

Mining of Web Server Logs in a Distributed Cluster Using Big Data Technologies

Savitha K

Dept. of Computer Science, Research Scholar
PSGR Krishnammal College for Women
Coimbatore, India.

Vijaya MS

Dept. of Computer Science, Associate Professor
GR Govindarajulu School Of Applied Computer Technology
Coimbatore, India.

Abstract—Big Data is an emerging growing dataset beyond the ability of a traditional database tool. Hadoop rides the big data where the massive quantity of information is processed using cluster of commodity hardware. Web server logs are semi-structured files generated by the computer in large volume usually of flat text files. It is utilized efficiently by Mapreduce as it process one line at a time. This paper performs the session identification in log files using Hadoop in a distributed cluster. Apache Hadoop Mapreduce a data processing platform is used in pseudo distributed mode and in fully distributed mode. The framework effectively identifies the session utilized by the web surfer to recognize the unique users and pages accessed by the users. The identified session is analyzed in R to produce a statistical report based on total count of visit per day. The results are compared with non-hadoop approach a java environment, and it results in a better time efficiency, storage and processing speed of the proposed work.

Keywords—Big data; Hadoop; Mapreduce; Session identification; Analytics

I. INTRODUCTION

A data is a collection of facts from the grids of web servers usually of unorganized form in the digital universe. Around 90% of data present in today's world are generated in last two years [1]. A large amount of the data available in the internet is generated either by individuals, groups or by the organization over a particular period of time. The volume of data becomes larger day by day as the usage of World Wide Web makes an interdisciplinary part of human activities. Rise of these data leads to a new technology such as big data that acts as a tool to process, manipulate and manage very large dataset along with the storage required.

Big Data is a high volume, high velocity and high variety information assets that demand cost-effective [2], innovative forums of information processing for enhanced insight and decision making. Big data, a buzz word in the business intelligence can handle petabytes or terabytes of data in a reasonable amount of time. Big data is distinct from large existing database which uses Hadoop framework for data intensive distributed applications.

Big Data analytics applies advanced analytical techniques of large datasets to discover hidden patterns and other useful information. It is performed using software tools mainly for predictive analysis and data mining. The growing number of technologies is used to aggregate, manipulate, manage and analyze big data. Some of the most prominent technologies are

[3] NoSQL databases that include Cassandra, MongoDB, redis, Hbase and Hadoop framework includes Hadoop, HDFS, Hive, Pig.

Data can come from a variety of sources and in a variety of types that includes not only structured traditional relational data, but also semi-structured and unstructured data. Machine generated data such as click stream logs and email logs of unstructured data are larger in magnitude when compared with human generated data that cannot fit in a traditional data warehouses for further analysis.

Numerous research works are carried out in web log mining, hadoop and some of them are reviewed below. Murat et al., [4] proposed the smart miner framework that extracts the user behavior from web logs. The framework used the smart session construction to trace the frequent user access paths. Sayalee Narkhede et al., [5] introduced the Hadoop-MR log file analysis tool that provides a statistical report on total hits of a web page, user activity, traffic sources. This work was performed in two machines with three instances of hadoop by distributing the log files evenly to all nodes.

Milind Bhandare et al., [6] put forth a generic log analyzer framework for different kinds of log files such as a database or file system. The work was implemented as a distributed query processing to minimize the response time for the users which can be extendable for some format of logs. Parallelization of Genetic Algorithm (PGA) was suggested by Kanchan Sharadchandra Rahate et al., [7]. PGA uses OlexGA package for classifying the document. The train model data is stored in HDFS and the test model categories the text document.

A framework for unstructured data analysis was proposed by Das et al., [8] using big data of public tweets from twitter. The tweets are stored in Hbase using Hadoop cluster through Rest Calls and text mining algorithms are processed for data analysis.

The semi structured log files are large datasets which are challenging to store, search, share, visualize and analyze. Almost 26% of web log types of data require big data technology to perform an analysis [9]. In order to improve the usage of a website and to track the user behavior, in online advertising and E-commerce the web log mining is performed using Hadoop.

The related works so far stated above performs the work with good scalability but fails to experiment the time efficiency between the different modes of hadoop and

necessity of the scalability. The proposed work analyses the working of both hadoop modes and the time efficiency in each mode specifically for semi structure log data, along which a statistical report is made.

This paper effectively utilizes the web logs of NASA website accessed by the user, to identify the session using Apache Hadoop Mapreduce in a distributed cluster. As hadoop does not enforce schema based storage, it processes the semi structured log files easily. The file holds 550MB dataset collected from various time period of the same year, and is stored in HDFS which is scattered in the cluster as blocks through high speed network. The work encompasses the use of both the modes along with non-hadoop approach. The identified session is analyzed based on date and number of times visited using R tool.

II. HADOOP MAPREDUCE

Hadoop is a flexible infrastructure for large scale computation and data processing on a network of commodity hardware. It allows applications to work with thousands of computational independent computers and petabytes of data. The main principle of hadoop is moving computations on the data rather the moving data for computation. Hadoop is used to breakdown the large number of input data into smaller chunks and each can be processed separately on different machines. To achieve parallel execution, Hadoop implements a MapReduce programming model.

MapReduce a java based distributed programming model consists of two phases: a massively parallel “Map” phase, followed by an aggregating “Reduce” phase. MapReduce is a programming model and an associated implementation for processing and generating large data sets [10]. A map function processes a key/value pair (k1,v1,k2,v2) to generate a set of intermediate key/value pairs, and a reduce function merges all intermediate values [v2] associated with the same intermediate key (k2) as in (1).

Maps are the individual tasks that transform the input records into intermediate records. A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks. The framework sorts the output of the map, which are then input to the reduce tasks. Both the input and the output of the processed job are stored in a file-system. Typically just zero or one output value is produced by the reducer. In MapReduce, a mapper and reducer is identified by the following signature,

$$\begin{aligned} \text{Map } (k1, v1) &\rightarrow [(k2, v2)] \\ \text{Reduce } (k2, [v2]) &\rightarrow [(k3, v3)] \end{aligned} \quad (1)$$

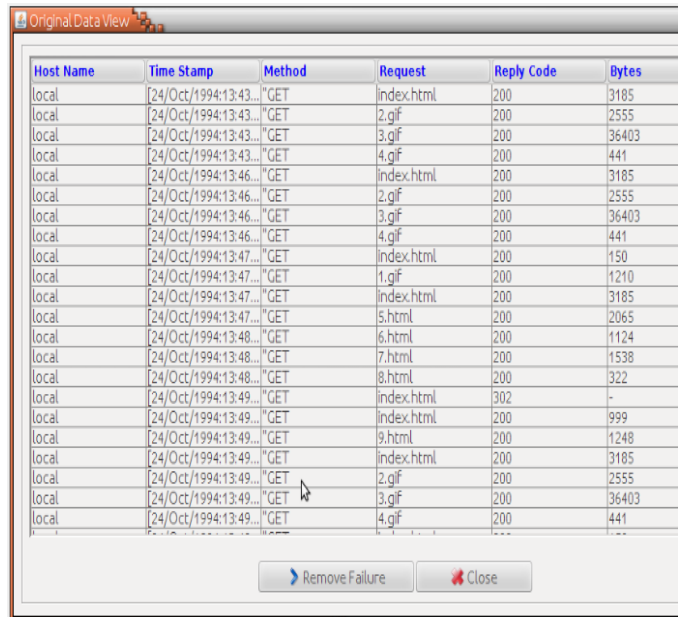
MapReduce suits applications where data is written once and read many times [12]. The data stored in a file system namespace contributes to HDFS which allows master-slave architecture.

The cluster consists of a single NameNode, a master that manages the file system namespace and regulates its access to files by clients. There can be a number of DataNodes usually one per node in the cluster which periodically report to NameNode, the list of blocks it stores.

HDFS replicates files for a configured number of times. It automatically re-replicates the data blocks on nodes that have failed. Using HDFS a file can be created, deleted, copied, but cannot be updated. The file system uses TCP/IP for communication between the clusters. A file is made of several blocks and the target machine holds each block that is chosen randomly on a block-by-block basis. Thus access to a file requires the cooperation of multiple machines, but supports file sizes larger than a single-machine DFS.

III. WEB LOG MINING USING NON HADOOP APPROACH

The current processing of log files goes through ordinary sequential ways in order to perform preprocessing, session identification and user identification. The developed non hadoop approach loads the log file dataset, to process each line one after another. The log field is then identified by splitting the data and by storing it in an array list as shown in Fig.1. The preprocessed log field is stored in the form of hash table, with key and value pairs, where key is the month and value is the integer representing the month.



Host Name	Time Stamp	Method	Request	Reply Code	Bytes
local	[24/Oct/1994:13:43...	*GET	index.html	200	3185
local	[24/Oct/1994:13:43...	*GET	2.gif	200	2555
local	[24/Oct/1994:13:43...	*GET	3.gif	200	36403
local	[24/Oct/1994:13:43...	*GET	4.gif	200	441
local	[24/Oct/1994:13:46...	*GET	index.html	200	3185
local	[24/Oct/1994:13:46...	*GET	2.gif	200	2555
local	[24/Oct/1994:13:46...	*GET	3.gif	200	36403
local	[24/Oct/1994:13:46...	*GET	4.gif	200	441
local	[24/Oct/1994:13:47...	*GET	index.html	200	150
local	[24/Oct/1994:13:47...	*GET	1.gif	200	1210
local	[24/Oct/1994:13:47...	*GET	index.html	200	3185
local	[24/Oct/1994:13:47...	*GET	5.html	200	2065
local	[24/Oct/1994:13:48...	*GET	6.html	200	1124
local	[24/Oct/1994:13:48...	*GET	7.html	200	1538
local	[24/Oct/1994:13:48...	*GET	8.html	200	322
local	[24/Oct/1994:13:49...	*GET	index.html	302	-
local	[24/Oct/1994:13:49...	*GET	index.html	200	999
local	[24/Oct/1994:13:49...	*GET	9.html	200	1248
local	[24/Oct/1994:13:49...	*GET	index.html	200	3185
local	[24/Oct/1994:13:49...	*GET	2.gif	200	2555
local	[24/Oct/1994:13:49...	*GET	3.gif	200	36403
local	[24/Oct/1994:13:49...	*GET	4.gif	200	441

Fig. 1. Log file stored as table with unique fields

The developed work is possible to run only on single computer with a single java virtual machine (jvm). A jvm has the ability to handle a dataset based on RAM i.e. if the RAM is of 2GB then a jvm can process dataset of only 1GB. Processing of log files greater than 1GB becomes hectic. The non hadoop approach is performed on java 1.6 with single jvm. Although batch processing can be found in these single-processor programs, there are problems in processing it due to limited capabilities. Therefore, it is necessary to use parallel processing approach to work effectively on massive amount of large datasets.

IV. LOG MINING USING HADOOP APPROACH

Hadoop framework access a large semi structure data in a parallel computation model. Log files usually generated from the web server comprise of large volume of data that cannot be

handled by a traditional database or other programming languages for computation. The proposed work aims on preprocessing the log file and keeps track on sessions accessed by the user, using Hadoop and the system architecture is shown in Fig.2. The work is divided into phases, where the storage and processing is made in HDFS.

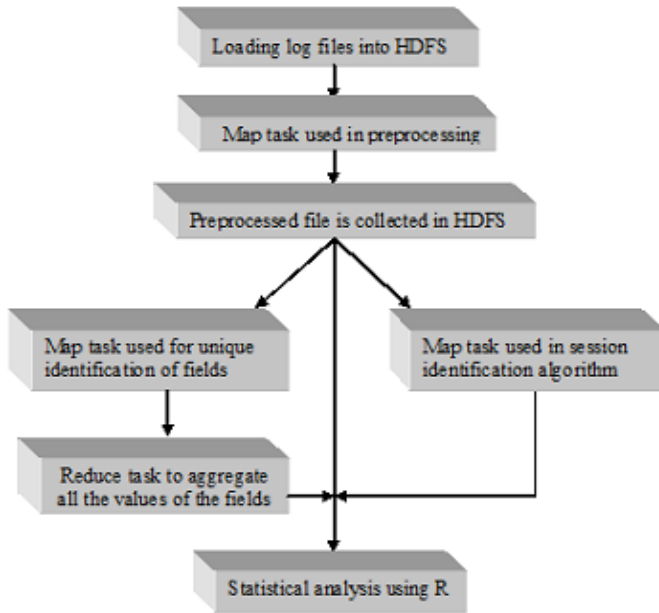


Fig. 2. System Architecture of the proposed work

Data cleaning is the first phase carried out in the proposed work as a preprocessing step in NASA web server log files. The NASA log file contains a number of records that corresponds to automatic requests originated by web robots, that includes a large amount of erroneous, misleading, and incomplete information. In the proposed work the web log file containing request from robots, spider and web crawlers are removed. These requests are a program that automatically downloads a complete website by following each hyperlink on every page. Request created by web robots are not considered as used data and consequently, it is filtered out from the NASA log data.

The entries that have status of “error” or “failure” have been removed. Also some access records generated by automatic search engine agent is identified and removed from the access log. The foremost important task carried out in data cleaning is the identification of status code. All the log lines even though satisfy the above constraints, only the lines holding the status code value of “200” is identified as correct log. The corresponding fields containing the correct status code are forwarded to the output text file.

The major advantage of this step is to eliminate an error that leads to accurate, error free log data, which produce a quality result and increase in efficiency. After applying data cleaning step, applicable resources are stored in the HDFS as text file, and feed back to the session identification algorithm as input file. The cleaned web log data is used for further

analysis of session identification utilized by the NASA users and also to identify unique user, unique URLs accessed.

Data cleaning, a preprocessing method is applied in the proposed work to filter and minimize the original size of data. The log file that resides in HDFS is given as input to the MapReduce job through FileInputFormat which is an abstract class. Input to map task is given as Long Writable and Reduce task is set to zero as there is no need for aggregation of values.

The TextInputFormat is used for text files to break the files into lines consisting of new line character. The key is the IP address and remaining all the fields are considered as values. Pattern Matching is used to separate the fields, and only log line of successful status count 200 is used for further processing. The processed line is checked to find out ‘GET’ method and records with “robots.txt” in the requested URL are identified and removed. The preprocessed log value includes the timestamp, date, browser version and total bytes sent for a request.

The session identification plays a major role in web log mining. The algorithm splits the sessions based on unique IP address and time spent by the user from login till logout. If the user exceeds time limit of 30 minutes, a new session number is generated for the similar IP address. The date function is used to calculate the time difference between the same IP address. The map function holds the session number as key and the remaining log fields as values. The reduce function is assigned to zero as no more aggregation is needed. The processed session file also resides in the HDFS that can be downloaded for further analysis.

The preprocessed log file is used to find the user identification, as Mapreduce in general identifies the unique values based on key value pair. The log file consists of different fields and the user is identified based on IP address, which is considered as key and their corresponding count as value. Once all the keys are found, the combiner is used to combine all the values of a specific key, the aggregated result is passed to the reduce task and total counts of IP accessed is listed as text file in HDFS.

Other than the identification of unique user, unique fields of date, url referred, and status code is also identified. These unique values is retrieved and used for further analysis in order to find the total url referred on a particular date or the maximum status code got successes on specific date.

V. RESULTS AND INTERPRETATIONS

The web server logs are mined for efficient session identification using Hadoop Mapreduce. This framework is used to compute the log processing in pseudo and fully distributed modes of cluster. The NASA web server logs gathered in four different files are used for processing in hadoop environment. The log data is collected from four different ASCII files of NASA with one line per request. Timestamps have 1 second resolution. The session identification algorithm is performed using the Mapreduce approaches. The log files include HTTP request of NASA web site. The process is analyzed in Ubuntu 12.04 OS with Apache Hadoop-0.20.2 [11].

A. Pseudo Distributed Mode

Hadoop framework consist of five daemons namely Namenode, Datanode, Jobtracker, Tasktracker, Secondary namenode. In pseudo distributed mode all the daemons run on local machine simulating a cluster. The job tracker and task tracker is set to idle in this mode. Preprocessing is done to eliminate the error that leads to accurate, error free log data, which produce a quality results and increase in efficiency. After applying data cleaning step applicable resources are stored in the HDFS as text file, and feed back to the session identification algorithm as input file. The cleaned web log data as shown in Fig.3 is used to analyze the session identification, unique user and unique URLs. Fig.4 depicts the storage of log data in localhost: 50070 which can be downloaded for further analysis.

```
*part-r-00000 *
jshep102.wustl.edu 28/Aug/1995 00:01:10 GET 200 HTTP/1.0 /pub/pribut/email2.gif 1209
130.34.146.126 28/Aug/1995 00:01:31 GET 200 HTTP/1.0 /pub/lischank/web/l1ne2.gif 267
130.34.146.126 28/Aug/1995 00:01:32 GET 200 HTTP/1.0 /pub/lischank/web/fwd.gif 275
130.34.146.126 28/Aug/1995 00:01:32 GET 200 HTTP/1.0 /pub/lischank/web/back.gif 883
130.34.146.126 28/Aug/1995 00:01:50 GET 200 HTTP/1.0 /pub/lischank/human.txt 35861
137.198.11.104 28/Aug/1995 00:01:31 GET 200 HTTP/1.0 /pub/atonicbk/catalog/adultcon.html 8785
151.161.17.54 28/Aug/1995 00:02:21 GET 200 HTTP/1.0 /pub/job/vk/vk-bk.jpg 4569
151.161.17.54 28/Aug/1995 00:02:21 GET 200 HTTP/1.0 /pub/job/vk/vendela.html 14961
160.15.28.30 28/Aug/1995 00:02:18 GET 200 HTTP/1.0 /pub/atonicbk/catalog/catalog.html 4654
160.15.28.30 28/Aug/1995 00:02:18 GET 200 HTTP/1.0 /pub/atonicbk/catalog/emboss.jpg 4741
160.15.28.30 28/Aug/1995 00:02:21 GET 200 HTTP/1.0 /pub/atonicbk/catalog/home.gif 813
160.15.28.30 28/Aug/1995 00:02:21 GET 200 HTTP/1.0 /pub/atonicbk/catalog/catalog.gif 693
160.15.28.30 28/Aug/1995 00:02:21 GET 200 HTTP/1.0 /pub/atonicbk/catalog/logo2.gif 12871
160.15.28.30 28/Aug/1995 00:02:23 GET 200 HTTP/1.0 /pub/atonicbk/catalog/new.gif 744
160.15.28.30 28/Aug/1995 00:02:24 GET 200 HTTP/1.0 /pub/atonicbk/catalog/orders.gif 880
164.24.med.unich.edu 28/Aug/1995 00:02:23 GET 200 HTTP/1.0 /pub/atonicbk/catalog/sleazbk.html 813
164.24.med.unich.edu 28/Aug/1995 00:02:24 GET 200 HTTP/1.0 /pub/atonicbk/catalog/home.gif 813
164.24.med.unich.edu 28/Aug/1995 00:02:24 GET 200 HTTP/1.0 /pub/atonicbk/catalog/logo2.gif 693
164.24.med.unich.edu 28/Aug/1995 00:02:24 GET 200 HTTP/1.0 /pub/atonicbk/catalog/catalog.gif 4654
164.24.med.unich.edu 28/Aug/1995 00:02:27 GET 200 HTTP/1.0 /pub/atonicbk/catalog/orders.gif 880
164.24.med.unich.edu 28/Aug/1995 00:02:27 GET 200 HTTP/1.0 /pub/atonicbk/catalog/new.gif 744
165.113.1.43 28/Aug/1995 00:01:04 GET 200 HTTP/1.0 /pub/atonicbk/home.html 4051
204.249.225.59 28/Aug/1995 00:00:34 GET 200 HTTP/1.0 /pub/rnharris/catalogs/dawsoac/intro.html 1710
204.249.225.59 28/Aug/1995 00:01:58 GET 200 HTTP/1.0 /pub/rnharris/catalogs/dawsoac/byron.gif 6423
204.253.122.109 28/Aug/1995 00:02:19 GET 200 HTTP/1.0 /pub/usenet-b/www/home.html 1613
204.56.56.285 28/Aug/1995 00:01:37 GET 200 HTTP/1.0 /pub/mikeblt/gaydc.html 6423
205.168.106.58 28/Aug/1995 00:01:01 GET 200 HTTP/1.0 /pub/bsleeper/CafeMac.GIF 26876
```

Fig. 3. Preprocessed Log Data

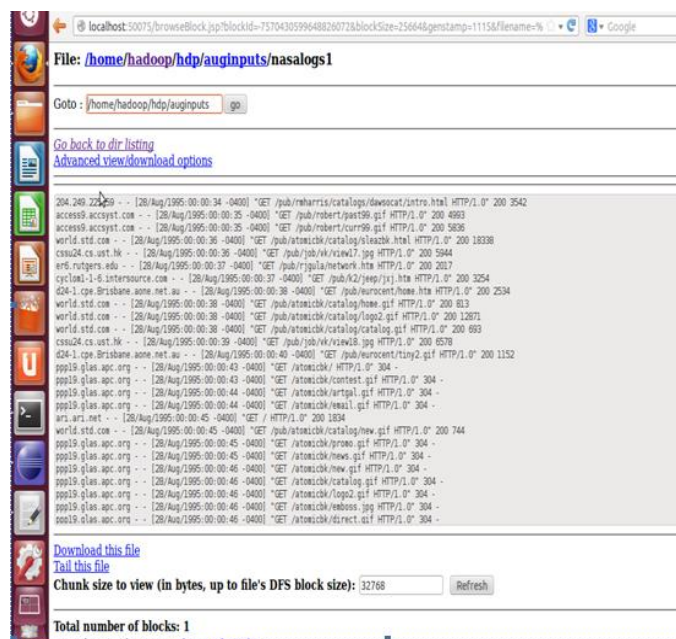


Fig. 4. File System storage in localhost

The Map phase maps one set of data to another set of data by a one-to-one rule. It is highly difficult to read the huge file without the use of HDFS where the files are loaded and processed. The total size of 516MB dataset is divided into counters or blocks of same size and processed using a single map task. The working of hadoop daemons and storage is made visible using different web user interfaces.

B. Fully distributed mode

The fully distributed mode makes use of all five daemons by assigning a job to each one. The cluster is created with two machines of same configuration with 2 GB RAM, Intel® Pentium® Dual CPU T3200 processor and Ubuntu 12.04 OS.

A cluster of two nodes is created with which one acts as slave and other as both master and slave where data is transferred in 100Mb/s speed. The master node contains daemons such as namenode, datanode, jobtracker and tasktracker. The slave node contains tasktracker and datanode. Hadoop clusters are capable to build with inexpensive computers. Even if one node fails, the cluster continues to operate without losing data by redistributing the work to remaining cluster.

The master node contains the IP address of the slave. The Name Node contains all the file system metadata of the developed cluster. The HDFS data is replicated twice as the cluster contains only one slave node. The slave node is identified by the master using its ip address and read the log files which are splitted into block size of 64MB that is default, as in Fig.5 and save it in its hard drive.

The log data is splitted into blocks and processed in two machines in parallel. The code is transmitted from the master to slave to process the work for the given block. The slave node mapper completes the preprocessing task and session identification process and shifts its data to master node. The job tracker holds the metadata of log files, the metadata includes 1 task level with 12 counters.

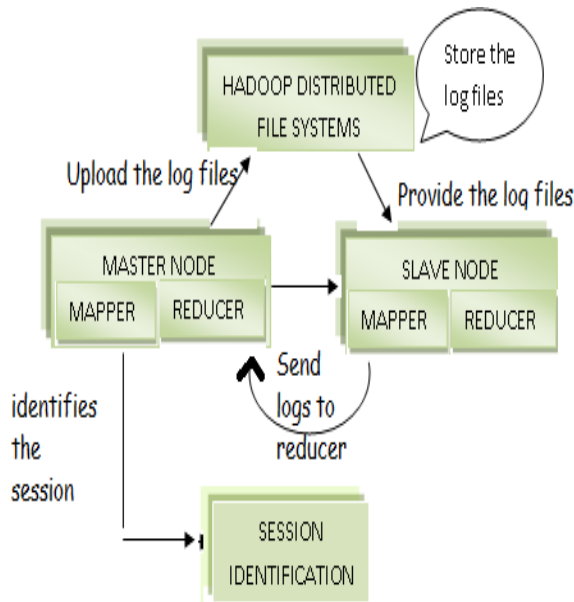


Fig. 5. Data processing in hadoop file system

The Mapreduce task in cluster is broken into record reader, mapper, combiner, and partitioner. The record reader passes the web log data to the mapper with ip address as key and the remaining fields as value. The map task groups all the data, and the combiner task is set to idle as the proposed work only uses the map task. The IP address 172.24.1.229 acts as a master and the IP 172.24.1.237 acts as master and slave. Both the machine communicates with each other using the IP. The namenode UI for the cluster can be tracked in the master node using the localhost: 54310 as shown in Fig.6 that contains the cluster summary of live and dead nodes.

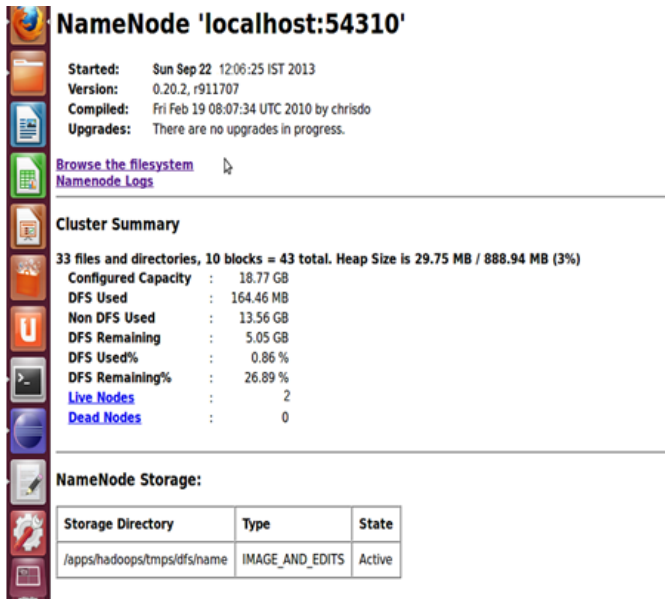


Fig. 6. Namenode UI with 2 live nodes

The preprocessing work executed in parallel results in 2.15 minutes and 3.04 minutes for session identification of 550MB NASA web log data file. The same work when performed in single node produces 2.48 minutes for preprocessing and 3.02 minutes for session identification. The performance evaluation of Non hadoop approach, Pseudo distributed mode and fully distributed mode is shown in Table-I and Fig.7, which proves that performing the work in distributed mode improves the time in few milliseconds. Expanding the cluster and using terabytes of data would result in better time efficiency.

TABLE I. PERFORMANCE OF DIFFERENT APPROACHES

NASA Server logs	Milliseconds	Minutes
Non hadoop approach	369391	6.15
Pseudo distributed mode	330001	5.50
Fully distributed mode	311400	5.21

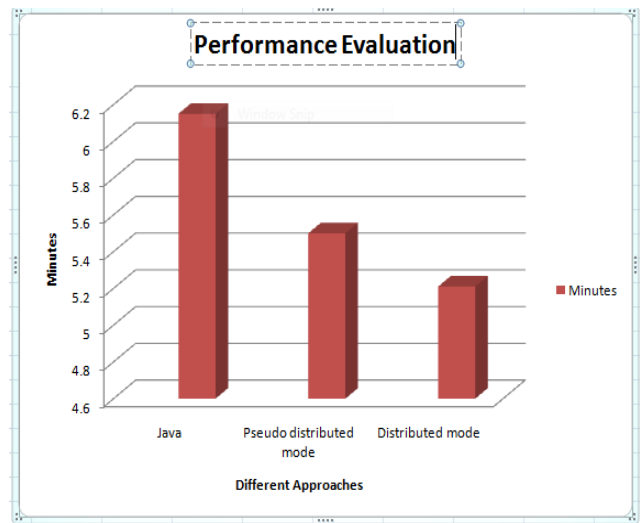


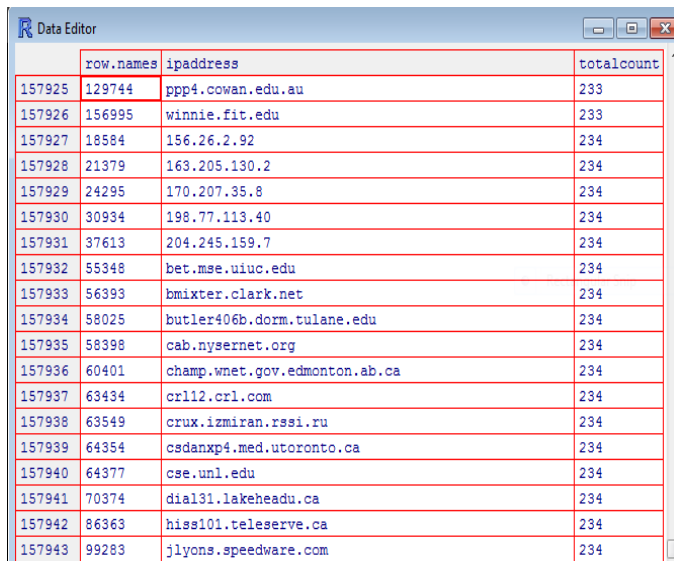
Fig. 7. Performance Evaluation of different approaches in hadoop

The preprocessed data is analyzed using R, a free statistical programming tool. The log files don't contain any specific format, due to which fields could not be separated easily. Data frames are used in the analysis of log files to read the content of the file using read.csv () [14], as comma separated values. Depending upon the format of the log file, the data in R is used to filter, analyze, or manipulate to make it more usable. Using data editor the data frame is created from the preprocessed log file with row ranging till 50, 00,000 as shown in Fig.8.

The package stringr is used to match the retrieved string from the pattern matching of logs. Stringr is used to ensure that function and argument names (and positions) are consistent. Each file that matches the regular expression is stored as data frame with column names. The individual column is taken back and stored in a separate table to find the unique values as shown in Fig.9.

	host	date	request	url
5205643	zww03.aers.psu.edu	30/Aug/1995>	-0400\tGET	/pub/mbrophy/pix/louie.jpg
5205644	zww03.aers.psu.edu	30/Aug/1995>	-0400\tGET	/pub/mbrophy/pix/sirfitz.jpg
5205645	zww03.aers.psu.edu	30/Aug/1995>	-0400\tGET	/pub/mbrophy/pix/louie.jpg
5205646	zww03.aers.psu.edu	30/Aug/1995>	-0400\tGET	/pub/mbrophy/pix/yahoo.jpg
5205647	zww03.aers.psu.edu	30/Aug/1995>	-0400\tGET	/pub/mbrophy/pix/yahoo.jpg
5205648	zzyzx.demon.co.uk	02/Jul/1995>	-0400\tGET	/shuttle/countdown/
5205649	zzyzx.demon.co.uk	02/Jul/1995>	-0400\tGET	/images/NASA-logosmall.gif
5205650	zzyzx.demon.co.uk	02/Jul/1995>	-0400\tGET	/images/KSC-logosmall.gif
5205651	zzyzx.demon.co.uk	02/Jul/1995>	-0400\tGET	/shuttle/countdown/count.gif
5205652	zzz.pe.u-tokyo.ac.>	13/Jul/1995>	-0400\tGET	/shuttle/missions/missions.html
5205653	zzz.pe.u-tokyo.ac.>	13/Jul/1995>	-0400\tGET	/images/launchedmedium.gif
5205654	zzz.pe.u-tokyo.ac.>	13/Jul/1995>	-0400\tGET	/images/NASA-logosmall.gif
5205655	zzz.pe.u-tokyo.ac.>	13/Jul/1995>	-0400\tGET	/images/KSC-logosmall.gif
5205656	zzz.pe.u-tokyo.ac.>	13/Jul/1995>	-0400\tGET	/shuttle/countdown/liftoff.html
5205657	zzz.pe.u-tokyo.ac.>	13/Jul/1995>	-0400\tGET	/shuttle/countdown/video/livevideo2.gif
5205658	zzz.pe.u-tokyo.ac.>	13/Jul/1995>	-0400\tGET	/htbin/cdr_clock.pl

Fig. 8. Data Frame with preprocessed data



	row.names	ipaddress	totalcount
157925	129744	ppp4.cowan.edu.au	233
157926	156995	winnie.fit.edu	233
157927	18584	156.26.2.92	234
157928	21379	163.205.130.2	234
157929	24295	170.207.35.8	234
157930	30934	198.77.113.40	234
157931	37613	204.245.159.7	234
157932	55348	bet.mse.uiuc.edu	234
157933	56393	bmixter.clark.net	234
157934	58025	butler406b.dorm.tulane.edu	234
157935	58398	cab.nysernet.org	234
157936	60401	champ.wnet.gov.edmonton.ab.ca	234
157937	63434	cr112.crl.com	234
157938	63549	crux.izmiran.rssi.ru	234
157939	64354	csdanxp4.med.utoronto.ca	234
157940	64377	cse.unl.edu	234
157941	70374	dial31.lakeheadu.ca	234
157942	86363	hiss101.teleserve.ca	234
157943	99283	jlyons.speedware.com	234

Fig. 9. Data Frame of unique IP address with count

VI. CONCLUSION

At present the Big data technology is successfully incorporated for all real domain problems such as web log analysis, fraud detection and text analysis. This paper reveals the importance of one of the Big data technology Hadoop, where the framework handles large amount of data in a cluster for web log mining. Data cleaning, the main part of preprocessing is performed to remove the inconsistent data. The preprocessed data is again manipulated using session identification algorithm to explore the user session. Unique identification of fields is carried out to track the user behavior. Both the algorithms process the NASA web server logs using Mapreduce task. The huge data is stored in HDFS as text files and is accessed around the cluster in blocks. From the results it is observed that the framework developed in hadoop has high performance when compared to the other approaches.

Future work can be extended to number of nodes for prediction analysis to find the next page that will be accessed by the user in a particular website. Also the resulting text files stored in HDFS can be binded with NoSQL databases, and other data mining tools for analysis.

REFERENCES

- [1] <http://www.web-datamining.net/>
- [2] Ruchi Verma, Sathyan R Mani, "Use of Big Data Tehnologies in Capital Markets," 2012 Infosys Limited, Bangalore, India.
- [3] James Manyika, Brad Brown et.al, "Big Data: The next frontier for Innovation, Competition, and Productivity," McKinsey Global Institute, June 2011.
- [4] Murat Ali , Ismail Hakki Toroslu, "Smart Miner: A New Framework for mining Large Scale Web Usage Data," WWW 2009, April 20-24. 2009 Madrid, Spain. ACM 978-1-60558-487-4/09/04.
- [5] Sayalee Narkhede and Tripti Baraskar, "HMR Log Analyzer: Analyze Web Application Logs Over Hadoop MapReduce," International Journal of UbiComp (IJU) vol.4, No.3, July 2013.
- [6] Milind Bhandare, Vikas Nagare et al., "Generic Log Analyzer Using Hadoop Mapreduce Framework," International Journal of Emerging Technology and Advanced Engineering (IJETA), vol.3, issue 9, September 2013.
- [7] Kanchan Sharadchandra Rahate et al., "A Novel Technique for Parallelization of Genetic Algorithm using Hadoop," International Journal of Engineering Trends and Technology (IJETT), vol.4, issue 8, August 2013.
- [8] T. K. Das et al., "BIG Data Analytics: A Framework for Unstructured Data Analysis," International Journal of Engineering and Technology (IJET) vol 5, No 1, Feb-Mar 2013.
- [9] Joseph McKendrick, "Big Data, Big Challenges, Big Opportunities: 2012 IOUG Big Data strategies survey," September 2012.
- [10] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," Google, Inc
- [11] "Hadoop", <http://hadoop.apache.org>.
- [12] Tom White, "Hadoop: The definitive Guide," Third Edition, ISBN: 978-1-449-31152-0-1327616795.
- [13] Ian Mitchell, Mark Locke and Aundy Fuller, "The White Book of Big Data"
- [14] [Http://en.wikipedia.org/wiki/R](http://en.wikipedia.org/wiki/R) (programming language).