PERFORMANCE ENHANCEMENT OF SQL-ON-HADOOP ENVIRONMENT USING PARAMETERS TUNING

Mansi Bhadoria[1], Dr.Sachin Goyal[2], Dr.Ratish Agarwal[3] ,Dr.Mahesh Pawar[4]
[1]DDIPG Scholar Department of Information Technology
[2]Assistant Professor Department of Information Technology
[3]Assistant Professor Department of Information Technology
[4]Assistant Professor Department of Information Technology
UNIVERSITY INSTITUTE OF TECHNOLOGY,
RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA, BHOPAL

Abstract— the conventional relational knowledge based systems cannot accommodate the necessity of analyzing data with enormous capacity and diverse formats, i.e., Big Data. Apache Hadoop since the 1st generation of ASCII text files massive knowledge resolution delivered a steady circulated knowledge space and resource supervision arrangement. Nevertheless, MapReduce framework, sole passage of employing the equivalent computing authority of Hadoop is that the API. Given a drag, one wishes to code a corresponding MapReduce platform in Java, which is time consuming. Moreover, Hadoop concentrates upon high output as an alternative of low latency. Therefore, Hadoop will be a poor fit interactive processing.

The necessity of interactive Big Data administering required decoupling of data storage from analysis. The simple SQL queries of traditional relational database systems is however the most accessible analyzing tool that people devoid of programming backdrop can also profit from. Consequently, Big Data SQL engines have been twirled off in the Hadoop Ecosystem. Hence , we can be managing apache hive which is in a hadoop ecosystem and functions similar as SQL so effortlessly work on bigdata by by means of SQL (Hive) on Hadoop (Storing and handling bigdata). In this dissertation we can also send off recommended that parameter tuning is also significant in terms of processing for the reason that accurate tuning takes fewer time and throughout which we can certainly analyze bigdata in a reduced amount of time

Keywords-- Hadoop, SQL, performance, tuning parameters, Hive, Mapreduce

I. INTRODUCTION

Hadoop could be a widespread open supply computer code framework that enables the distributed process of massive scale information sets [1]. It employs the MapReduce paradigm to divide the computation tasks into elements that may be distributed to a goods cluster and so, provides horizontal scaling [2-9]. The MapReduce functions of Hadoop manipulates (key,value) pairs as format. The input is retrieved in chunks from Hadoop Distributed file system (HDFS) and appointed to 1 of the mappers which will method information in parallel and turn out the (k1,v1) pairs for the reduce step. Then, (k1,v1) try goes through shuffle section that assigns constant k1 pairs to constant reducer. The reducers associate the pairs with constant k1 values into teams also implements aggregation operations.

HDFS is that the underlying files storage system of Hadoop. As a result of its simplicity, reliable, fault-tolerance along with potency Hadoop has gained important support from each business and academia; but, there square measure some limitations in terms of its interfaces and performance [10]. Querying the information with Hadoop as during ancient RDBMS infrastructure is one in every of the foremost common issues that Hadoop user face.

Hadoop Hive is associate degree open supply SQL-based distributed warehouse system that is planned to resolve the issues mentioned higher than of by providing associate degree SQL-like abstraction together with Hadoop framework for querying information hold on during a cluster [11].

When users demand to acquire acquaintance from each MapReduce and SQL, mapping SQL statements to MapReduce tasks will become awfully tough job. This job is done by Hive by decoding queries to MapReduce jobs; in this manner, make use of the scalability of Hadoop whereas presenting a well-known SQL abstraction. These attributes of Hive create it an acceptable tool for information warehouse applications wherever giant scale information is analyzed, quick response times don't appear to be needed, and there's no ought to update information oftentimes [4].

Nearly all information warehouse applications are enforced maltreatment SQL-based RDBMSs, Hive take down the barrier to traversing these applications to Hadoop, thus, those already grasp SQL will simply employ Hive. Since Hive relies on a question-at-a-time model and processes every query severally, supplying multiple queries in shut measure decreases performance of Hive as an outcome of its implementation model. From this angle, it's vital to notice , no study, to date, that includes the Multiple-query improvement (MQO) technique for Hive to scale back the whole accomplishment phase of a batch of queries [1].

HADOOP

The Apache Hadoop project develops ASCII text file software package for ascendible, reliable, distributed computing. The Apache Hadoop [5] library is a framework that allows for the distributed process of huge data sets on the thousands of connected pc's called cluster of procedure freelance computers and enormous quantity (terabytes, petabytes) of information. Hadoop was derived from Google filing system (GFS) and Google's Map scale back. Apache Hadoop is considerate alternative for twitter analysis because it works for distributed immense information. Hadoop runs applications exploitation the MapReduce rule, wherever the information is processed in parallel on completely different clusters nodes.

APACHE HIVE

Facebook created Hive for analyzing giant datasets. It's a most generally adopted information reposition application which might offer the relative model and SQL interface. Hive infrastructure runs on the topmost of Hadoop. It in the main helps in providing outline of the info, question and analysis of the unstructured information. Since its incubation in 2008, Apache Hive is taken into account as normal for Batch and Interactive SQL workloads on information in Hadoop. Hive offers Hadoop users the broadest set of SQL linguistics at computer memory unit scale with interactive response times.

II. LITERATURE REVIEW

In [1] Hadoop is currently the factual customary for storage and treating huge knowledge, not just for unstructured knowledge however conjointly for a few structured knowledge. Consequently, providing SQL analysis practicality to the large knowledge resided in HDFS becomes countless and a lot of vital. Hive could be a foremost setup that assists SQL-like study to the facts in HDFS. In contrast, the efficiency of the primitive design of Hive isn't adequate. This results in the short occurrence of dozens of SQL-on-Hadoop systems that try and support cooperative SQL question process to the information hold on in HDFS. This paper first offers a concise technical view on modern struggle of SQL on-Hadoop systems. Then we incline to scrutinize and judge the functioning of 3 representative SQL-on-Hadoop systems, supported the TPCH benchmark. In keeping with the results, we tend to show that such systems will profit a lot of from applications of the many parallel query process techniques that are wide studied within the ancient massively data processing databases.

During this paper, they first review struggles of SQL on Hadoop systems in recent past. Then we incline to inspect 3 representative systems mistreatment the TPC-H benchmark. We discover that by applying progressive question process techniques (such as columnar storage, MPP design, be a part of optimization, and vectorized question execution) that are extensively studied by info community for several years. It's expected that with a lot of advanced parallel info techniques applied, the performance of SQL-on-Hadoop systems are often more improved. Providing high performance SQL analysis practicality to the information hold on in HDFS can attract a lot of and a lot of users to use SQL on Hadoop systems for interactive analysis as an alternate of proprietary DBMSs.

In [2], the huge knowledge is vital for mining the precious is data to boost the system performance. To attain this goal, analysis establishments and net firms develop three-type script question tools that are severally Hive supported MapReduce, Spark SQL supported RDD and Impala primarily based distributed question engine. In this paper, we tend to analyze the impact of the file format for the question time, and that we conduct that compression will scale back the number of information, thus on improve the question time. It's the most effective option to take RCFile compressed by Snappy for Hive, and it's the most effective option to take Parquet for Impala.

Moreover, Impala holds the quickest question speed in comparison with Spark SQL and Hive. Second we tend to discuss that the file format impact on both memory in addition to the central processor unit. Impala takes the file format of Parquet show sensible performance. Then we discover Parquet generated by completely different question tools show different performance. Finally, we discover the question velocity of Impala taken the file format of Parquet generated by Spark SQL is that the quickest.

In [3], this paper set forth how to regulate the definite job performance development of Spark SQL and Hive, and make an analogy of them concurrently. First, with the help of ten SQL queries, comparison among Hive and Spark SQL will be determined. By inspecting the signification of distinct compression strategies and file formats on the performance in diverse query types, we presume that Parquet is supported by Spark SQL more effectively, while it does not exhibit evident advantages for Parquet in Hive as in Spark SQL. Snappy has a better outcome on the intermediary data compression, and in regard to ORC, Parquet integrated with Snappy has the outstanding performance. Second, in Hive, we will alter the default configuration, remodel the number of Map Reduce, optimize the join strategy; dispose of the effects of data skew, making Hive performance enhanced by 10% to 75% or more depending on the workload types. Also, we optimize Spark SQL through the improvement of parallelism and join methods.

In 1980's The (Multiple Query Optimization) MQO problem was proposed and come up with an optimal global query plan adopting the MQO was shown to be NP-Hard problem [13].In a study during devising multiple queries to establish the aggregate of intra-operator parallelism in parallel databases considering the factors like memory usage, I/O load variables and CPU utilization to deprecate the overall execution interval of declustered join procedures. Beynon suggested a proxy- established infrastructure for managing data profound applications [15]. A data incorporation structure that cut down the communication expenditure by a numerous query reconstruction algorithm is put forward by [16].

IGNITE [17] are meaningful studies that put to work the micro machine theory for query operators to drain the total execution time of a query set. A unique MQO structure is advised for the current SPARQL query engines. For gigantic data analysis, a cascade-style optimizer can be preferred for Scope, Microsoft's system.

In recent past, a substantial load of research and commercial venture has centred on merging MapReduce and structured database technologies. Primarily, by two ways we can either add up MapReduce features to a parallel database or summating database technologies to MapReduce. The second way is more engaging as no broadly obtainable open source parallel database system exists, whereas MapReduce is accessible as an open source project. A SQL abstraction across MapReduce platform is provided by Hive, Pig, Scope, and HadoopDB projects to enlighten the programmers with complicated queries. SQL/MapReduce is a modern project that accounts MapReduce to handle user-defined functions (UDF).

In recent past, intriguing studies for unstructured data have been achieved manipulate MQO to MapReduce frameworks for example MRShare treats a bunch of input queries as a distinct query.In spite of some foremost MQO studies to cut down the execution time of MapReduce-based optimal clustering of single queries determining its efficiency, to our observation there is no study coinciding to ours that is associated to the MQO of Hadoop Hive by employing *insert query* records.

III PROBLEM DEFINITION

The data that we have a leaning for generating was growing in no time - as example we have a tendency to grow from a 25TB data set in 2008 to 800TB information set these days. The arrangement at that time was thus incompetent that certain daily process jobs were taking on daily to method and so the condition of concerns was merely getting worse with every passing day. We have a tendency to tend to had pressing have to be compelled to wish for infrastructure that might scale in conjunction with our data. As a result we have a tendency to tend to started exploring Hadoop as a technology to modify our scaling needs but writing a mapreduce program really troubles one presently a days for a flowery draw back that need terribly high end programming skills and to boot takes code maintenance time.

IV PROPOSED WORK

For analzing these large and complex data we use Hive which is a popular query languages like SQL and as a result users ended up spending hours (for writing mapreduce code) to write programs for even simple analysis.

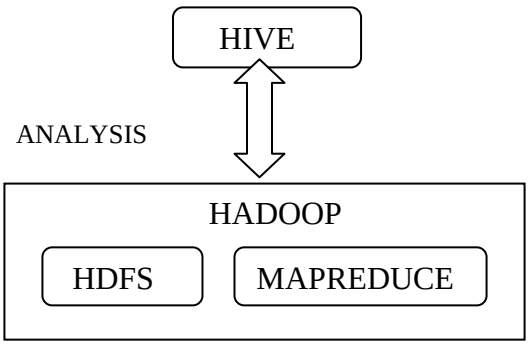


Figure.1. Workflow Diagram of SQL-on-Hadoop system

This paper we optimize the query performance of SQL-on-Hadoop by tuning the hive configuration properties. In this we can take a base system with default configuration properties in which hive is working on top of the hadoop, and we proposed a proposed system model in which we can tune the configuration properties in hive-site.xml in hive configuration folder.

Our Steps or Algorithm Steps will follow:

Step 1: First we integrate apache hive on top of the hadoop with default parameters and these system is called as base system
Step 2: store large dataset into HDFS and than analyse using hive on base system.
Step 3: Now tune the base system by setting many properties in hive configuration and he system in known as proposed system.
Step 4: Analyse the same dataset by using hive on proposed system and compare the execution time of both the system.

Enable Parallel Execution

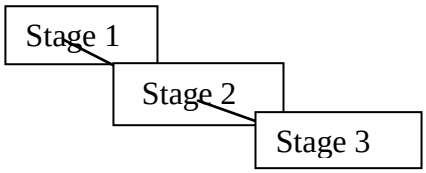
In this we proposed an parallel execution in which Hive converts a query into one or more stages. Stages could be a MapReduce stage, sampling stage, a merge stage, a limit stage. By default, Hive executes these stages one at a time. A particular job may consist of some stages that are not dependent on each other and could be executed in parallel, possibly allowing the overall job to complete more quickly. Parallel execution can be enabled by setting below properties.

Parameter	Default Value	Optimize Value
hive.exec.parallel	False	True
hive.exec.parallel.thread.number	Nil	8

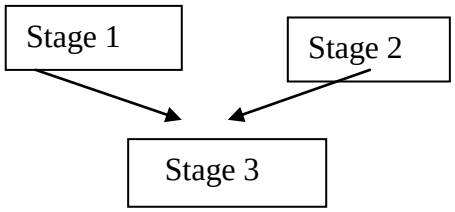
Table 1 Proposed & Improved parameter list

Parallel Execution

If hive.exec.parallel = false



If hive.exec.parallel = true



Optimize Limit Operator

In this we proposed an optimization work on limit operator because maximum times the client or analyst wants a limited record as an output like top 10 recent records so that kind of query we uses limit operator in our query. By default LIMIT operator still executes the entire query, and then only returns limited results. Because this behavior is generally wasteful, it can be avoided by setting below properties.

Parameter	Default Value	Optimize Value
hive.limit.optimize.enable	False	True
hive.limit.row.max.size	Nil	100000
hive.limit.optimize.limit.file	Nil	10
hive.limit.optimize.fetch.max	Nil	50000

Table 2. Proposed & Improved parameter list for limit operator

Single Reduce for Multi Group By

In any analyst task maximum time we can uses group by clause in our query to generates an output , So we need to optimize query which consist an group by clause for which we can uses single reducer for multi group by . By enabling single reducer task for multi group by operations, we can combine multiple GROUP BY operations in a query into a single MapReduce job. Execution can be enabled by setting below properties.

Parameter	Default Value	Optimize Value
hive.multigroupby.simplereducer	False	True

Table-3 Proposed & Improved parameter list for single reduce for multi group by

V. EXPERIMENTAL & RESULT ANALYSIS

All the experiments are perform on google cloud platform (GCP) on which we developed a heterogeneous clusters of five nodes. Cluster is implemented on ubuntu with hadoop is configure on it and top of the hadoop hive is working with default configuration,So to achieve this we are going to follow the following methods:

- Loading datasets into HDFS.
- Analyze the dataset by using Hive on Base System
- Analyze the dataset by using Hive on proposed System.

Loading Dataset into HDFS

For loading dataset into HDFS we first create an heterogenous cluster of hadoop , for which we can use a google cloud services to create a hadoop cluster. we can use Google Cloud Platform to create a cluster for five nodes, After that we can start the cluster and on master node we can operate all the functions so we just ssh to the master and the

ssh create a connection between master terminal to browser. After connection to the VM, we can access the terminal of master node and just load the dataset into HDFS by using hadoop put command and the data which is stored into the HDFS are shown in figure 1.

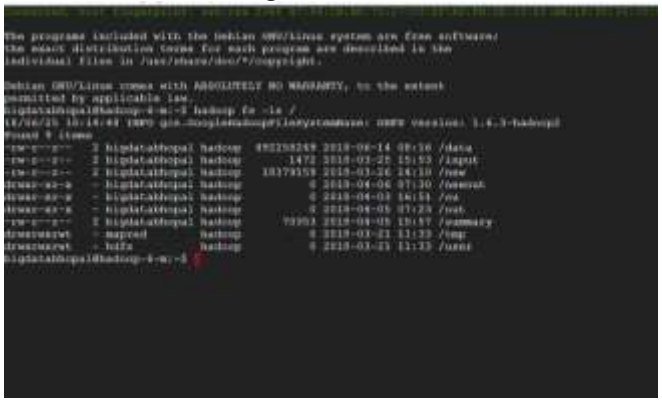


Figure 1. Dataset is loaded into HDFS

Using Parallel Execution

Analyzing the dataset by using Hive on Base System

After storing the dataset into HDFS, we are started hive over hadoop and for analyzing these data we first create a table for storing a dataset because hive works same as SQL so before analyzing we first create and stored the dataset into table. We can take h1b visa dataset for analyzing using hive. So we can analyze the h1b dataset using hive on base system means the hive is configure on default configuration parameters on top of the hadoop. we can launch a SQL query on the table and the result and time taken by the query are shown in figure 2.

Figure 2. Time taken by SQL query on base system

Analyzing the dataset by using Hive on Proposed System

After analyzing on base system be tune the hive configuration and enables various parallel thread executions mechanism on hive by adding a configuration property in hive-site.xml file on hive configuration folder. After tuning the based system we get our proposed system and we analyze the same dataset and launch the same SQL query on proposed system and the output and time taken by this are shown in figure 3.

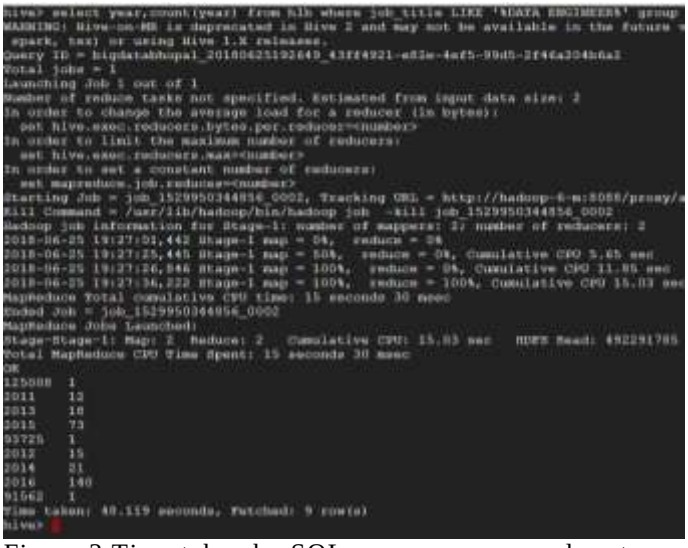


Figure 3 Time taken by SQL query on proposed system

By comparing result of both the system on structured or unstructured dataset , we can say that SQL query result are same means both are very accurate in terms of result but there is a difference between query execution time, the proposed system takes less time as compared to base system but the time difference is small because we can take 400 MB size of dataset for which only two mapper has been launched but when the dataset size is in GB's, TB's than the difference to large because parallel execution takes less time as compared to serial execution.

Name	Time duration in seconds on structured data	Time duration in seconds on unstructured data
Base System	55.87	155.261
Proposed system	48.11	148.122

Table 4. Time Taken by Query on base and proposed system

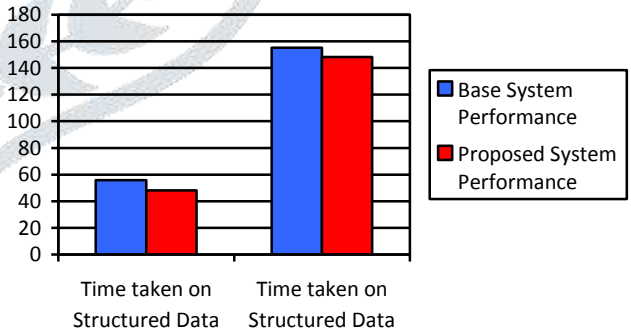


Figure 4. Time taken by both systems

Using Limit Optimization

In this we proposed an optimization work on limit operator because maximum times the client or analyst wants a limited record as an output like top 10 recent records so that kind of query we uses limit operator in our query. By default LIMIT operator still executes the entire query, and then only returns limited results. So we can launch a query on proposed system.

Name	Time duration in seconds on structured data	Time duration in seconds on unstructured data
Base System	129.823	63.211
Proposed system	124.364	60.876

Table 5. Time Taken by Query on base and proposed system

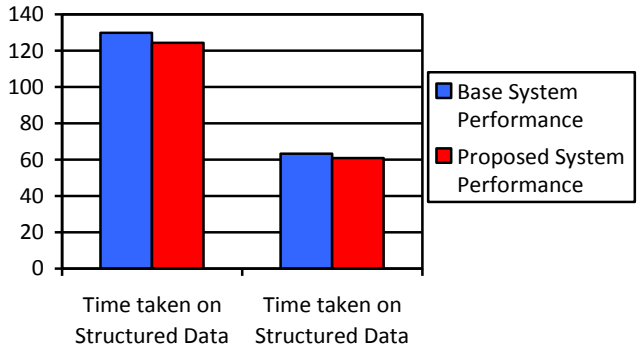


Figure 5. Time taken by both systems

Using Single Reduce For Multi Group By

In any analyst task maximum time we can uses group by clause in our query to generates an output , So we need to optimize query which consist an group by clause for which we can uses single reducer for multi group by . By enabling single reducer task for multi group by operations, we can combine multiple GROUP BY operations in a query into a single MapReduce job.

Name	**Time duration in seconds on structured data**	**Time duration in seconds on unstructured data**
Base System	129.823	155.261
Proposed system	119.494	143.704

Table 6. Time Taken by Query on base and proposed system

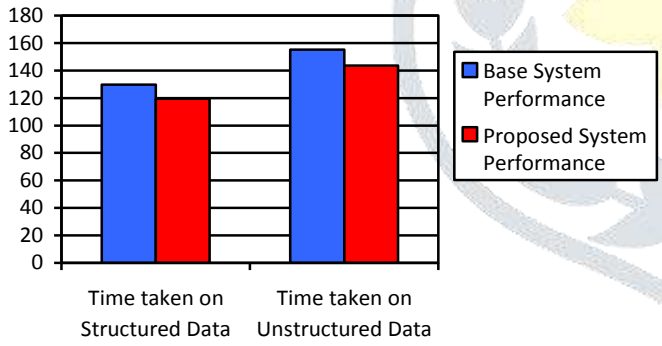


Figure 6. Time taken by both system

VI CONCLUSION:

The necessity of interactive Big Data administering required decoupling of data storage from analysis. The simple SQL queries of traditional relational database systems is however the most accessible analyzing tool that people devoid of programming backdrop can also profit from. Consequently, Big Data SQL engines have been twirled off in the Hadoop Ecosystem. Hence , we can be managing apache hive which is in a hadoop ecosystem and functions similar as SQL so effortlessly work on bigdata by means of SQL (Hive) on Hadoop (Storing and handling bigdata). In this dissertation we can also send off recommended that parameter tuning is also significant in terms of processing for the reason that accurate tuning takes fewer time and throughout which we can certainly analyse bigdata in a reduced amount of time . Subsequently we can catch hold of base system with default configuration parameters and a put forward a system in which hive can be tune by us by designing properties on hive configuration in addition to we send off a SQL query on both the system as well as we can say that proposed system accomplish query 13.78 % faster as compared to base system.

REFERENCES

[01] Xiongpai Qin, Yueguo Chen*, Jun Chen, Shuai Li, Jiesi Liu, Huijie Zhang, “The Performance of SQL-on-Hadoop Systems: An Experimental Study” in IEEE, 2017 IEEE 6th International Congress on Big Data.

[02] Divya Kamath , Praveen Srinivas , Ashika Gopal , Lanchana B.V , Suma V, " *Profiling Tool for Apache HIVE Run-time Query*" in Proceedings of the IEEE 2017 International Conference on Computing Methodologies and Communication (ICCMC)

[03] Karan Sachdeva, Japtej Singh Lamba, Vishal Sinha, Neetu Singh, " Comparison of Data Processing Tools in Hadoop" in 2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT).

[04] Xiaopeng Li, Wenli Zhou, “Performance Comparison of Hive, Impala and Spark SQL” in 2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics.

[05] Man Zhang1, Fang Liu, Yutong Lu, Zhiguang Chen, “Workload Driven Comparison and Optimization of Hive and Spark SQL’ in 2017 4th International Conference on Information Science and Control Engineering.

[06] Bing Liu. Web data mining; Exploring hyperlinks, contents, and usage data, chapter 11: Opinion Mining. Springer, 2006.

[07] Sandeep Kumar Dewangan*, Shikha Pandey† and Toran Verma, “A Distributed Framework for Event Log Analysis using MapReduce” in 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)

[08] McKinsey, Big Data: The Next Frontier for Innovation, Competition, and Productivity, McKinsey & Company, 2011, http: //www.mckinsey.com/.

[09] White Paper Big Data Analytics Extract, Transform, and Load Big Data with Apache Hadoop-Intel corporation.

[10] Prof. Manjunath R., Tejus, Channabasava R.K, Prof. Balaji S., “A BigData MapReduce Hadoop Distribution Architecture for Processing Input Splits to solve the Small Data Problem”in 978-1-5090-2399-8/16/2016 IEEE

[11] M.Santhanakumar and C.Christopher Columbus, “Web Usage Analysis of Web pages UsingRapidminer”, WSEAS Transactions on computers, EISSN: 2224-2872, vol.3, May 2015.

[12] ShailyG.Langhnoja ,MehulP.Barot and DarshakB.Mehta, “Web Usage Mining Using Association Rule Mining on Clustered Data for Pattern Discovery “,International Journal of Data Mining Techniques and Applications, vol.2 ,Issue.1, June 2013.

[13] Web server logs ://http. Sever side log.org.

[14] Nanhay Singh, Achin Jain, Ram and Shringar Raw, “Comparison Analysis of Web Usage Mining Using Pattern Recognition Techniques”, International Journal of Data Mining & Knowledge Process(IJDKP) vol.3, Issue.4, July 2013.

[15] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, “The Hadoop Distributed File System,” in the 26th IEEE Symposium on Mass Storage Systems and Technologies, pp. 1-10, May 2010.

[16] J.Srivastava et al, “Web usage Mining: Discoveryand Applications of usage patterns from Web Data“, ACM SIGKDD Explorations, vol.1, Issue. 2, pp.12-23, 2000.