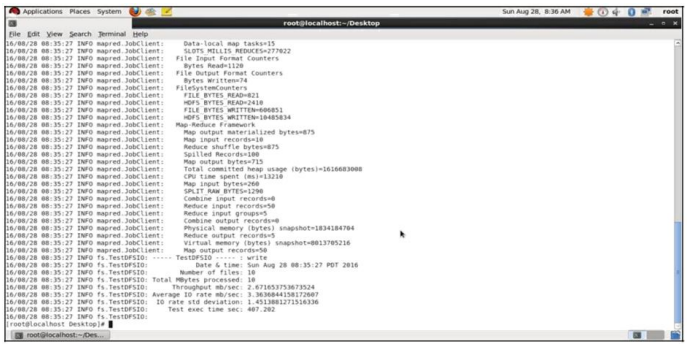

Fig.4.9.TestDFSIO Benchmarking for Hadoop Fair Scheduler

4.3.3 Hadoop Capacity Schedulers: The following test results are the resultant of theworking of Hadoop capacity scheduler algorithm on the Big Data sets for heterogeneous clusters in Hadoop environment.

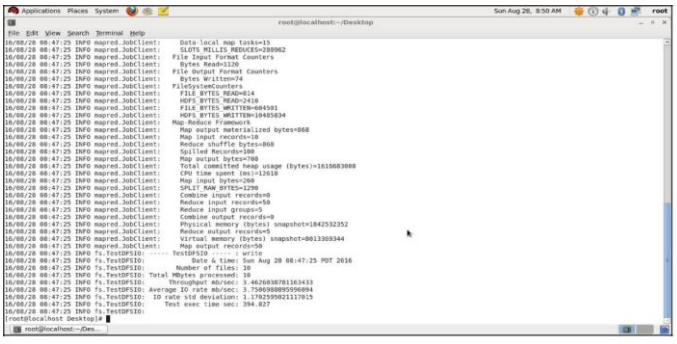

Fig.4. 10 TestDFSIO Benchmark for Hadoop Capacity Scheduler

V. CONCLUSION

This work and experiments conducted under this work have emulated quite surprising results, some of them being the choice of schedulers to schedule jobs, placement of data in similarity matrix, clustering before scheduling queries and moreover, iterative, mapping and reducing and binding the internal dependencies together to avoid query stalling and execution times. The experiment conducted also establishes

the fact that if a process is defined to handle the different use case scenarios, one could overall reduce the cost of computing and can benefit on relying on distributed systems for fast executions.

5.1 Limitations

The Big Data set has many probing worries for computing and data storage. Since the major part of big data analytics is related to dynamic nature of data, data is deleted, manipulated and retrieved frequently. This ad hic based processing of data includes streaming data in and out of the storage systems as the requirement but this also introduces a large chunk of processing. MapReduce and Hadoop version 1 are limited to processing and storing data at same ends where as in the forthcoming versions, these are taken care by YARN. It simplifies the processing, and the architecture followed in the version 2 of Hadoop clearly defines the limits and scopes of each individual part. Here the processing is solely the task of Yarn, computing the part of MapReduce and storage the part of data nodes.

The proposed work has however followed the footsteps of Hadoop version 1, yet it is limited to processing and management in the same unit if MapReduce.

5.2 Future Scope

The authors from different areas have provided a sufficient guidance on the setup, measurement and working of Hadoop ecosystems. The future of this methodology is to be utilized in systems that have Hadoop and MapReduce at its core. The algorithm could be tinkered to suit the needs of Heterogeneous computing environment to suit the big data sets.

REFERENCES

[1] Zhuo Liu, “Efficient Storage Design and Query Scheduling for Improving Big Data Retrieval and Analytics”, Ph.D. dissertation, C.S.E. Dept., A.U., Auburn, Al, 2015

[2] Fernando G. Tinetti1, Ignacio Real, Rodrigo Jaramillo, and Damián Barry, “Hadoop Scalability and Performance Testing in Heterogeneous Clusters”, PDPTA 2015.

[3] Zongben Xu, Yong Shi, “Exploring Big Data Analysis: Fundamental Scientific Problems”, Springer, Dec. 2015.

[4] Felipe Viegas, Wellington Martins, Leonardo Rocha, “Parallel Lazy Semi-Naïve Bayes Strategies for Effective and Efficient Document Classification”, CIKM’15,ACM. ISBN 978-1-4503-3794-6/15/10, 2015.

[5] Jong-Yeol Yoo, Dongmin Yang, “Classification Scheme of Unstructured Text Document using TF-IDF and Naïve Bayes Classifier”, Advanced Science and Technology Letters Vol.111 (COMCOMS 2015), pp.263-266, 2015.

[6] Fong-Hao Liu, Ya-Ruei Liou, Hsiang-Fu Lo, Ko-Chin Chang, and Wei-Tsong Lee, “The Comprehensive Performance Rating for Hadoop Clusters on Cloud Computing Platform”, IJIEE Vol. 4, No. 6, November 2014.

[7] David C. Anastasiu, George Karypis, Jeremy Iverson, Shaden Smith, “Big Data Frequent Pattern Mining”, 2014

[8] A Fahad, “A survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis”, 2014

[9] Zheyi Rong, Jeroen De Knijf, “Direct Out-Of-Memory Distributed Parallel Frequent Pattern Mining”, BigMine’13, ACM 978-1-4503-2324-6/13/08, 2013

[10] Wang Teng, BPAR: A Bundle-Based Parallel Aggregation Framework for Decoupled I/O Execution, 2014

[11] B. Thirumala Rao, N.V. Sridevi, V. Krishna Reddy, L.S.S. Reddy, “Performance Issues of Heterogeneous Hadoop Clusters in Cloud Computing”, GJSCT, Vol. XI Issue VII, May 2011.

[12] EMC Corporation, “Virtualizing Hadoop in Large-Scale Infrastructures”, 2014.

[13] Charu Aggarwal, Jiawei Han, “An Introduction to Frequent Pattern Mining”, in Frequent Pattern Mining, ISBN 978-3-319-07820-5, 2014.

[14] Giannakouris - Salalidis Victor, Plerou Antonia, Sioutas Spyros, “CSMR: A Scalable Algorithm for Text Clustering with Cosine Similarity and MapReduce”, IFIP Advances in Information and Communication Technology, September 2014.

[15] Jun Liu, Tianshu Wu, and Ming Wei Lin and Shuyu Chen,”An Efficient Job Scheduling for MapReduce Clusters”, International Journal of Future Generation Communication and Networking Vol. 8, No. 2 (2015), pp. 391-398

[16] T.K.Das, P.Mohan Kumar, “Big Data Analytics: A Framework for Unstructured Data Analysis”, IJET ISSN: 0975-4024, Vol 5 No 1 Feb-Mar 2013.

[17] Florica Novăcescu, “Big Data in High Performance Scientific Computing”, “EFTIMIE MURGU” RESIŁA, ANUL XX, NR. 1, ISSN 1453 – 7397, 2013.

[18] Jian Xue, Jinyu Li, and Yifan Gong, "Restruccturing of Deep Neural Network Acoustic Models with Singular Value Decomposition", ISCA, INTERSPEECH 2013.

[19] Bin Li, Yuan Guoyong, “Improvement of TF-IDF Algorithm Based on Hadoop Framework”, The 2nd International Conference on Computer Application and System Modeling, 2012.

[20] Mert Akdere, “Learning based Query Performance Modeling and Prediction”, 2011

[21] Hui Gao, Jun Jiang, Li She, Yan Fu, “A new agglomerative Hierarchical clustering algorithm implementation based on the MapReduce Framework”, IJDCTA, Volume 4, Number 3, June 2010.

[22] Jiong Xie, Shu Yin, Xiaojun Ruan, Zhiyang Ding, Yun Tian, James Majors, Adam Manzanares, and Xiao Qin, “Improving MapReduce Performance through Data Placement in Heterogeneous Hadoop Clusters”, April 2010.

[23] Jian Wan, Wenming Yu, Xianghua Xu, “Design and Implement of Distributed Document Clustering Based on MapReduce”, ISBN 978-952-5726-07-7, 2009.

[24] Matei Zaharia, Andy Konwinski, Anthony D. Joseph, Randy Katz, Ion Stoica, "Improving MapReduce Performance in Heterogeneous Environments", OSDI 2008

[25] Jeffrey Dean, Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", OSDI 2004

[26] Badrul Sarwar, George Karypis, Joseph Konstan, John Riedl "ItemBased Collaborative Filtering Recommendation Algorithms", ACM 1581133480/01/0005, 2001.

[27] Krushnarah Kamtekar, "Performance Modeling of Big Data", May 2015.

[28] Amol Jagtap, "Categorization of the Documents using K-Means and MapReduce", International Journal of Innovative Research in Science, Engineering and Technology, ISSN: 2319-8753, 2015.