International Journal of Research in Advent Technology, Vol.2, No.5, May 2014
E-ISSN: 2321-9637

Analysis and Performance Evaluation of Large Data Sets Using Hadoop

Krishnakumar.K[1], Ganesh Karthikeyan.V[2]
PG Student of CSE[1], Assistant Professor of CSE[2]
Adithya Institute of Technology, Coimbatore
mail2krishna1990@gmail.com[1],ganeshkarthikeyan_v@adithyatech.com[2]

Abstract - The Hadoop MapReduce is designed for Distributed and parallel processing to process very large data reliably, and to stream those data at very high bandwidth to user application. Hadoop Distributed File System (HDFS) and Mapreduce paradigm provides a parallelization and distributed processing for its ease-of-use, analyzing unstructured and structured data. This provides scalability, reliable storage, and fault-tolerance. MapReduce specify Map and Reduce functions to make a huge task to parallelize and execute on a large cluster of Commodity machines. Here the immense of data is loaded in HDFS for data processing, evaluating, indexing, and resource utilization. By using the loaded data in hadoop the classification and clustering is done using machine learning algorithm. In data processing all the missing values and relations is identified and performance evaluation has been executed. This system handles identifying error, load balancing, utilizing system resources, less cost and high performance.

Index term – Hadoop, MapReduce, Classification, Clustering, Machine learning algorithm.

I. INTRODUCTION

With the evolution of information technology, enormous expanses of data have become increasingly obtainable at outstanding volumes. Amount of data being gathered today is so much that, 90% of the data in the world nowadays has been created in the last two years [3]. The Internet impart a resource for compiling extensive amounts of data, Such data have many sources including large business enterprises, social networking, social media, telecommunications, scientific activities, data from traditional sources like forms, surveys and government organizations, and research institutions [8].

Large scale data is complex and difficult to process using on-hand database management tools, desktop statistics, database management systems or traditional data processing applications and visualization packages. The traditional method in data processing had only smaller amount of data and processing was very slow [12].

The fast increase and update of big data brings a new challenge to process large amount of data. The term Big Data refers to volume, variety, velocity and veracity. This provides the functionalities of Apprehend, analysis, storage, sharing, transfer and visualization [14]. Hadoop Distributed File System (HDFS) and Mapreduce paradigm provides a Parallelization and distributed processing for its ease-of-use, analyzing unstructured and structured data.

In this work, we explore a method of large data analysis and processing using Hadoop MapReduce paradigm. The immense of data is loaded in HDFS for data processing, evaluating, indexing, and resource utilization. By using the loaded data in hadoop the classification and clustering is done and workload is handled and throughput is increased.

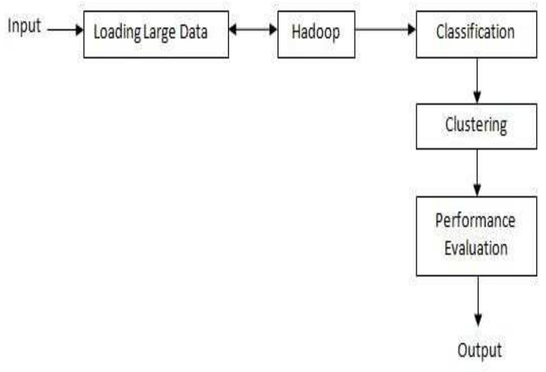

Fig 1: System Architecture

II. LITERATURE SURVEY

Abhishek Verma, Ludmila Cherkasova, Roy H. Campbell [1], described about the data processing of MapReduce jobs for minimizing their execution time. This is to swell their utilization of MapReduce clusters to prune their cost and to optimize the execution of Mapreduce jobs on the Cluster. Here a subset of production workloads of unstructured Information that consists of MapReduce jobs with no dependency and we recognize that the order in which these jobs are executed can have an outstanding impact on their all inclusive completion time and the cluster resource utilization. This description allow to apply the classic Johnson algorithm that was intended for building an optimal two-stage job schedule for finding the shortest path in directed weighted graph. Performance of the constructed schedule through an immeasurable set of simulations over a varies pragmatic workloads and cluster-size dependent.

Jiong Xie, FanJun Meng, HaiLong Wang, HongFang Pan, JinHong Cheng, Xiao Qin [2], described about the scheduling scheme for data processing in Hadoop. Here the prefetching mechanism and predictive schedule mechanism is introduced into MapReduce model while keep compact with the Hadoop. Exhaustive data application running on a Hadoop MapReduce cluster estimates the processing time of each task and preloads an amount of data to the memory before the new task is assigned to the node. Here CPU and IO workload are underutilized while they are working on a data exhaustive application. In MaReduce model, HDFS is adjusting to support huge file sizes. HDFS range from gigabytes to terabytes in size. This spilt the big files to various separation and dispense to hundreds of nodes in a single cluster, The HDFS center store the index information, called Meta data to manage several partition [13].

Asha T, Shravanthi U.M, Nagashree N, Monika M [3], described about the Machine learning algorithm on Hadoop at the core of data analysis**.** Machine Learning Algorithms are recursive and sequential and the accuracy of Machine Learning Algorithms depend on size of the data where, considerable the data more accurate is the result. Reliable framework for Machine Learning is to work for bigdata has made these algorithms to disable their ability to reach the fullest possible. Machine Learning Algorithms need data to be stored in single place because of its recursive. Here the comparison of the performance of ID3 decision tree algorithm, K-means clustering and K-Nearest Neighbor algorithm had been implemented on both serial and parallel implementation using Hadoop has been done. The general and exact technique for parallel programming of a large class of machine learning algorithms for multicore processors is based on a map-reduce paradigm. This is to achieve speedup in the multi-core system and this cannot be used on a single core system [8].

Jian Wan, Wenming Yu, Xianghua Xu [4], described about the implementation of document clustering in distributed system using Hadoop MapReduce. The document clustering for large collection can be efficiently implemented with MapReduce. Hadoop implementation provides an appropriate and supple framework for distributed computing on a cluster of commodity machines. Implementation of tfidf and K-Means algorithm on MapReduce is presented here. More importantly, productivity and effectiveness of the algorithm has been enhanced. Document pre-processing and new iteration algorithm is used calculate tfidf weight. MapReduce is designed in order to assess how important a term is to a document in a collection. Then, KMeans clustering is implemented on MapReduce to partition all documents into k clusters in which each documents belongs to the cluster with the same meaning. Ignoring highest document frequencies can not only speed up the algorithm on MapReduce. This enhances the scalability of processing huge data [13].

Weiyi Shang, Zhen Ming Jiang, Hadi Hemmati, Bram Adams, Ahmed E. Hassan, Patrick Martin[5], described about deploying the big data application in Hadoop clouds. Big Data Analytics Applications is used to analyze enormous parallel processing frameworks. These applications typically build up them using a little model of data in a pseudo-cloud environment. Afterwards, they arrange the applications in a large-scale cloud situation with notably more processing organize and larger input data. Runtime analysis and debugging of such applications in the deployment stage cannot be easily addressed by usual monitoring and debugging approaches. BDA Apps in the cloud follows some stages where, implement BDA App in a pseudo cloud environment using a small data, deploy the application on a larger data set and processing in a real-life setting and verify the execution of the application to make sure all data are processed successfully to achieve better performance. This approach drastically reduces the verification effort when verifying the deployment of BDA Apps in the cloud [10].

III RELATED WORKS

Hadoop MapReduce process huge amount of data in parallel on distributed systems in reliable, fault-tolerant manner. A MapReduce job splits the input into independent chunk and completely processed in parallel. Then the output from the map is sorted and given as input to the reduce tasks. This takes care of scheduling, monitoring and re-executes the failed tasks. In the MapReduce programming paradigm, the basic unit of *Map* and *Reduce* functions is a (key, value) pair [5]. Operations on a set of pairs perform in three stages: Map, Shuffle and the Reduce stage. In Map stage, the mapper considers the single input (key, value) pair, and produces the output as any number of new (key, value) pairs[9].

$$\text{Map}(k1,v1) \rightarrow \text{list}(k2,v2) \qquad (1)$$

The mapper generates a sequence of tuples, (k1, v1), (k2, v2)…., In Shuffle stage, MapReduce sends all the values associated with an individual key to the same machine. In reduce stage, the reducer takes all the values with a single key k, and outputs a multiset of (key, value) pairs with the same key k, all of the maps need to finish before the reduce stage can begin[14].

$$\text{Reduce}(k2, \text{list}(v2)) \rightarrow \text{list}(v3) \qquad (2)$$

The Naive Bayes classifier assumes that the value of an exacting feature is unrelated to the existence or deficiency of any other feature, given the class variable. Naive Bayes classifiers work well in many complex real-world situations. An advantage of naive Bayes requires only small amount of training data to estimation like means and variances of the variables required for classification [11]. The Naive Bayes classifier selects the most likely classification *Cnb* as

$$C_{nb} = argmax_{C_j \epsilon C}\, P(C_j) \prod_i P(X_i \mid C_j) \qquad (3)$$

where *X* = *X*1;*X*2……,*Xn* denotes the set of attributes, *C* = *C*1;*C*2…..,*Cd* denotes the finite set of possible class labels, and *Cnb* denotes the class label output by the Naive Bayes classifier , In a text categorization task Naive Bayes classifier is,

$$C(d) = argmax_{C_j \epsilon C}\, P(C_j) \prod_{w_i \epsilon d} P(w_i \mid C_j) \qquad (4)$$

where *d* denotes a document represented as a stream of words *w* = *w*1,*w*2….,*wn*, *C* = *C*1,*C*2….,*Cd* denotes the finite set of possible class labels, and *C*(*d*) denotes the class label output by the Naive Bayes classifier.

K-means is one of the simplest unsupervised learning algorithms that explain the well known clustering problem. The major scheme is to describe k centroids, one for each cluster [3]. K-means clustering solves

$$\arg min_c \sum_{i=1}^{k} \sum_{x \epsilon c_i} d(X, \mu_i) = \arg min_c \sum_{i=1}^{k} \sum_{x \epsilon c_i} ||X - \mu_i||_2^2 \qquad (5)$$

Where, ***ci*** is the set of points that fit in to cluster *i*. The K-means clustering uses the square of the Euclidean distance

$$d(X, \mu_i) = ||X - \mu_i||_2^2 \qquad (6)$$

The K-means algorithm simply hopes to locate the global minimum, perhaps getting fixed in a special solution.

IV PROPOSED METHOD

An Objective of proposed System is to the underutilization of CPU processes, the growing importance of MapReduce performance and to establish an efficient data analysis framework for handling the large data Drift in the workloads from enterprise through the exploration of data handling mechanism like parallel database such as Hadoop. Our aids are threefold: first, we propose a Efficient framework for task and resource provisioning solution. Second, typical wide ranging workloads: parallel database is utilized for parallel process like evaluating the jobs, loading job and building the indexes for statistical study in the data centers, Third we use mapreduce programming model for effective processing and utilizing the hybrid structure to process the dataset with less utilization time and high throughput.

The advantage of the proposed system incorporates modeling and analyzing statistics data. This system handles the identifying error, and supporting dynamic load balancing, therefore, to help server users attain beloved computing results by competently utilizing system resources in terms of less cost, high performance and trade-offs among cost and performance.

As Enhancement, we propose a classification technique in the mapreduce Models for data evolution through the

Ensemble classifier in the efficient data analysis framework to cater the workloads and processing of it in commodity clusters in the data centres which yields better performance in terms of High throughput through efficient selective forwarding technique, efficient in terms of fault handling.

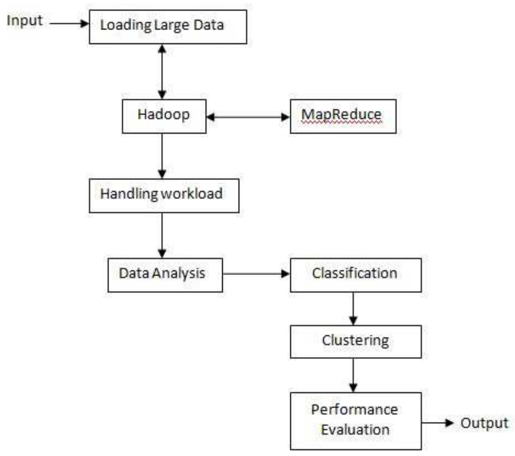

Fig 2: Methodology Architecture

V EXPERIMENTAL SETUP AND RESULTS

The Proposed analysis and performance using hadoop were experimented under certain hardware and software constraints and requirements. The hardware platform for the experiments is Intel Core 2 Duo CPU, 2.27 GHz, RAM 2GB, and total storage 250GB Hard Disk. The software development has been carried out in Hadoop-0.20.2, Eclipse hellio, Java 1.7, Microsoft Windows 7 Ultimate 32-bit operating system environment. The Hadoop package contains the necessary Java Archive files and scripts to start Hadoop.

Table 1: Parameters and Description

parameter	description
(k,v)	Key and value
(T1, Tm)	Execution time of the single and parallel task.
T(1,D)	Execution time of the single task with respect to scale of data
T(m,mD)	Execution time of the parallel task with respect to scale of data
T(m,nD)	Execution time of the single task with respect to size
T(m,D)	Execution time of the parallel task with respect to size

It was observed that the MapReduce tool is much efficient in data optimization and very reliable since it reduces the time of data access or loading by more than 50%. Experiments show the computation performance of the MapReduce prototype tool based on the measurement of speedup, scaleup and the sizeup**.**

Table 2: Experimental Result

Data Size\ Execution time	Single node
1x (148)	7
2x (296)	12
4x (592)	15
8x (1184)	20

Each run executed the entire data mining process, Mapping, splitting, shuffle, combine, and Reducing attempts and producing the output which was a selected field's content with a computed answer.

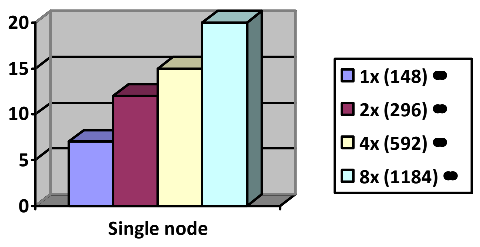

Fig 3: Comparative Result for Data and Node

VI PERFORMANCE ANALYSIS

Performance improvements of large data sets using hadoop mainly consider the trend in time consumption with the increase in the volume of data, and we try to show the difference in run time between a single PC implementation and the parallel Hadoop implementation. Three measurements were used for the evaluations: speedup, scaleup, and sizeup. Speedup tries to evaluate the ability of the parallelism to improve the execution time. Scaleup evaluates the ability of parallelism to grow both the MapReduce system and the data size, that is, the scalability of the MapReduce tool. Sizeup evaluates the ability of the parallelism to handle growth. It measures how much longer it takes to execute the parallel tasks, when the data size is n-times larger than the original datasets.

Table 3: Scaleup, Sizeup, Speedup Results

Cluster Nodes	Scaleup	Sizeup	Speedup
2	0.95	2	2.06
4	0.89	3	3.86
8	0.72	4	5.32
16	0.59	7.86	6.36

Our results compares with the various results output during the experimental study, including the evaluations of the Speedup, Sizeup and Scaleup ability of the parallelism to handle growth.

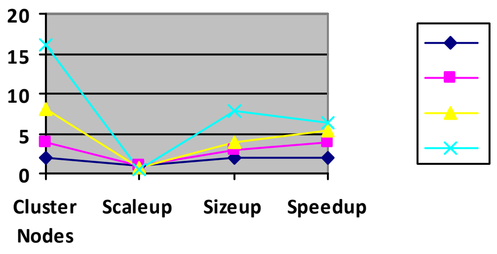

Fig 4: Comparative Results for Scaleup, Sizeup, Speedup

The speedup increases significantly as the sizeup of the cluster grows inversely to the scaleup. It is evidently shown that this is unreliable and inefficient, expensive when the same computation is done in a cluster without the use of Hadoop MapReduce technique in a distributed file system.

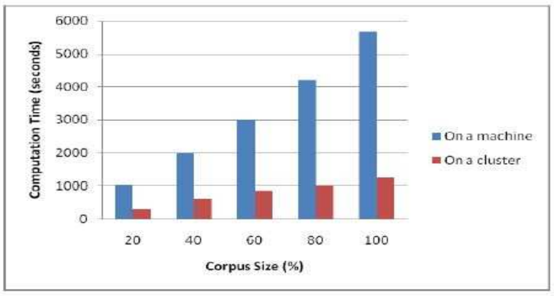

Fig 5: Comparative Result for Computation Time

We can see that the running time of the data on a Hadoop cluster relatively much less on a single machine

VII CONCLUSION

The study successfully developed and implemented a MapReduce for data mining and data optimization. In this work, we consider the problem of finding a large data that minimizes the overall completion time, performance, load balancing, speed of a given data. The evaluation results show satisfactory performance in the parameters such as speedup, scaleup, performance, time and sizeup. Further work on improving this methodology by making use of latest tools, procedures and technologies to solve issues of data optimization, reliability, scalability and data security, among other issues fault tolerance in cluster computing in distributed system.

REFERENCES

[1] Verma, Abhishek, Ludmila Cherkasova, and R. Campbell. "Orchestrating an Ensemble of MapReduce Jobs for Minimizing Their Makespan." (2013): 1-1.

[2] Xie, Jiong, FanJun Meng, HaiLong Wang, HongFang Pan, JinHong Cheng, and Xiao Qin. "Research on Scheduling Scheme for Hadoop Clusters." *Procedia Computer Science* 18 (2013): 2468-2471.

[3] Asha, T., U. M. Shravanthi, N. Nagashree, and M. Monika. "Building Machine Learning Algorithms on Hadoop for Bigdata." *International Journal of Engineering and Technology* 3, no. 2 (2013).

[4] Wan, Jian, Wenming Yu, and Xianghua Xu. "Design and implement of distributed document clustering based on MapReduce." In *Proceedings of the Second Symposium International Computer Science and Computational Technology (ISCSCT), Huangshan, PR China*, pp. 278-280. 2009.

[5] Shang, Weiyi, Zhen Ming Jiang, Hadi Hemmati, Bram Adams, Ahmed E. Hassan, and Patrick Martin. "Assisting developers of big data analytics applications when deploying on hadoop clouds." In *Proceedings of the 2013 International Conference on Software Engineering*, pp. 402-411. IEEE Press, 2013.]

[6] Shvachko, Konstantin, Hairong Kuang, Sanjay Radia, and Robert Chansler. "The hadoop distributed file system." In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pp. 1-10. IEEE, 2010.

[7] Zhao, Yaxiong, and Jie Wu. "Dache: A data aware caching for big-data applications using the MapReduce framework." In *INFOCOM, 2013 Proceedings IEEE*, pp. 35-39. IEEE, 2013.

[8] Begoli, Edmon, and James Horey. "Design Principles for Effective Knowledge Discovery from Big Data." In *Software Architecture (WICSA) and European Conference on Software Architecture (ECSA), 2012 Joint Working IEEE/IFIP Conference on*, pp. 215-218. IEEE, 2012.

[9] Borthakur, Dhruba, Jonathan Gray, Joydeep Sen Sarma, Kannan Muthukkaruppan, Nicolas Spiegelberg, Hairong Kuang, Karthik Ranganathan et al. "Apache Hadoop goes realtime at Facebook." In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pp. 1071-1080. ACM, 2011.

[10] Moturi, Christopher A., and Silas K. Maiyo. "Use of MapReduce for Data Mining and Data Optimization on a Web Portal." *International Journal of Computer Applications* 56 (2012).

[11] Markonis, Dimitrios, Roger Schaer, Ivan Eggel, Henning Müller, and Adrien Depeursinge. "Using MapReduce for Large-Scale Medical Image Analysis." In *HISB*, p. 1. 2012.

[12] Vaidya, Madhavi. "Parallel Processing of cluster by Map Reduce." *International Journal of Distributed & Parallel Systems* 3, no. 1 (2012).

[13] Almeer, Mohamed H. "Cloud Hadoop Map Reduce For Remote Sensing Image Analysis." *Journal of Emerging Trends in Computing and Information Sciences* 3, no. 4 (2012): 637-644.

[14] Zhang, Junbo, Jian-Syuan Wong, Tianrui Li, and Yi Pan. "A comparison of parallel large-scale knowledge acquisition using rough set theory on different MapReduce runtime systems." *International Journal of Approximate Reasoning* (2013)

[15] Plantenga, Todd D., Yung Ryn Choe, and Ann Yoshimura. "Using performance measurements to improve mapreduce algorithms." *Procedia Computer Science* 9 (2012): 1920-1929.